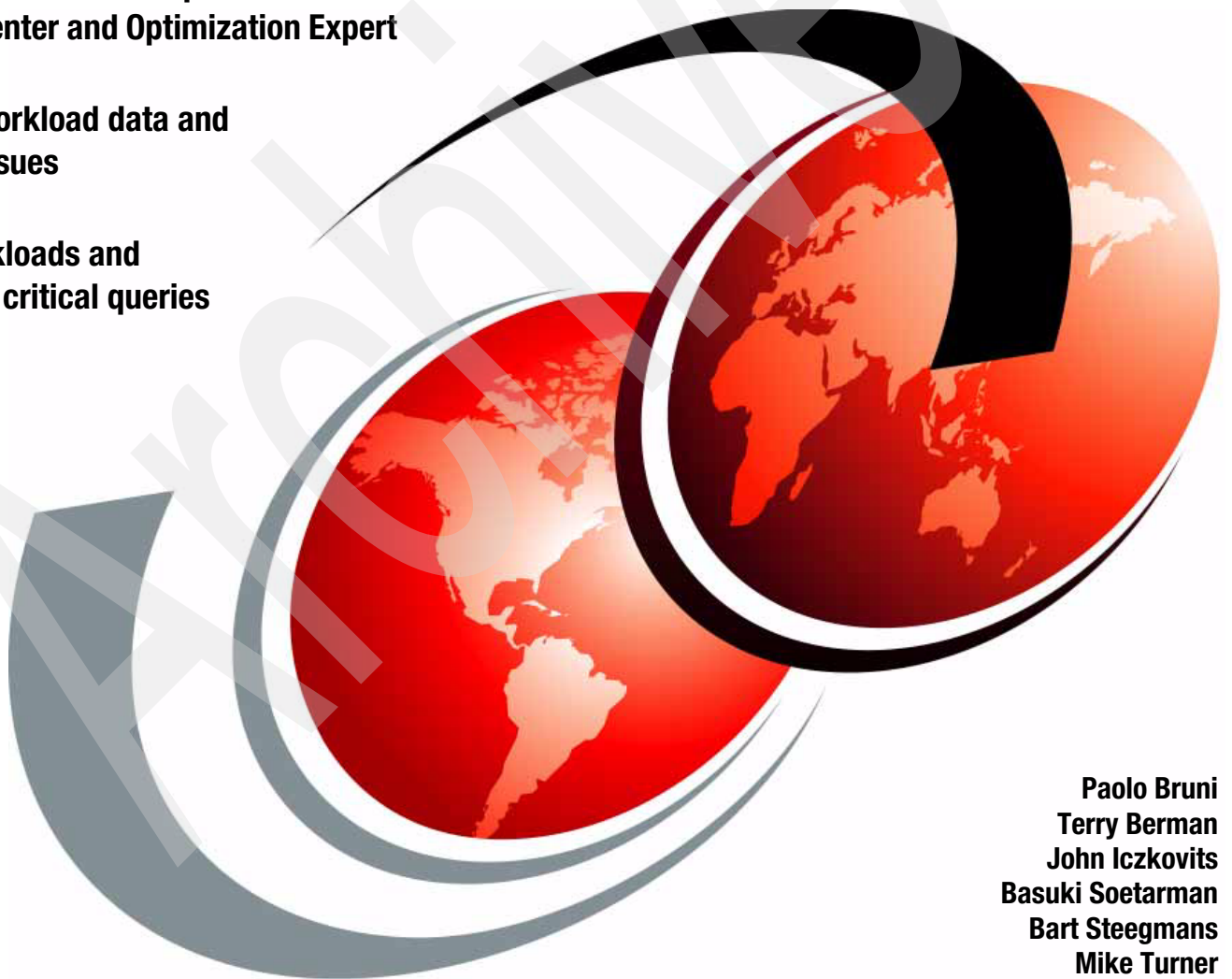


# IBM DB2 9 for z/OS: New Tools for Query Optimization

Install and customize Optimization  
Service Center and Optimization Expert

Collect workload data and  
identify issues

Tune workloads and  
individual critical queries



Paolo Bruni  
Terry Berman  
John Iczkovits  
Basuki Soetarman  
Bart Steegmans  
Mike Turner

**Redbooks**





International Technical Support Organization

**IBM DB2 9 for z/OS: New Tools for Query Optimization**

December 2007

Archived

**Note:** Before using this information and the product it supports, read the information in “Notices” on page xxi.

### **First Edition (December 2007)**

This edition applies to Version 1 Release 1 of IBM DB2 Optimization Expert for z/OS, program number 5655-S19 and Version 1 Release 1 of IBM Optimization Service Center for DB2 for z/OS. Optimization Service Center is offered as part of Version 1 Release 1 of the DB2 Accessories Suite for z/OS, product number 5655-R14. Both products are used with IBM DB2 Version 9.1 for z/OS, program number 5635-DB2.

© Copyright International Business Machines Corporation 2007. All rights reserved.

Note to U.S. Government Users Restricted Rights -- Use, duplication or disclosure restricted by GSA ADP Schedule Contract with IBM Corp.



# Contents

<b>Figures</b> .....	ix
<b>Tables</b> .....	xvii
<b>Examples</b> .....	xix
<b>Notices</b> .....	xxi
Trademarks .....	xxii
<b>Preface</b> .....	xxiii
The team that wrote this book .....	xxiii
Become a published author .....	xxv
Comments welcome .....	xxvi
<b>Part 1. Introduction</b> .....	1
<b>Chapter 1. Query optimization</b> .....	3
1.1 Autonomic computing .....	4
1.1.1 DB2 and autonomic computing .....	4
1.1.2 Total cost of ownership .....	5
1.1.3 Quality of service .....	5
1.1.4 Optimization Service Center or Optimization Expert .....	5
1.1.5 Customer adaptation .....	7
1.1.6 Extensible architecture .....	7
1.1.7 Customer value .....	8
1.1.8 Conclusion .....	9
1.2 The delivery of query optimization .....	9
1.2.1 Optimization Service Center .....	10
1.2.2 Optimization Expert .....	26
1.2.3 Product functions and DB2 support .....	29
<b>Chapter 2. Sample query optimization</b> .....	31
2.1 Starting an Optimization Service Center session .....	32
2.2 Creating a project .....	36
2.3 Investigating a query by typing the text .....	38
2.4 Investigating a query from a package .....	56
2.5 Investigating a query from the dynamic statement cache .....	66
2.6 Using the Statistics Advisor .....	71
<b>Part 2. Installation and customization</b> .....	79
<b>Chapter 3. Installing Optimization Service Center</b> .....	81
3.1 Data server and DB2 requirements .....	82
3.2 Workstation requirements .....	86
3.3 Installing Optimization Service Center on a workstation .....	86
3.3.1 Installing Optimization Service Center from the product CD .....	87
3.3.2 Installing Optimization Service Center from SMP/E .....	87
3.3.3 Installing Optimization Service Center from the Web download .....	88

3.4 Step-by-step install instructions .....	88
3.5 Performing a silent install .....	92
<b>Chapter 4. Installing Optimization Expert</b> .....	95
4.1 Data server and DB2 requirements .....	96
4.2 Workstation requirements .....	100
4.3 Installing Optimization Expert on a workstation .....	101
4.3.1 Installing Optimization Expert from the product CD .....	101
4.3.2 Installing Optimization Expert from SMP/E .....	102
4.3.3 Purchasing and downloading DB2 Optimization Expert from the ShopzSeries Web page .....	102
4.4 Step-by-step install instructions .....	103
4.5 Performing a silent install .....	107
<b>Chapter 5. Tools configuration and customization</b> .....	109
5.1 Workstation configuration .....	110
5.1.1 Identify a DB2 system .....	110
5.1.2 Connect to a DB2 system .....	113
5.1.3 Bind packages on the DB2 system (optional) .....	115
5.1.4 Enable the EXPLAIN function .....	119
5.1.5 Create user tables for workload information .....	127
5.2 Security considerations .....	128
<b>Part 3. Critical query detection</b> .....	129
<b>Chapter 6. Profile monitoring</b> .....	131
6.1 Profile monitor capabilities .....	132
6.1.1 Monitor profiles .....	132
6.1.2 Monitor pushout .....	135
6.1.3 Consolidation .....	137
6.1.4 Scheduling: Client-side or administrative (server-side) .....	137
6.1.5 Compared to other monitoring solutions .....	138
6.1.6 Data sharing limitation .....	139
6.1.7 Restarting monitoring after a DB2 restart .....	140
6.2 Operating the profile monitor .....	140
6.2.1 Viewing monitors .....	140
6.2.2 Displaying monitoring options currently in effect .....	141
6.2.3 Setting up a new profile monitor .....	142
6.2.4 Starting all monitor profiles .....	142
6.2.5 Monitor results .....	143
6.2.6 Monitoring workload projects .....	144
6.2.7 Why are you not seeing all the results? .....	145
6.2.8 Important considerations .....	145
6.3 Profile monitor DB2 tables .....	145
6.3.1 Profile monitor definitions .....	146
6.3.2 DB2 tables containing monitoring results .....	146
<b>Chapter 7. Workloads</b> .....	149
7.1 Defining a new workload .....	150
7.1.1 Workload sources .....	150
7.1.2 Tasks .....	152
7.1.3 Workloads and workload projects .....	152
7.1.4 Defining a new workload .....	152
7.1.5 Gather explain information .....	154

7.2 Working with workloads	154
7.2.1 Workload list	154
7.2.2 Workloads - operations	155
7.2.3 Workload project	156
7.2.4 Workload statements	158
7.2.5 Workload schedule tasks and history	159
<b>Part 4. Solution components</b>	<b>161</b>
<b>Chapter 8. Query tools</b>	<b>163</b>
8.1 Providing a query to analyze	164
8.1.1 Selecting a query from a workload	164
8.1.2 Selecting a query using a query project	164
8.1.3 Context setting	166
8.1.4 Explaining an SQL statement	168
8.2 Access Plan Graph	171
8.3 Query Annotation tool	180
8.3.1 Annotation	183
8.4 Query reports	191
8.4.1 Table report	193
8.4.2 Index reports	195
8.4.3 Predicate reports	196
8.5 Gather service information	198
8.5.1 Specifying SQL statement and tables to be used	198
8.5.2 Specifying which data to generate	200
8.5.3 Sending Service SQL information to IBM	203
8.6 Visual Plan Hint	204
8.6.1 Introducing optimization hints	204
8.6.2 Introducing Visual Plan Hint	204
8.6.3 Invoking Visual Plan Hint	205
8.6.4 Visual Plan Hint join graph	206
8.6.5 Visual Plan Hint general toolbar	207
8.6.6 VPH hint area	208
8.6.7 Implementing an optimization hint using Visual Plan Hint	209
8.6.8 Validating the hint	211
8.6.9 Refining the hint	212
8.6.10 Implementing the hint	217
8.6.11 Visual Plan Hint graph reports	220
<b>Chapter 9. Statistics Advisor</b>	<b>223</b>
9.1 Access Path statistics introduction	224
9.1.1 BASE Statistics	224
9.1.2 COLUMN statistics	225
9.2 Statistics Advisor analysis	227
9.3 Invoking Statistics Advisor	231
9.3.1 EXPLAIN as a pre-requisite	232
9.3.2 Statistics Advisor options	233
9.3.3 Statistics Advisor output panel	233
9.4 Sample Statistics Advisor scenarios	236
9.4.1 Conversion to DB2 V9	236
9.4.2 Typical RUNSTATS - is it sufficient?	236
9.4.3 Checking for consistent statistics	239
9.4.4 Workload Statistics Advisor sample	240
9.4.5 Statistics first-cut recommendations for very large workloads	241

<b>Chapter 10. Index Advisor</b>	243
10.1 Index Advisor overview	244
10.2 Query Index Advisor	244
10.2.1 Running Query Index Advisor	244
10.2.2 Query Index Advisor recommendations	247
10.2.3 Acting on the recommendations	250
10.2.4 Setting Query Index Advisor options for the current project	253
10.2.5 Setting Query Index Advisor option defaults for new projects	256
10.3 Workload Index Advisor	257
10.3.1 Before running Workload Index Advisor	257
10.3.2 Running Workload Index Advisor	258
10.3.3 Workload Index Advisor recommendations	265
10.3.4 What-If Analysis	268
10.3.5 Acting on the recommendations	271
10.3.6 Setting Workload Index Advisor options for the current project	272
10.3.7 Setting Workload Index Advisor option defaults for new projects	278
<b>Chapter 11. Access Path Advisor and Query Advisor</b>	283
11.1 Creating a workload from a file	284
11.2 Access Path Advisor	292
11.2.1 Running the Access Path Advisor	292
11.2.2 Access Path Advisor recommendations	295
11.2.3 Acting on the recommendations	297
11.2.4 Setting Access Path Advisor options	300
11.3 Query Advisor	300
11.3.1 Running the Query Advisor	300
11.3.2 Query Advisor recommendations	302
11.3.3 Acting on the recommendations	305
11.3.4 Setting Query Advisor options	307
11.4 Workload Query Advisor	307
11.4.1 Running the Workload Query Advisor	307
11.4.2 Workload Query Advisor recommendations	312
11.4.3 Acting on the recommendations	319
11.4.4 Setting Workload Query Advisor options	319
<b>Part 5. Tuning scenarios</b>	321
<b>Chapter 12. Analysis of a problem query</b>	323
12.1 Not optimal access path because of data correlation	324
12.1.1 The query	324
12.1.2 The current access path	324
12.1.3 Comparing optimizer estimate to real number	325
12.1.4 Query breakdown	327
12.1.5 Detecting column correlation	334
12.1.6 Collecting correlation statistics	336
12.1.7 Enhanced access path after correlation statistics	337
12.1.8 Data access cost	339
12.1.9 Conclusion	340

<b>Chapter 13. Analysis of a problem workload</b> .....	343
13.1 The selected workload .....	344
13.2 Connect to Optimization Service Center or Optimization Expert. ....	344
13.3 Defining a monitor profile .....	344
13.4 Activating monitoring and measuring the initial run. ....	349
13.4.1 Starting the monitor profile .....	350
13.4.2 Executing the workload that you want to capture .....	351
13.4.3 Obtaining monitoring profile data .....	351
13.5 Analyzing the results .....	353
13.6 Running Workload Statistics Advisor .....	354
13.7 Implementing the Workload Statistics Advisor recommendations. ....	363
13.8 Re-measuring the workload after Workload Statistics Advisor recommendations ..	363
13.8.1 Using Workload Index Advisor .....	365
13.9 Implementing the indexes suggested by Workload Index Advisor .....	372
13.9.1 Creating the indexes .....	372
13.9.2 Gathering statistics for the newly created indexes .....	372
13.10 Re-measuring the workload after Workload Index Advisor recommendations ....	373
13.11 Comparing all three workload runs .....	375
13.12 Conclusion .....	376
<b>Chapter 14. Common practices</b> .....	379
14.1 Using Optimization Service Center or Optimization Expert in application's life cycle	380
14.1.1 Usage during application design and development .....	380
14.1.2 Usage during testing or quality assurance .....	381
14.1.3 Usage during production phase .....	381
14.2 Common practice for tuning steps .....	384
14.2.1 Tuning a single query .....	384
14.2.2 Tuning a query workload. ....	388

<b>Part 6. Appendixes</b>	395
<b>Appendix A. Tips for optimizing very large static workloads</b>	397
A.1 Report all column expressions (index candidates)	398
A.1.1 Mass-explain procedure	398
A.1.2 Reporting column expressions	400
A.2 Automated RUNSTATS column lists	401
A.2.1 Mass-explain procedure	401
A.2.2 Columns in predicates (need COLUMN statistics)	401
A.2.3 Columns and literal values (need FREQVAL)	403
<b>Appendix B. Administrative Scheduler</b>	405
B.1 Purpose of the Administrative Scheduler	406
B.2 Setup of the Administrative Scheduler	406
B.3 Security for the Administrative Scheduler	406
B.4 Sample SELECT to display the task list	407
B.5 Additional information	408
<b>Appendix C. Relevant DB2 objects</b>	409
C.1 Security requirements for Optimization Service Center or Optimization Expert	410
C.2 Job DSNTIJOS	412
C.3 Job AOCDDL	417
C.4 Packages required by Optimization Service Center	419
C.5 Additional packages required by Optimization Expert	423
<b>Related publications</b>	425
IBM Redbooks publications	425
Other publications	425
Online resources	426
How to get IBM Redbooks publications	426
Help from IBM	426
<b>Index</b>	427

# Figures

1-1 Optimization Service Center or Optimization Expert . . . . .	6
1-2 Customer adaptation . . . . .	7
1-3 Autonomic computing . . . . .	8
1-4 The Accessories Suite . . . . .	10
1-5 Optimization Service Center install error if DB2 Connect not installed . . . . .	11
1-6 Optimization Service Center Welcome window . . . . .	12
1-7 DB2 Subsystems Configuration window . . . . .	13
1-8 Adding a subsystem . . . . .	14
1-9 Subsystem Properties panel . . . . .	15
1-10 New subsystem details . . . . .	16
1-11 New subsystem added successfully . . . . .	17
1-12 Obtaining the information needed to define a subsystem . . . . .	17
1-13 Connect to a subsystem . . . . .	18
1-14 Connect to a subsystem - User ID panel . . . . .	19
1-15 Checking the Optimization Service Center objects already installed . . . . .	20
1-16 Enable Explain panel . . . . .	21
1-17 The options for setting up the Explain tables . . . . .	22
1-18 Defining the names of the table spaces . . . . .	23
1-19 Create table space subpanel . . . . .	24
1-20 Confirmation of table space creation . . . . .	25
1-21 Warning that some tables already exist . . . . .	26
1-22 Optimization Expert Query Advisor . . . . .	27
1-23 OE Index Advisor . . . . .	28
2-1 Startup on DB2 Subsystem Configurations window . . . . .	32
2-2 Specify User ID and password . . . . .	33
2-3 Successful connection . . . . .	34
2-4 Connected and ready to work . . . . .	35
2-5 Create new query project . . . . .	36
2-6 Project to tune a single query . . . . .	37
2-7 Prepare to identify target query . . . . .	38
2-8 Identify target query . . . . .	39
2-9 Type in the SQL statement . . . . .	40
2-10 The Run Query option . . . . .	41
2-11 Query result . . . . .	42
2-12 Selecting the Access Plan Graph option . . . . .	43
2-13 Viewing the Access Plan Graph . . . . .	44
2-14 View the raw plan table rows . . . . .	45
2-15 The Access Plan Graph . . . . .	46
2-16 Access Plan Graph detail . . . . .	47
2-17 Table information . . . . .	48
2-18 Viewing the column definitions for a table . . . . .	49
2-19 Viewing column detail . . . . .	50
2-20 Changing the view of a column . . . . .	51
2-21 Showing attribute explanations . . . . .	52
2-22 Viewing index information . . . . .	53
2-23 Join information . . . . .	54
2-24 Help for NLJOIN object . . . . .	55
2-25 Infopop . . . . .	56

2-26	Choosing the Catalog as the query source . . . . .	57
2-27	Selecting a view . . . . .	58
2-28	Create View panel. . . . .	59
2-29	Access Path Filters tab . . . . .	60
2-30	Choose sort sequence . . . . .	61
2-31	Choose data to display . . . . .	62
2-32	The SQL statements retrieved from the DB2 Catalog . . . . .	63
2-33	Seeing the statement text . . . . .	64
2-34	The Access Plan Graph . . . . .	65
2-35	Selecting the dynamic statement cache as the source of SQL. . . . .	66
2-36	Selecting a view of the dynamic statement cache . . . . .	67
2-37	Statements returned from the cache. . . . .	68
2-38	Customizing the view . . . . .	69
2-39	Filter on primary Auth ID . . . . .	69
2-40	Selecting a statement . . . . .	70
2-41	Access Plan Graph from dynamic cache statement . . . . .	71
2-42	Selecting an Advisor . . . . .	72
2-43	Statistics Advisor asks for source of explain data . . . . .	73
2-44	Statistics Advisor recommendations . . . . .	74
2-45	Statistics Advisor details . . . . .	75
2-46	Statistics Advisor second recommendation . . . . .	76
2-47	Completion of Runstats. . . . .	77
2-48	Result of rerunning the Statistics Advisor . . . . .	78
3-1	Optimization Service Center Install - Welcome window . . . . .	89
3-2	Optimization Service Center install - License agreement window . . . . .	89
3-3	Optimization Service Center install - Save options window . . . . .	90
3-4	Optimization Service Center install - Directory window. . . . .	90
3-5	Optimization Service Center install - Directory and space verification window. . . . .	91
3-6	Optimization Service Center install - Progress window. . . . .	91
3-7	Optimization Service Center install - Successful install window . . . . .	92
3-8	Optimization Service Center install - Successful install window . . . . .	92
4-1	Optimization Expert Install - Welcome window . . . . .	103
4-2	Optimization Expert install - License agreement window . . . . .	103
4-3	Optimization Expert install - Save options window . . . . .	104
4-4	Optimization Expert install - Directory window . . . . .	104
4-5	Optimization Expert install - Directory and space verification window . . . . .	105
4-6	Optimization Expert install - Progress window . . . . .	105
4-7	Optimization Expert install - Successful install window. . . . .	106
4-8	Optimization Expert - Verifying version installed. . . . .	106
5-1	Configure Subsystem window. . . . .	110
5-2	Subsystem Location window. . . . .	112
5-3	Connect to Subsystem window. . . . .	113
5-4	Testing the connection window. . . . .	114
5-5	Configure Subsystems window with successful results . . . . .	115
5-6	Connect - Package not found window . . . . .	116
5-7	Connect - Bind window . . . . .	117
5-8	Connect - EXPLAIN tables are not enabled . . . . .	118
5-9	Connect - Enable EXPLAIN window . . . . .	119
5-10	Connect - Create EXPLAIN table . . . . .	120
5-11	Connect - Override EXPLAIN options window . . . . .	121
5-12	Connect - display successful CREATE of EXPLAIN tables . . . . .	122
5-13	Connect - EXPLAIN version mismatch window . . . . .	123
5-14	Connect - EXPLAIN tables are ready for next step. . . . .	124



5-15 Connect - existing EXPLAIN tables . . . . .	125
5-16 Connect - EXPLAIN tables created window . . . . .	126
5-17 Connect - Manage Users window . . . . .	127
5-18 Manage Users window . . . . .	128
6-1 Profile monitor architecture . . . . .	132
6-2 Client or Administrative Scheduling . . . . .	138
6-3 Enable the Administrative Scheduler . . . . .	138
6-4 View monitors panel . . . . .	141
6-5 Subsystem monitoring . . . . .	141
6-6 Monitor options . . . . .	143
6-7 Sample monitor results . . . . .	143
6-8 Workload totals . . . . .	144
7-1 Workloads and workload projects . . . . .	152
7-2 Workload Source Filter panel . . . . .	153
7-3 Workloads List . . . . .	154
7-4 Workload Project . . . . .	157
7-5 Workload Statements . . . . .	158
7-6 View SQL Statement . . . . .	158
8-1 Selecting a Query Tool from a workload . . . . .	164
8-2 Selecting a Query Tool in a query project . . . . .	165
8-3 Query tools invocation from project window . . . . .	166
8-4 Specify Query Context invocation . . . . .	167
8-5 Query context options . . . . .	168
8-6 Preferences Re-EXPLAIN window . . . . .	169
8-7 Explain Option Confirmation window . . . . .	170
8-8 Specifying Explain options using the Query text window . . . . .	171
8-9 Access Plan Graph . . . . .	172
8-10 Search Node . . . . .	173
8-11 Bookmarks and History . . . . .	174
8-12 Access Plan Graph buttons . . . . .	174
8-13 View SQL statement . . . . .	175
8-14 Graph Overview window . . . . .	176
8-15 Print options window . . . . .	177
8-16 Report Options window . . . . .	178
8-17 Access Path Report in HTML format . . . . .	179
8-18 Change Settings window . . . . .	180
8-19 Original query in Query Annotation . . . . .	181
8-20 Transformed query in Query Annotation . . . . .	182
8-21 Annotated query . . . . .	183
8-22 Customize Annotations window . . . . .	184
8-23 Annotation sort and order predicate options . . . . .	185
8-24 Annotation highlighting . . . . .	186
8-25 Access path graph . . . . .	188
8-26 Transformed annotated query . . . . .	190
8-27 Query reports tab . . . . .	191
8-28 Open report files . . . . .	192
8-29 HTLM table report . . . . .	193
8-30 Specify objects . . . . .	199
8-31 Generate options . . . . .	200
8-32 Generate Report result . . . . .	202
8-33 Initial VPH tab . . . . .	206
8-34 Join graph with added predicates . . . . .	207
8-35 VPH top toolbar . . . . .	207

8-36 VPH Hint area . . . . .	208
8-37 Default join sequence . . . . .	208
8-38 Hint Customization Rule window . . . . .	209
8-39 Invalid association message . . . . .	210
8-40 Leading table disabled message . . . . .	210
8-41 Hint area after join sequence change . . . . .	211
8-42 Plant Hint Summary window . . . . .	212
8-43 ACCOUNTS table hints customization rule. . . . .	213
8-44 VPH Hints Criteria area. . . . .	214
8-45 hint summary after index change . . . . .	215
8-46 Customizing prefetch . . . . .	216
8-47 Changed hint criteria . . . . .	216
8-48 Hint customization rule for CUSTOMERS. . . . .	217
8-49 Adding a new rule . . . . .	217
8-50 Hint Deployment Options . . . . .	218
8-51 Hint deployment for DSNTEP2 . . . . .	218
8-52 Hint successfully deployed . . . . .	219
8-53 Hint already exists. . . . .	219
8-54 Visual Plan Hint window after implementing the hint . . . . .	220
8-55 Visual Plan Hint Report. . . . .	221
9-1 Type of statistics . . . . .	224
9-2 Statistics Advisor. . . . .	227
9-3 Workload Weighting Options . . . . .	230
9-4 Workload Project Run Advisors . . . . .	232
9-5 Workload Tools - Statistics Advisor . . . . .	232
9-6 Workload Statistics Advisor - Schedule . . . . .	232
9-7 Workload EXPLAIN required . . . . .	232
9-8 Statistics Advisor result panels . . . . .	234
9-9 Statistics Advisor Detail Report. . . . .	235
9-10 High priority - Run repair RUNSTATS. . . . .	238
9-11 Statistics Advisor Sample Scenario - Detail report extract . . . . .	239
9-12 Workload Statistics Advisor - Monitor sample. . . . .	240
9-13 Workload Statistics Advisor - Recommendations . . . . .	240
9-14 Workload Statistics Advisor - Sample COMPLETE RUNSTATS . . . . .	241
10-1 Specifying a single SQL statement . . . . .	244
10-2 The SQL statement. . . . .	244
10-3 The access plan graph . . . . .	245
10-4 Run the Index Advisor. . . . .	246
10-5 Asking for the source of Explain information. . . . .	246
10-6 Local EXPLAIN information not complete. . . . .	247
10-7 Index Advisor recommendations. . . . .	248
10-8 Viewing the Create Index DDL . . . . .	249
10-9 Index filtering and screening . . . . .	250
10-10 Using Optimization Expert to create the index . . . . .	251
10-11 Statistics Advisor recommendations . . . . .	252
10-12 The new Access Plan Graph . . . . .	253
10-13 The Project tab . . . . .	254
10-14 The Advisors tab . . . . .	255
10-15 Query Index Advisor Options panel . . . . .	255
10-16 Tools menu Preferences option . . . . .	256
10-17 The Preferences panel . . . . .	257
10-18 Selecting a workload. . . . .	258
10-19 The new project . . . . .	259

10-20	The text of the SQL statements	259
10-21	Run the Index Advisor.	260
10-22	EXPLAIN information needed	261
10-23	Notification that Explain task has completed	261
10-24	Switching to monitor tasks	262
10-25	Task list.	262
10-26	Explain information needed	263
10-27	Progress indicator.	264
10-28	Index Advisor recommendations.	265
10-29	Scrolling the index list	266
10-30	Show DDL.	267
10-31	Selecting an index for investigation	267
10-32	Related SQL panel	268
10-33	Specifying What-If options	269
10-34	Specifying new What-If options.	269
10-35	Results of What-If analysis	270
10-36	Open the Advisors tab	272
10-37	The Advisors tab	273
10-38	Workload Index Advisor options - Tab 1	274
10-39	Workload Index Advisor options - Tab 2	275
10-40	Workload Index Advisor options - Tab 2, scrolled down	276
10-41	Workload Index Advisor options - Tab 3	277
10-42	Workload Index Advisor options - Tab 4	278
10-43	Tools Preferences menu	279
10-44	Workload Index Advisor preferences	279
10-45	Index Policies tab	280
10-46	Tune Workload panel	280
11-1	Sample file of SQL statements	284
11-2	Create a new workload project	285
11-3	Name the project.	285
11-4	Set the workload name	286
11-5	Define the workload source	287
11-6	Specifying the input file and the table Creator ID	288
11-7	Capture workload immediately	289
11-8	Setting up the workload	290
11-9	The workload is captured	290
11-10	Building the statement list	291
11-11	The statement list	292
11-12	Selecting a statement and running Access Path Advisor	293
11-13	Access Path Advisor running	293
11-14	Access Path Advisor warnings	294
11-15	The query text	295
11-16	The Access Path Advisor warning detail.	296
11-17	The SQL statement text	296
11-18	The Description box text	296
11-19	The Explanation box text	297
11-20	Run all the Query Advisors	298
11-21	Result of running all Advisors	298
11-22	The Statistics Advisor recommendations	299
11-23	The Index Advisor recommendations	300
11-24	Typing the SQL statement to be analyzed	301
11-25	The SQL statement text	301
11-26	Run Query Advisor	302

11-27	Query Advisor recommendations . . . . .	303
11-28	Seeing the detailed recommendation . . . . .	303
11-29	Recommendation description . . . . .	304
11-30	Recommendation explanation . . . . .	304
11-31	The revised SQL statement . . . . .	305
11-32	A new project for the revised query . . . . .	306
11-33	No recommendation for the revised statement . . . . .	306
11-34	Query Advisor preferences panel . . . . .	307
11-35	Run the Workload Query Advisor . . . . .	308
11-36	Explain is needed . . . . .	308
11-37	Schedule Explain panel . . . . .	309
11-38	Schedule Explain second tab . . . . .	310
11-39	Checking the Explain task. . . . .	311
11-40	Workload Query Advisor is running . . . . .	311
11-41	Workload Query Advisor recommendations summary . . . . .	312
11-42	The Query Advisor tab . . . . .	313
11-43	See more of the SQL statement text. . . . .	313
11-44	Individual statement recommendation . . . . .	314
11-45	Recommendation details . . . . .	315
11-46	Recommendation description . . . . .	315
11-47	Recommendation explanation . . . . .	316
11-48	Recommendations for another SQL statement. . . . .	317
11-49	The first recommendation for the second statement. . . . .	317
11-50	Description of first recommendation . . . . .	318
11-51	Explanation of the first recommendation. . . . .	318
11-52	The second recommendation for the second statement. . . . .	319
11-53	Workload Query Advisor preferences panel . . . . .	320
12-1	Access plan graph using the PLAN_TABLE . . . . .	324
12-2	Query output cardinality . . . . .	326
12-3	Available indexes in Node Descriptor . . . . .	328
12-4	Generate index report. . . . .	328
12-5	Predicate to index mapping . . . . .	330
12-6	Index filtering for CUSTIX4 with parameter markers. . . . .	331
12-7	Matching versus total filter factor . . . . .	333
12-8	Colgroup statistics in node descriptor . . . . .	335
12-9	Improved access path. . . . .	338
12-10	Accessing the CUSTOMER table . . . . .	339
12-11	Data access after additional MCARD statistics. . . . .	340
13-1	Define a monitor profile. . . . .	345
13-2	Define a 'normal' monitor . . . . .	345
13-3	Define monitor source. . . . .	346
13-4	Specify monitor granularity . . . . .	347
13-5	Monitor start option . . . . .	348
13-6	Monitor definition complete. . . . .	349
13-7	Start monitor . . . . .	350
13-8	Start monitor from the client . . . . .	350
13-9	Verifying the monitor is started . . . . .	351
13-10	Monitoring snapshot . . . . .	352
13-11	Profile snapshot selection. . . . .	352
13-12	Stop monitor warning . . . . .	353
13-13	Open monitor results. . . . .	353
13-14	Initial results . . . . .	354
13-15	Start Workload Statistics Advisor . . . . .	355

13-16	Explain not run warning	355
13-17	Schedule EXPLAIN task	356
13-18	Explain task ended	356
13-19	Workload Statistics Advisor results	357
13-20	Invoke query annotation from monitored statement list	358
13-21	Query #5 annotated version	359
13-22	Access plan graph for query block accessing DISTARRANG	360
13-23	Run after implementation of Workload Statistics Advisor RUNSTATS	364
13-24	Access plan graph of Q5 after Workload Statistics Advisor	365
13-25	Start Workload Index Advisor	366
13-26	Workload Index Advisor results	367
13-27	SQL statement expected to benefit from the suggested index	368
13-28	Access plan graph of Q7	369
13-29	PLAN_TABLE output of Q7	369
13-30	Annotated version of the Q7 query	370
13-31	Workload Index Advisor Index options - Index policies	371
13-32	CREATE INDEX DDL from Workload Index Advisor	372
13-33	Workload Index Advisor results	373
13-34	Access plan graph of Q7 after Workload Index Advisor run	374
13-35	Initial run with statements sorted by SQL statement text	375
13-36	Run after Workload Statistics Advisor with statements sorted by SQL statement text	375
13-37	Workload Index Advisor results with statements sorted by SQL statement text	376
13-38	Workload performance evolution	376
14-1	Profile monitor overhead	384
14-2	Missing maximum column frequency statistics	386
14-3	Index Advisor recommendations	386
14-4	Access Path Advisor recommendation	387
14-5	Workload Query Advisor - summary result	389
14-6	Workload Query Advisor - query list	389
14-7	Workload Query Advisor - query recommendation	390
14-8	Workload Statistics Advisor - recommendations	391
14-9	Workload Statistics Advisor - details	392
14-10	Workload Index Advisor - recommendations	393

Archived

# Tables

1-1 Family product functional reference . . . . .	29
3-1 Stored procedures required by Optimization Service Center . . . . .	83
3-2 DB2 objects for Optimization Service Center Java procedures . . . . .	84
4-1 Stored procedures required by Optimization Expert . . . . .	97
4-2 DB2 objects for Optimization Expert Java procedures . . . . .	98
6-1 Externalizing monitor data . . . . .	135
6-2 Monitor results . . . . .	144
7-1 Workload Summary Status . . . . .	155
8-1 Annotation option interactions. . . . .	184
9-1 Single-column statistics . . . . .	225
9-2 Multi-column statistics. . . . .	226
12-1 Filtering of different query components. . . . .	327
12-2 Filtering of different query components. . . . .	330
12-3 Filtering of different query components. . . . .	332
12-4 Filtering of different query components. . . . .	332
12-5 Filtering of different query components. . . . .	338
13-1 Number of qualified rows for the tables of query block 2 . . . . .	360
13-2 Number of qualified rows for the tables of query block 2 after Workload Statistics Advisor . . . . .	365
A-1 EXPLAIN tables needed. . . . .	398
C-1 DSNTIJOS objects . . . . .	412
C-2 AOCCDL objects . . . . .	417

Archived



# Examples

3-1 V91CWLMJ started task . . . . .	85
4-1 V91CWLMJ started task . . . . .	99
5-1 DSNL004I display during DB2 start up . . . . .	111
6-1 Profile monitor definitions . . . . .	146
6-2 SQL to fetch Optimization Expert monitoring results . . . . .	146
8-1 Saved query with annotation. . . . .	187
8-2 SQL statement . . . . .	188
8-3 CREATE VIEW statement . . . . .	189
8-4 Text table report . . . . .	193
8-5 Text index report. . . . .	195
8-6 Text predicate report (adapted) . . . . .	197
8-7 SQL statement . . . . .	205
8-8 Hint successfully used by DSNTEP2 at execution time . . . . .	219
9-1 BETWEEN predicate SQL . . . . .	236
9-2 BETWEEN predicate RUNSTATS . . . . .	236
9-3 Statistics Advisor Sample Scenario - RUNSTATS . . . . .	237
9-4 Statistics Advisor Sample Scenario - SQL . . . . .	237
9-5 Statistics Advisor Sample Scenario - EXPLAIN . . . . .	237
9-6 Conflict statistics demonstration - Manipulation . . . . .	239
9-7 Conflict statistics demonstration - Conflict Report. . . . .	239
12-1 SQL statement . . . . .	324
12-2 Annotated query . . . . .	325
12-3 SQL statement using literals . . . . .	326
12-4 COUNT(*) of complete result . . . . .	327
12-5 Available indexes report . . . . .	329
12-6 Determine amount of index filtering . . . . .	330
12-7 Filtering per predicate . . . . .	332
12-8 Queries to determine correlation. . . . .	334
12-9 Column group statistics in the table report . . . . .	335
12-10 RUNSTATS COLGROUP(CITY,STATE) . . . . .	336
12-11 Table report after COLGROUP RUNSTATS. . . . .	337
12-12 RUNSTATS to gather correlations statistics on (ZIPCODE,CITY, STATE) . . . . .	340
13-1 RUNSTATS justification for DISTARRANG table . . . . .	361
13-2 RUNSTATS statement for DISTARRA . . . . .	363
A-1 Generating predicate information for a set of packages. . . . .	399
A-2 Reporting all column expressions used as predicates . . . . .	400
A-3 Report of column expressions . . . . .	400
A-4 Columns used in predicates - Target COLUMN statistics . . . . .	401
A-5 Columns used with literal values . . . . .	403
B-1 Example of a failed Admin Scheduler startup in the MSTR started task . . . . .	406
B-2 Task list display . . . . .	407

Archived

# Notices

This information was developed for products and services offered in the U.S.A.

IBM may not offer the products, services, or features discussed in this document in other countries. Consult your local IBM representative for information on the products and services currently available in your area. Any reference to an IBM product, program, or service is not intended to state or imply that only that IBM product, program, or service may be used. Any functionally equivalent product, program, or service that does not infringe any IBM intellectual property right may be used instead. However, it is the user's responsibility to evaluate and verify the operation of any non-IBM product, program, or service.

IBM may have patents or pending patent applications covering subject matter described in this document. The furnishing of this document does not give you any license to these patents. You can send license inquiries, in writing, to:

*IBM Director of Licensing, IBM Corporation, North Castle Drive, Armonk, NY 10504-1785 U.S.A.*

**The following paragraph does not apply to the United Kingdom or any other country where such provisions are inconsistent with local law:** INTERNATIONAL BUSINESS MACHINES CORPORATION PROVIDES THIS PUBLICATION "AS IS" WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESS OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF NON-INFRINGEMENT, MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE. Some states do not allow disclaimer of express or implied warranties in certain transactions, therefore, this statement may not apply to you.

This information could include technical inaccuracies or typographical errors. Changes are periodically made to the information herein; these changes will be incorporated in new editions of the publication. IBM may make improvements and/or changes in the product(s) and/or the program(s) described in this publication at any time without notice.

Any references in this information to non-IBM Web sites are provided for convenience only and do not in any manner serve as an endorsement of those Web sites. The materials at those Web sites are not part of the materials for this IBM product and use of those Web sites is at your own risk.

IBM may use or distribute any of the information you supply in any way it believes appropriate without incurring any obligation to you.

Information concerning non-IBM products was obtained from the suppliers of those products, their published announcements or other publicly available sources. IBM has not tested those products and cannot confirm the accuracy of performance, compatibility or any other claims related to non-IBM products. Questions on the capabilities of non-IBM products should be addressed to the suppliers of those products.


This information contains examples of data and reports used in daily business operations. To illustrate them as completely as possible, the examples include the names of individuals, companies, brands, and products. All of these names are fictitious and any similarity to the names and addresses used by an actual business enterprise is entirely coincidental.

## COPYRIGHT LICENSE:

This information contains sample application programs in source language, which illustrate programming techniques on various operating platforms. You may copy, modify, and distribute these sample programs in any form without payment to IBM, for the purposes of developing, using, marketing or distributing application programs conforming to the application programming interface for the operating platform for which the sample programs are written. These examples have not been thoroughly tested under all conditions. IBM, therefore, cannot guarantee or imply reliability, serviceability, or function of these programs.

# Trademarks

The following terms are trademarks of the International Business Machines Corporation in the United States, other countries, or both:

Redbooks (logo) ®  
z/OS®  
zSeries®  
DB2 Connect™  
DB2®

DRDA®  
IBM®  
IMS™  
MVS™  
OS/390®

QMFTM  
Redbooks®  
RACF®  
System z™  
WebSphere®

The following terms are trademarks of other companies:

Snapshot, and the Network Appliance logo are trademarks or registered trademarks of Network Appliance, Inc. in the U.S. and other countries.

Adobe, and Portable Document Format (PDF) are either registered trademarks or trademarks of Adobe Systems Incorporated in the United States, other countries, or both.

Java, JDBC, JDK, JRE, and all Java-based trademarks are trademarks of Sun Microsystems, Inc. in the United States, other countries, or both.

Internet Explorer, Microsoft, Windows, and the Windows logo are trademarks of Microsoft Corporation in the United States, other countries, or both.

Pentium, Intel logo, Intel Inside logo, and Intel Centrino logo are trademarks or registered trademarks of Intel Corporation or its subsidiaries in the United States, other countries, or both.

UNIX is a registered trademark of The Open Group in the United States and other countries.

Other company, product, or service names may be trademarks or service marks of others.

# Preface

The cost-based optimizer of DB2® for z/OS® has continually evolved since its initial inception. Support has included new optimization algorithms, join methods, complex relational data structures, such as star schemas. DB2 for z/OS V8 has provided access path visualization and a Statistics Advisor function through a new Visual Explain.

DB2 9 for z/OS starts addressing the challenge of reducing cost of ownership by extending query optimization through expert-based query and workload analysis, server enhancements, and an Index Advisory function. These functions are delivered with new functions in DB2 9 for z/OS and two tools: IBM® DB2 Optimization Expert for z/OS and IBM Optimization Service Center for DB2 for z/OS.

This book helps you understand the installation, customization, and usage aspects of the tools. You are guided through scenarios of gradually increasing complexity where the functions of the tools are exploited for query optimization.

## The team that wrote this book

This book was produced by a team of specialists from around the world working at the International Technical Support Organization, San Jose Center.

**Paolo Bruni** is a DB2 Information Management Project Leader at the International Technical Support Organization, San Jose Center, California. In this capacity, he has authored several IBM Redbooks® publications on DB2 for z/OS and Data Management tools, and has conducted workshops and seminars worldwide. During Paolo's many years with IBM, in development and in the field, his work has been mostly related to database systems. Before joining the ITSO in 1998, Paolo was a Certified Consultant IT Architect with IBM Italy.

**Terry Berman** is a Database Manager at DST Systems, Inc. in Kansas City, Missouri. He is in charge of Level 2 concentration on DB2/ISV support issues, such as performance, availability, security, recovery, and software feature deployment. Over a period of 13 years in his group, Database Engineering, he has built up an extensive performance repository, providing comprehensive measurement-based reporting ("database intelligence"). Operating as an internal software vendor, his group supports a wide range of internal tools and automations that benefit DBAs (access path statistics management, for example), programmers (catalog visibility), and sysprogs (an alert monitor). He has presented several times at IDUG, most recently on query optimization topics.

**John Iczkovits** is a Consulting IT Specialist with IBM Advanced Technical Support Americas, Dallas Systems Center, Texas. He provides DB2 for z/OS and OS/390® technical support and consulting services. His areas of expertise include DB2 data sharing, performance, availability, and DB2 synergy with disk and tape. His work in IBM includes experience supporting DB2 as a Systems Engineer, Database Administrator, IT Specialist, Consultant, and project lead. He has an operations background and more than 25 years of IT experience, ranging from database products, such as DB2 and IMS™ to MVS™ systems support, including storage management. He has also co-authored IBM Redbooks publications, Redpapers, white papers, and presented at SHARE, the DB2 Tech Conference, the Information On Demand Conference, the zEXPO Conference, and at local DB2 users groups.

**Basuki Soetarman** is a Senior Software Engineer in DB2 for z/OS product development in Silicon Valley Laboratory, San Jose, California. He has been with IBM for 17 years with experience in various projects, such as Prolog compiler and interpreter development; data access class library development for DB2; inventor, developer, team lead, and architect for Content Management and Federated Content Management; member of an international expert group on XQuery API for Java™ (XQJ). His recent activities are in the development of DB2 for z/OS Query Optimization tools for both the client and server components.

**Bart Steegmans** is a Senior DB2 Product Support Specialist from IBM Belgium, currently working remotely for the Silicon Valley Laboratory in San Jose, providing technical support for DB2 for z/OS performance problems. Previously, Bart was on assignment as a Data Management for z/OS project leader at the ITSO, San Jose Center, from 2001 to 2004. He has over 19 years of experience in DB2. Before joining IBM in 1997, Bart worked as a DB2 system administrator at a banking and insurance group. His areas of expertise include DB2 performance, database administration, and backup and recovery.

**Mike Turner** is an independent consultant based in the United Kingdom. He has over 35 years of experience in the mainframe database management field. His areas of expertise include DB2 performance, recovery, and data sharing. He has taught extensively on these topics and co-authored *DB2 for z/OS and OS/390 Version 7 Performance Topics*, SG24-6129.

A photo of the team is shown in Figure 1.

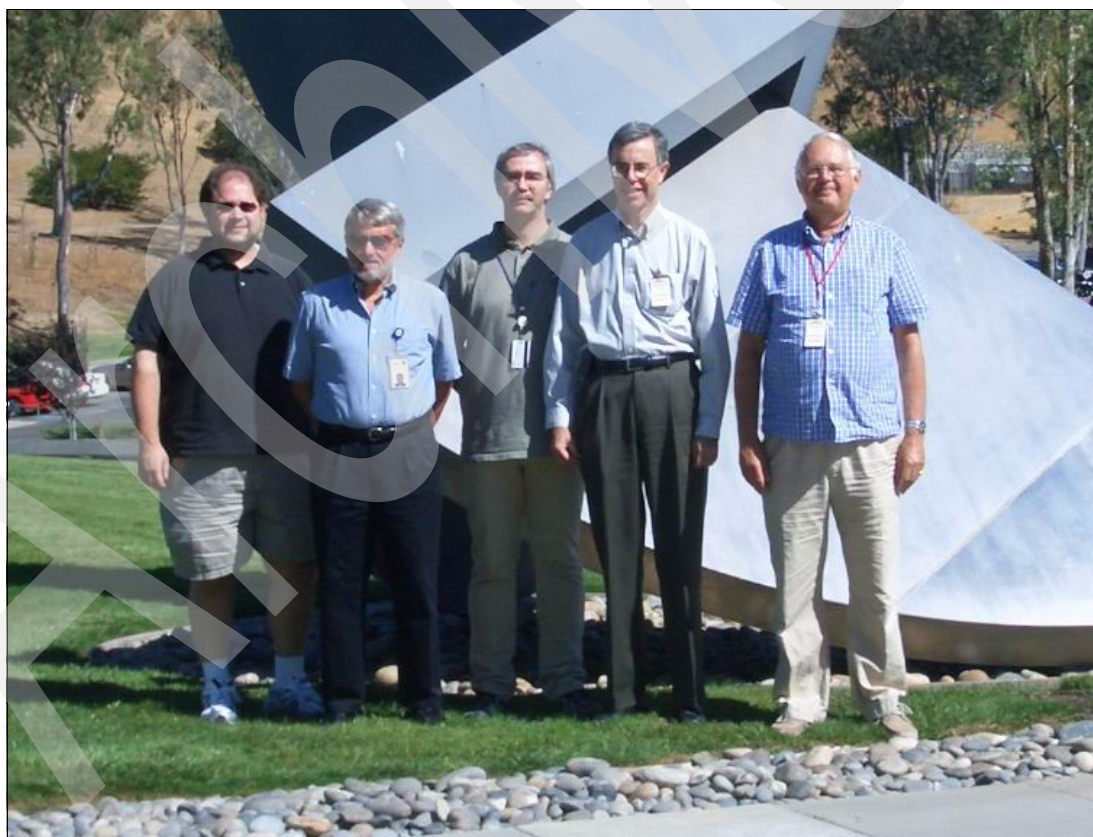


Figure 1 From left to right: John, Paolo, Bart, Terry, and Mike (photo courtesy of Basuki Soetarman)

Special thanks to **Zhuo Zhang** for the technical and liaison support provided on-demand throughout the project, bridging skill gaps, language barriers, and time zones.

Thanks to the following people for their contributions to this project:

Rich Conway  
Bob Haimowitz  
Roger Kolp  
Deanna Polm  
Sangam Racherla  
International Technical Support Organization

Brian Baggett  
Patrick Bossman  
Frank Bower  
Jay Bruce  
Chih-jieh Chang  
Gene Fuh  
Dengfeng Gao  
Nick Gariaeff  
James Guo  
Arno Huang  
Illong Huang  
Bruce McAlister  
Paul McWilliams  
David Moy  
Adarsh Pannu  
Terry Purcell  
Dave Raiman  
Yan Stein  
Yoichi Tsuji  
Dan Weis  
Ping Yu  
IBM Silicon Valley Lab

Dong Sheng Chen  
Qing Li  
Jun Liu  
Wei Qin  
Kun Peng Ren  
Tai Wei Shi  
Qiang Song  
Fang Xing  
Kang Xu  
Xiao Yi Wang  
Xin Yu Wang  
Shuo Wu  
Zhuo Zhang  
IBM China Software Development Lab

Rick Butler  
Bank of Montreal (BMO) Financial Group, Toronto

## Become a published author

Join us for a two- to six-week residency program! Help write a book dealing with specific products or solutions, while getting hands-on experience with leading-edge technologies. You

will have the opportunity to team with IBM technical professionals, Business Partners, and Clients.

Your efforts will help increase product acceptance and customer satisfaction. As a bonus, you will develop a network of contacts in IBM development labs, and increase your productivity and marketability.

Find out more about the residency program, browse the residency index, and apply online at:

[ibm.com/redbooks/residencies.html](http://ibm.com/redbooks/residencies.html)

## Comments welcome

Your comments are important to us!

We want our books to be as helpful as possible. Send us your comments about this book or other IBM Redbooks publications in one of the following ways:

- Use the online **Contact us** form found at:

[ibm.com/redbooks](http://ibm.com/redbooks)

- Send your comments in an e-mail to:

[redbooks@us.ibm.com](mailto:redbooks@us.ibm.com)

- Mail your comments to:

IBM Corporation, International Technical Support Organization  
Dept. HYTD Mail Station P099  
2455 South Road  
Poughkeepsie, NY 12601-5400





# Part 1

## Introduction

Part 1 introduces the need for DB2 query optimization, the IBM planned solution, the tools which provide it, and a simple example to illustrate the functions of the Optimization Service Center. This part contains the following chapters:

- ▶ Chapter 1, “Query optimization” on page 3
- ▶ Chapter 2, “Sample query optimization” on page 31

Archived



# Query optimization

This chapter describes the business need for DB2 query optimization and outlines the solution provided by IBM.

We introduce the concept of autonomic computing and the intended strategy for DB2 query optimization. We look at new tools, IBM DB2 Optimization Expert for z/OS and IBM Optimization Service Center for DB2 for z/OS, and position them in terms of functions.

This chapter contains the following topics:

- ▶ “Autonomic computing” on page 4
- ▶ “The delivery of query optimization” on page 9

## 1.1 Autonomic computing

The purpose of *autonomic computing* is to drive functionality into products that addresses two critical IT requirements: cost and complexity of operation.

The architecture of the current IBM System z™ includes many autonomic computing features that go well beyond the traditional heterogeneous workload management on a single server. There are more functions or constructs that help with automated decisioning for resource balancing and allocation. IBM intends to eventually deliver hands-off performance management across all systems and storage in an enterprise.

Server consolidation and vendor-packaged solutions have contributed to the need of skilled work forces, but in-house expertise on resource balancing and optimization is often lacking.

Consequently, an IBM initiative, which started in 2001 for server self-optimization and includes an automated knowledge bank of best practices for resource configuration and management, can bring enormous benefits to IT organizations.

For the definition of the initiative and the activities in progress, check the following Web sites at:

<http://www.research.ibm.com/autonomic/overview/>  
<http://www.ibm.com/autonomic/>

In April 2003, IBM introduced the autonomic computing blueprint, which is based on open standards and designed around developing self-managing systems that use intelligent control loops to collect information from the system, then make decisions and appropriate system adjustments. IBM also started working with the IBM business partner community on other autonomic computing issues, providing independent software vendors (ISVs) with the latest technologies and supporting them in bringing new applications to market with the latest capabilities built around open standards.

Autonomic computing is a pervasive effort that includes all IBM products, software, and hardware. IBM also provides an on-site assessment methodology for identifying the critical points and suggesting how to incorporate autonomic tools and what effects these tools will have on the immediate IT environment.

### 1.1.1 DB2 and autonomic computing

Since the introduction of the cost-based optimizer in the initial release of DB2 in 1983, IBM has continually evolved optimization technology. In recent years, DB2 for z/OS has extended support for complex relational data structures, such as star schemas and access path visualization with Visual Explain. New optimization algorithms, join methods, and access methods have been added to keep pace with query complexities. In DB2 for z/OS Version 8, Statistics Advisor was added to Visual Explain to provide a rich client for tuning queries. IBM plans to continue to extend optimization through expert-based query and workload analysis, server enhancements, and a workload history warehouse.

Over the next ten years, the role of the database administrators (DBAs) is expected to change significantly, as they find themselves in a very dynamic environment where maintaining quality of service (QoS) becomes more and more mission critical across the enterprise and total cost of ownership (TCO) increasingly impacts the enterprise bottom line. The focus on database availability, while minimizing costs, will only increase in the coming years. DBAs need to fundamentally change how they work to keep pace.

DB2 for z/OS is an ideal offering to fill future enterprise QoS requirements. For over twenty years of product evolution, DB2 for z/OS has been a trailblazer for providing reliability, availability, and serviceability. As enterprises face increasing TCO challenges during the next decade, DB2 for z/OS will address these challenges as a top priority by reducing costs through optimized innovations.

### 1.1.2 Total cost of ownership

Total cost of ownership (TCO) is broken down into hardware, software, people, and other related costs. Ten years ago, less than half of overall system expenditures were for hardware costs, while people costs represented about a quarter of the total cost. Today, that position is shifting, with people costs representing over a third of total costs while hardware costs now represent about a fifth. Clearly, the largest TCO impact is addressing people costs. A rich set of specialized DBA tools using best practices and industry standards is required to address rising people costs and increasing complexities.

### 1.1.3 Quality of service

A future challenge to quality of service (QoS) will also emerge over the years. As large enterprises grow through mergers and acquisitions, enter into new markets, and maintain a true global presence, the complexity of managing and tuning the various workloads will increase dramatically and may outpace the ability of humans to manage it all. People will become the limiting factor. Any initiative to reduce TCO must also address the growing management and tuning complexities.

Challenges that current and future DBAs are facing include:

- ▶ Reacting to QoS situations, such as a high priority query that has exceeded its service goals.
- ▶ Assisting application developers in designing and deploying new applications.
- ▶ Monitoring existing workloads to optimize query performance as needed.

Query performance problem determination, physical database design, and query optimization require highly specialized skills. A rich set of autonomic, platform-specific tools is needed to lower TCO.

Query performance tuning over time will become a task that cannot be done by DBAs. Even with expert DBA skills, issues cannot be resolved in a reasonable amount of time. Identifying the poorly performing queries is challenging enough. Collecting the critical information needed for analysis or diagnosis after the fact may be too late. The relevant information may be already lost. Even if the data is still present, the APIs do not exist today to extract the information.

### 1.1.4 Optimization Service Center or Optimization Expert

**Acknowledgement:** The text in this section is loosely based on the article “The Evolution of DB2 for z/OS Optimization” by Jay Bruce in IBM Systems Magazine, May-June 2006 issue.

The combined functions of IBM Optimization Service Center for DB2 for z/OS and IBM DB2 Optimization Expert for z/OS, see Figure 1-1 on page 6, propose to lower TCO, reduce specialized skills, and simplify workload management by providing a set of reactive, proactive,

and in the future, autonomic tools to optimize query performance and physical database design:

- ▶ Query-based analysis to react to serviceability issues.
- ▶ Workload-based analysis to proactively optimize physical database design.
- ▶ Autonomic workload performance monitoring and optimization.

DBAs and system programmers can take advantage of the Optimization Service Center or Optimization Expert's query-based analysis to react to specific performance problems. Users are guided through query performance problem determination. Power users can drill into query analysis and even set optimizer hints through an easy-to-use graphical interface. In critical situations, users can send queries directly to IBM service along with all required documentation to enable a quick turnaround.

Optimization Service Center or Optimization Expert extend query analysis into workload optimization. Users define workloads from a variety of sources, including QMF™, Packages/Plans, statement caches, and many others. You can proactively analyze a workload to optimize the physical database design. Expert-based plug-ins can recommend physical design changes, including indexes, clustering of data, query changes, and RUNSTATS suggestions to provide better information to the Optimizer. You can implement these recommendations immediately, scheduled via stored procedures or saved in a data set for later execution.

The Optimization Service Center or Optimization Expert Query Warehouse comes into play at this point. As workloads are defined and tuned, Optimization Service Center or Optimization Expert can collect performance data through new DB2 for z/OS performance data capture enhancements, directly from the DB2 for z/OS statement cache, or through open interfaces. The performance data provides a historical perspective on execution statistics, access path changes, and database design changes, allowing the user to contrast performance before and after the change. You can repeat this process iteratively until no further optimization is recommended.

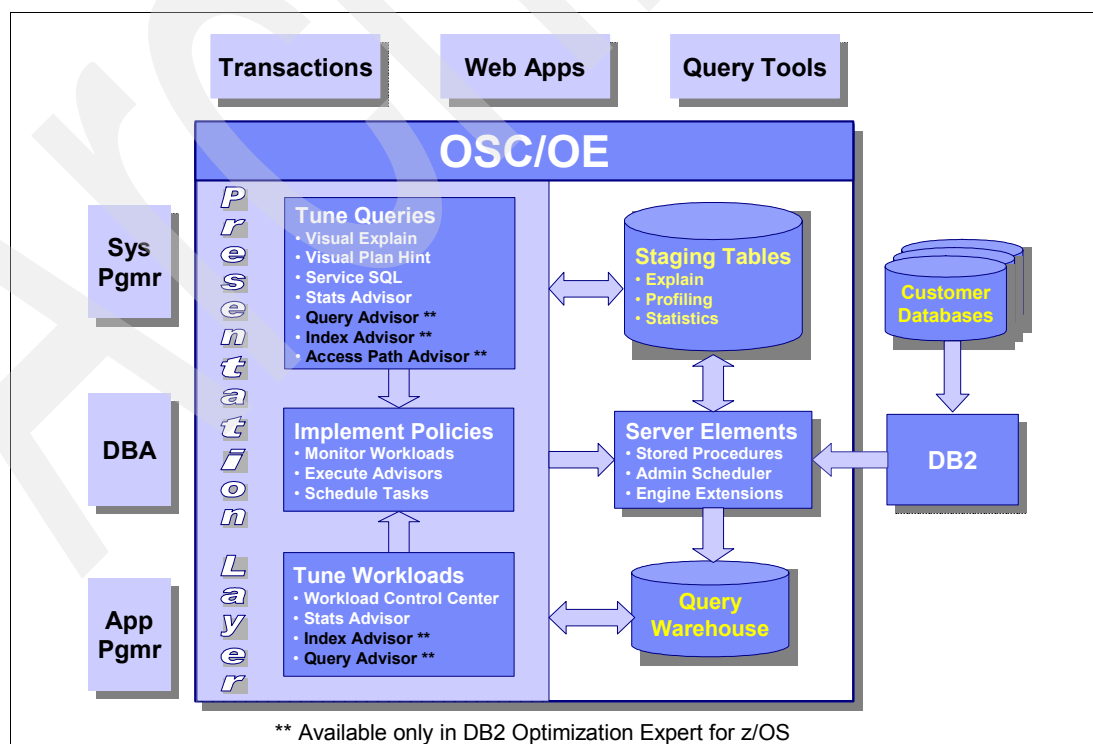


Figure 1-1 Optimization Service Center or Optimization Expert

As mentioned, users can schedule recommended tasks using stored procedures supplied with DB2 for z/OS. Users can also schedule times when runtime execution statistics are gathered to minimize the statistics overhead. These execution statistics can then be loaded into the Optimization Service Center or Optimization Expert Query Warehouse.

### 1.1.5 Customer adaptation

Deployment of Optimization Service Center or Optimization Expert follows the natural customer adaptation flow (Figure 1-2). The first Optimization Service Center or Optimization Expert release extends functionality previously provided by Visual Explain and Statistics Advisor in DB2 for z/OS V8 and introduces the Query Warehouse. Users can plug in an expert Index Advisor into Optimization Service Center or Optimization Expert. Later on, other Advisor plug-ins will be available. Future users can exploit policy-driven autonomies.

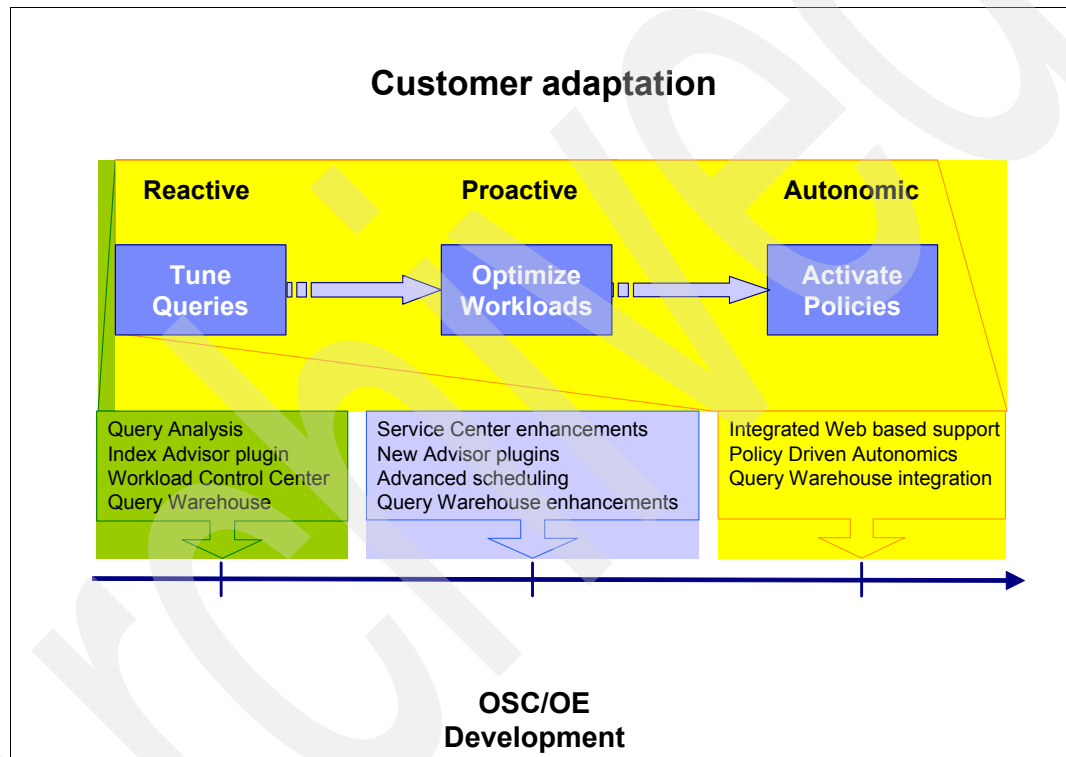


Figure 1-2 Customer adaptation

### 1.1.6 Extensible architecture

Optimization Service Center or Optimization Expert are built on the Eclipse Rich Client Platform and will continue to be delivered as a series of plug-ins to allow for installation level integration with IBM's next generation of database design, application development, and performance monitoring tools. Additionally, Optimization Service Center or Optimization Expert provide a full set of APIs that allows functionality to be imbedded in both IBM and third party tools. The Optimization Service Center or Optimization Expert Query Warehouse will evolve to support an open warehouse schema. In the future, a Web-based console will extend monitoring functions.

Optimization Service Center or Optimization Expert will allow DB2 for z/OS customers to transition from manual query tuning and problem identification using performance monitoring

tools, through automated analysis and recommendations, to policy-driven autonomic optimization based on best practices and business requirements (Figure 1-3).

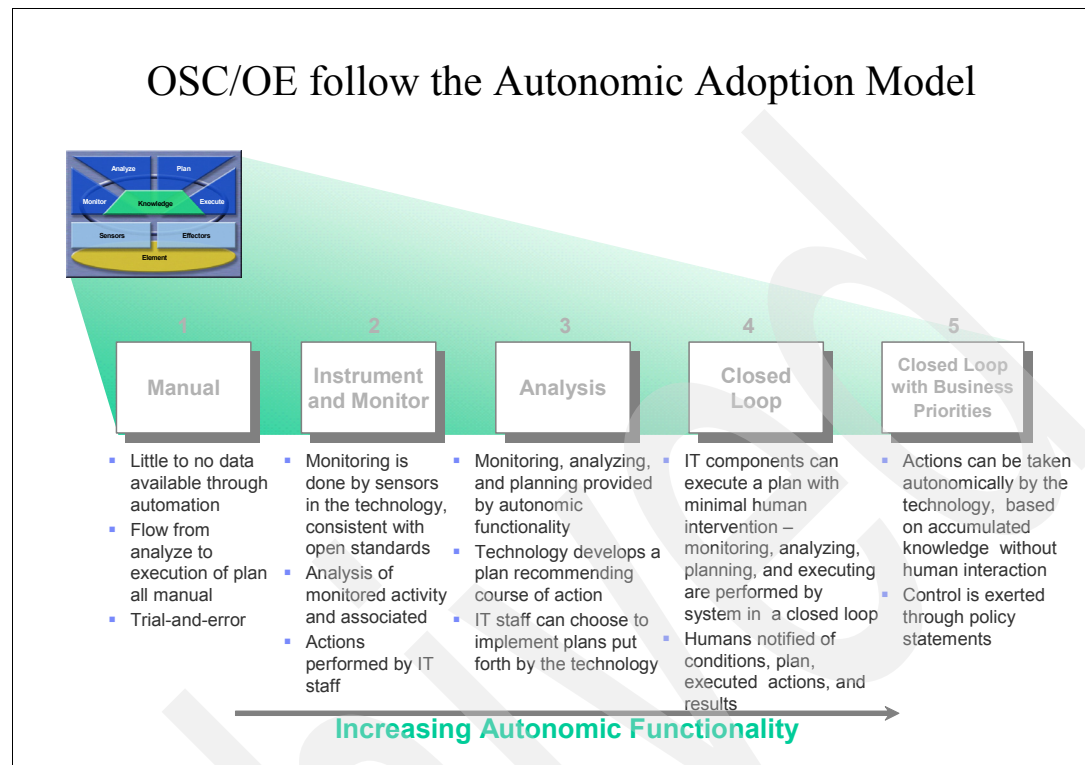


Figure 1-3 Autonomic computing

### 1.1.7 Customer value

Customer value is derived from all aspects of Optimization Service Center or Optimization Expert. Many customers have experienced performance problems related to query optimization. These problems range from a single query that has exceeded service level criteria, to application outages and migration obstacles. Optimization Service Center or Optimization Expert address each of these areas by providing a set of easy-to-use tools to proactively identify and resolve the underlying problem. As customers deploy the autonomic features, they realize value from increased availability, optimum use of hardware and software, and reduced personnel costs.

Deployment of Optimization Service Center or Optimization Expert in customer DB2 for z/OS environments is a natural evolution of current tooling and is in alignment with industry trends in both autonomic computing and open standards. Optimization Service Center or Optimization Expert easily integrates with a customer's existing and future tools and processes. By reducing the need for specialized DBA skills and integrating with IBM's next generation database design and application development tools, Optimization Service Center or Optimization Expert can minimize costs associated with developing and deploying new DB2 for z/OS applications or maintaining existing ones.

Optimization Service Center or Optimization Expert addresses such key value areas as it:

- ▶ Lowers TCO by optimal exploitation of DB2 for z/OS.
- ▶ Increases QoS by proactively solving problems before they occur.
- ▶ Decreases reliance on specialized DBA skills through autonomies and open tools.



### 1.1.8 Conclusion

Optimization Service Center or Optimization Expert addresses critical problem areas that are solved by the use of monitoring tools for problem identification, and then with manual tuning efforts that result in reactive problem resolution and labor intensive solutions. By building on an open standards platform, providing a rich API, and integrating with IBM and third party performance tools, they provide a cost-effective and much needed solution. Through exploitation of autonomic computing, Optimization Service Center or Optimization Expert maximizes the value of DB2 for z/OS while reducing labor costs and specialized DBA skills.

## 1.2 The delivery of query optimization

New functions in DB2 9 for z/OS, the Optimization Service Center product and the Optimization Expert tool, are the foundation blocks for DB2 for z/OS automated query optimization. This section introduces the query optimization functions of these new DB2 related products.

The DB2 Optimization Service Center for DB2 for z/OS V1.1 is available as a no-charge tool. It is part of the DB2 Accessories Suite (Figure 1-4 on page 10).

DB2 Accessories Suite for z/OS, V1.2 (5655-R14) consists of two features that are made up of a series of independent components designed to enhance your use of the DB2 for z/OS data server. Its initial release, with the general availability of DB2 9 for z/OS, provides support for:

- ▶ Enhanced SQL tuning (Optimization Service Center)
- ▶ Spatial data support
- ▶ Unicode and internationalization operations

DB2 Accessories Suite V1.2 and its Subscription and Support offering (5655-R30), announced on December 4, 2007, have two new features. You must specify one or both of these features when ordering:

- ▶ OmniFind™ Text Search Server is delivered in the enhanced DB2 Accessories Suite feature
- ▶ Data Studio capabilities are delivered in the new Data Studio feature

The FMID for the Optimization Service Center is H2AG110. It has both a workstation component and a mainframe component. The workstation component is available as a free download from the Web at:

<http://www.ibm.com/software/data/db2/zos/downloads>

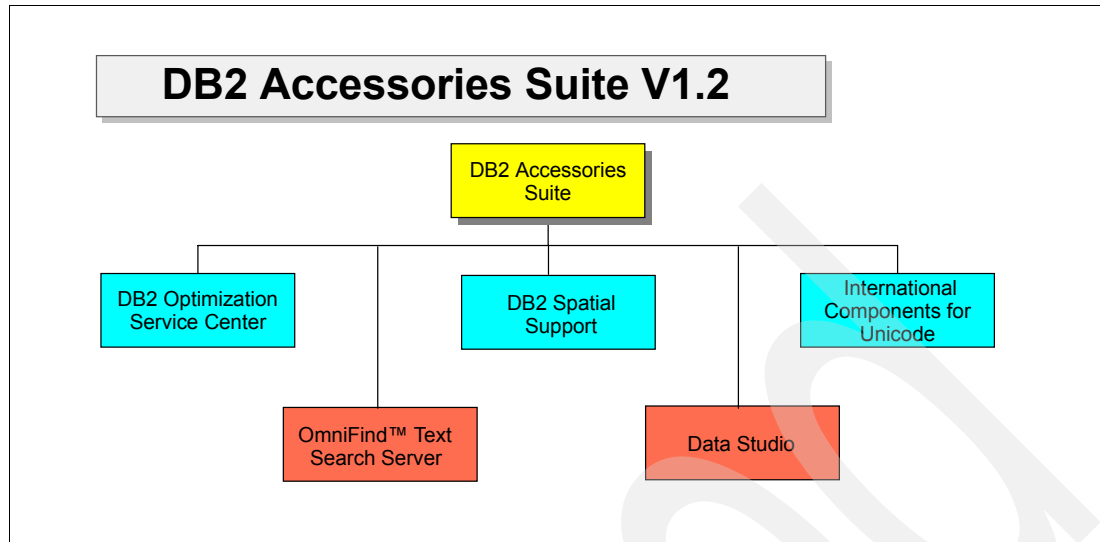


Figure 1-4 The Accessories Suite

The DB2 Optimization Expert for z/OS V1.1 (program number 5655-S19), which you can purchase, requires DB2 9 for z/OS functions. It includes all the functions of Optimization Service Center and adds additional functions for the expert user.

The next sections introduce the main functions of the two products. They provide an introduction to the setup, connectivity, and examples.

### 1.2.1 Optimization Service Center

DB2 Optimization Service Center includes functions previously available for DB2 V8 in Visual Explain and can:

- ▶ Extract SQL statements from either the DB2 global dynamic statement cache or from the package information held in the DB2 catalog for analysis.
- ▶ Produce an access plan graph for an SQL statement.
- ▶ Advise on the need to gather additional statistics using RUNSTATS.

In addition to these features, previously provided by Visual Explain, Optimization Service Center can:

- ▶ Re-format and annotate the optimizer transformed SQL statement.
- ▶ Monitor selected groups of SQL statements as they are executed and capture them as a workload for subsequent analysis.
- ▶ Advise on the need to gather additional statistics based on a workload rather than a single SQL statement.
- ▶ Provide improved query reports to support analysis of a query, such as table, index, indexed columns report, and index/predicate reports.

#### Setting up Optimization Service Center

The installation of both mainframe and workstation components of the Optimization Service Center is documented in Chapter 3, "Installing Optimization Service Center" on page 81.

Install DB2 Connect™ Personal Edition (or later) on your workstation before you install the Optimization Service Center workstation component. If it is not and you run the workstation install, you receive an error message as shown in Figure 1-5. The Optimization Service Center install is aborted. You need to install DB2 Connect and then try again to install Optimization Service Center.

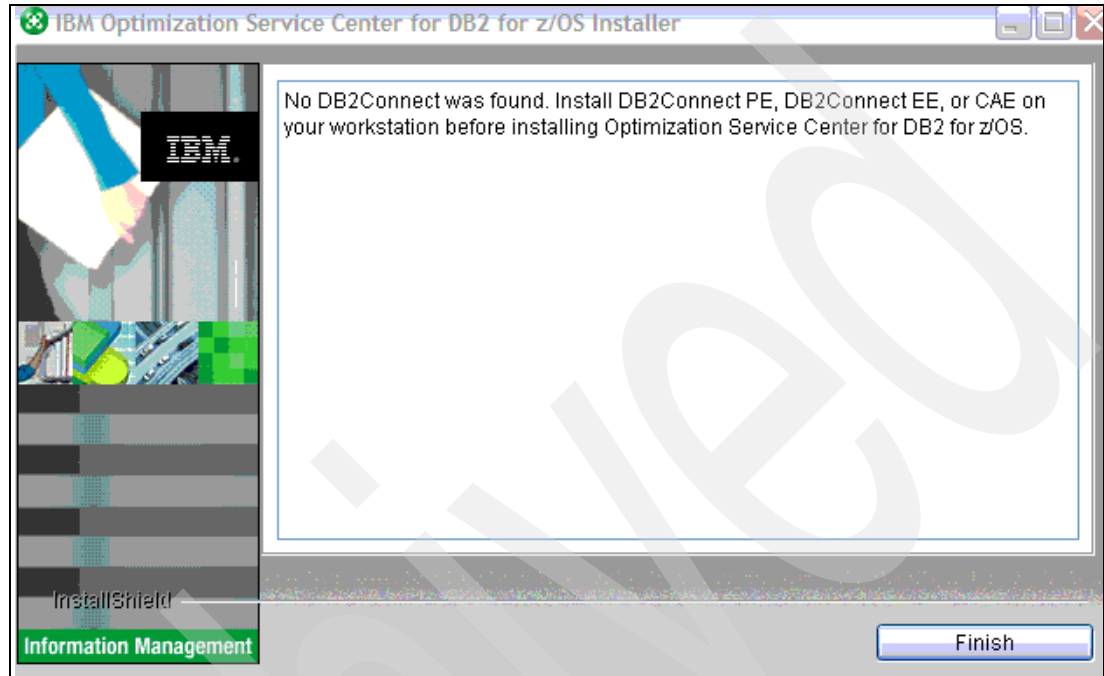


Figure 1-5 Optimization Service Center install error if DB2 Connect not installed

When you first start Optimization Service Center, you see the Welcome window as shown in Figure 1-6.

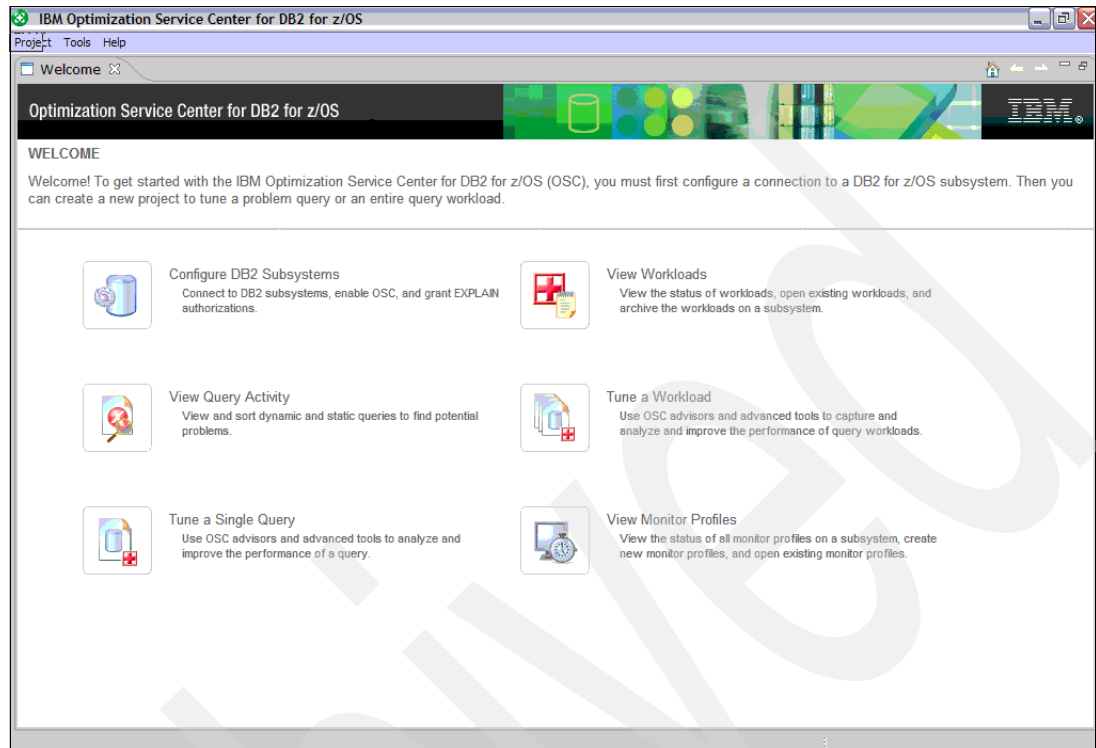


Figure 1-6 Optimization Service Center Welcome window

Configure the DB2 for z/OS subsystem that you wish to work with. Click **Configure DB2 Subsystems** on the Welcome window. You then see the DB2 Subsystem Configurations window, as shown in Figure 1-7.

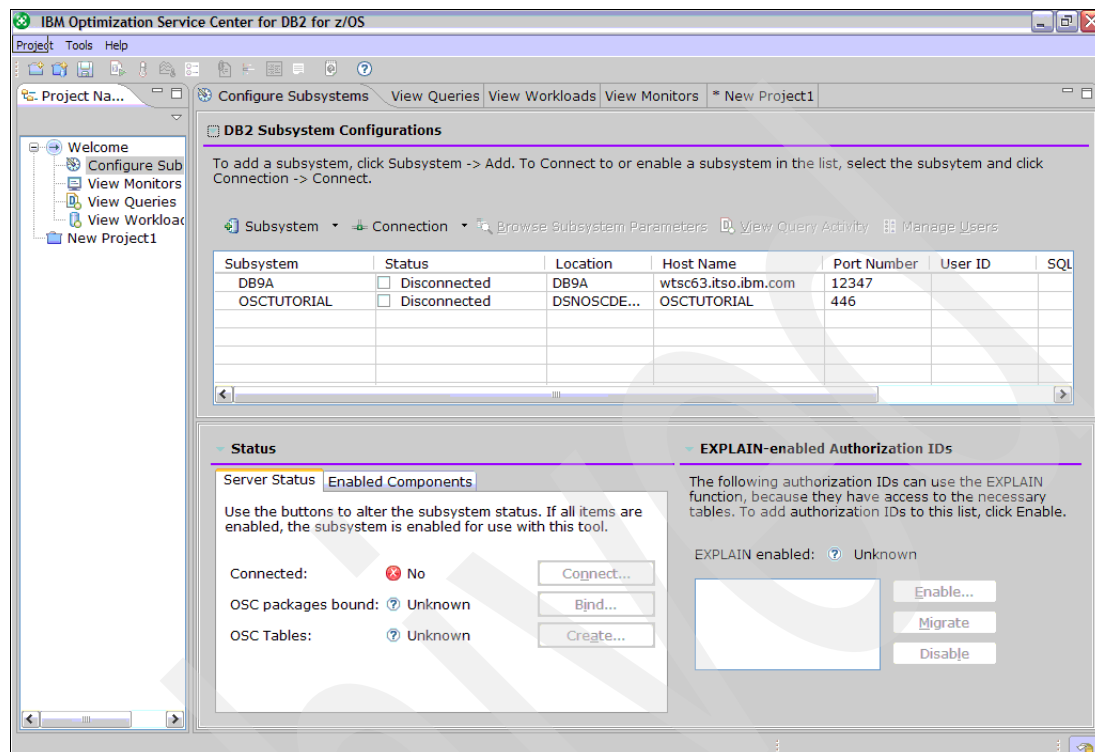


Figure 1-7 DB2 Subsystems Configuration window

The window lists the DB2 subsystems currently known to DB2 Connect. In the example shown, there is one subsystem, DB9A, already defined to DB2 Connect. The subsystem named OSCUTORIAL is a dummy subsystem used when running the Optimization Service Center tutorial.

If the DB2 subsystem that you want to use is not shown, then it is not defined to DB2 Connect. You can add the subsystem from this Optimization Service Center window without using a DB2 Connect tool, such as Configuration Assistant. Click on the word **Subsystem** on the left above the list of known subsystems, as shown in Figure 1-8.

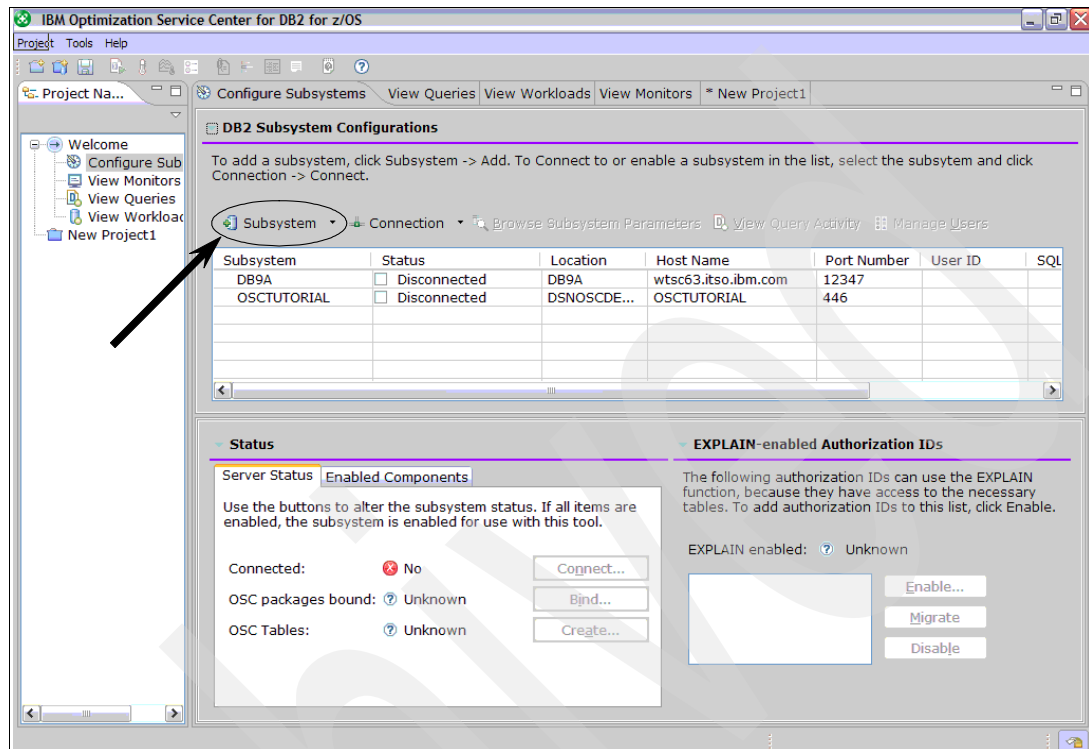


Figure 1-8 Adding a subsystem

On the drop-down menu, select the **Add** option. A Subsystem Properties panel opens as shown in Figure 1-9.

The screenshot shows a window titled "Subsystem Properties" with a close button in the top right corner. Inside the window, the "Subsystem Location" tab is selected. Below the tab name, there is instructional text: "Specify the following information about the subsystem that you want to configure and click Catalog. After the subsystem is cataloged, click Next." To the left of the main form area is a "Steps" panel with two items: "1. Location" (which is expanded with a small triangle icon) and "2. Connect". The main form area contains several input fields: "Subsystem alias:" followed by a text box; "Location:" followed by a text box; "Host name:" followed by a text box; "Port number:" followed by a text box containing the value "446"; and "Comment:" followed by a large text area. Below the text area is a "Catalog" button. At the bottom of the window, there is a row of four buttons: "< Back", "Next >", "Finish", and "Cancel".

Figure 1-9 Subsystem Properties panel

Type in the name by which you wish to refer to the subsystem in the first box, the location name in the second box, the host name, and the port number, as shown in Figure 1-10.

**Subsystem Properties**

**Subsystem Location**

Specify the following information about the subsystem that you want to configure and click Catalog.  
After the subsystem is cataloged, click Next.

**Steps**

- 1. Location
- 2. Connect

Subsystem alias:

Location:

Host name:

Port number:

Comment:

Figure 1-10 New subsystem details



Click **Finish** and the new subsystem is defined to Optimization Service Center and to DB2 Connect, as shown in Figure 1-11.

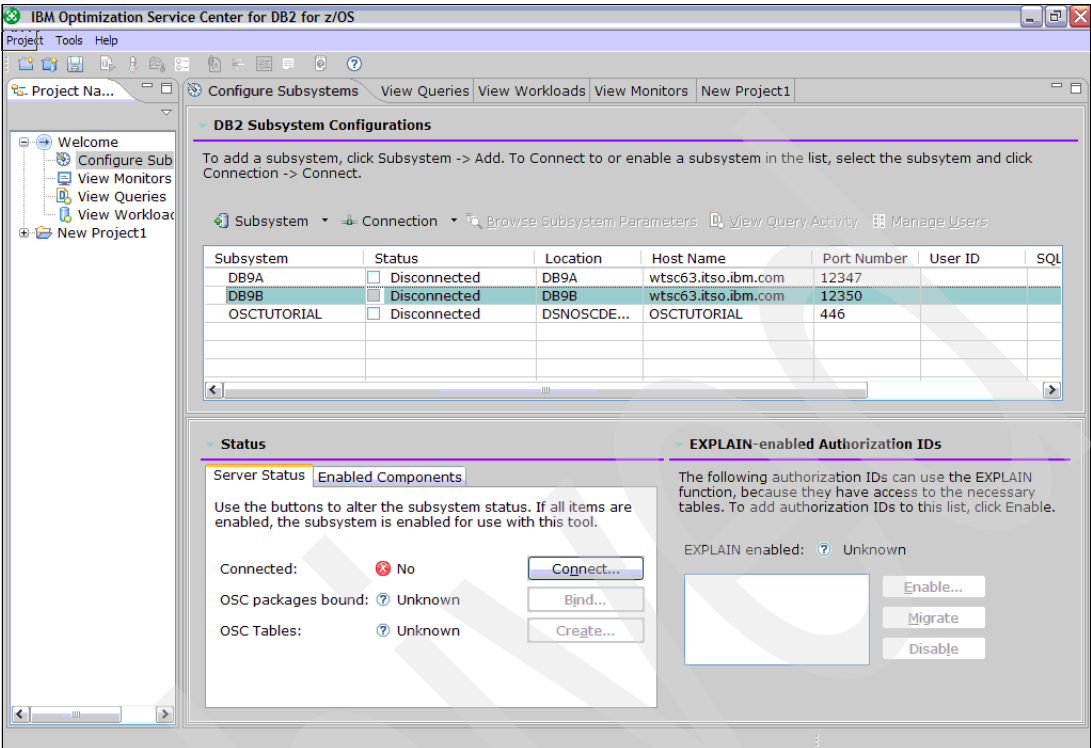


Figure 1-11 New subsystem added successfully

You can find the information needed to define a subsystem from the DDF startup messages, as shown in Figure 1-12. The Domain name is entered as the host name and the TCP Port number as the port number.

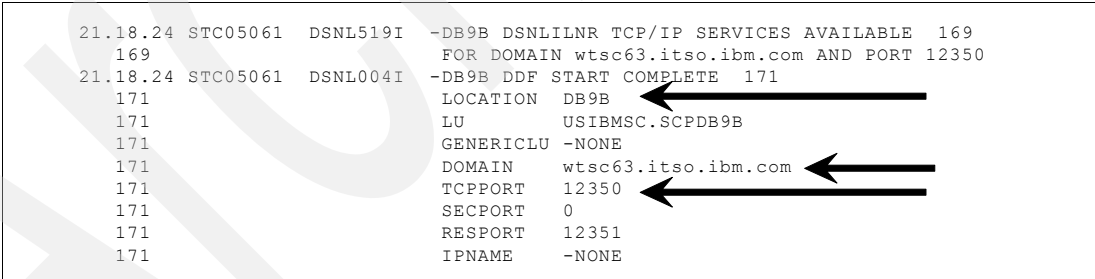


Figure 1-12 Obtaining the information needed to define a subsystem

## Connecting to a DB2 subsystem

Let us connect to subsystem DB9A and see what happens. Select the subsystem from the list and click **Connect** in the Server Status panel (see Figure 1-13).

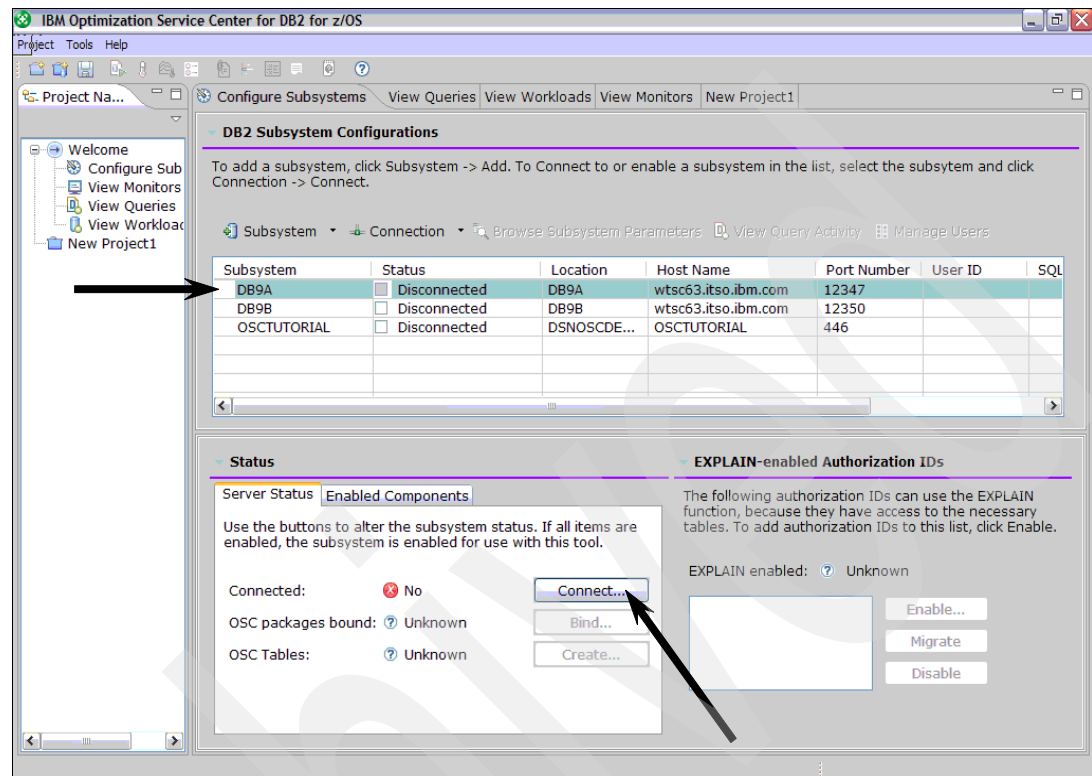


Figure 1-13 Connect to a subsystem

A panel opens asking for your TSO user ID and password (see Figure 1-14). You can optionally specify a secondary Authorization ID as the SQL ID. This SQL ID is used for the creator of the Explain tables, which will be used by the Optimization Service Center. If you check the box marked “Remember login ID”, then Optimization Service Center remembers your user ID next time you connect to this subsystem.

**Subsystem Properties**

**Connect to subsystem**

Specify the authorization information for the subsystem and click Connect. When the subsystem is connected, click Next or Finish.

**Steps**

- 1. Location
- 2. Connect

Complete the following fields and click Connect.

User ID:

Password:

SQL ID:

☒ Remember login ID

The following fields describe the OSC status of this subsystem:

Connected: ✖ No

OSC packages bound: ? Unknown

EXPLAIN enabled: ? Unknown

OSC Tables: ? Unknown

Figure 1-14 Connect to a subsystem - User ID panel

When you click **Connect**, Optimization Service Center checks to see which components are already installed on the subsystem and reports these as shown in Figure 1-15. In this example, we have successfully connected to the DB2 subsystem and the DB2 packages and tables required by Optimization Service Center have already been installed on the subsystem. When Explain is enabled, the message saying that our authorization ID is EXPLAIN-disabled means that the user-specific EXPLAIN tables required by Optimization Service Center do not exist yet.

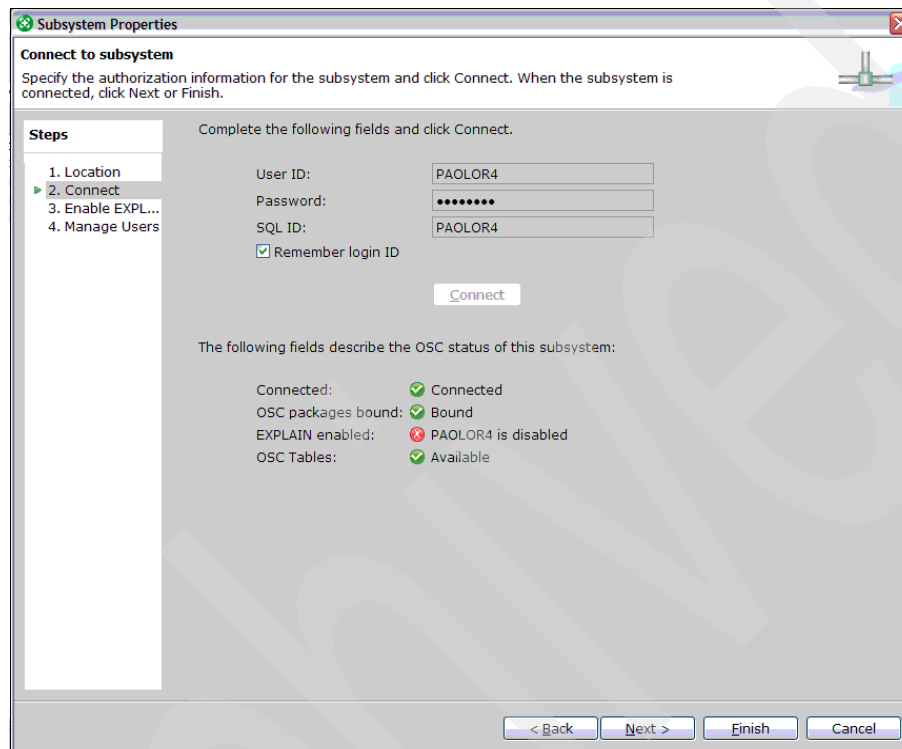


Figure 1-15 Checking the Optimization Service Center objects already installed

If you click **Next**, you see the Enable Explain panel, as shown in Figure 1-16. To use the Explain facility of the Optimization Service Center, you must either create a set of Optimization Service Center Explain tables for your User ID or you can use an existing set of tables.

**Subsystem Properties**

**Enable EXPLAIN**

Select an action to enable one or more SQL IDs to use the EXPLAIN function. Complete the related fields and click the appropriate button to complete the action. Then click Next or Finish.

**Steps**

1. Location
2. Connect
3. Enable EXPLAIN
4. Manage Users

Action: Create new EXPLAIN tables

SQL ID: PAOLOR4

Qualifier: PAOLOR4

Database name: DSNDB04 New...

4 KB table space: DSNSUMTS New...

8 KB table space: FXNTS New...

32 KB lob table space: LOBTS1 New...

32 KB lob table space: LOBTS2 New...

Create Tables

The following authorization IDs can use the EXPLAIN function on this subsystem, because the necessary EXPLAIN tables exist and they have access to them:

BART  
PAOLOR2  
PAOLOR1

The following fields describe the OSC status of this subsystem:

Connected: ✓ Connected

OSC packages bound: ✓ Bound

EXPLAIN enabled: ✗ PAOLOR4 is disabled

< Back Next > Finish Cancel

Figure 1-16 Enable Explain panel

You can see the options available by opening the drop-down list at the top of the panel, as shown in Figure 1-17.

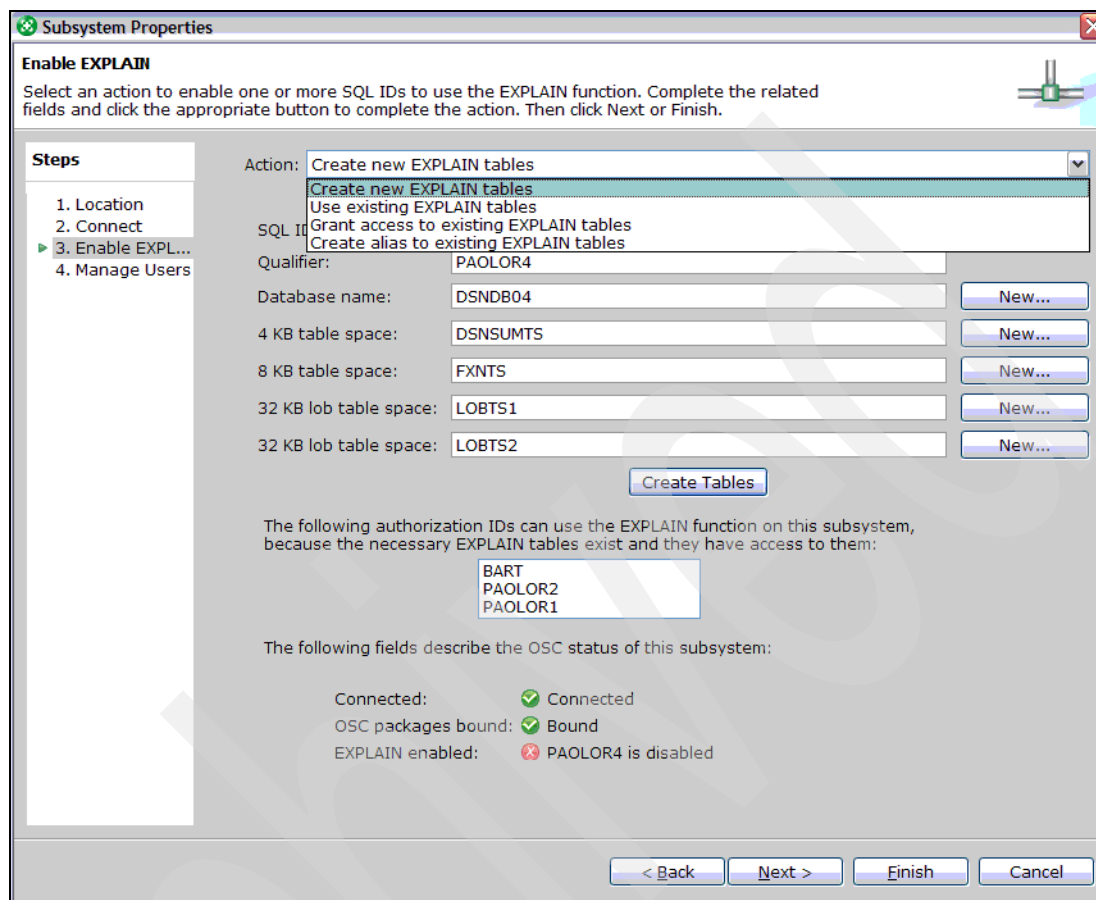


Figure 1-17 The options for setting up the Explain tables

If you wish to create a new set of Explain tables, you must specify the names of the database and table spaces to use. If they do not exist, you can create them by clicking **New** to the right of each object. You must have DB2 privileges needed to create the objects. The names of the user-specific tables used by Optimization Service Center are:

- ▶ DSN\_DETCOST\_TABLE
- ▶ DSN\_FILTER\_TABLE
- ▶ DSN\_FUNCTION\_TABLE
- ▶ DSN\_PGRANGE\_TABLE
- ▶ DSN\_PGROUPTABLE
- ▶ DSN\_PREDICAT\_TABLE
- ▶ DSN\_PTASK\_TABLE
- ▶ DSN\_QUERY\_TABLE
- ▶ DSN\_SORTKEY\_TABLE
- ▶ DSN\_SORT\_TABLE
- ▶ DSN\_STATEMENT\_CACHE\_TABLE
- ▶ DSN\_STATEMENT\_TABLE
- ▶ DSN\_STRUCT\_TABLE
- ▶ DSN\_VIEWREF\_TABLE
- ▶ PLAN\_TABLE

Figure 1-18 shows the Enable Explain panel with the names filled in for User ID PAOLOR4. The ID PAOLOR4 is defined to DB9A as SYSADM and can, therefore, create all the objects that Optimization Service Center requires. The database and the 4 KB table space already exist, but you need to create the 8 KB and the two LOB table spaces.

**Subsystem Properties**

**Enable EXPLAIN**

Select an action to enable one or more SQL IDs to use the EXPLAIN function. Complete the related fields and click the appropriate button to complete the action. Then click Next or Finish.

**Steps**

1. Location
2. Connect
3. Enable EXPLAIN
4. Manage Users

Action: Create new EXPLAIN tables

SQL ID: PAOLOR4

Qualifier: PAOLOR4

Database name: sg7421db New...

4 KB table space: plants New...

8 KB table space: plan8kts New...

32 KB lob table space: planLTS1 New...

32 KB lob table space: planLTS2 New...

Create Tables

The following authorization IDs can use the EXPLAIN function on this subsystem, because the necessary EXPLAIN tables exist and they have access to them:

BART  
PAOLOR2  
PAOLOR1

The following fields describe the OSC status of this subsystem:

Connected: ✓ Connected

OSC packages bound: ✓ Bound

EXPLAIN enabled: ✗ PAOLOR4 is disabled

< Back Next > Finish Cancel

Figure 1-18 Defining the names of the table spaces

To create the 8 KB table space, click **New** to the right of the table space name. A subpanel opens, as shown in Figure 1-19.

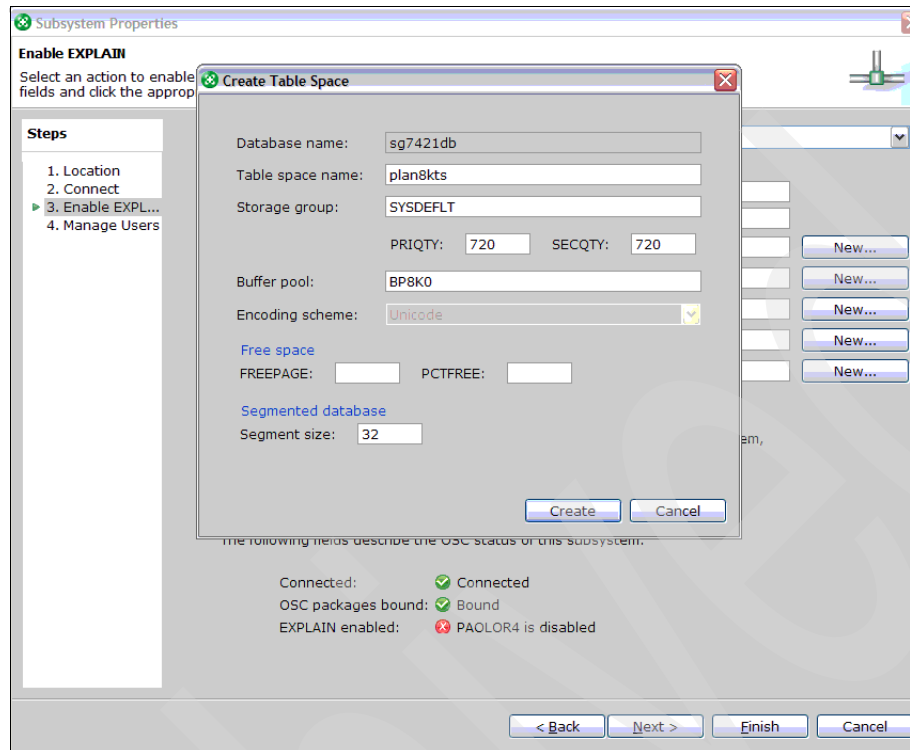


Figure 1-19 Create table space subpanel



You can change the parameters or accept the defaults. When you click **Create**, a confirmation is received indicating that the table space was successfully created, as shown in Figure 1-20. The two LOB table spaces are created in a similar manner.

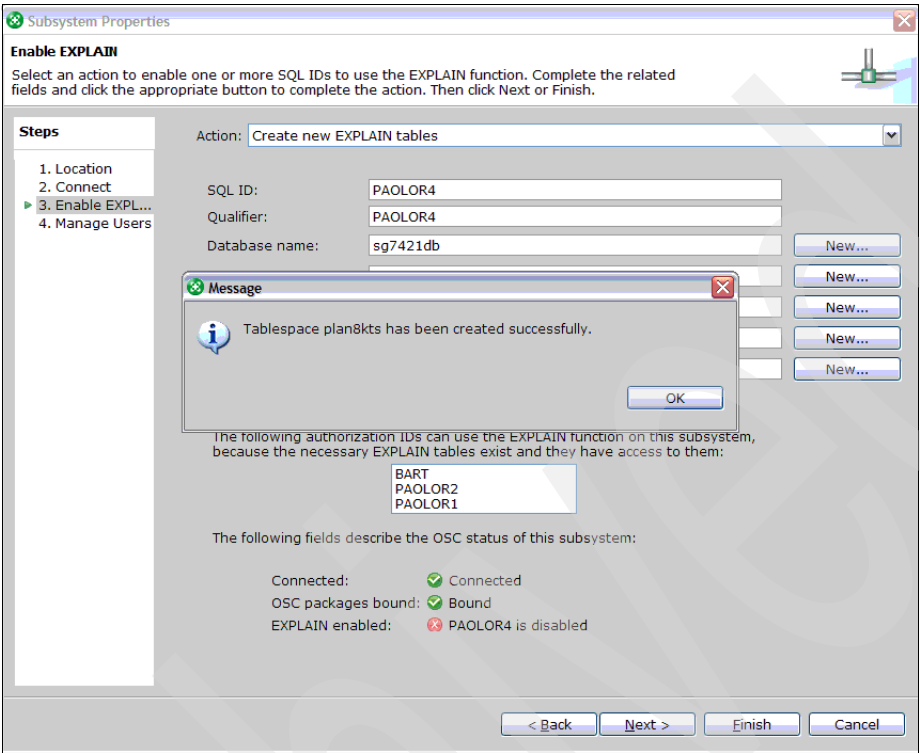


Figure 1-20 Confirmation of table space creation

When all table spaces exist, the Optimization Service Center user-specific tables are created by clicking **Create Tables** (see Figure 1-18 on page 23). In our example, two of the tables, the PLAN\_TABLE and the DSN\_STATEMNT\_TABLE, already existed and a warning message was displayed, as shown in Figure 1-21. Click **OK** to clear the warning message.

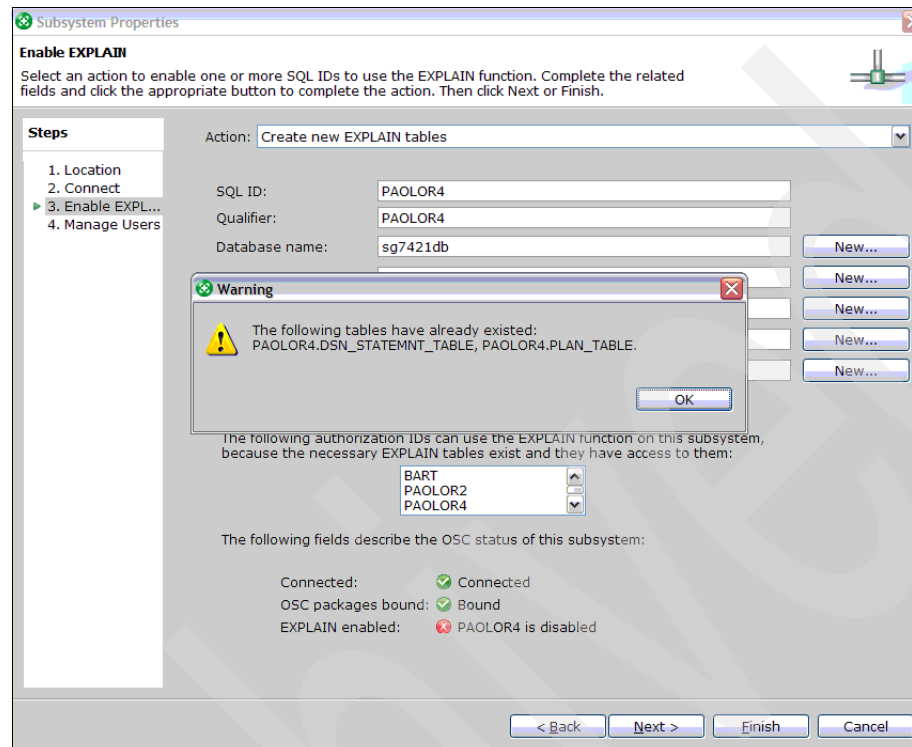


Figure 1-21 Warning that some tables already exist

Back on the Enable Explain panel, click **Finish** to complete the connection to and setup of the subsystem DB9A. If you click **Next** instead of **Finish**, you are taken to a Manage Users window, which is for system administrator use only.

You are now ready to use the Optimization Service Center.

## 1.2.2 Optimization Expert

DB2 Optimization Expert for z/OS includes all the functions of Optimization Service Center, but Optimization Expert also offers a comprehensive set of index (Query and Access Path Advisors) to improve system performance and lower total cost of ownership.

You can use the Optimization Expert to identify and analyze groups of statements and receive expert advice about how to improve the performance of entire SQL workloads. Optimization Expert makes it easy for you to get and implement expert tuning recommendations. After you have examined the workload and identified a problem query, you can run any or all of the expert Advisors.

The Query Advisor recommends ways to rewrite an SQL query to improve performance (Figure 1-22), such as:

- ▶ Considering many different conditions.
- ▶ Recommending best practice fixes to common query writing mistakes.

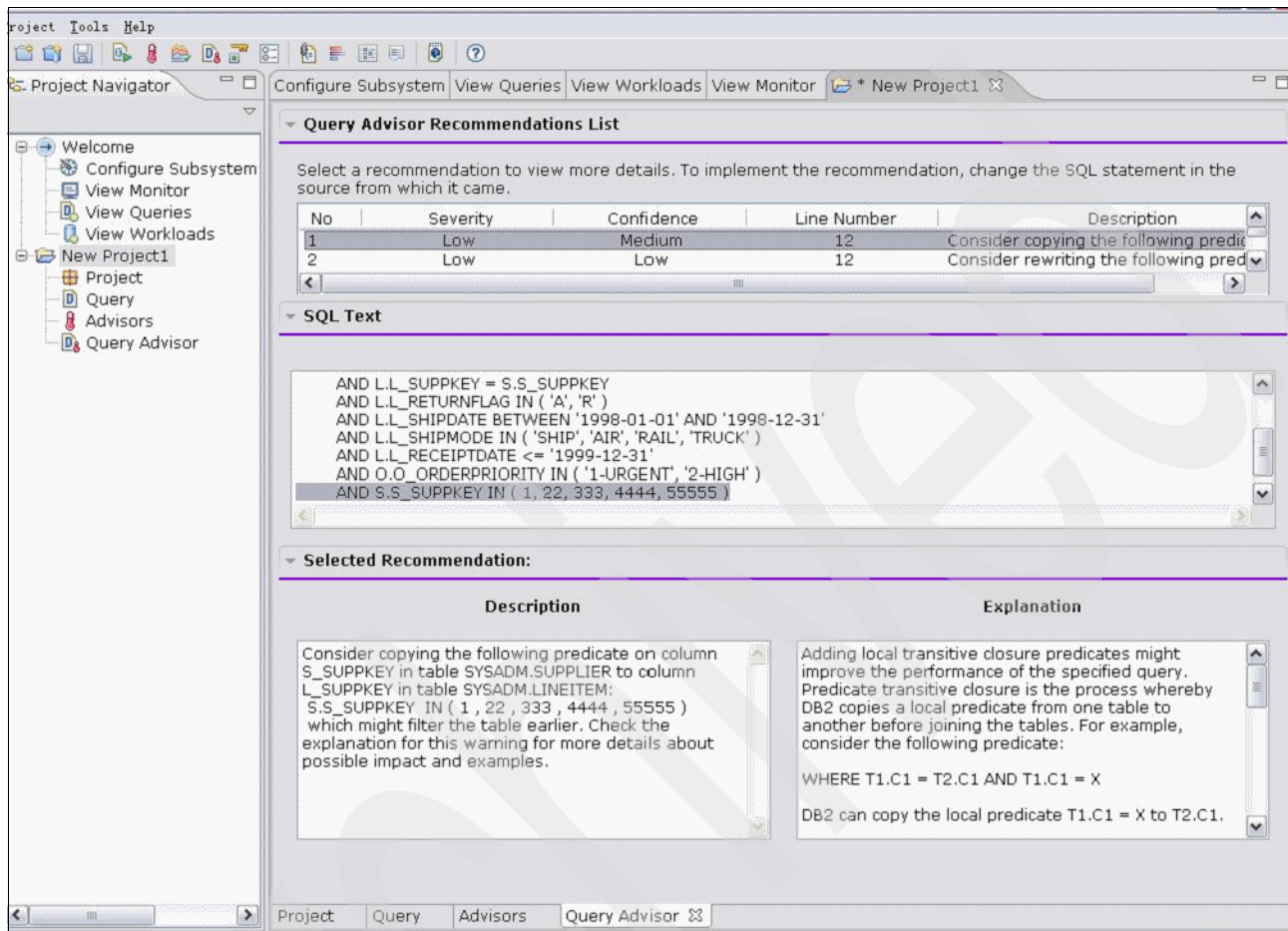


Figure 1-22 Optimization Expert Query Advisor

Optimization Expert considers a number of different conditions and recommends best practice fixes to common query writing mistakes.

The Index Advisor recommends indexes that you might create or modify to enhance the performance of either a single SQL statement or a workload (Figure 1-23 on page 28).

In addition, CREATE INDEX statements are generated so that you can:

- ▶ Implement the recommendations.
- ▶ Execute directly from your workstation on your DB2 for z/OS subsystem.

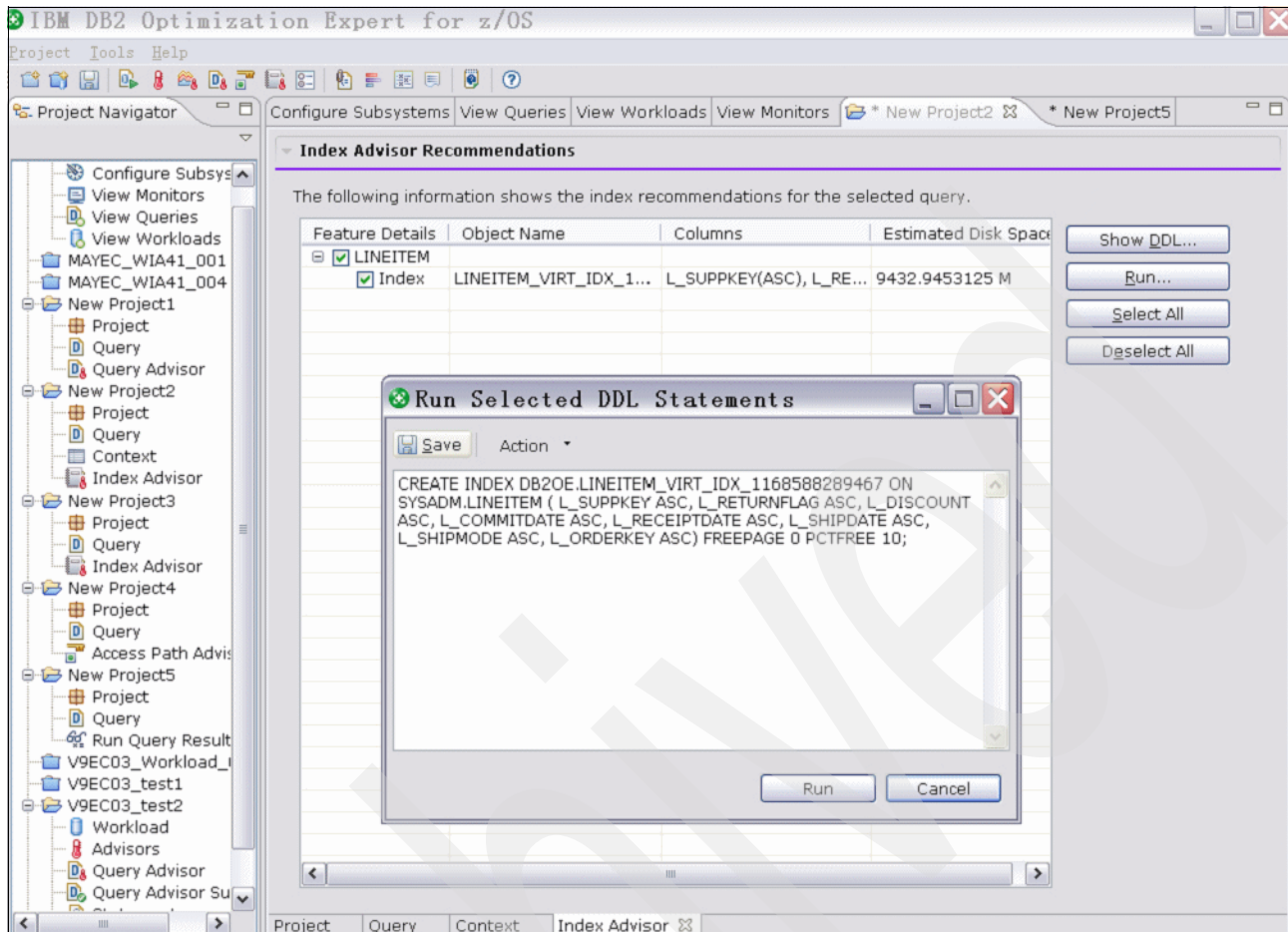


Figure 1-23 OE Index Advisor

Optimization Expert includes all the functions of Optimization Service Center and adds an Access Path Advisor that warns about poor access paths and suggests actions that might result in a better access path.

Optimization Expert is built on Eclipse, an award-winning, open-source platform for the construction of powerful software development tools and rich desktop applications. Leveraging the Eclipse plug-in framework to integrate technology on the desktop saves technology providers time and money. They can focus their efforts on delivering differentiation and value for their offerings. For more information about Eclipse, see their site at:

<http://www.eclipse.org>

## Setting up Optimization Expert

The installation of both mainframe and workstation components of Optimization Expert is documented in Chapter 4, "Installing Optimization Expert" on page 95. If you already have Optimization Service Center installed on your workstation, you can install Optimization Expert as well. The default location for the Optimization Expert install is different from that used by Optimization Service Center and you can, therefore, run both products independently from the same workstation.

As with Optimization Service Center, install DB2 Connect (Personal Edition or later) on your workstation before you install the Optimization Expert workstation component.

## Connecting to a DB2 subsystem

The process of connecting to a DB2 subsystem is identical to that described for Optimization Service Center (see “Connecting to a DB2 subsystem” on page 18).

### 1.2.3 Product functions and DB2 support

To support the functions for query optimization, several enhancements have been implemented to DB2 9 for z/OS:

- ▶ Profile-based Query Performance normal/exception monitoring
- ▶ Virtual Index “What-if” Optimizer support
- ▶ Server-based Administrative Scheduler
- ▶ Expanded Explain facility

The tools are built on Eclipse architecture with a new *outside in* design process.

As mentioned, the Optimization Service Center is a functional replacement of Visual Explain with the addition of the Workload Control Center. The Optimization Expert encompasses all Optimization Service Center functions, including Index, Query, and Access Path Advisors.

Table 1-1 summarizes the different functions as supported by the three products (Visual Explain, Optimization Service Center, and Optimization Expert) and points to the relevant sections in this book.

Table 1-1 Family product functional reference

Function	Visual Explain	Optimization Service Center	Optimization Expert
Queries from Cache, Catalog	Y	Y 2.5, “Investigating a query from the dynamic statement cache” on page 66 “Dynamic statement cache” on page 150	Y 2.5, “Investigating a query from the dynamic statement cache” on page 66 “Dynamic statement cache” on page 150
Queries Formatter, Annotation		Y 8.3, “Query Annotation tool” on page 180	Y 8.3, “Query Annotation tool” on page 180
Access Plan Graph	Y	Y 8.2, “Access Plan Graph” on page 171	Y 8.2, “Access Plan Graph” on page 171
Visual Plan Hint		Y 8.6, “Visual Plan Hint” on page 204	Y 8.6, “Visual Plan Hint” on page 204
Query Statistic Advisor	Y	Y 9.2, “Statistics Advisor analysis” on page 227	Y 9.2, “Statistics Advisor analysis” on page 227
Workload Statistics Advisor		Y 9.3, “Invoking Statistics Advisor” on page 231	Y 9.3, “Invoking Statistics Advisor” on page 231

Function	Visual Explain	Optimization Service Center	Optimization Expert
Profile based monitoring		Y Chapter 6, "Profile monitoring" on page 131	Y Chapter 6, "Profile monitoring" on page 131
Query Reports	Y	Y 8.4, "Query reports" on page 191	Y 8.4, "Query reports" on page 191
Service SQL	Y	Y 8.5, "Gather service information" on page 198	Y 8.5, "Gather service information" on page 198
Query Index Advisor			Y 10.2, "Query Index Advisor" on page 244
Workload Index Advisor			Y 10.3, "Workload Index Advisor" on page 257
Query Advisor			Y 11.3, "Query Advisor" on page 300
Workload Query Advisor			Y 11.4, "Workload Query Advisor" on page 307
Access Path Advisor			Y 11.2, "Access Path Advisor" on page 292

## Sample query optimization

This chapter describes a basic scenario on using the Optimization Service Center to investigate an SQL statement.

This chapter contains the following topics:

- ▶ “Starting an Optimization Service Center session” on page 32
- ▶ “Creating a project” on page 36
- ▶ “Investigating a query by typing the text” on page 38
- ▶ “Investigating a query from a package” on page 56
- ▶ “Investigating a query from the dynamic statement cache” on page 66
- ▶ “Using the Statistics Advisor” on page 71

## 2.1 Starting an Optimization Service Center session

When you start the Optimization Service Center for the first time after you have configured it, you see the DB2 Subsystem Configurations window as shown in Figure 2-1. If you see a different window, switch to the DB2 Subsystem Configurations window in one of two ways:

- ▶ Double-click the **Configure Subsystems** option from the Project Navigator panel on the left of the window.
- ▶ Single-click the **Configure Subsystems** tab from the tabs near the top of the window.

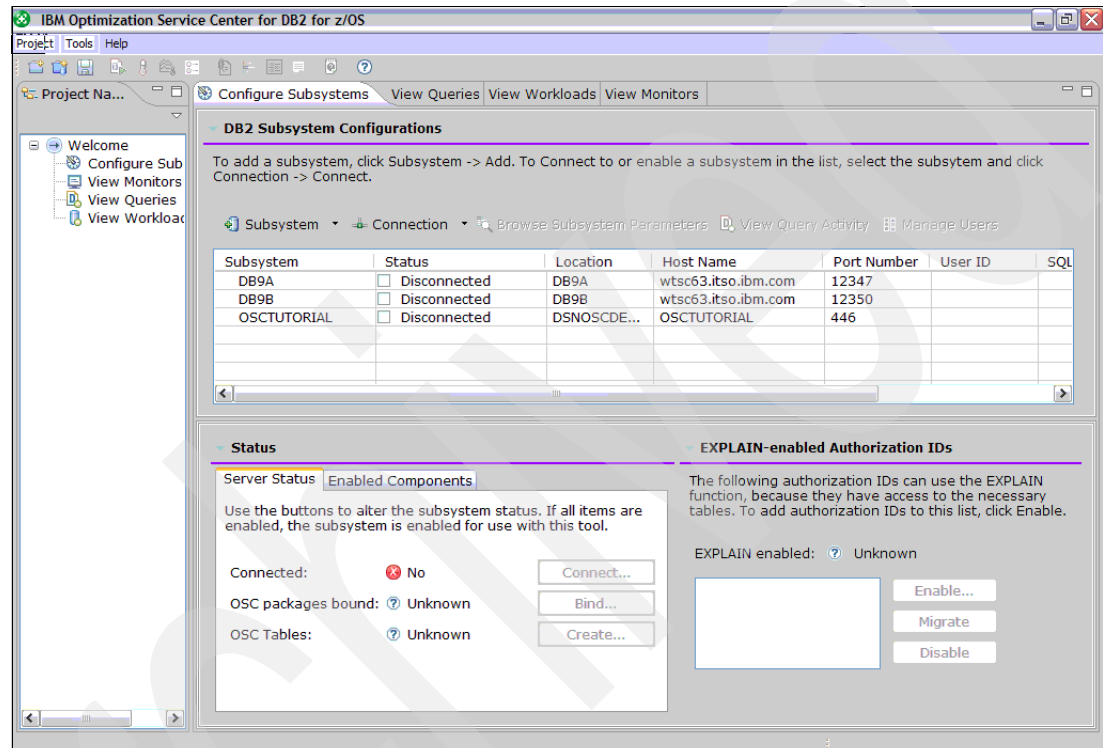


Figure 2-1 Startup on DB2 Subsystem Configurations window

1. Select the DB2 subsystem that you wish to work with from the list of available subsystems in the middle of the window. Click **Connect** in the Server Status panel at the bottom left of the window to establish a connection to the subsystem. We will connect to DB9A.



2. A panel opens asking for your TSO user ID and password as shown in Figure 2-2. If you previously asked Optimization Service Center to remember your user ID, then you only need to enter your password. As soon as you enter your password, the **Connect** button becomes active. Click it to connect to the subsystem.

**Subsystem Properties**

**Connect to subsystem**

Specify the authorization information for the subsystem and click Connect. When the subsystem is connected, click Next or Finish.

**Steps**

- 1. Location
- 2. Connect

Complete the following fields and click Connect.

User ID: PAOLOR4

Password:

SQL ID: PAOLOR4

☒ Remember login ID

**Connect**

The following fields describe the OSC status of this subsystem:

Connected: No

OSC packages bound: Unknown

EXPLAIN enabled: Unknown

OSC Tables: Unknown

< Back   Next >   Finish   Cancel

Figure 2-2 Specify User ID and password

3. The panel changes to show that you are connected and that all Optimization Service Center objects are defined and available, as shown in Figure 2-3.

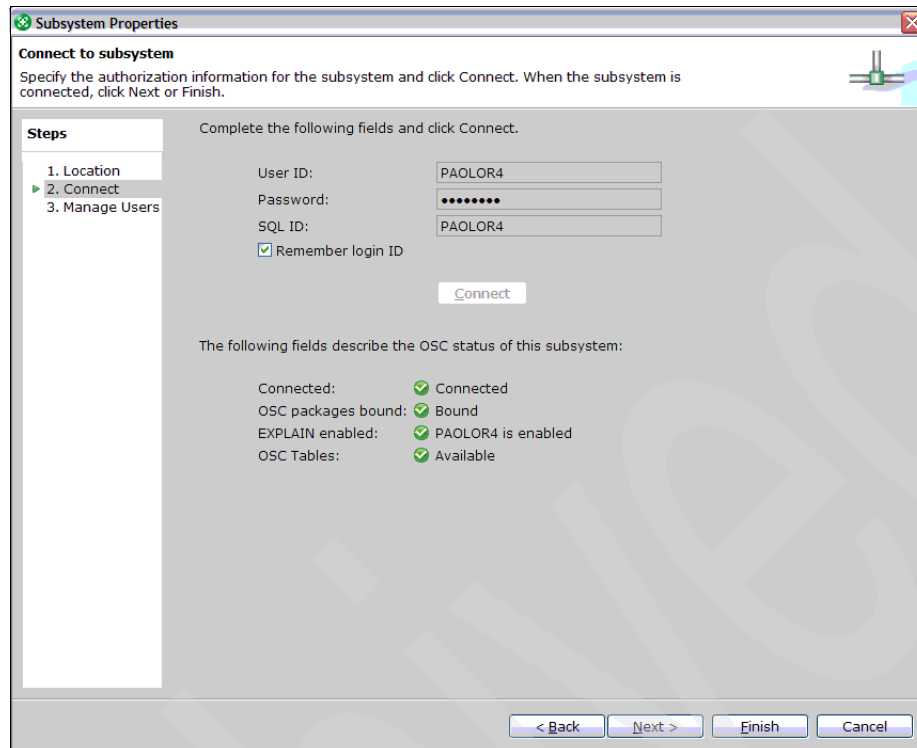


Figure 2-3 Successful connection

4. Click **Finish** to return to the DB2 Subsystem Configurations window with the connection to the DB9A subsystem activated, as shown in Figure 2-4.

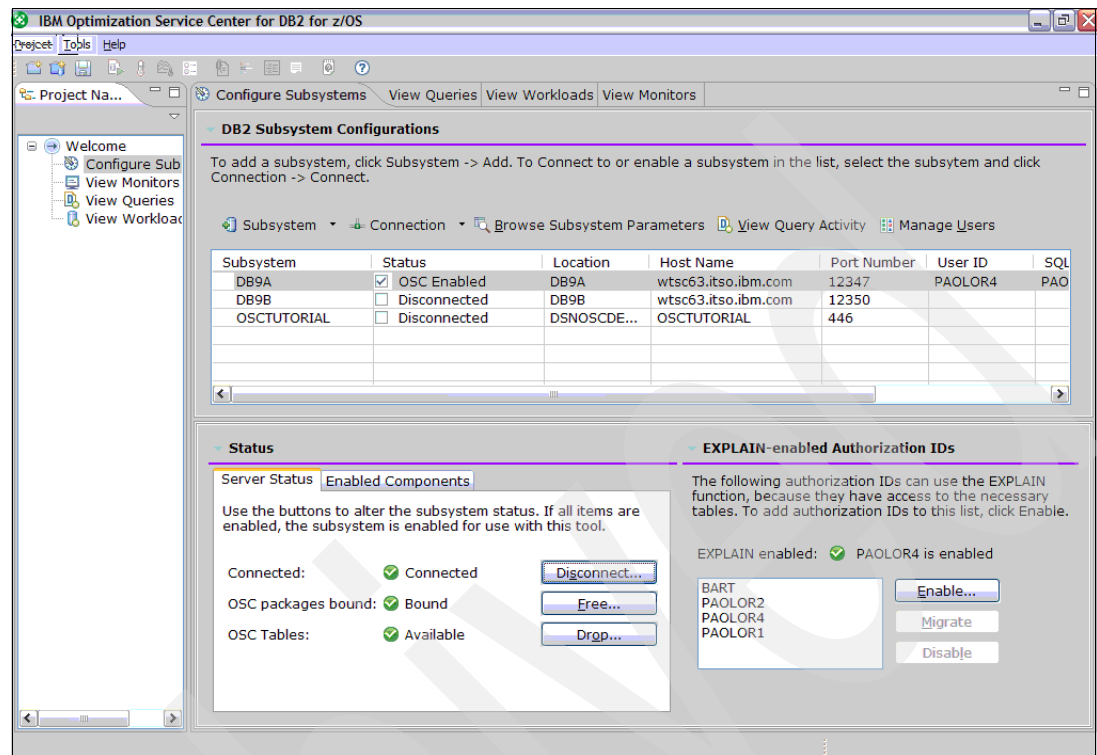


Figure 2-4 Connected and ready to work

## 2.2 Creating a project

To create a project:

1. Using the menu bar at the top of the Optimization Service Center window, select the **Project** drop down menu, and from that menu select the **New Query Project** option (Figure 2-5).

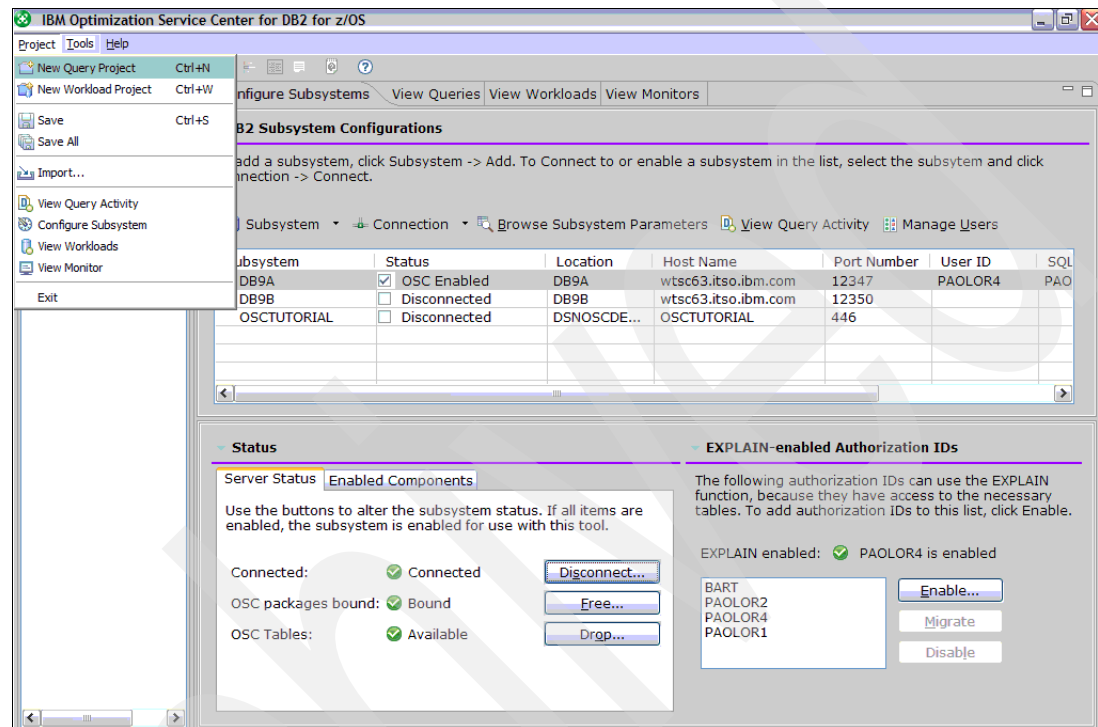


Figure 2-5 Create new query project

2. A new project, named New Project 1, is created to allow you to tune a single SQL statement, as shown in Figure 2-6.

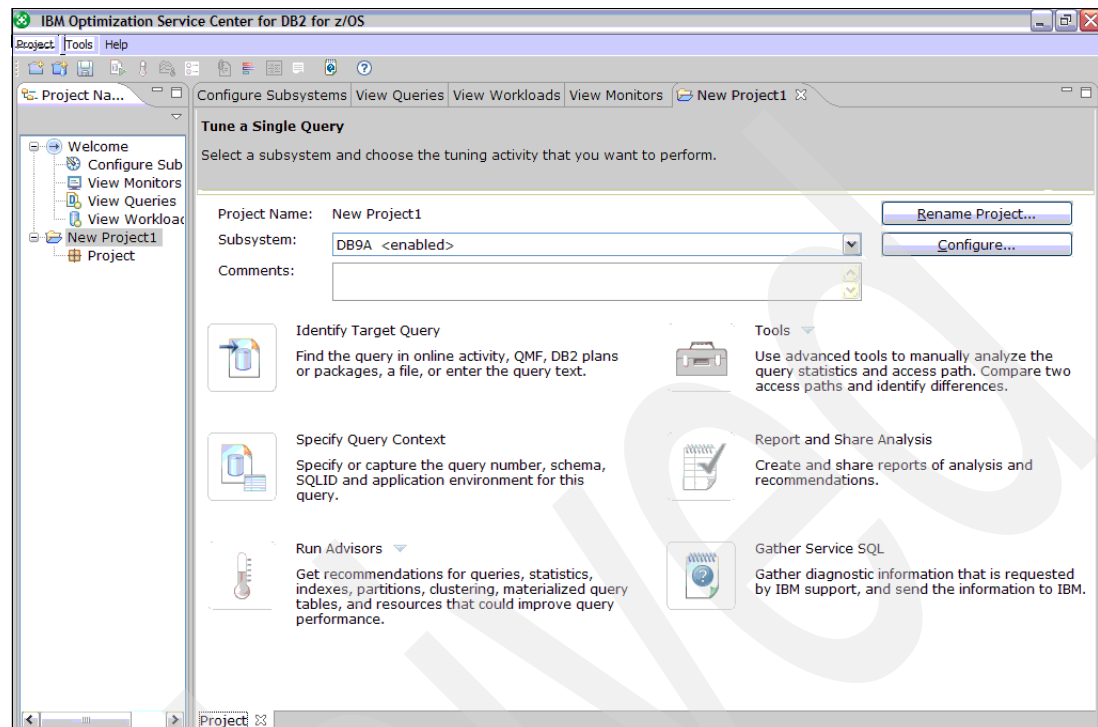


Figure 2-6 Project to tune a single query

You can use the **Rename Project** button to the right of the project name to give the project a name that is meaningful. You can also add a comment. The project automatically chooses DB9A as the DB2 subsystem to work with because that is the only subsystem you are currently connected to. Change the name of the project to Join01 and add a comment.

## 2.3 Investigating a query by typing the text

To investigate a query:

1. Click **Identify Target Query** as shown in Figure 2-7.

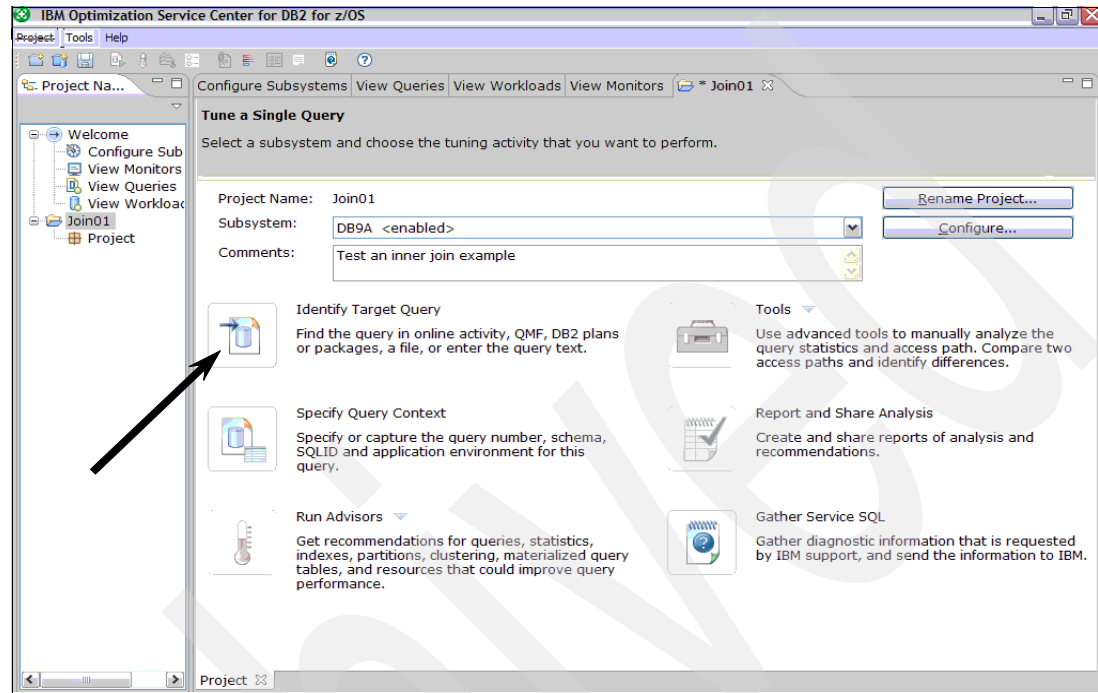


Figure 2-7 Prepare to identify target query

2. A window opens allowing you to specify the source of the query you wish to work with, as shown in Figure 2-8.

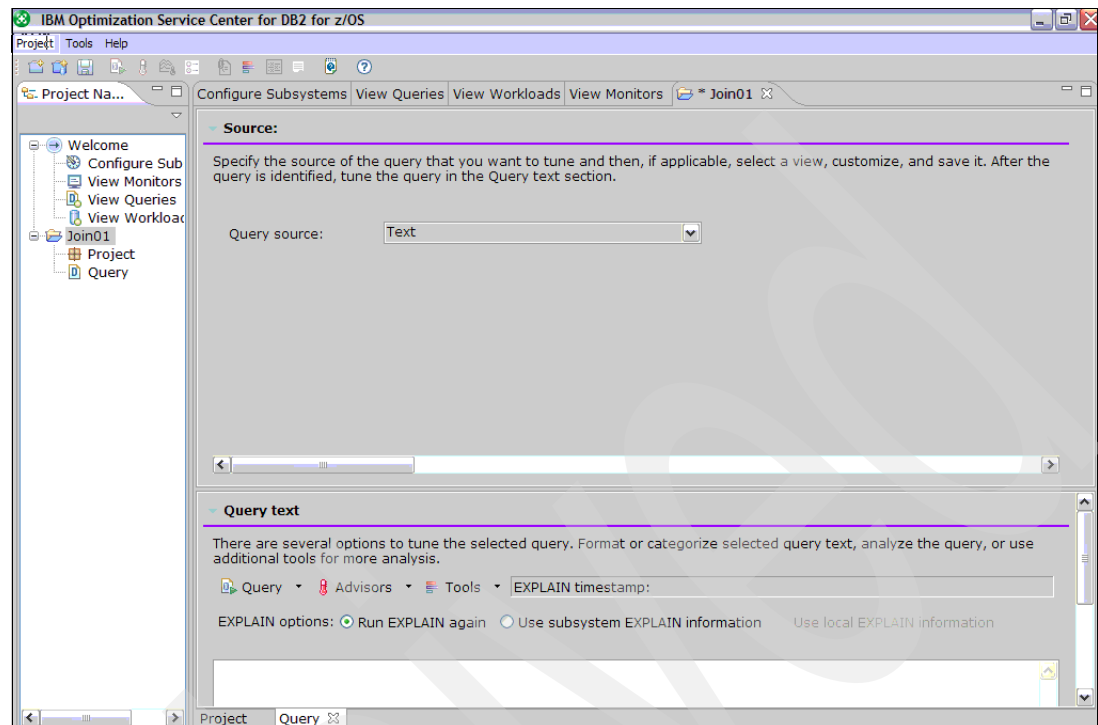


Figure 2-8 Identify target query

The query source is shown as Text. This is the option to use if you wish to type the SQL statement into the Optimization Service Center yourself. You can type the text of the SQL statement into the box at the bottom of the panel. Note that the bottom panel has a scrollbar on the right side that you can use to see more of the text box.

Figure 2-9 shows the text of an inner join of two tables, PAOLOR4.DEPT and PAOLOR4.EMP. These are copies of the IBM sample tables DEPT and EMP.

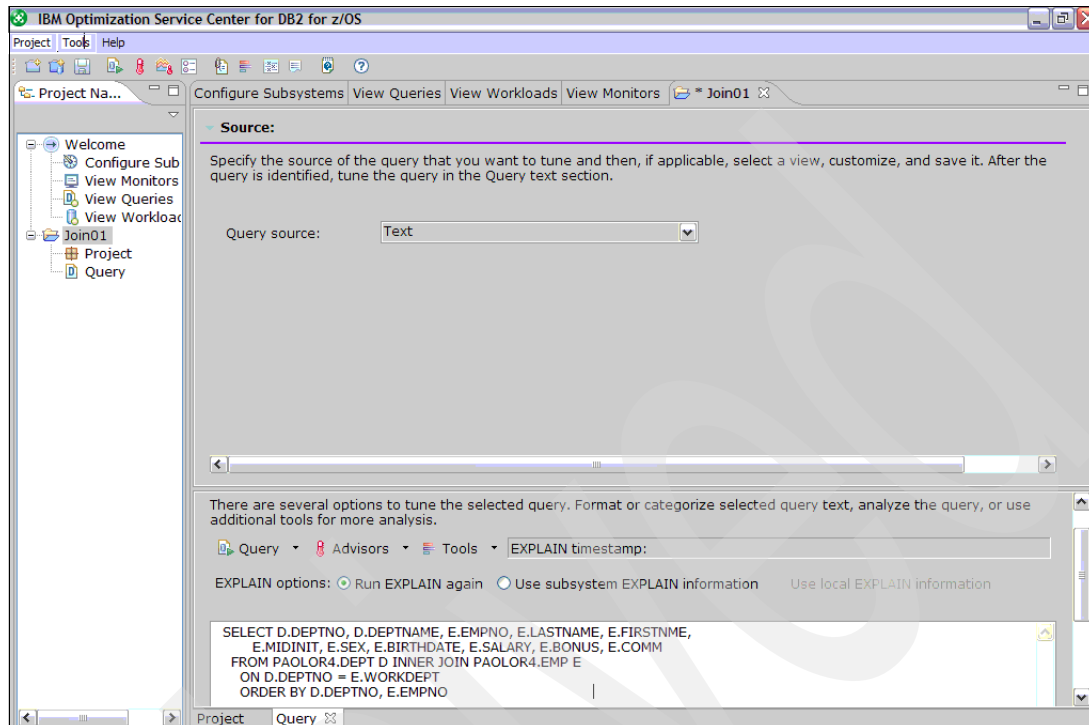


Figure 2-9 Type in the SQL statement

3. You can now choose an option from one of the three pull-down menus available in the bottom panel, above the text box. The menus are labelled Query, Advisors, and Tools.



4. You can run the query and see the result by choosing the **Run Query** option from the Query drop down menu as shown in Figure 2-10.

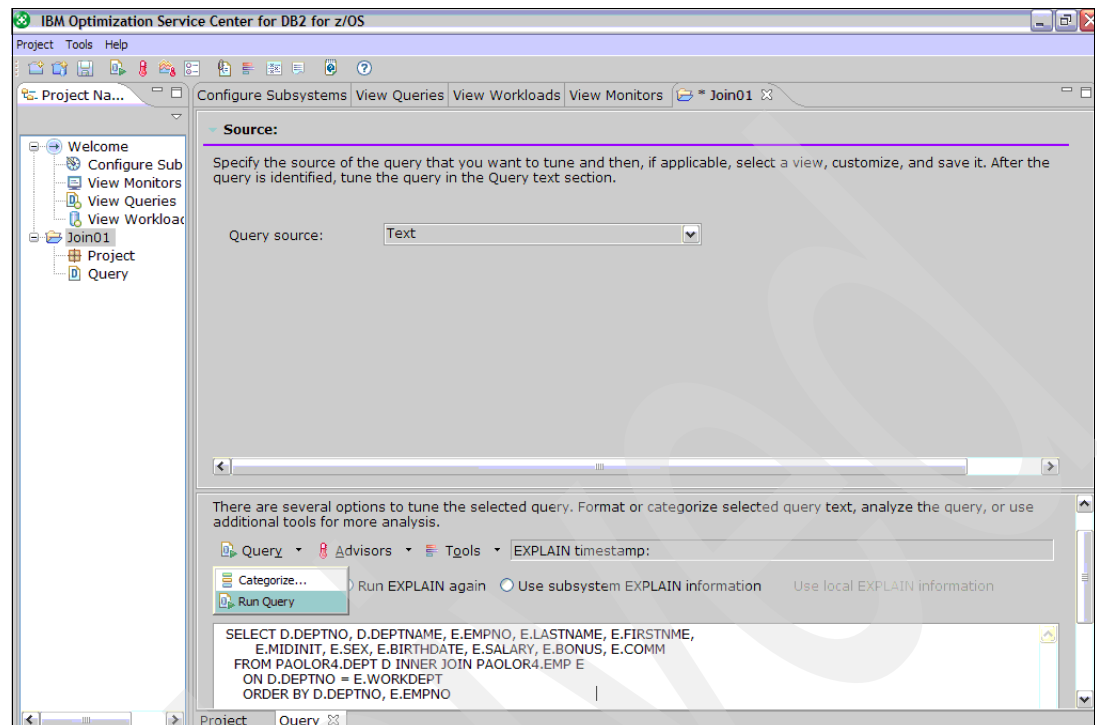


Figure 2-10 The Run Query option

- The query is run and the result shown in a result tab as shown in Figure 2-11. Notice the three tabs at the bottom of the window, labelled Project, Query, and Run Query Result. The highlighted active tab is the Run Query Result tab. By clicking the **Query** tab, you can return to the window where you typed in the query text. Clicking the **Project** tab takes you to the window where you defined the project. Notice also that the same three options are available in the left-hand Project Navigator pane under the name of the project (Join01). Double-click the option in the Project Navigator pane as an alternative way to switch between these three screens.

All of the rows are displayed. The number of rows is 42.

DEPTNO	DEPTNAME	EMPNO	LASTNAME	FIRSTNAME	MIDINIT
A00	SPIFFY COMPUTER SERVICE DIV.	000010	HAAS	CHRISTINE	I
A00	SPIFFY COMPUTER SERVICE DIV.	000110	LUCCHESI	VINCENZO	G
A00	SPIFFY COMPUTER SERVICE DIV.	000120	O'CONNELL	SEAN	
A00	SPIFFY COMPUTER SERVICE DIV.	200010	HEMMINGER	DIAN	J
A00	SPIFFY COMPUTER SERVICE DIV.	200120	ORLANDO	GREG	
B01	PLANNING	000020	THOMPSON	MICHAEL	L
C01	INFORMATION CENTER	000030	KWAN	SALLY	A
C01	INFORMATION CENTER	000130	QUINTANA	DOLORES	M
C01	INFORMATION CENTER	000140	NICHOLLS	HEATHER	A
C01	INFORMATION CENTER	200140	NATZ	KIM	N
D11	MANUFACTURING SYSTEMS	000060	STERN	IRVING	F
D11	MANUFACTURING SYSTEMS	000150	ADAMSON	BRUCE	
D11	MANUFACTURING SYSTEMS	000160	PIANKA	ELIZABETH	R
D11	MANUFACTURING SYSTEMS	000170	YOSHIMURA	MASATOSHI	J
D11	MANUFACTURING SYSTEMS	000180	SCOUTTEN	MARILYN	S
D11	MANUFACTURING SYSTEMS	000190	WALKER	JAMES	H
D11	MANUFACTURING SYSTEMS	000200	BROWN	DAVID	
D11	MANUFACTURING SYSTEMS	000210	JONES	WILLIAM	T
D11	MANUFACTURING SYSTEMS	000220	LUTZ	JENNIFER	K
D11	MANUFACTURING SYSTEMS	200170	YAMAMOTO	KIYOSHI	
D11	MANUFACTURING SYSTEMS	200220	JOHN	REBA	K
D21	ADMINISTRATION SYSTEMS	000070	PULASKI	EVA	D
D21	ADMINISTRATION SYSTEMS	000230	JEFFERSON	JAMES	J
D21	ADMINISTRATION SYSTEMS	000240	MARINO	SALVATORE	M
D21	ADMINISTRATION SYSTEMS	000250	SMITH	DANIEL	S
D21	ADMINISTRATION SYSTEMS	000260	JOHNSON	SYBIL	V
D21	ADMINISTRATION SYSTEMS	000270	PEREZ	MARIA	L
D21	ADMINISTRATION SYSTEMS	200240	MONTEVERDE	ROBERT	M
E01	SUPPORT SERVICES	000050	GEYER	JOHN	B
E11	OPERATIONS	000090	HENDERSON	EILEEN	W
E11	OPERATIONS	000280	SCHEIDER	FRANK	D

Figure 2-11 Query result

6. Lets switch back to the Query window and use the Access Plan Graph option from the Tools menu (as shown in Figure 2-12) to look at the access path graph.

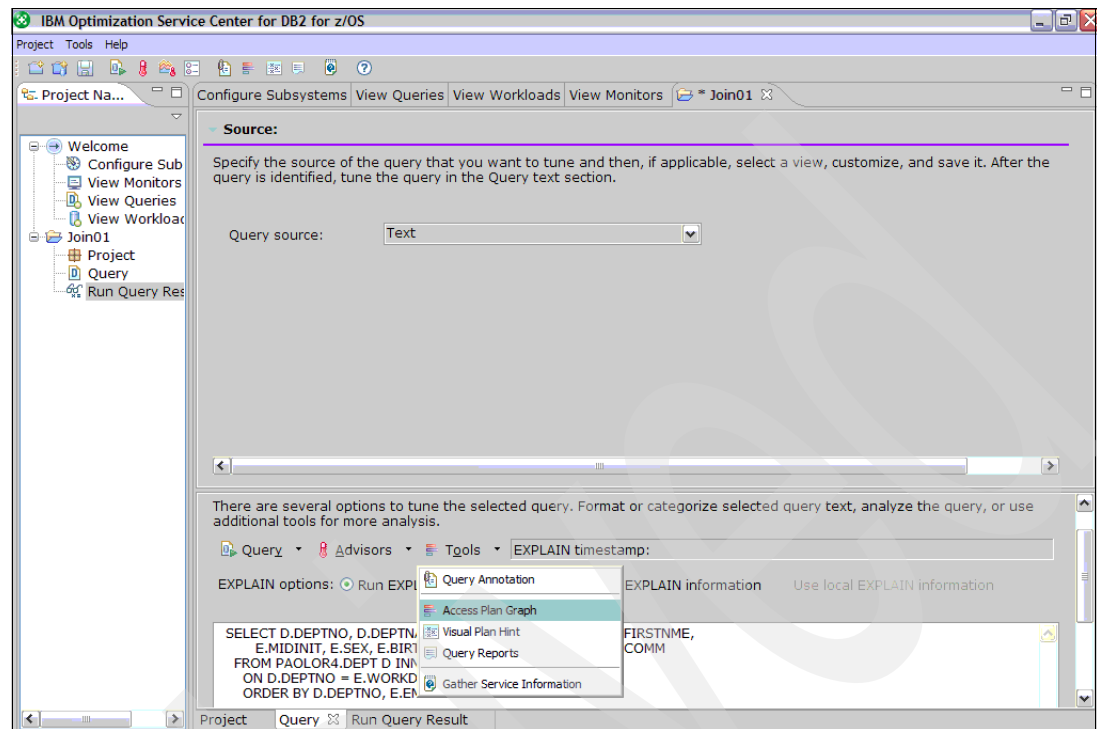


Figure 2-12 Selecting the Access Plan Graph option

It takes a few seconds for Optimization Service Center to generate the access path graph and you receive status messages while the process is running. When the graph is ready, it displays as shown in Figure 2-13.

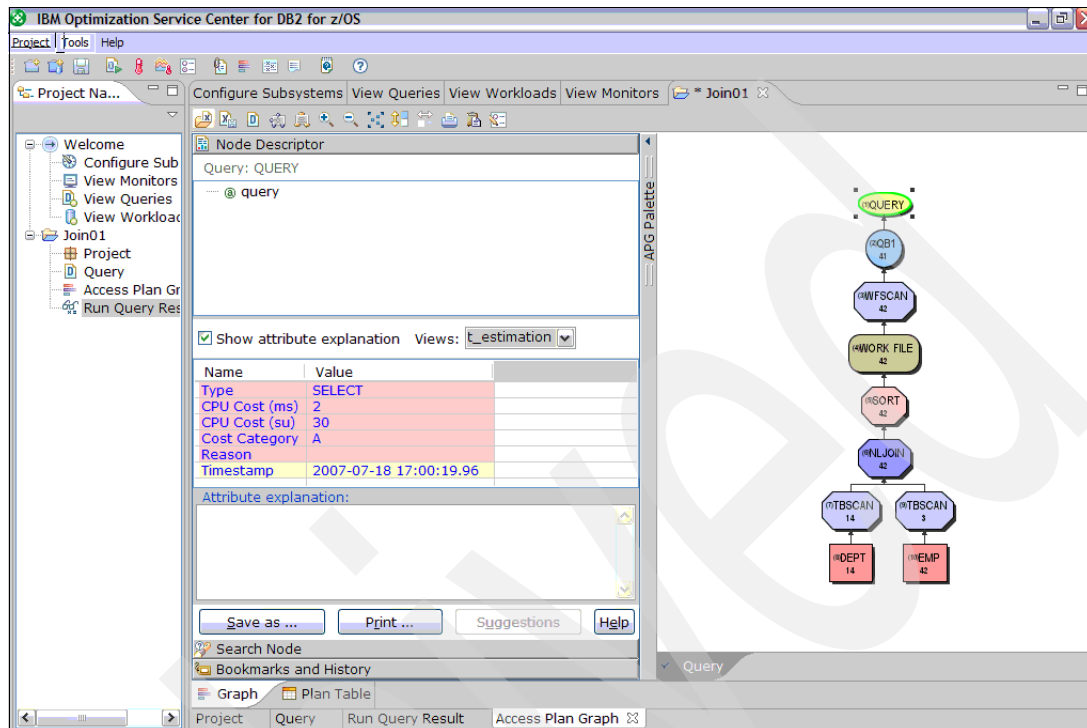


Figure 2-13 Viewing the Access Plan Graph

- Note that an extra tab has been added to the bottom of the window. There are now four tabs: Project, Query, Run Query Result, and Access Plan Graph. The Access Plan Graph is the active tab. Above these tabs are two other tabs that are part of the Access Plan Graph window. They are labelled Graph and Plan Table. If you prefer the raw plan table rows, select the Plan Table tab, as shown in Figure 2-14. You can use the scroll bar at the bottom of the plan table rows to scroll to the right and see more plan table columns.

QUERYNO	QBLOCKNO	PLANNO	PARENT_QBLOCKNO	METHOD	MERGE_JOIN_COLS	CREATOR	TNAME
112	1	1	0	0		PAOLOR4	DEPT
112	1	2	0	1		PAOLOR4	EMP
112	1	3	0	3			

Figure 2-14 View the raw plan table rows

The short format eliminates some `PLAN_TABLE` columns, such as `APPLNAME`, `PROGNAME`, and so on. You can see these columns by selecting the long format from the select format radio bar.

Lets go back to the Access Plan Graph and look at some of the information it gives us. For a detailed explanation of the Access Plan Graph, see Chapter 8, “Query tools” on page 163.

First, look at the graph itself on the right-hand side of Figure 2-13 on page 44. It is shown in detail in Figure 2-15 on page 46. Like Visual Explain, the graph is read left to right, bottom to top. In our example, you see that the first table accessed is the `DEPT` table, which is read using a table space scan. A nested loop join is performed to join the `EMP` table, which is also accessed using a table space scan. A sort is performed on the result of the join and the result is written to a Logical Work File (LWF). The LWF is then read with a workfile scan to return the result. The numbers in each graphic object are the cardinalities, which are the number of rows processed. You can see that the `DEPT` table has 14 rows (according to the Catalog statistics) and all 14 are read by the table space scan and fed into the join. Each access into the `EMP` table (which contains 42 rows) for a single `DEPT` table row returns on average 3 rows. The join produces 42 result rows that are fed into the sort. All these numbers are the optimizer's estimates and might not reflect the actual numbers of rows processed at each stage.

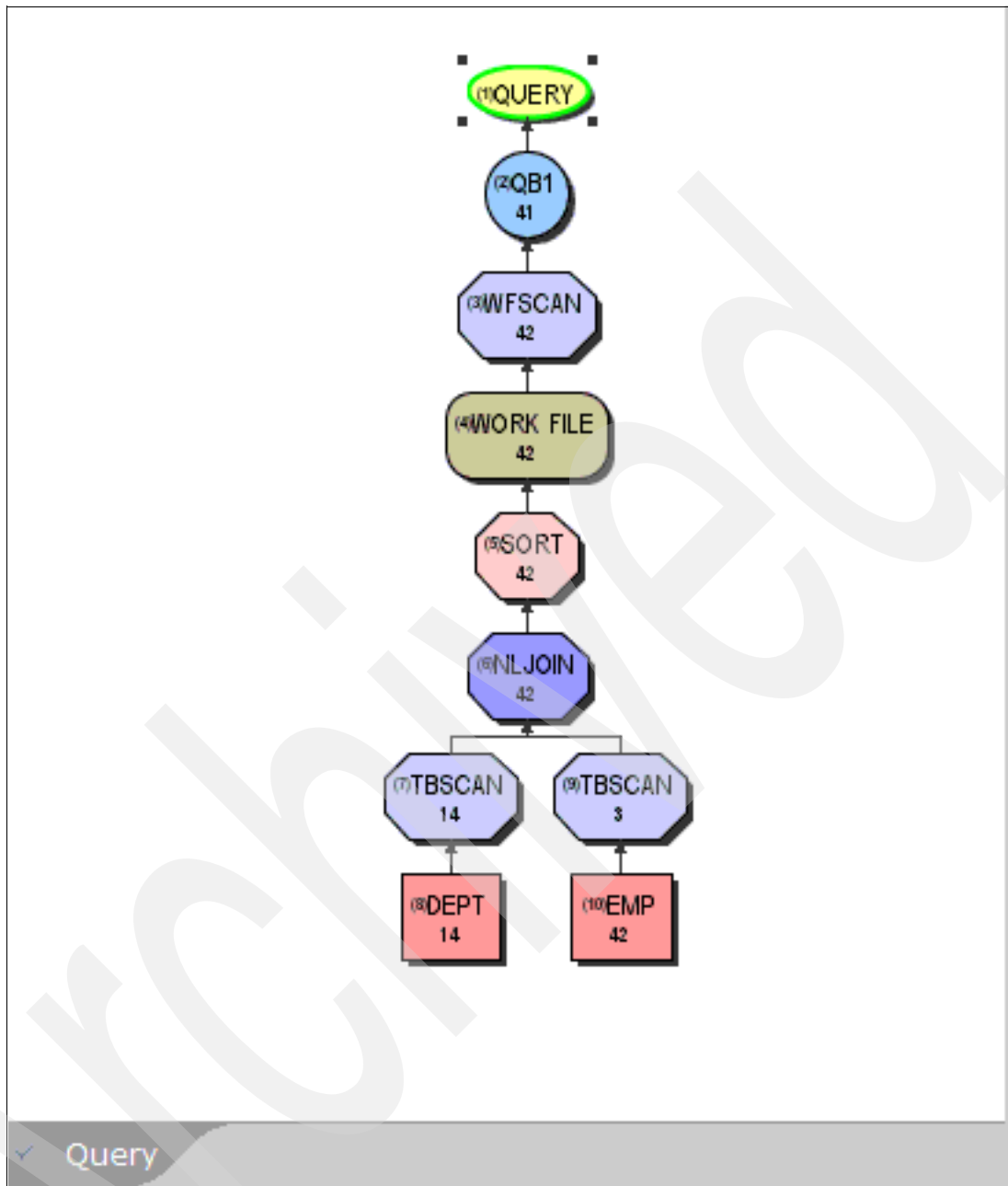


Figure 2-15 The Access Plan Graph

Now look at other information you can see on the Access Plan Graph tab. In Figure 2-16, you can see both the graph and the detailed information to the left of the graph in the Node Descriptor panel.

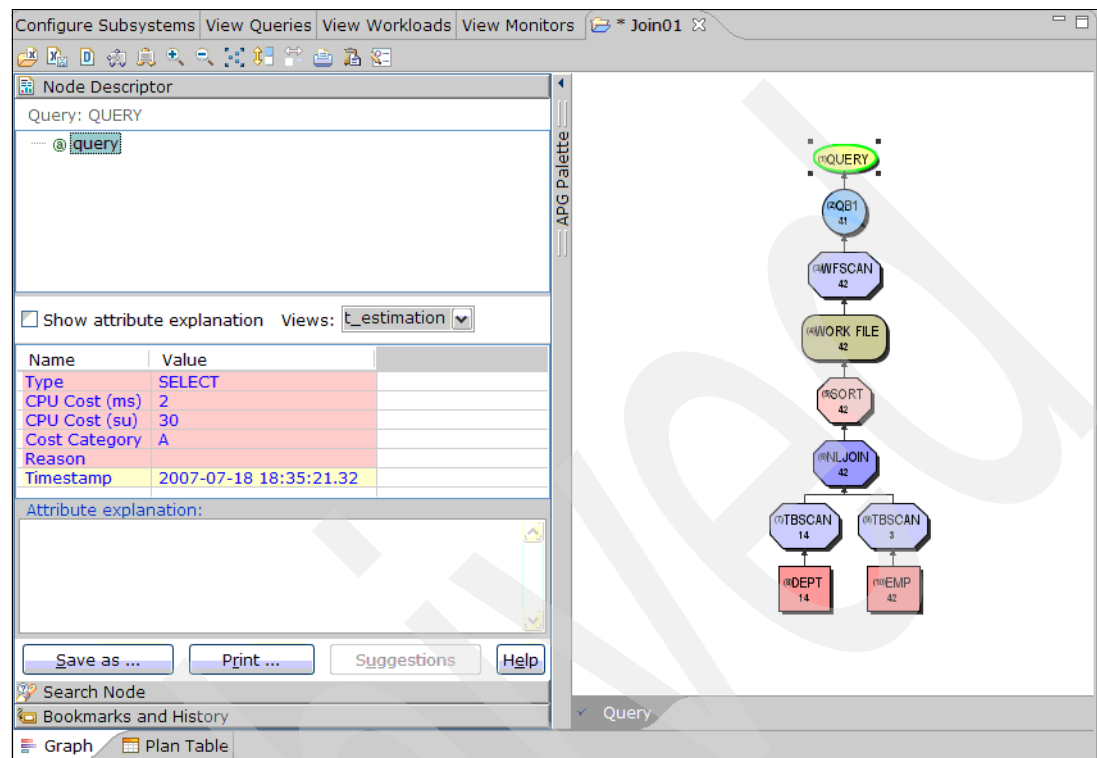


Figure 2-16 Access Plan Graph detail

Initially, the top node of the graph is selected (note the four dots surrounding it). In the middle pane on the left-hand side, you can see overall information about the query. It shows that the graph is for a Select statement, the estimated CPU cost is 2 milliseconds or 30 service units, and that the estimate is considered to be accurate (cost category A). The time stamp shows when the Explain was performed to produce the graph.

Look at the information available for other nodes on the graph. If you click the node at the bottom left that represents the DEPT table, the information in the Node Descriptor panel changes, as shown in Figure 2-17.

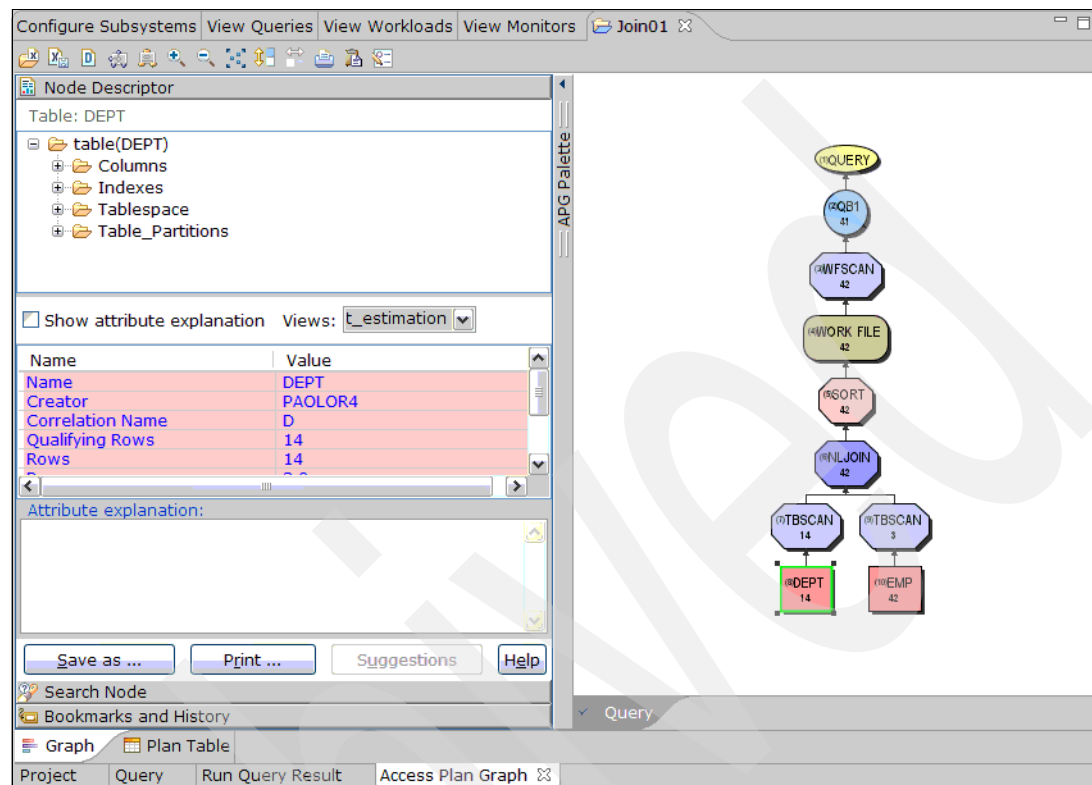


Figure 2-17 Table information

The middle pane of the Node Descriptor panel shows the name of the table (DEPT) and the creator ID (PAOLOR4) and correlation name used in the SQL statement (D). The number of qualifying rows is the number expected to qualify after applying local predicates. This is the same as the total number of rows in the table (14) in our example because the SQL statement has a no WHERE clause. Notice the scroll bar at the right of the middle pane. This shows that there is more information that you can view by scrolling down.



Now look at the top pane in the Node Descriptor panel. It shows a directory structure with the table at the top. Below it are folders containing information about columns, indexes, table spaces, and partitions. If you want to see the column definitions for the table, click the + box to the left of the name Columns. This opens up the columns folder so that you can look at the column definitions, as shown in Figure 2-18.

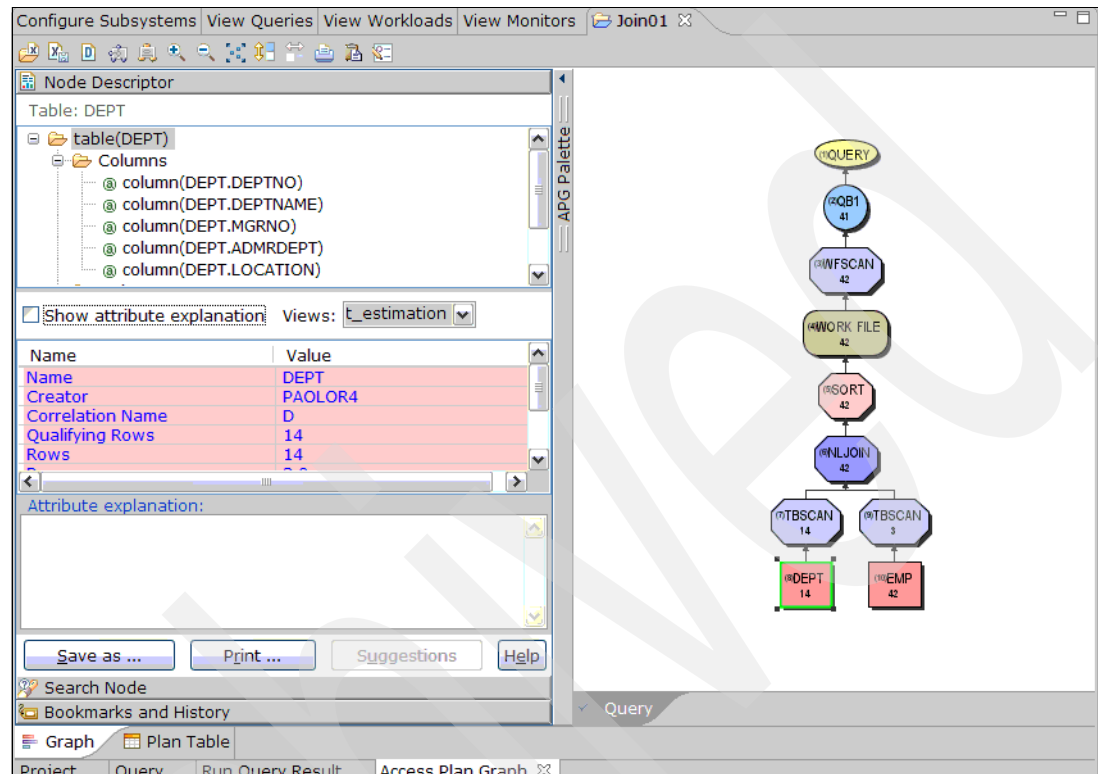


Figure 2-18 Viewing the column definitions for a table

You can see the top pane now shows a list of the column names for table DEPT. The middle pane is still showing information about the table because the table is still selected in the top pane of the Node Descriptor panel. To see information about a specific column, select it in the top pane and the middle pane will show the column information.

In Figure 2-19, you have selected the DEPTNO column in the top pane. If you look at the middle pane, you can see detailed information about the column. Its cardinality is 14 (there are 14 different values), the second highest value is "I22" and the second lowest is "B01", and it cannot take Null values. All this information is read from the DB2 Catalog. Some of it is data placed there by the Runstats utility. The scroll bar to the right of the middle panel indicates that more information is available, including the time stamp of the most recent Runstats.

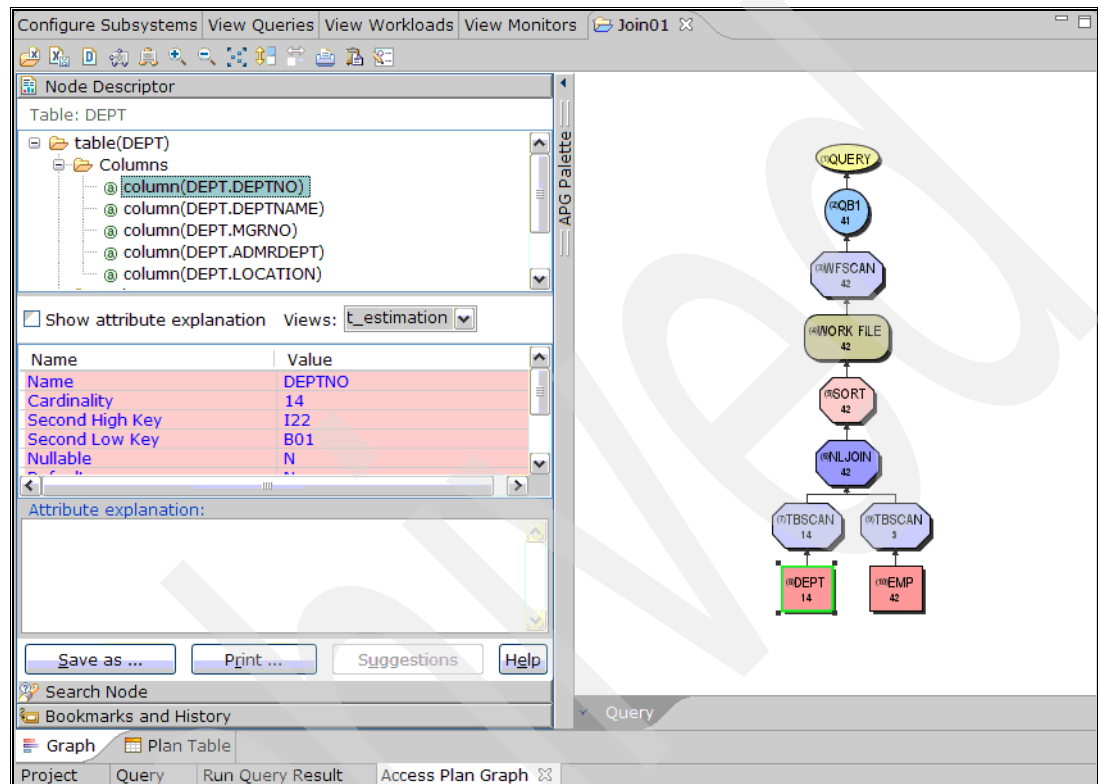


Figure 2-19 Viewing column detail

The middle pane is currently showing a subset of the information about column DEPTNO. It is showing us a Cost Estimation view that shows the information the optimizer is interested in when costing the access path. There are other views available. If you click the pull-down box at the top right of the middle pane (where it says "Views:"), you can see the other views that are available for a column, as shown in Figure 2-20 on page 51. You can see that, for a column, the available views are Cost Estimation, Space, and All. The views available vary, depending on the object selected in the access graph and in the top pane of the Node Descriptor panel. If you select the All view of the column, you can see the column definition - CHAR(3).

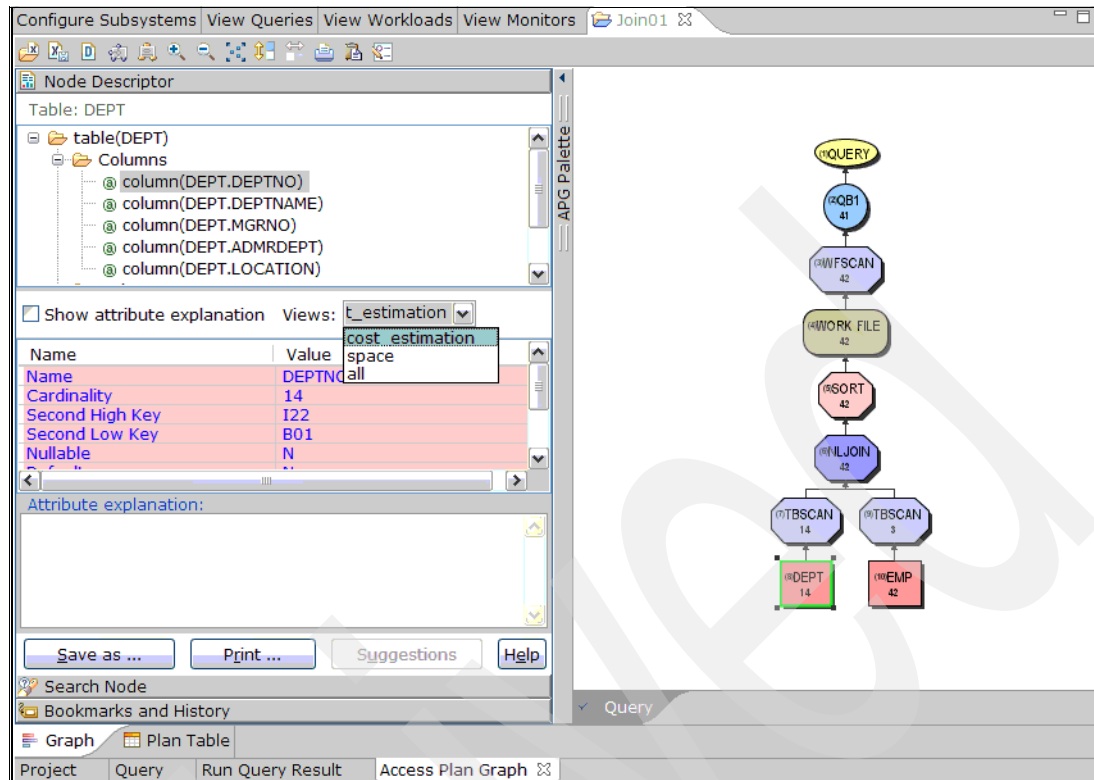


Figure 2-20 Changing the view of a column

Lets look at one other feature in the Node Descriptor panel. On the top left of the middle pane is a tick box labelled Show attribute explanation. If you click that to activate it, then the bottom pane shows a brief description of any information that you select in the middle pane. In Figure 2-21 on page 52, you have activated this feature and then selected the Cardinality information in the middle pane. The bottom pane now shows a brief description of the column cardinality.

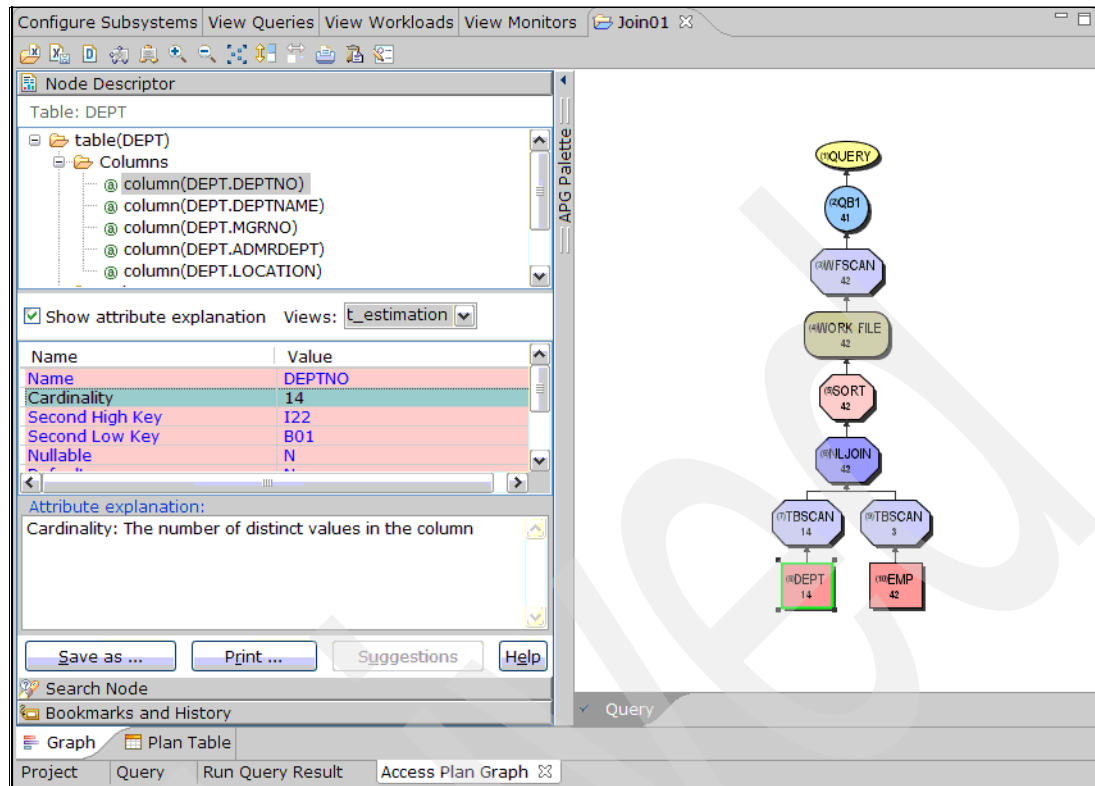


Figure 2-21 Showing attribute explanations

Lets look at the indexes defined on the table DEPT. If, in the top pane, you click the - box to the left of Columns, it closes the list of column names. You then click the + box by the Indexes folder and you see that there is one index defined on the DEPT table called XDEPT1. Click the + box by the index to show a folder called Indexkeys. Click the + box by Indexkeys to see the columns that make up the index key.

The result of this sequence of operations is shown in Figure 2-22. It shows that the index XDEPT1 is based on a single column, DEPTNO. You selected the key column in the top pane to display information about it in the middle pane and selected the Ordering information in the middle pane to get an explanation in the bottom pane. You can see that key values are stored in the index in ascending sequence. The information displayed in the Node Descriptor panel is all derived from the DB2 Catalog tables.

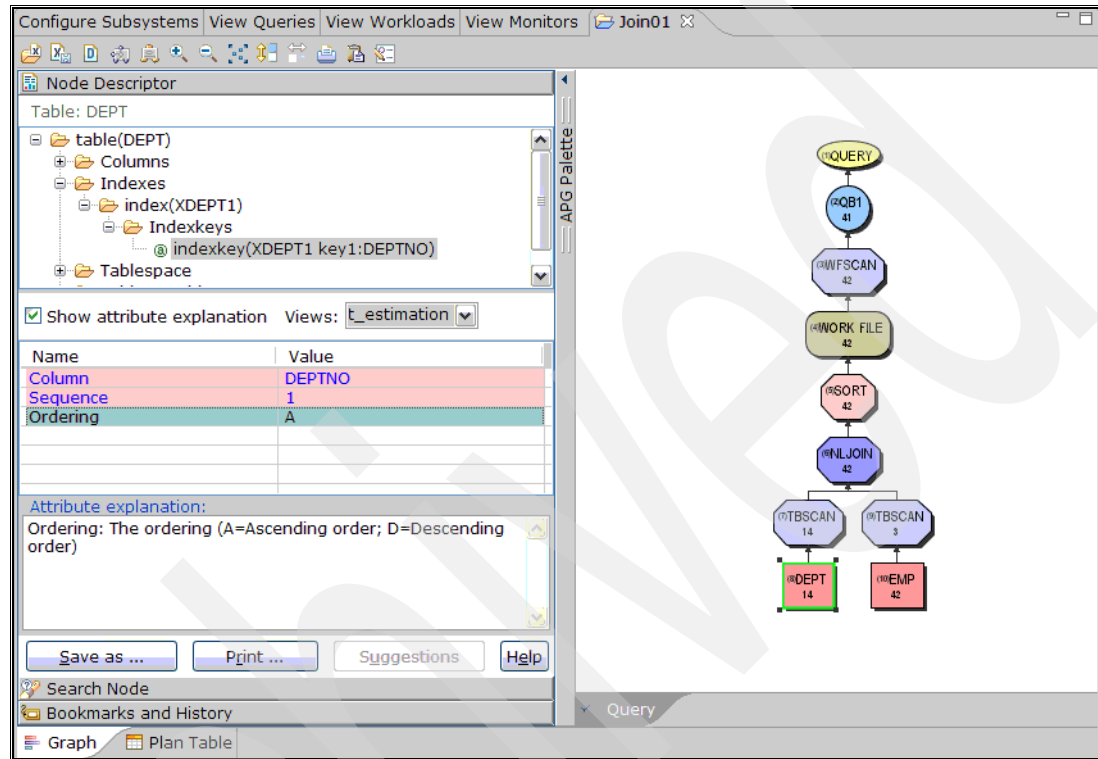


Figure 2-22 Viewing index information

Click the **NLJOIN** object in the graph and then look at the available information. Figure 2-23 shows the result. You can see that the join predicate is D.DEPTNO = E.WORKDEPT, and that each access to the inner table (EMP) is expected to return 3 rows.

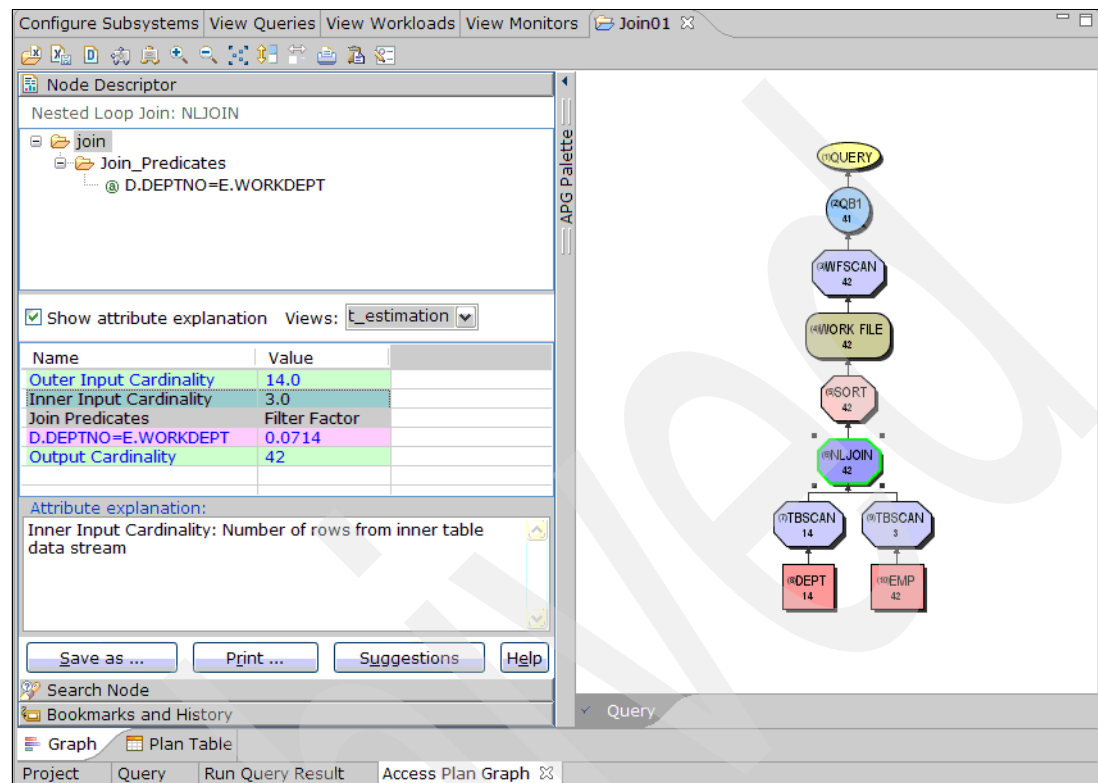


Figure 2-23 Join information

If you do not know that the NLJOIN object represents a Nested Loop Join, you can find a brief description by clicking **Help** in the Node Descriptor panel while the NLJOIN node is selected, as shown in Figure 2-24.

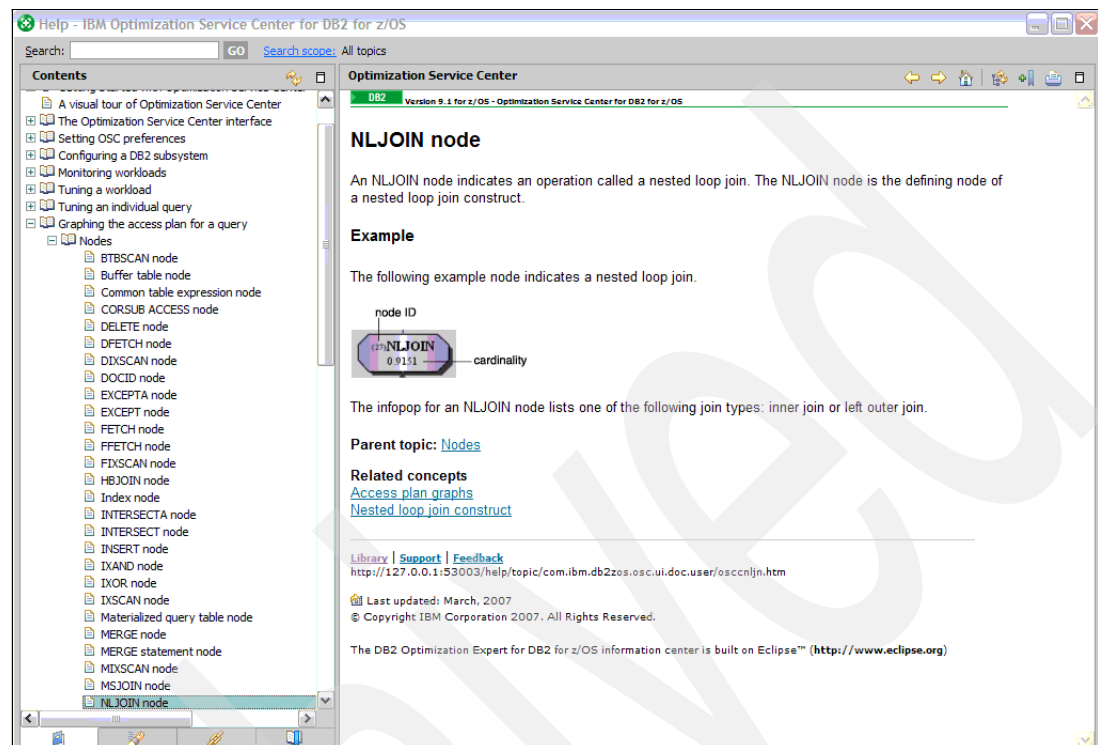


Figure 2-24 Help for NLJOIN object

One final way to get information from the access plan graph is via pop-ups, called *infopops*. If you hover the cursor over any node on the graph, an infopop appears with a subset of the information available from the Node Descriptor panel.

In Figure 2-25, we have hovered the cursor over the Sort node. You can see that 42 rows are expected to be sorted. The sort is required because ORDER BY was specified in the SQL statement. You can determine the sort keys, sort key size, and sort data size and use this information to size the sort pool and sort work file size.

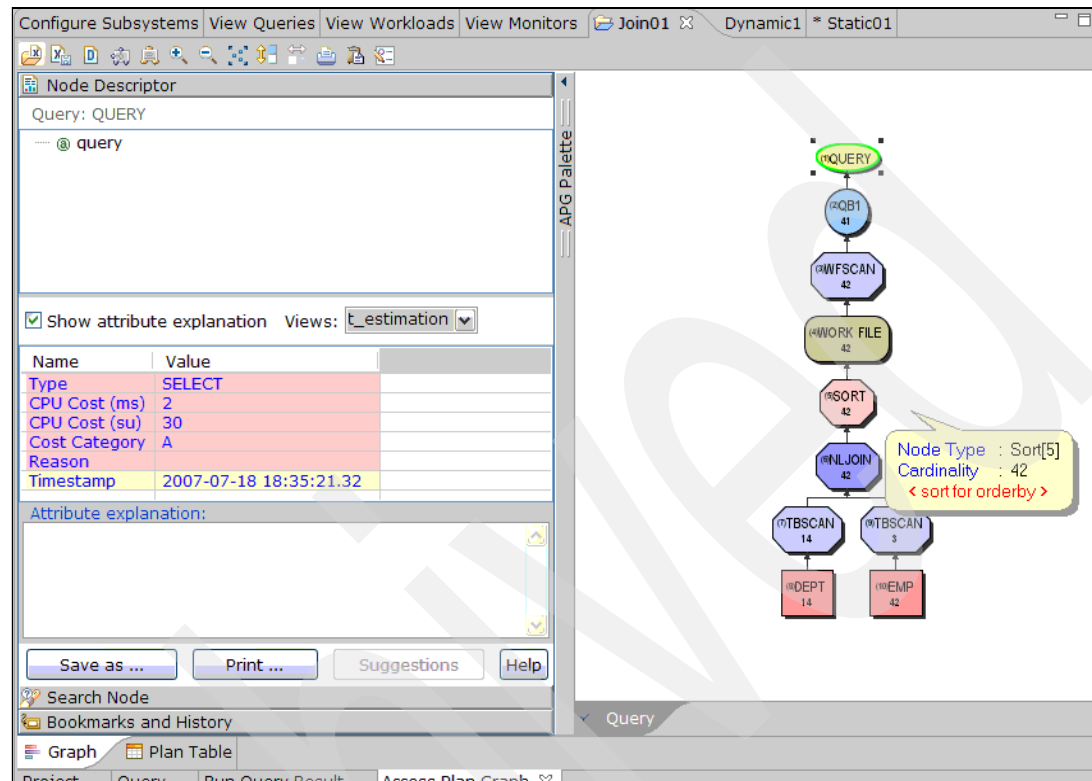


Figure 2-25 Infopop

## 2.4 Investigating a query from a package

In the previous section, we looked at typing the query we want to investigate directly into the Optimization Service Center. Now we extract an SQL statement from a package bound on the DB2 subsystem. We will create a new project called Static01 and click **Identify Target Query**. You can refer back to Figure 2-7 on page 38 to see how we did it for the previous project.



To investigate a query from a package:

1. This time, use the pull-down menu to select Catalog plan or package as the query source (see Figure 2-26). This is the option to use if you wish to extract the SQL statement from a DB2 package or plan.

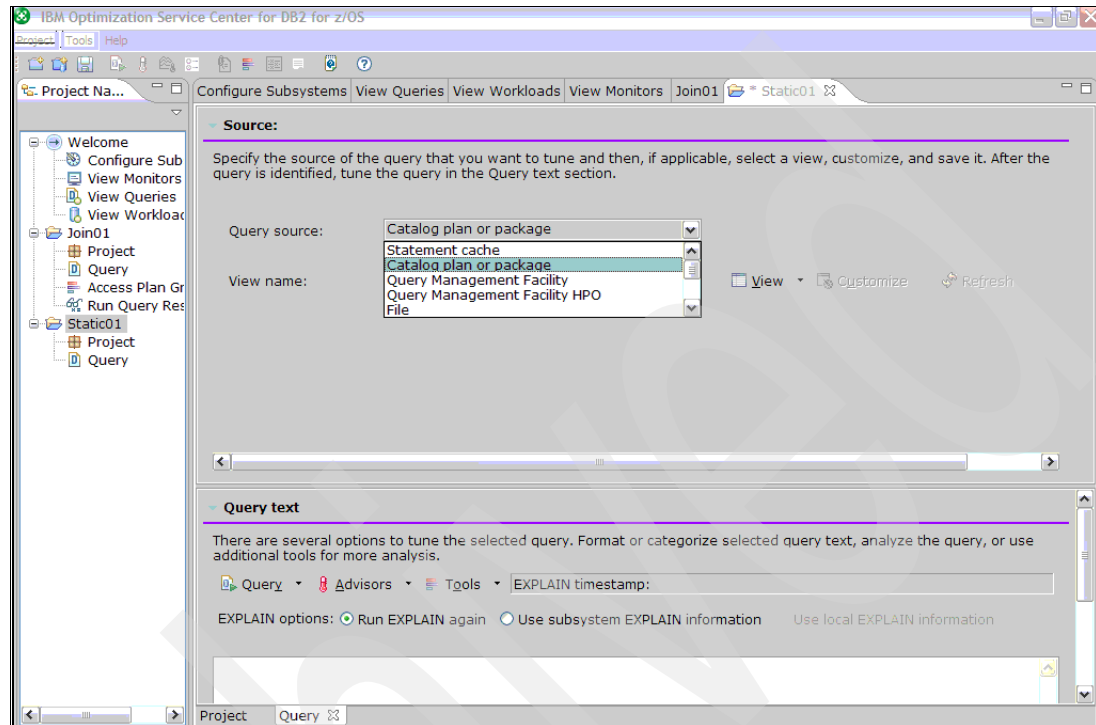


Figure 2-26 Choosing the Catalog as the query source

Now define a View to restrict the number of SQL statements extracted from the DB2 Catalog. If you use the default view, the Optimization Service Center produces an empty list. To create a new view, you can either use the pull-down list next to the description View name and select **<New view...>**, as shown in Figure 2-27, or click the **View** icon to the right of the view name and select **New** from the drop down menu.

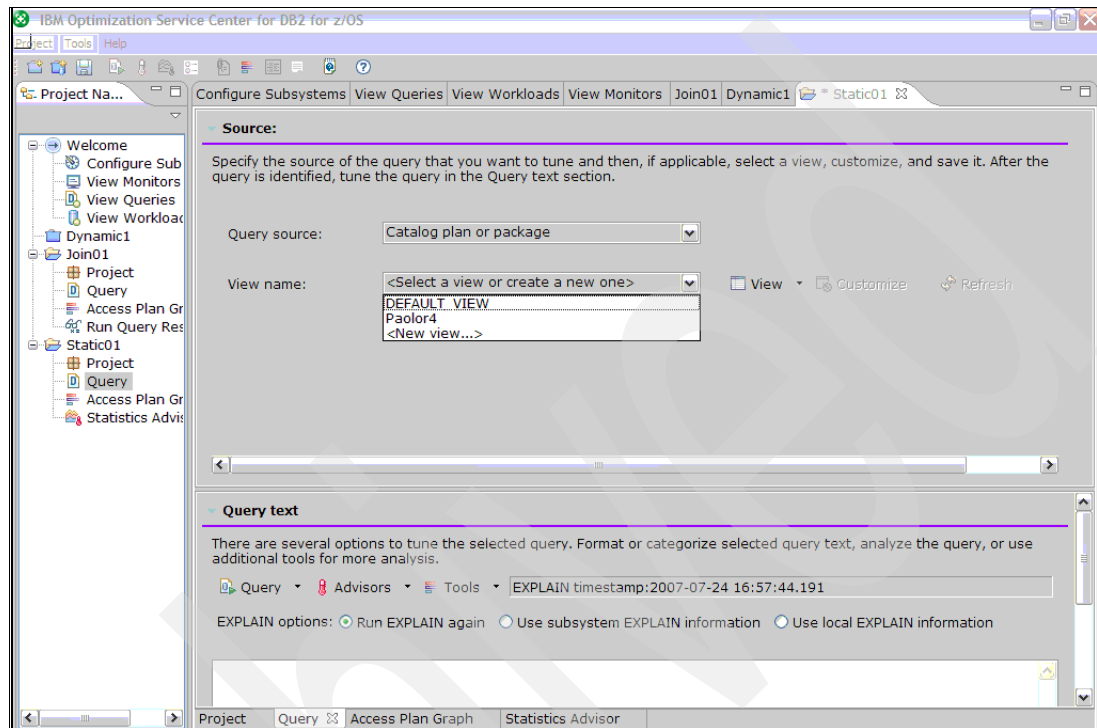


Figure 2-27 Selecting a view

The Create View panel opens as shown in Figure 2-28. In the figure, you typed in a name for the View name (Paolor4) and on the Packages tab, you double-clicked the Value column for OWNER and typed PAOLOR4. This extracts SQL statements from all packages with Owner Auth ID PAOLOR4. You can specify other criteria, as shown in the figure, to restrict the packages that are processed.

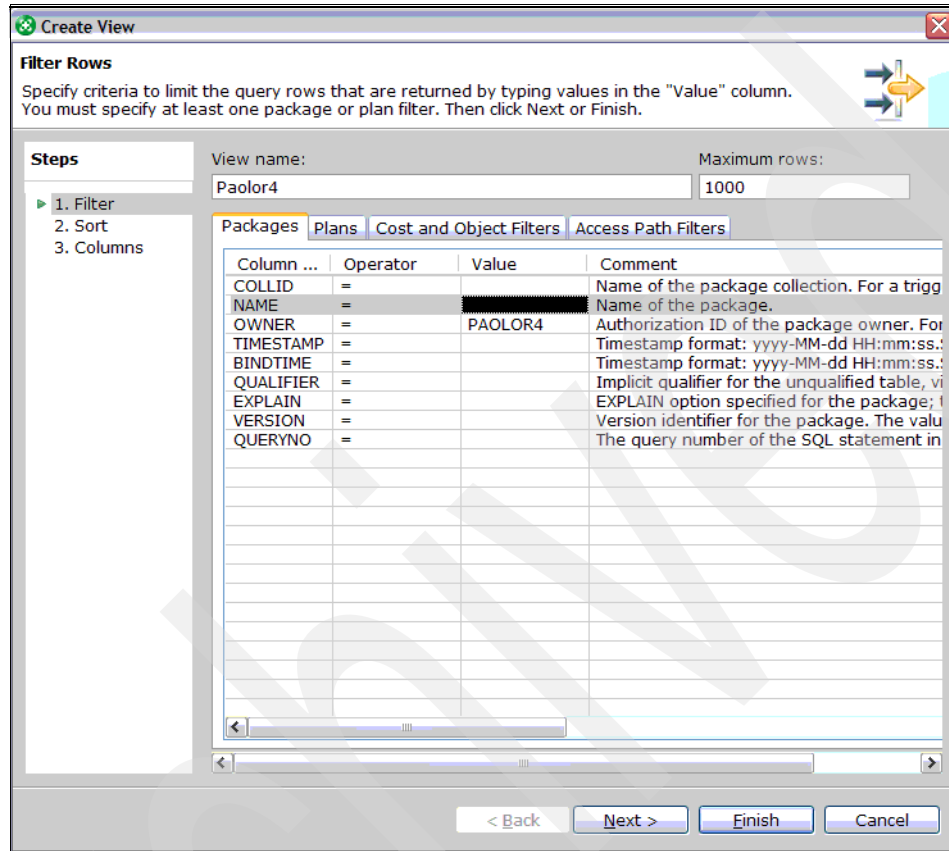


Figure 2-28 Create View panel

2. You can click the **Plans** tab to specify restrictive criteria for plans. Leave the Plan options blank if you are selecting the Package filter criteria. Make sure that all options are deselected on the Access Path Filter tab initially (see Figure 2-29), as selecting options here might cause SQL statements to be discarded.

**Note:** If you specify selection options for both packages and plans, for example specifying owner for both, you might get the SQL statements from your packages listed twice. Optimization Service Center finds them once based on the package filter specifications. It then finds the plans and locates all the packages accessible through those plans. If it finds the same packages, they get included twice.

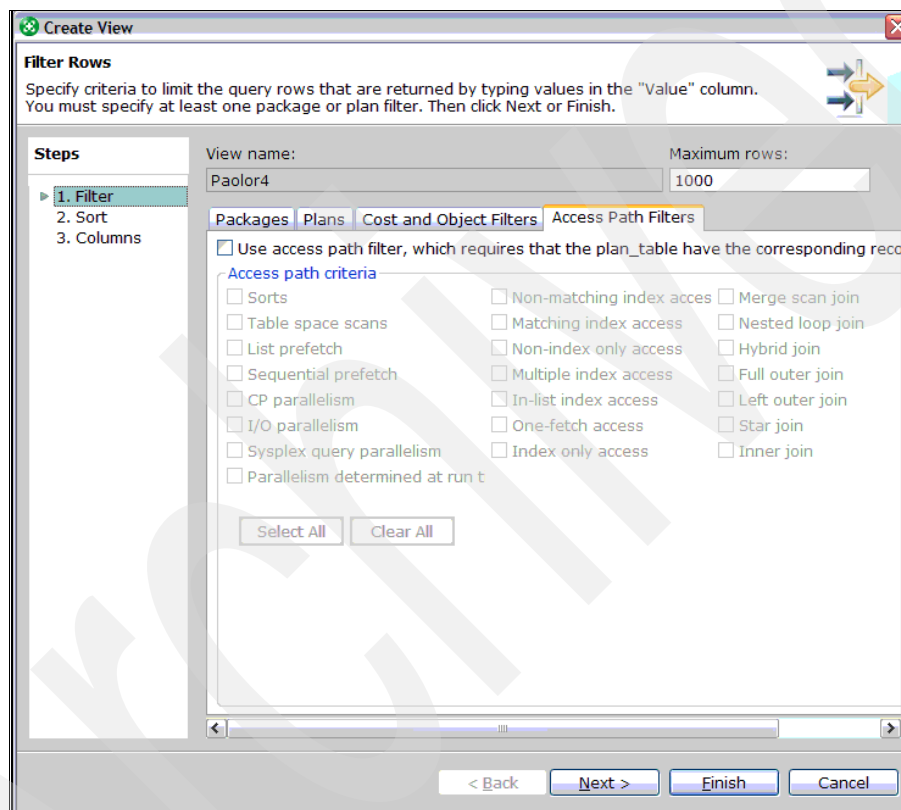


Figure 2-29 Access Path Filters tab

3. Specify the filter criteria for the view and click **Next** to specify the ordering of the SQL statements. In Figure 2-30, you selected the column STMTNO and clicked the right single arrow button to move it to the right-hand pane.

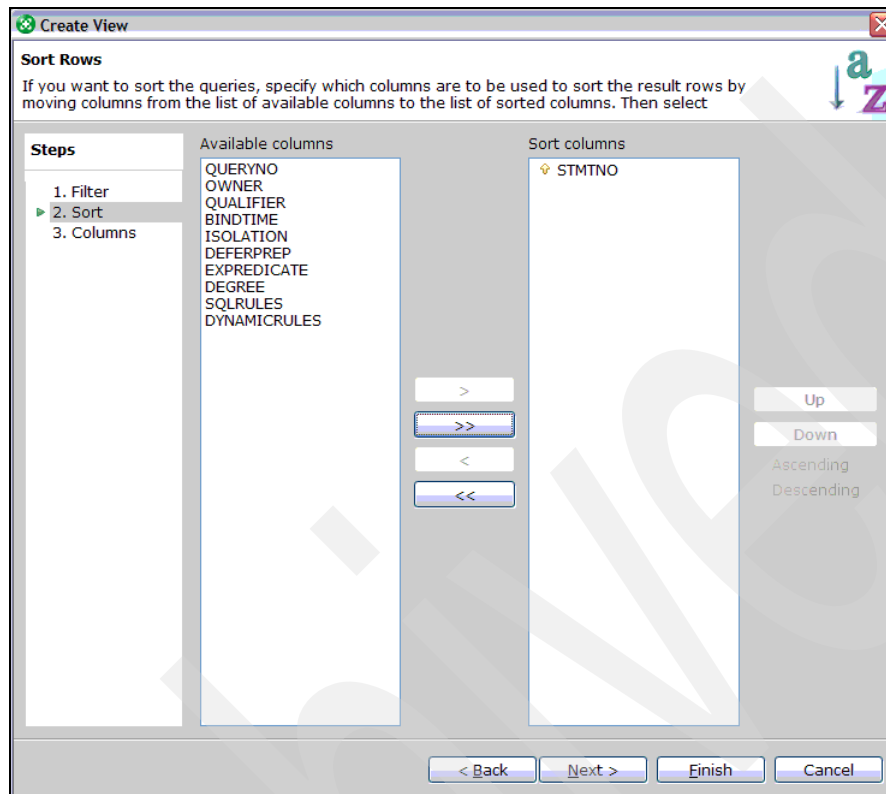


Figure 2-30 Choose sort sequence

4. The SQL statements returned from the DB2 Catalog are sorted by statement number. Click **Next** to select the data to be returned for each SQL statement as shown in Figure 2-31. Take the defaults, so click **Finish** to complete the view definition. As soon as you click Finish, Optimization Service Center starts to access the DB2 Catalog. You see a progress indicator during the access.

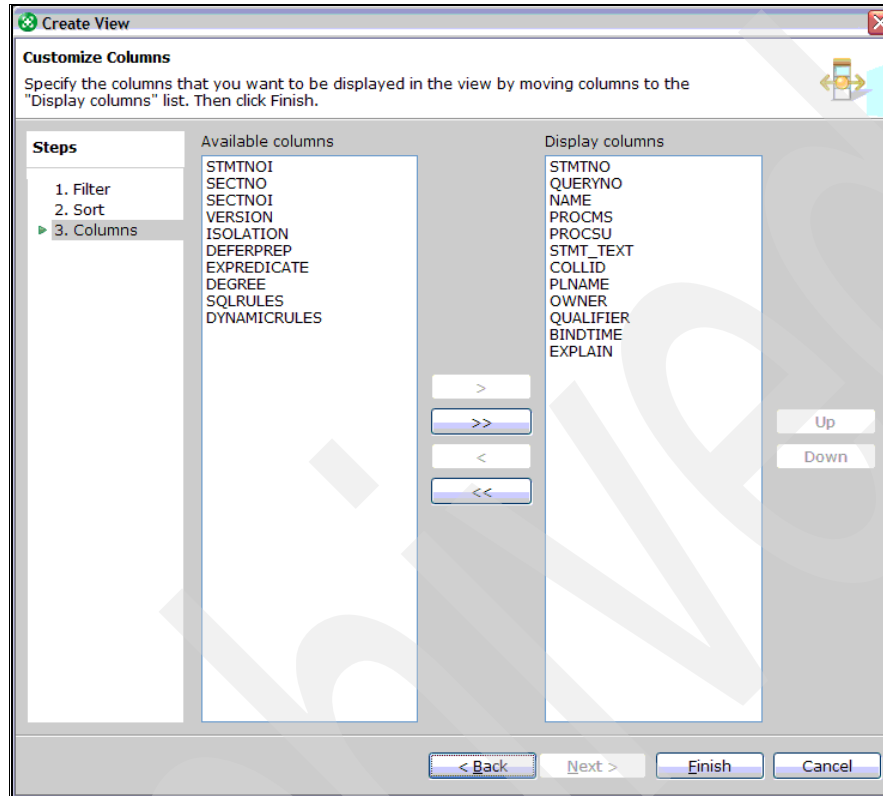


Figure 2-31 Choose data to display

When it has retrieved the statements from the packages with owner PAOLOR4 (in our example), it returns to the Source panel with the SQL statements listed (see Figure 2-32).

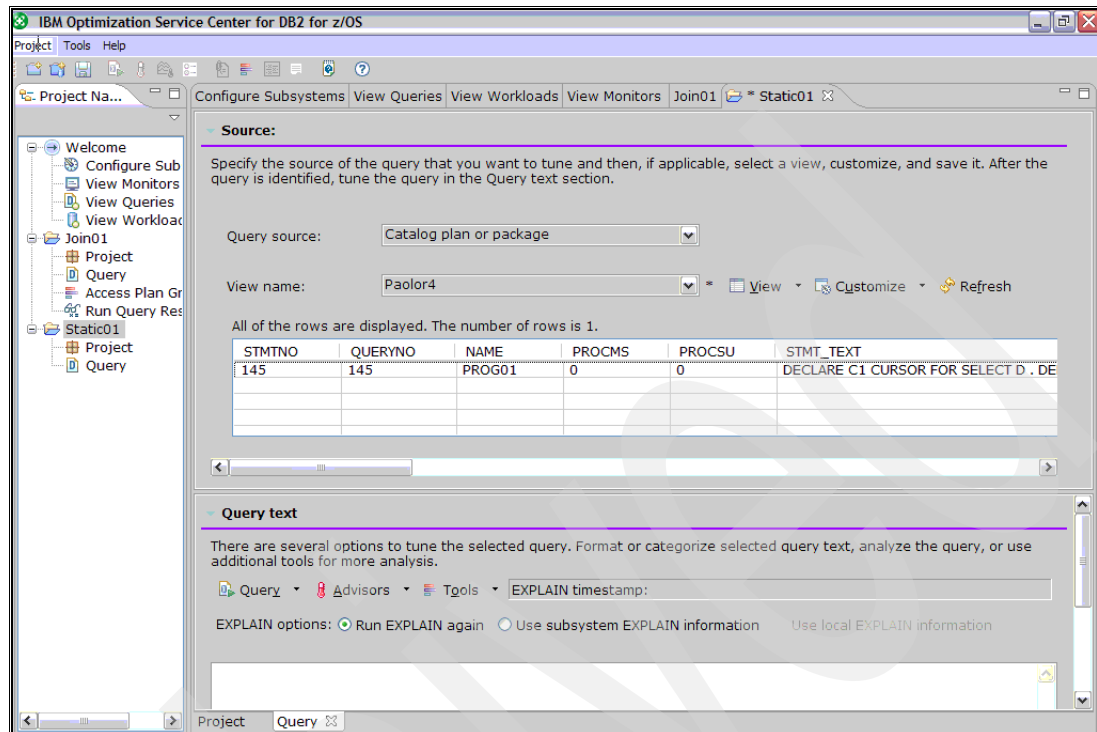


Figure 2-32 The SQL statements retrieved from the DB2 Catalog

In our example, there is only one SQL statement from a single package. You can drag the scroll bar below the list of statements to the right to see more information about the statements. If you click one statement, the text of the statement appears in the Query Text panel, as shown in Figure 2-33.

Now that you have selected a single SQL statement, you can do the same thing as when you typed in a single statement's text (see section 2.3, "Investigating a query by typing the text" on page 38). For example, you can see the Access Plan Graph by clicking **Tools** on the Query text panel and selecting **Access Plan Graph** from the drop-down menu.

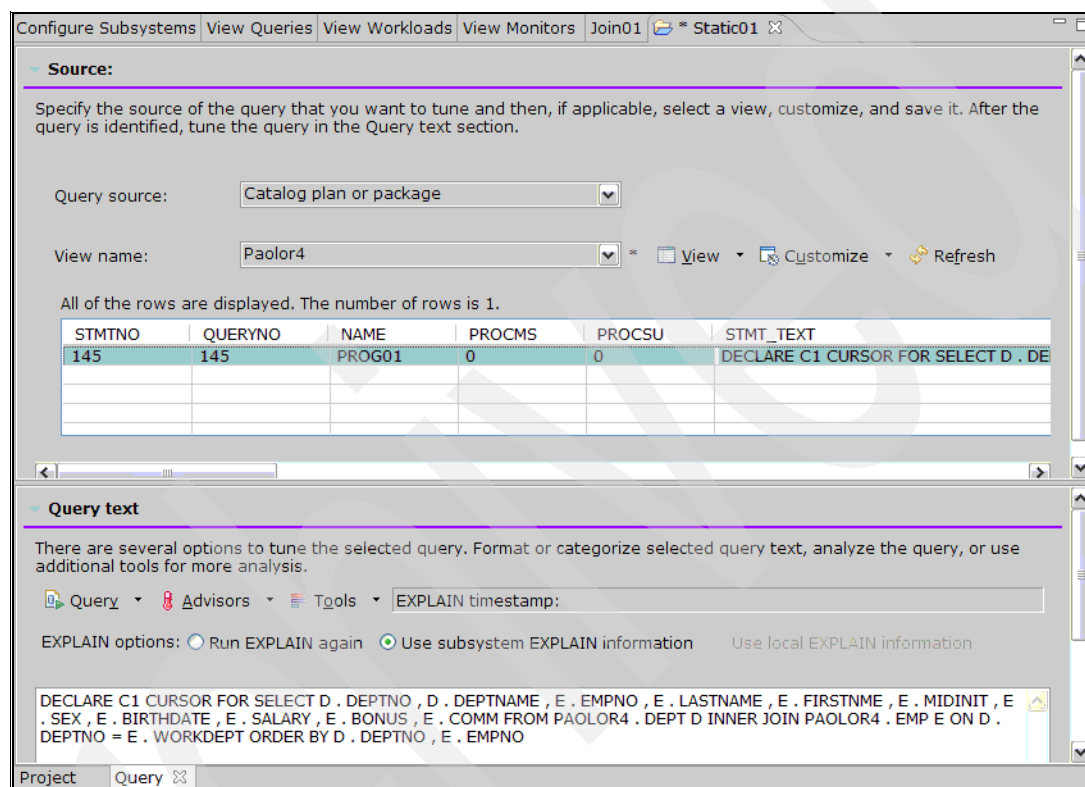


Figure 2-33 Seeing the statement text



This produces a graph as shown in Figure 2-34. This is the same graph as our previous example because the statement in the package is the same as the one you typed in previously.

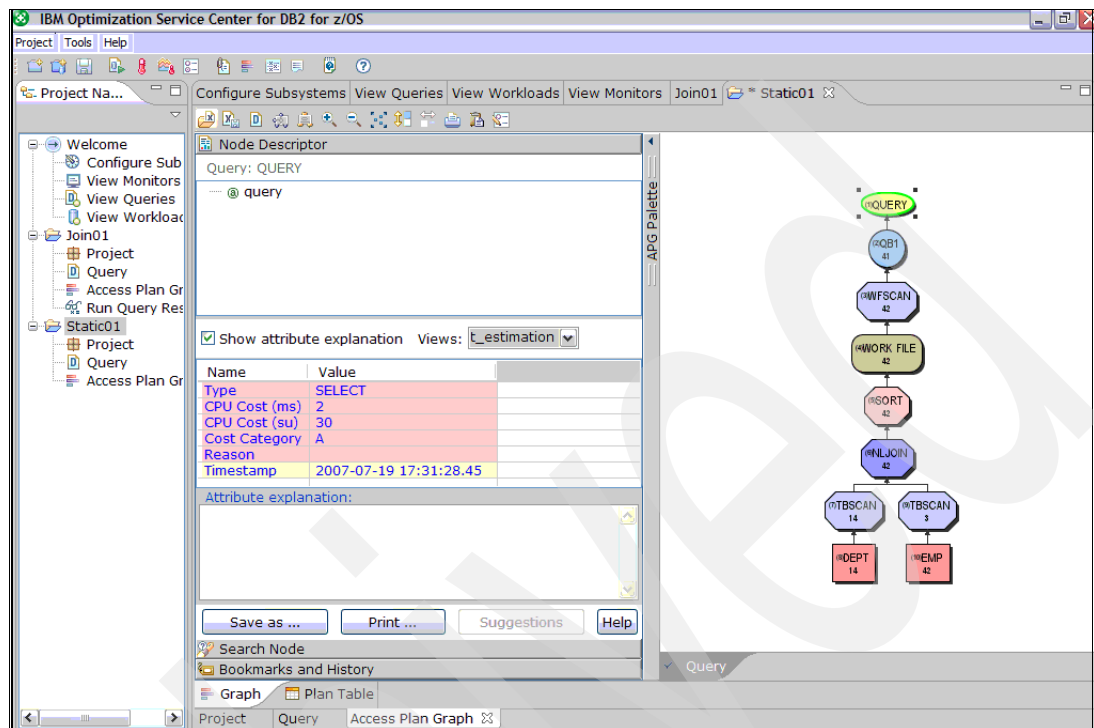


Figure 2-34 The Access Plan Graph

## 2.5 Investigating a query from the dynamic statement cache

Now we will extract an SQL statement from the global dynamic statement cache. We create a new project, called Dynamic1, and click **Identify Target Query**:

1. From the Query source pull-down menu, select the **Statement cache** as shown in Figure 2-35.

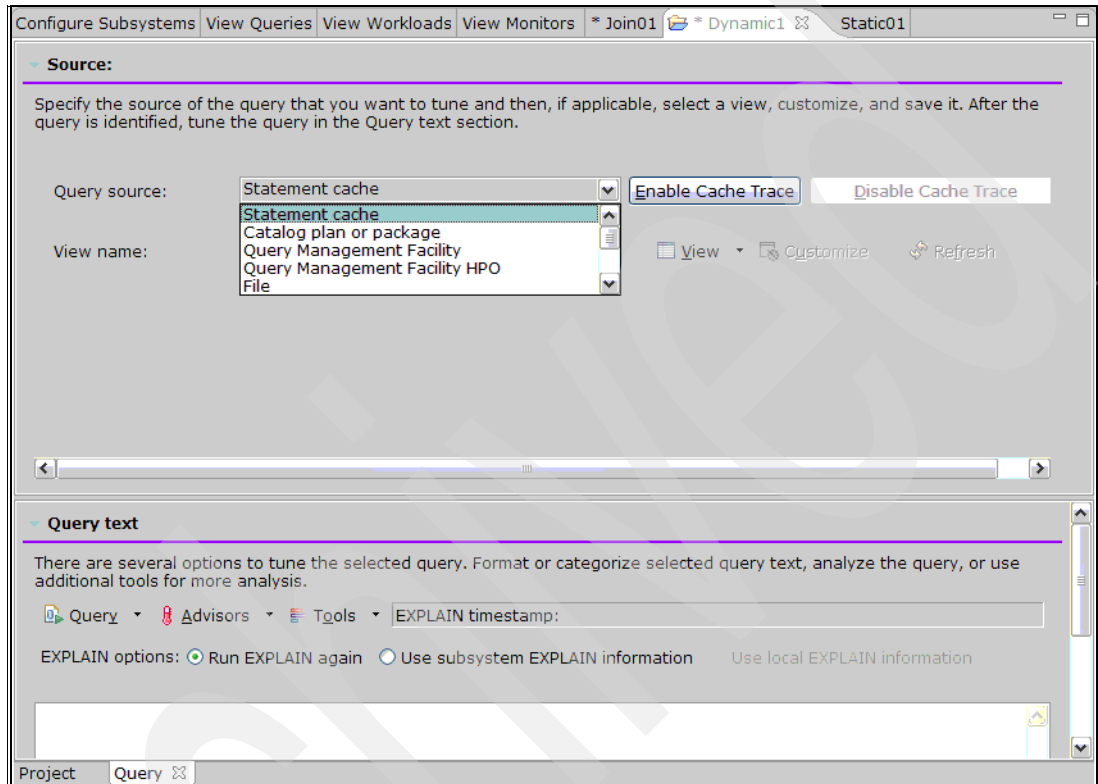


Figure 2-35 Selecting the dynamic statement cache as the source of SQL

2. Select a view as shown in Figure 2-36.

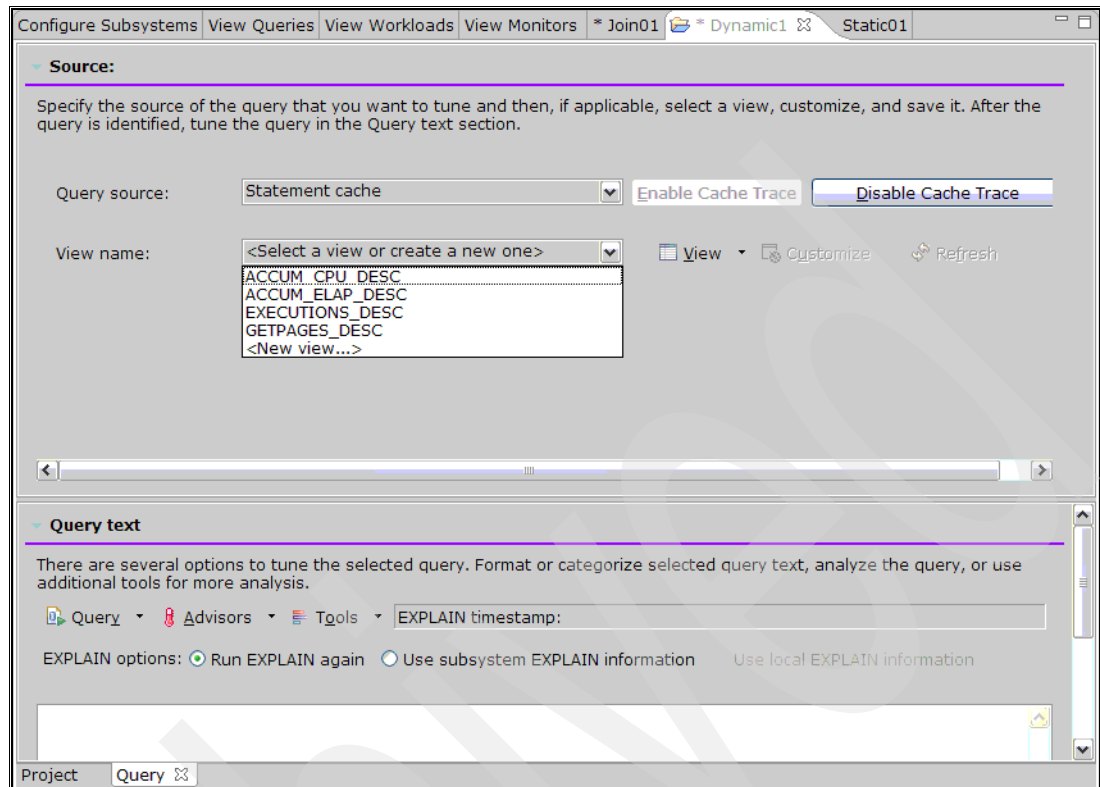


Figure 2-36 Selecting a view of the dynamic statement cache

3. Select the predefined view **ACCUM\_ELAP\_DESC**, which gives you statements from the global dynamic statement cache in descending order of accumulated elapsed time. As soon as you select the view, Optimization Service Center extracts statements from the cache as shown in Figure 2-37. They are listed in the middle of the panel and you can scroll down to see further statements.

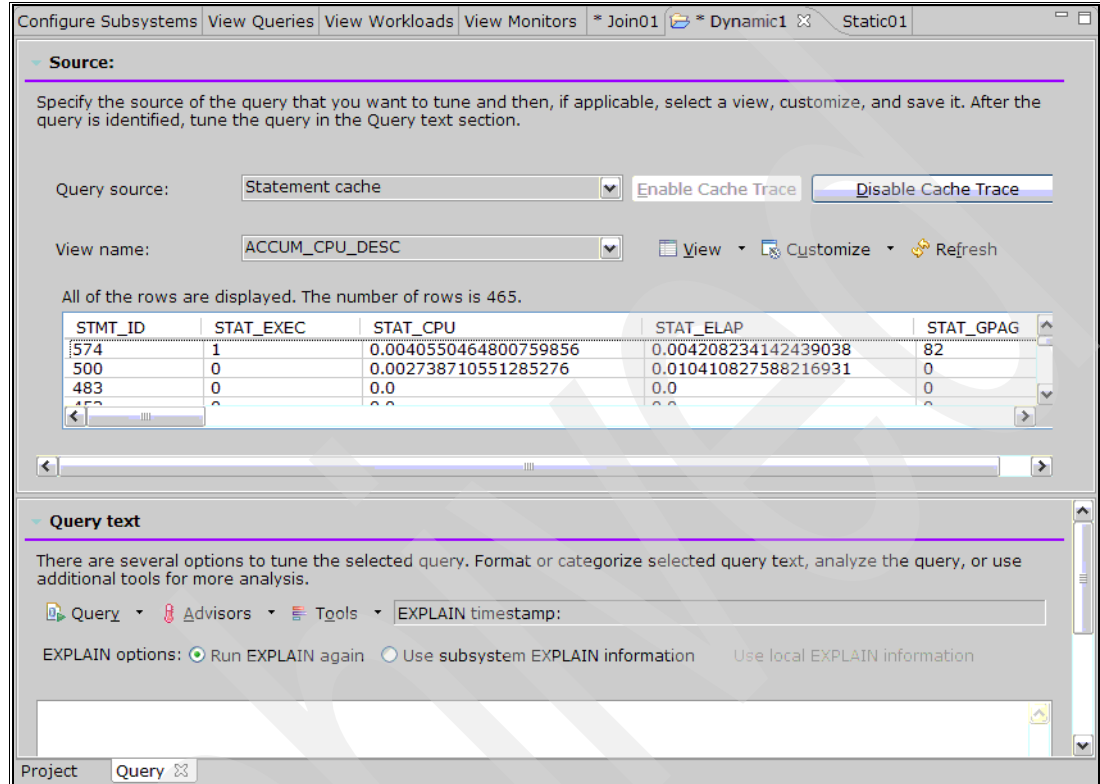


Figure 2-37 Statements returned from the cache

You can restrict the statements returned by clicking **Customize** next to the view name (see Figure 2-38 on page 69). We can specify filtering criteria to restrict the statements returned from the cache, how the statements are to be ordered, and which information is to be listed for each statement.

Notice that the view capability in Optimization Service Center or Optimization Expert is a vast improvement over Visual Explain: you can create a customized view (what columns to present, the sort order, the order the columns are presented in), and you can *save* this view for repeated use. You can design several different views to snap and display the contents of the cache in a customized and re-usable display. Also, after the rows are displayed, you can click the column header to sort the displayed rows in the order of the column selected.

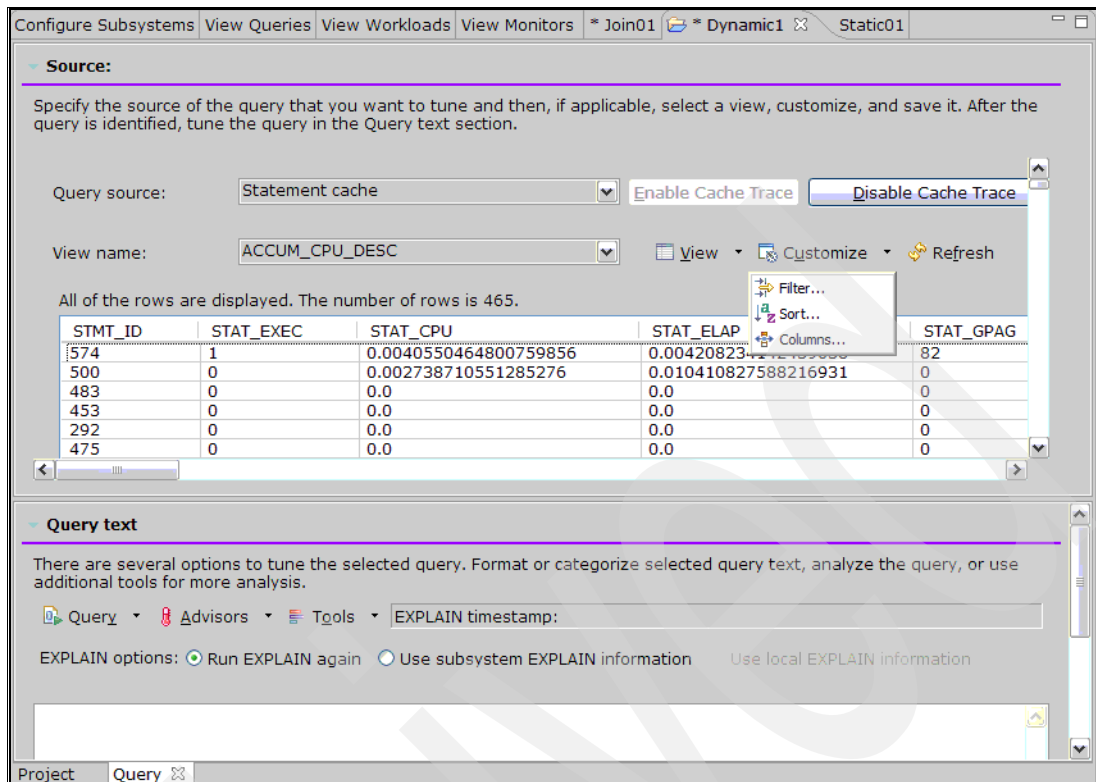


Figure 2-38 Customizing the view

4. Lets restrict the statements to those run under the Authorization ID PAOLOR4. Select the **Filter** option from the Customize drop down menu and specify the Primary Auth ID by double-clicking the **Value** column next to PRIMAUTH and typing the value PAOLOR4 as shown in Figure 2-39.

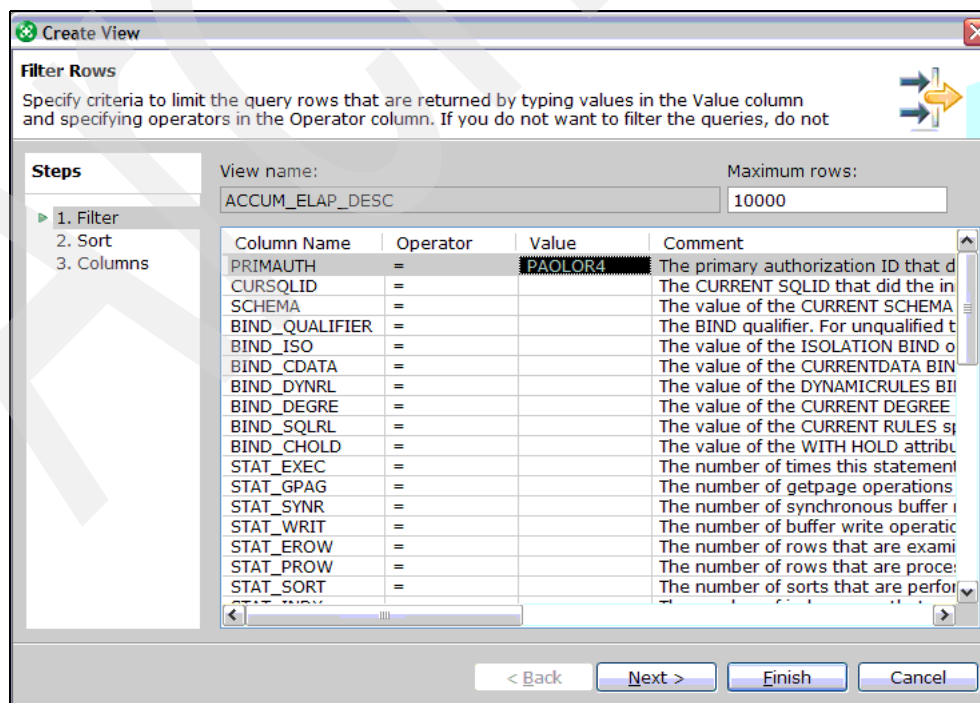


Figure 2-39 Filter on primary Auth ID

- When you click **Finish**, the list of statements is refreshed and now includes only those run by PAOLOR4. If you then select one statement by clicking on it, you see the statement code appear in the bottom Query text panel as shown in Figure 2-40.

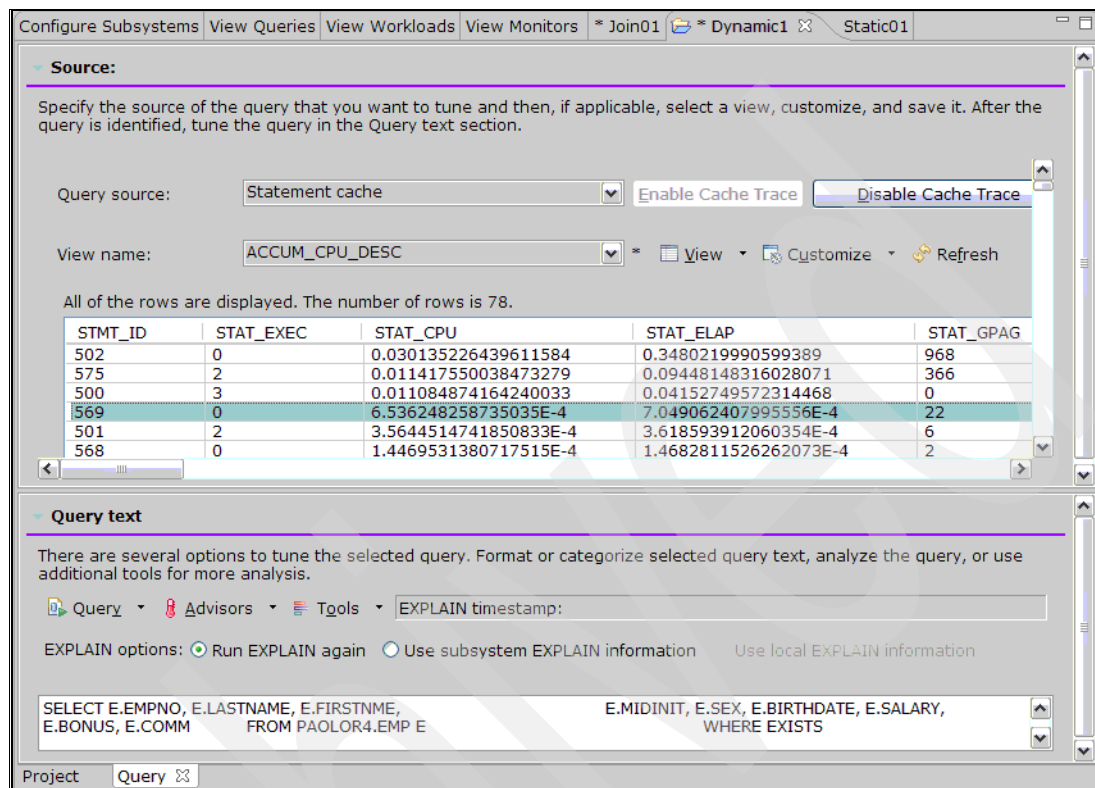


Figure 2-40 Selecting a statement

- From here, you can explain the statement and produce an access plan graph by selecting the **Access Plan Graph** option from the tools drop-down menu, as you have done before for other SQL statements from other sources. The resulting graph is shown in Figure 2-41 on page 71.

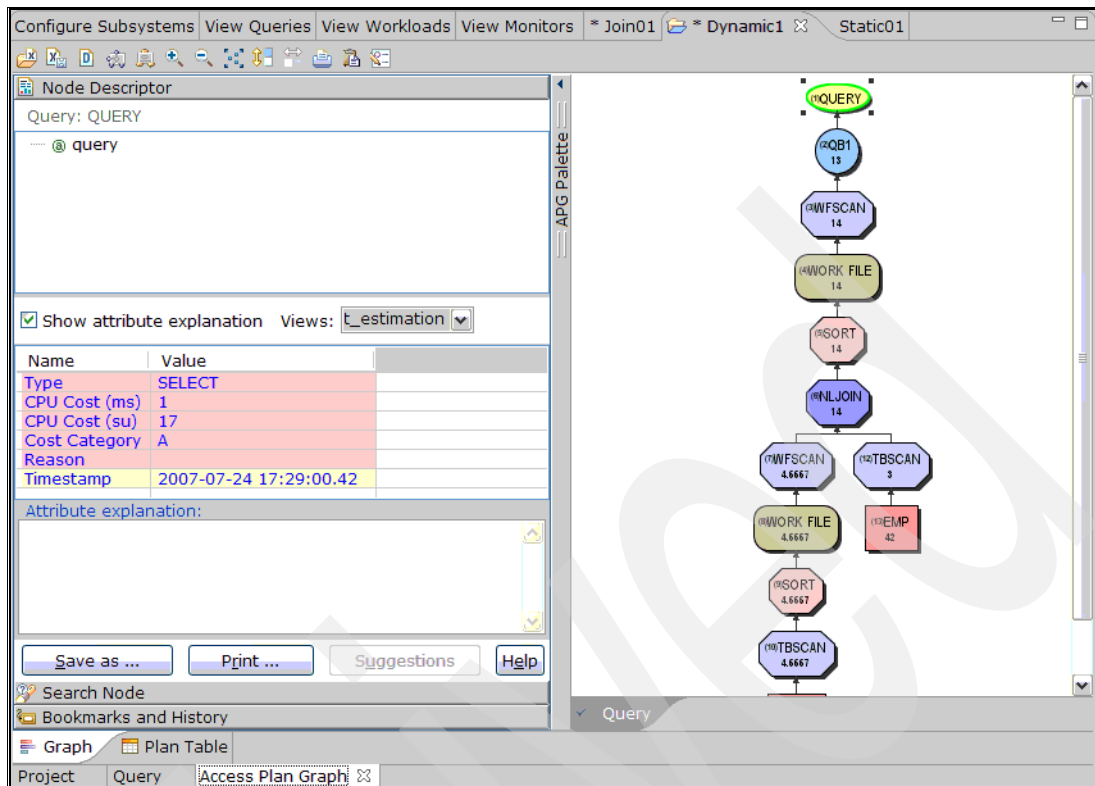


Figure 2-41 Access Plan Graph from dynamic cache statement

## 2.6 Using the Statistics Advisor

You can ask Optimization Service Center to look at the DB2 Catalog statistics in the context of a single SQL statement and advise whether gathering extra statistics might produce a better access path. Return to the Query tab for the static SQL project.

If you click **Advisors**, you see a drop down menu as shown in Figure 2-42.

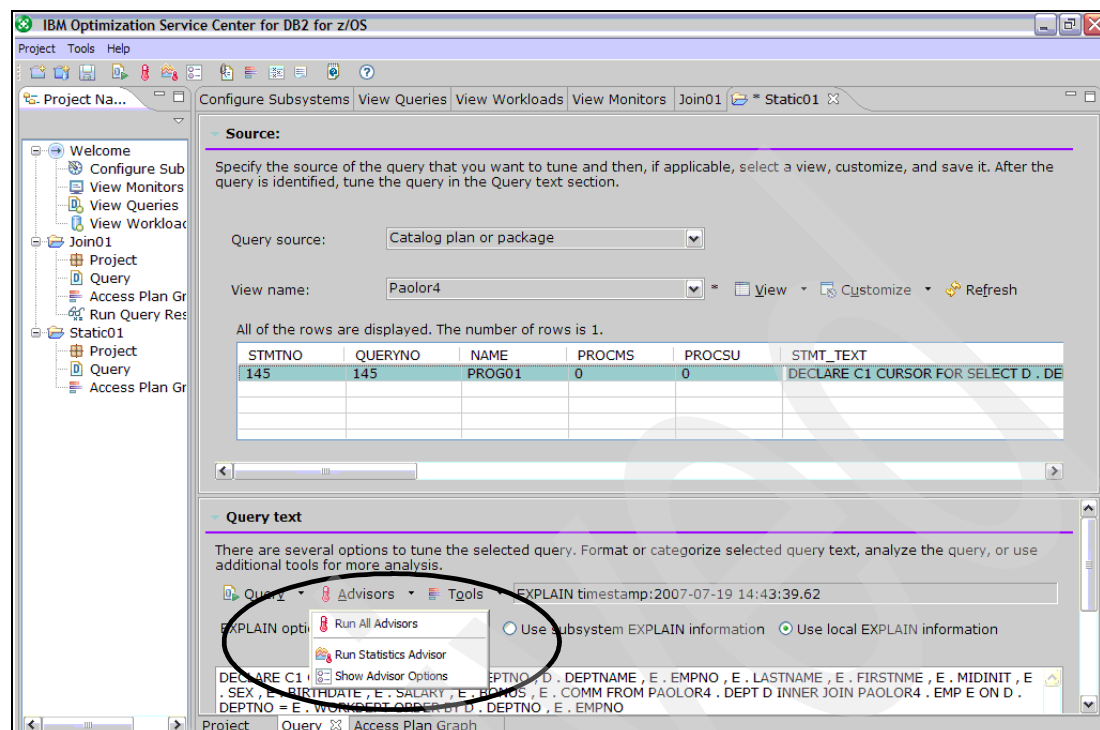


Figure 2-42 Selecting an Advisor

There is only one Advisor supplied with Optimization Service Center, the Statistics Advisor. Other advisors are provided as part of the Optimization Expert product.



To run the Statistics Advisor:

1. Select **Run Statistics Advisor** from the drop-down menu. A pop-up panel asks how the explain information is obtained, as shown in Figure 2-43.

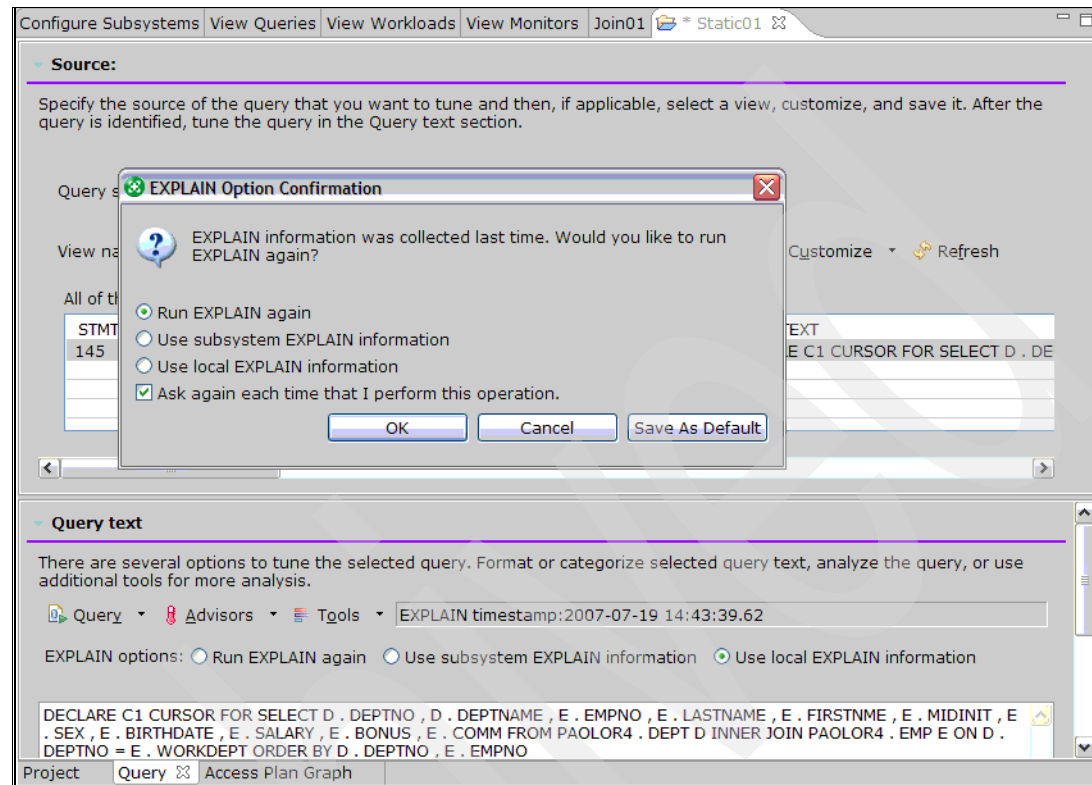


Figure 2-43 Statistics Advisor asks for source of explain data

2. Accept the default option to run Explain again. The Statistics Advisor runs Explain and then makes its recommendations, as shown in Figure 2-44.

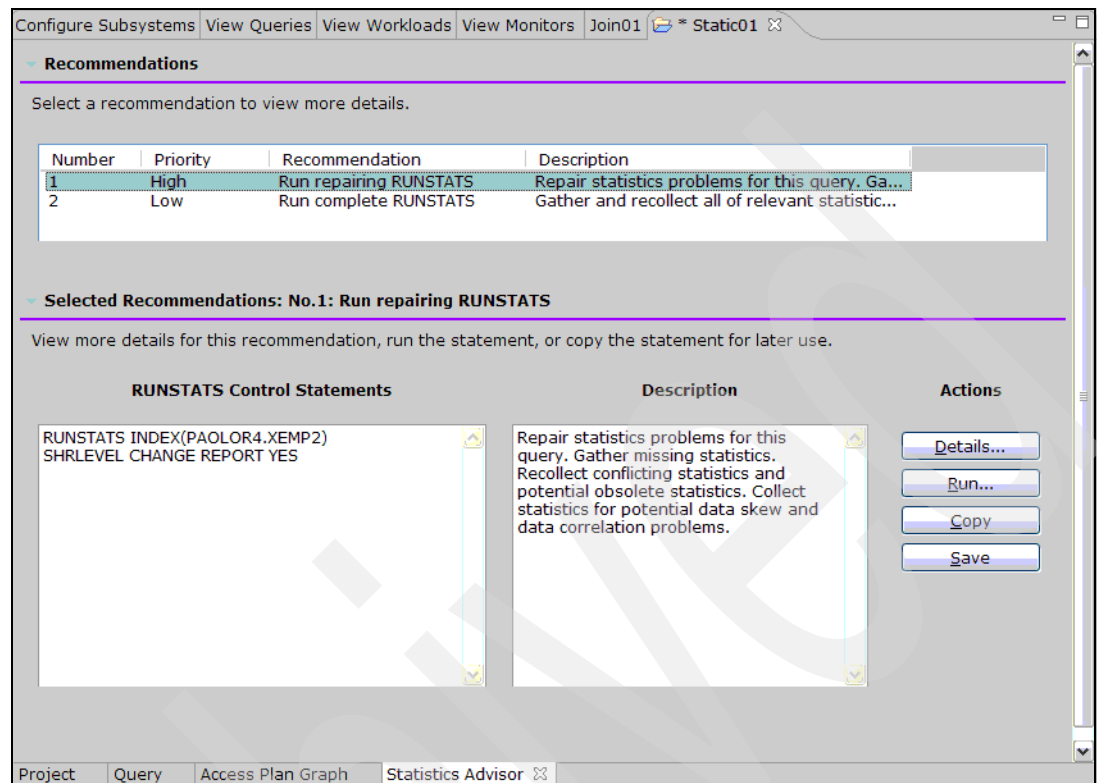


Figure 2-44 Statistics Advisor recommendations

The Statistics Advisor makes two recommendations:

- a. The first (number 1) is a high priority recommendation to run the Runstats utility against an index PAOLOR4.XEMP2. The description tells you that the index might have missing statistics. If you click **Details** on the right side and then scroll down the top panel, you can see that statistics have never been gathered for the index as shown in Figure 2-45 on page 75. This index was created after the statistics were gathered.

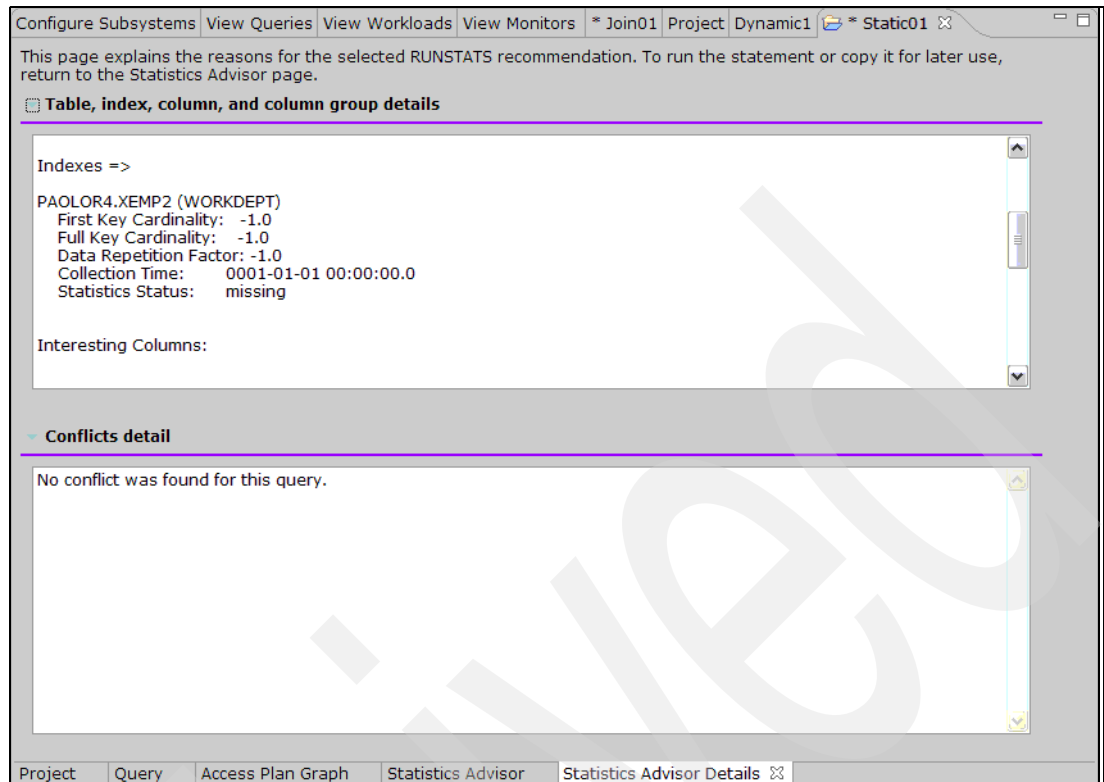


Figure 2-45 Statistics Advisor details

- b. If you return to the Statistics Advisor tab and highlight the second low priority recommendation, you see that it recommends the statistics to be gathered on a regular basis (see Figure 2-46 on page 76).

Statistics Advisor has given us two recommendations for the SQL statement: the first is a high priority recommendation to fix an immediate problem, and the second is a long term recommendation identifying statistics to be gathered on a regular basis. Collecting the complete statistics ensures that the statistics for the index are consistent with the table.

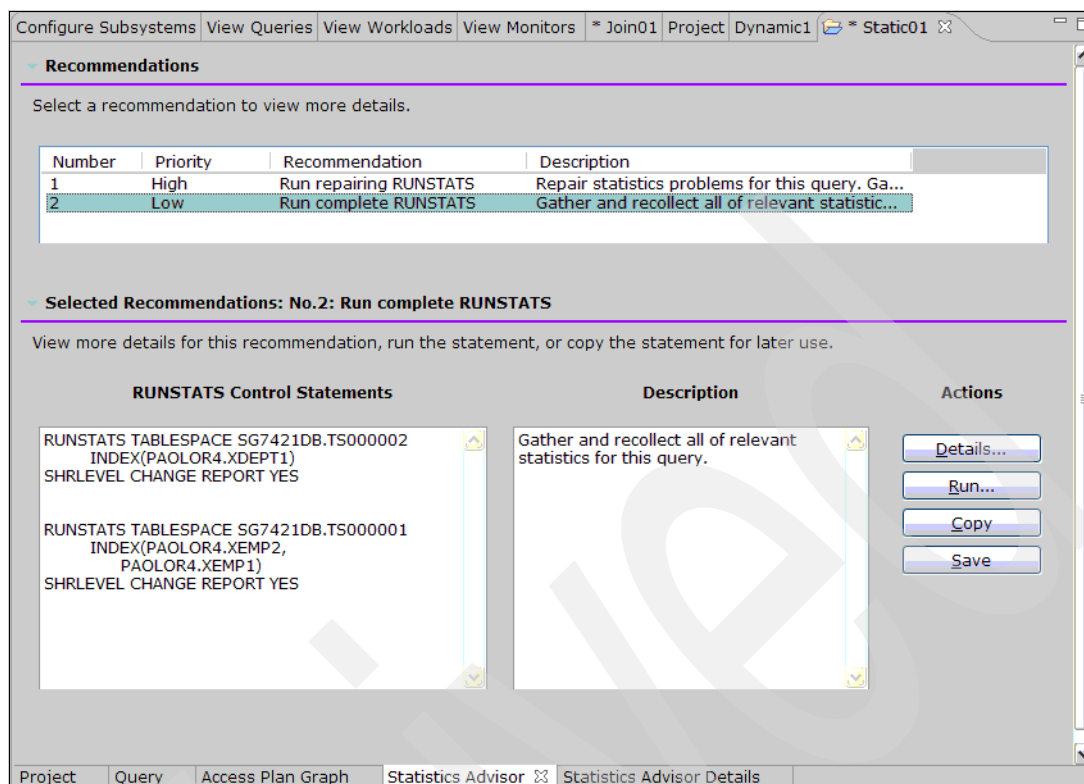


Figure 2-46 Statistics Advisor second recommendation

3. Select the first recommendation again. This shows the Runstats to be run immediately. If we click **Run** at the right, Optimization Service Center runs the Runstats. On completion, you are asked if you wish to rerun the Statistics Advisor (Figure 2-47).

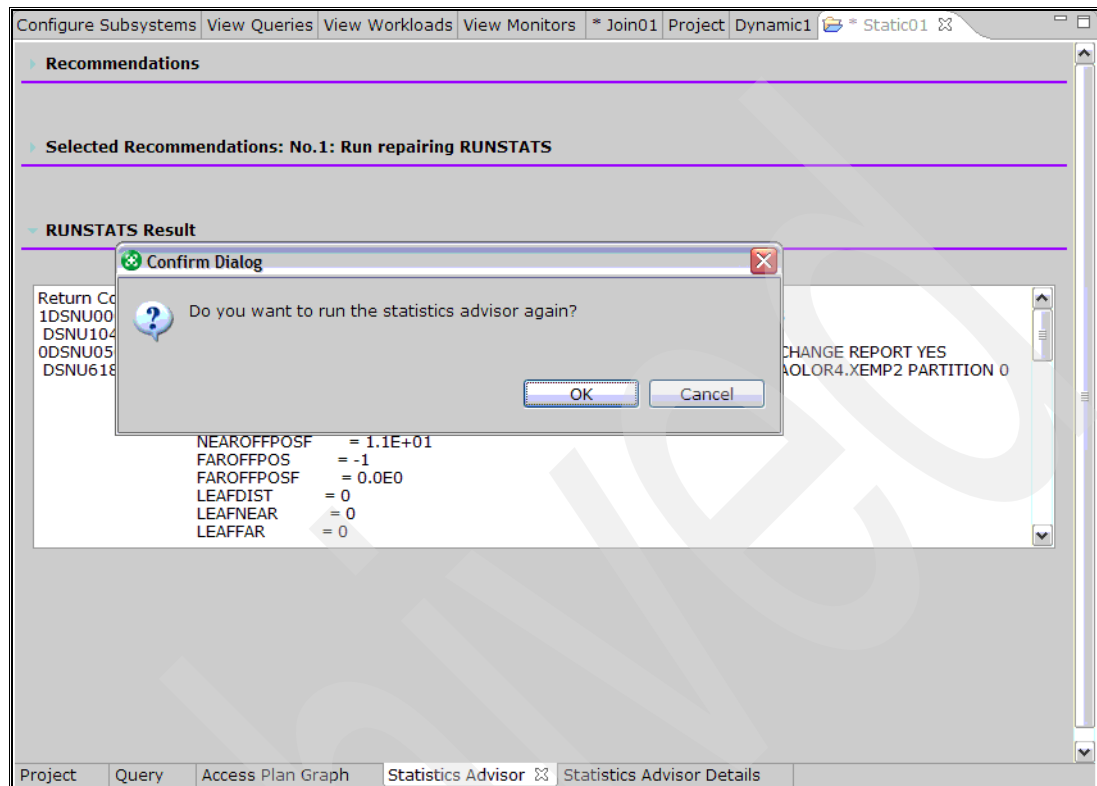


Figure 2-47 Completion of Runstats

4. Rerun the Statistics Advisor. The result is shown in Figure 2-48 on page 78. You now have no high priority recommendation.

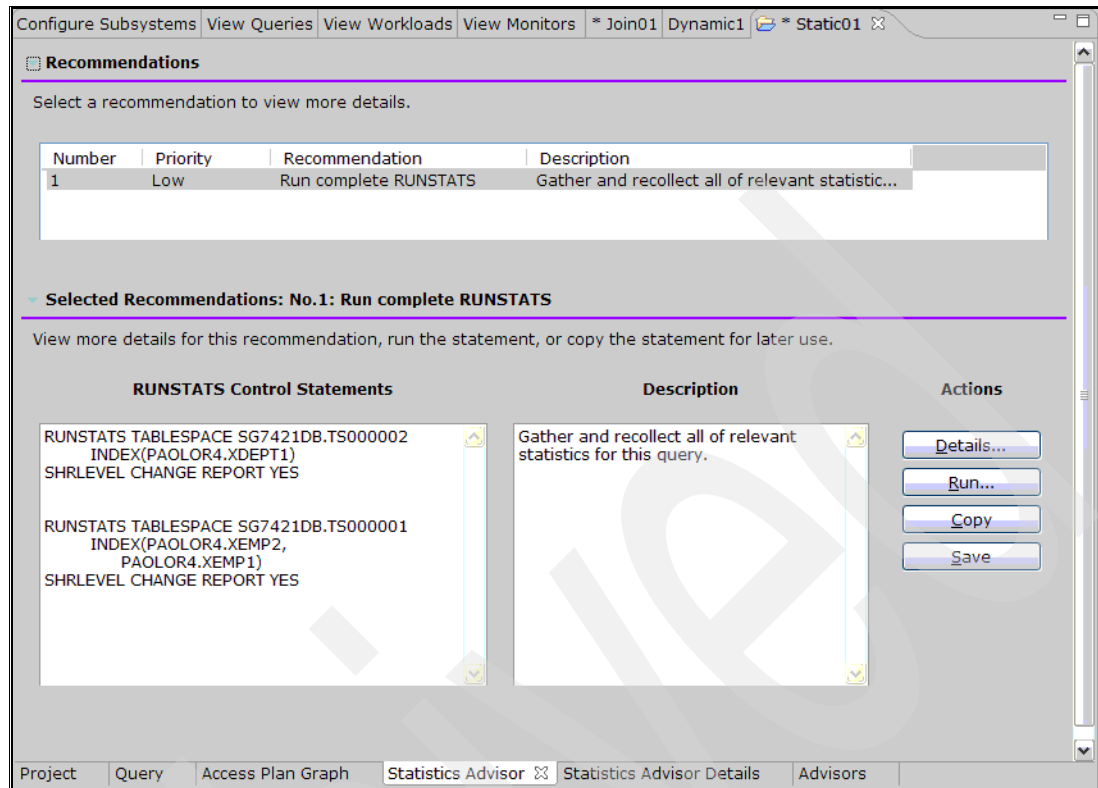


Figure 2-48 Result of rerunning the Statistics Advisor

You can recreate the Access Plan Graph to see if the new statistics have changed the access path.

The Statistics Advisor can also make recommendations based on a set of SQL statements rather than a single SQL statement. You can extract the set of SQL statements from a file, the DB2 catalog, the global dynamic statement cache, or a number of other sources. This is discussed in Chapter 9, "Statistics Advisor" on page 223.



## Part 2

# Installation and customization

This part describes the install process of both tools, the implications of using them in an existing DB2 environment, and provides recommendations on managing these new objects. This part contains the following chapters:

- ▶ Chapter 3, “Installing Optimization Service Center” on page 81
- ▶ Chapter 4, “Installing Optimization Expert” on page 95

Archived



# Installing Optimization Service Center

This chapter discusses how to install BM DB2 Optimization Service Center for DB2 for z/OS. You can install Optimization Service Center or Optimization Expert on the same workstation because they coexist.

You can install Optimization Service Center on a Windows® workstation, create a connection to DB2, and perform tuning tasks to improve the performance of SQL queries and workloads that run on that DB2.

This chapter contains the following topics:

- ▶ “Data server and DB2 requirements” on page 82
- ▶ “Workstation requirements” on page 86
- ▶ “Installing Optimization Service Center on a workstation” on page 86
- ▶ “Step-by-step install instructions” on page 88
- ▶ “Performing a silent install” on page 92

## 3.1 Data server and DB2 requirements

Before you install Optimization Service Center, ensure that each data server and DB2 system satisfy the hardware and software requirements.

For complete and detailed information about requirements for enabling a data server and DB2 system to work with Optimization Service Center, see *Program Directory for DB2 Accessories Suite for z/OS*, GI10-8749.

Ensure that your data server and DB2 system meet the following prerequisites:

- ▶ Use DB2 Version 9.1 for z/OS.
- ▶ Enable Distributed Data Facility (DDF).
- ▶ Run two jobs, which are delivered with DB2 Version 9.1 for z/OS, to enable Optimization Service Center. The sample JCL jobs DSNTIJOS and DSNTIJRA are provided in the SDSNSAMP target library.

You can use the DSNTIJOS job to complete the following tasks on the DB2 system:

- Create objects that are used specifically by Optimization Service Center, including tables, table spaces, and indexes.
- Bind Optimization Service Center packages.
- Create stored procedures that are used by Optimization Service Center.
- Create aliases that are used by Optimization Service Center.
- Grant the required execute privileges to packages and stored procedures.

**Note:** DSNTIJOS requires that buffer pools BP8K1 and BP16K1 have a VPSIZE > 0 before submitting.

You can use the DSNTIJRA job to create a RACF® group that is required by Optimization Service Center.

**Note:** If Optimization Expert was installed prior to Optimization Service Center, jobs DSNTIJOS and DSNTIJRA are not required to run.

To use the Optimization Service Center monitoring function, we recommend the Administrative Scheduler. You can use a client side scheduling to run the monitor, but that requires an Optimization Service Center or Optimization Expert workstation to be running and connected to DB2 all the time. The Administrative Scheduler is a separate started task brought up by DB2. However, DB2 shutdown will not close the separate started task. A DB2 crash will not affect the Administrative Scheduler active task. The MODIFY command is required for all Administrative Scheduler started task changes. For more information about the Administrative Scheduler, see Appendix B, “Administrative Scheduler” on page 405 and the following resources:

- *DB2 Version 9.1 for z/OS Administration Guide*, SC18-9840, Chapter 14, “Scheduling administrative tasks”
- *DB2 Version 9.1 for z/OS Command Reference*, SC18-9844:
  - Chapter 43, “MODIFY admtproc,APPL=SHUTDOWN”
  - Chapter 44, “MODIFY admtproc,APPL=TRACE”
  - Chapter 72, “START admtproc”
  - Chapter 79, “STOP admtproc (z/OS)”

- A WLM environment is set up for running stored procedures.
- The DB2-supplied stored procedures shown in Table 3-1 are installed and enabled.

**Note:** As seen in the “Job that installs it” column, Optimization Service Center requires that you run job DSNTJXP to create stored procedure DSN8.DSN8EXP. Run job DSNTIJCC to create stored procedure SYSPROC.DSNACCMD.

Table 3-1 Stored procedures required by Optimization Service Center

Stored procedure name	What the procedure does	Job that installs it
SYSPROC.DSNWZP	Returns the current settings of the system parameters and allows you to view them in the Optimization Service Center client.	DSNTIJSG
SYSPROC.DSNUTILU	Enables you to invoke utilities, such as the RUNSTATS utility directly from the Optimization Service Center client.	DSNTIJSG
DSN8.DSN8EXP	Invokes the EXPLAIN statement, allowing you to run EXPLAIN from Optimization Service Center.	DSNTEJXP
SYSPROC.OSC_RUNSQL	Uses a stored procedure to run EXPLAIN for the statement cache with SYSADM authority and captures workloads from the dynamic statement cache.	DSNTIJOS
SYSPROC.OSC_EXECUTE_TASK	Enables you to capture, run EXPLAIN for, and transform workloads and monitor profiles from Optimization Service Center.	DSNTIJOS
SYSPROC.ADMIN_COMMAND_DB2	Stored procedure executes one or more DB2 commands on a connected DB2 system.	DSNTIJSG
SYSPROC.ADMIN_TASK_SCHEDULE	Schedules and updates workload control center tasks in the Administrative Scheduler.	DSNTIJSG
SYSPROC.ADMIN_TASK_REMOVE	Removes workload control center tasks from the Administrative Scheduler.	DSNTIJSG
SYSPROC.DSNACCMD	Executes DB2 commands (for instance, to enable cache trace).	DSNTIJCC

- Java environment:

For Optimization Service Center to use a required Java Stored Procedure, you must enable the Java environment on the data server. For more information about how to set up the Java environment, see *DB2 Version 9.1 for z/OS Application Programming Guide and Reference for JAVA*, SC18-9842, “Prerequisites and setup” in *DB2 for z/OS and OS/390: Ready for Java*, SG24-6435, and *DB2 for z/OS Stored Procedures: Through the CALL and Beyond*, SG24-7083.

The Java stored procedures, corresponding global tables, and packages (DSNTIJMS) are listed in Table 3-2 on page 84.

Table 3-2 DB2 objects for Optimization Service Center Java procedures

Procedure	Global tables	Packages
SYSIBM.SQLCOLPRIVILEGES	SYSIBM.SQTCOLPRIVILEGES	DSNAPCC.DSNACPR8
SYSIBM.SQLCOLUMNS	SYSIBM.SQTCOLUMNS	DSNAPCC.DSNACOL8
SYSIBM.SQFOREIGNKEYS	SYSIBM.SQTFORIGNKEYS	DSNAPCC.DSNAFNK8
SYSIBM.SQPRIMARYKEYS	SYSIBM.SQTPRIMARYKEYS	DSNAPCC.DSNAPRK8
SYSIBM.SQPROCEDURECOLS	SYSIBM.SQTSPECIALCOLUMNS	DSNAPCC.DSNAPCO8
SYSIBM.SQPROCEDURES	SYSIBM.UINDEXES	DSNAPCC.DSNAPRC8
SYSIBM.SQLSPECIALCOLUMNS	SYSIBM.SQTPROCEDURECOLS	DSNAPCC.DSNAPC8
SYSIBM.SQLSTATISTICS	SYSIBM.SQTPROCEDURES	DSNAPCC.DSNASTA8
SYSIBM.SQTABLEPRIVILEGES	SYSIBM.SQTSTATISTICS	DSNAPCC.DSNATBP8
SYSIBM.SQTABLES	SYSIBM.SQTTABLEPRIVILEGES	DSNAPCC.DSNATBL8
SYSIBM.SQLGETTYPEINFO	SYSIBM.SQTTABLES	DSNAPCC.DSNATYP8
SYSIBM.SQLUDTS	SYSIBM.SQTABLETYPES	DSNAPCC.DSNAUDT8
SYSIBM.SQLCAMESSAGE	SYSIBM.SQLTYPEINFO	
	SYSIBM.SQTUDTS	

– JDK™ v1.4:

Ensure that JDK version 1.4 is installed under the folder in the UNIX® Service System that is specified by the JAVA\_HOME environment variable.

– JDBC/SQLJ driver for z/OS:

Ensure that the driver file (db2jcc.jar) is installed under the folder in the UNIX Service System that is specified by CLASSPATH.

– JCC properties file:

Ensure that the SQLJ/JDBC™ runtime properties file is set up correctly. It contains settings for the JDBC/SQLJ Driver for OS/390, and is specified by the DB2SQLJPROPERTIES environment variable.

DB2SQLJSSID: Specifies the name of the system to which a JDBC or an SQLJ application connects.

DB2SQLJPLANNAME: Specifies the name of the plan that is associated with a JDBC or an SQLJ application. The plan is created by the bind process.

For example:

```
# Any lines starting with the pound sign '#'
# are comments. Please see the DB2 for z/OS
# Application Programming Guide and Reference
# for Java for the description of these settings.
#
DB2SQLJSSID=V91C
DB2SQLJPLANNAME=DSNJDBC
DB2SQLJ_TRACE_FILENAME=/tmp/mytrc
DB2SQLJ_TRACE_BUFFERSIZE=1024
```

**Note:** UNIX Service System files and commands are case-sensitive.

– WLM-environment for Java stored procedures

Ensure that the NUMTCB parameter of the WLM application environment for Java is not greater than 5.

Recommendation: Use a value of 1 to ensure that sufficient memory is available.

- Started-task job for Java stored procedures:

Ensure that a started-task job for Java stored procedures exists in the system PROCLIB. For example, the started task JCL for V91CWLMJ in SYS1.PROCLIB is shown in Example 3-1.

*Example 3-1 V91CWLMJ started task*

---

```
//V91CWLMJ PROC DB2SSN=V91C,NUMTCB=1,APPLENV=V91CWLMJ
//JPROC EXEC PGM=DSNX9WLM,TIME=1440,
// PARM='&db2ssn,&NUMTCB,&APPLENV',
// REGION=0M
//STEPLIB DD DISP=SHR,DSN=V91C.SDSNEXIT
// DD DISP=SHR,DSN=DSN910.NEW.SDSNLOAD
// DD DISP=SHR,DSN=DSN910.NEW.SDSNLOD2
//JAVAENV DD DISP=SHR,DSN=V91C.JAVAENV
//JAVAOUT DD PATH='/V1R7/USR/db2910/JAVAOUT.TXT',
// PATHOPTS=(ORDWR,OCREAT,OAPPEND),
// PATHMODE=(SIRUSR,SIWUSR,SIRGRP,SIWGRP,SIROTH,SIWOTH)
//JAVAERR DD PATH='/V1R7/USR/db2910/JAVAERR.TXT',
// PATHOPTS=(ORDWR,OCREAT,OAPPEND),
// PATHMODE=(SIRUSR,SIWUSR,SIRGRP,SIWGRP,SIROTH,SIWOTH)
//JSPDEBUG DD SYSOUT=*
//SYSTSPRT DD SYSOUT=*
//CEEDUMP DD SYSOUT=*
//SYSPRINT DD SYSOUT=*
//OUT1 DD SYSOUT=*
//SYSABEND DD DUMMY
//DSNTRACE DD SYSOUT=*
```

---

- JDBC/SQLJ environment variables:

Ensure that JDBC/SQLJ environment variables are created under the UNIX Service System, such as V91C.JAVAENV in the preceding example job. In the sample below, dsnwccsp.jar is the Java stored procedure JAR file that was shipped as part of DB2 for z/OS. JCC\_HOME is the home directory of the JCC driver, and JAVA\_HOME is the home directory of JRE™. You might need to change JCC\_HOME and JAVA\_HOME, depending on where your JCC driver and JRE are installed.

```
ENVAR("CLASSPATH=/usr/lpp/db2910_base/lib/IBM/dsnwccsp.jar",
"TZ=PST08",
"JCC_HOME=/usr/lpp/db2/devbase",
"JAVA_HOME=/usr/lpp/java140/J1.4"),
MSGFILE(JSPDEBUG,,,,ENQ),
XPLINK(ON)
```

**Note:** You must include all the environment variables (ENVAR) in this file. Ensure that the total length of all the entries put together, excluding blanks, does not exceed 245 bytes. Exceeding 245 bytes for ENVAR produces unpredictable results and errors. For entries that exceed the 245 byte limit, place most of the ENVAR entries into an HFS file, then point ENVAR to that file. The HFS file has no limitation size. For more information, see section 17.2.6, “Setting up the JAVAENV data set for Java stored procedure execution,” *DB2 for z/OS Stored Procedures: Through the CALL and Beyond*, SG24-7083.

## 3.2 Workstation requirements

Ensure that your workstation meets the following hardware requirements:

- ▶ 500 MB of space on your hard disk drive
- ▶ 1 GB of memory
- ▶ A Pentium® IV processor equivalent or above

Recommendation: For optimal performance, use faster processors with more memory.

Ensure that you have the following software installed on your workstation:

- ▶ One of the following Microsoft® Windows operating systems:
  - Windows 2000
  - Windows XP
  - Windows 2003
- ▶ Microsoft Internet Explorer® Version 5.5 or later
- ▶ Adobe® SVG Viewer 3.0. Optimization Service Center prompts you to download and install this component when you first access the visual plan hints function. You must download the SVG Viewer from the Adobe Web site.

**Note:** According to the Adobe Web site, Adobe will discontinue support for the SVG Viewer as of January 1, 2008. Optimization Service Center or Optimization Expert developers are working on a solution.

- ▶ IBM DB2 Connect Application Server, Enterprise, Personal, or Unlimited Edition at the Version 9.1 Fix Pack 1, 8.2 Fix Pack 6, or 8.1 Fix Pack 13 level, or later.

**Note:** You can use a limited-use license for DB2 Connect Personal Edition Version 9, which is included with IBM DB2 Version 9.1 for z/OS, for connections from your client workstation (where Optimization Service Center resides) to DB2. Customers requiring additional licenses need to inquire about an upgraded version of DB2 Connect.

The configuration with DB2 Connect residing in a middle layer is currently not supported.

- ▶ An IP network connection to DB2.

Recommendation: For faster access, do not use a dial-up connection. Use a broadband network connection.

## 3.3 Installing Optimization Service Center on a workstation

You can install Optimization Service Center on a workstation based on three options from:

- ▶ The product CD
- ▶ SMP/E
- ▶ The Web download

**Note:** Optimization Service Center is a no-cost tool. Optimization Expert is a purchased tool.

Before installing Optimization Expert, ensure that your workstation meets the hardware and software requirements listed in 3.2, “Workstation requirements” on page 86.

Changes to DB2 might be required before you can install and use Optimization Service Center. Such changes include enabling DDF and setting up a WLM environment. After you confirm that DB2 is configured properly, you can install Optimization Service Center on your workstation. You must have administrator privileges on the workstation before you can install Optimization Service Center.

Two versions of Optimization Service Center were available while writing this book, the GA version and a fix pack version. Installing either version is acceptable. However, the fix pack version was preferable because of the fixes available. For this version of Optimization Service Center, you can either install the GA version, wait until it completes successfully, and then install the fix pack version, or just install the fix pack version, which is a full code replacement for the GA version.

From an install point of view, the GA and fix pack version were basically the same. If documentation does not exist for your install level to state whether to install the GA version prior to a fix pack version, contact support prior to the install to verify. In our situation, installing the fix pack version of Optimization Service Center without the GA version was fully acceptable. This was not true for Optimization Expert, which had the same scenario. However, the GA version was required to be installed prior to the fix pack version. The GA version of Optimization Expert is required because of licensing requirements, which do not exist for Optimization Service Center.

**Note:** You can install Optimization Service Center through the mainframe or workstation environment. If the install uses the workstation approach, the Workload Control Center (WCC) stored procedures are not created. In this case, there are drawbacks to monitoring.

For customers requiring strong control over their environment, we strongly recommend the mainframe approach using SMP/E.

### 3.3.1 Installing Optimization Service Center from the product CD

If your workstation has a CD-ROM drive, you can install Optimization Service Center from the product CD:

1. Insert the Optimization Service Center product CD into your workstation's CD drive.
2. Navigate to the Windows location of your CD drive to view the contents of the product CD.
3. Double-click **setup.exe** and follow the instructions to complete the installation wizard. Upon successful completion, Optimization Service Center is installed.
4. To launch Optimization Service Center from your Windows desktop, click **Start → All Programs → IBM Optimization Service Center for DB2 for z/OS → Optimization Service Center**. Optimization Service Center opens, and the Welcome page displays.

### 3.3.2 Installing Optimization Service Center from SMP/E

You can use SMP/E to transfer the Optimization Service Center installation package to a workstation from the z/OS Hierarchical File System (HFS):

1. To receive, apply, and accept the installation files to HFS using SMP/E, follow the instructions in the *Program Directory for IBM DB2 Accessories Suite for z/OS*, GI10-8749.

**Note:** Review all SMP/E hold data before proceeding.

2. Transfer the installation package from the /usr/lpp/osc/IBM/ HFS directory to your workstation.
3. On your workstation, locate the installation files, double-click **setup.exe**, and follow the instructions to complete the installation wizard. Upon successful completion, Optimization Service Center is installed.
4. To launch Optimization Service Center from your Windows desktop, click **Start → All Programs → IBM Optimization Service Center for DB2 for z/OS → Optimization Service Center**. Optimization Service Center opens and the Welcome page displays.

### 3.3.3 Installing Optimization Service Center from the Web download

Optimization Service Center is available from the World Wide Web as a no-charge, downloadable feature of the DB2 Accessories Suite for z/OS:

1. Go to the Web site at: <http://www.ibm.com/software/data/db2/zos/downloads/osc.html>
2. Click **Download Optimization Service Center for DB2 for z/OS**.
3. You are redirected to the IBM support Web site. After signing in, choose which version of Optimization Service Center you want to download.
4. Unzip the files in the compressed folder to a temporary directory.
5. Locate the extracted files, double-click **setup.exe**, and follow the instructions to complete the installation wizard. Upon successful completion, Optimization Service Center is installed.
6. To launch Optimization Service Center from your Windows desktop, click **Start → All Programs → IBM Optimization Service Center for DB2 for z/OS → Optimization Service Center**. Optimization Service Center opens, and the Welcome page displays.

## 3.4 Step-by-step install instructions

Perform the following install instructions:

1. After double-clicking **setup.exe**, the installation wizard begins. The first prompted window is the welcome window, as shown in Figure 3-1 on page 89.



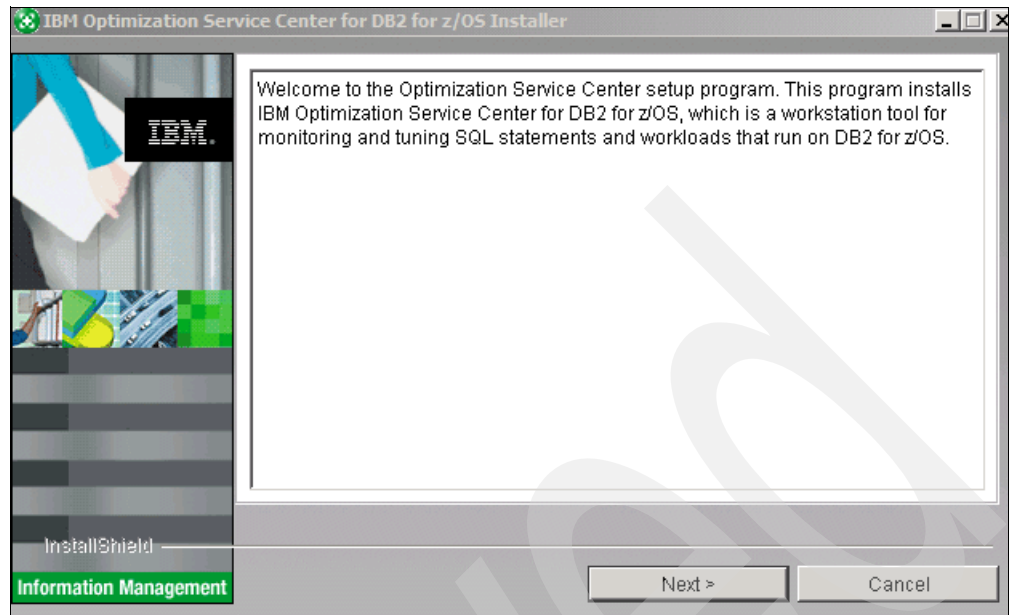


Figure 3-1 Optimization Service Center Install - Welcome window

2. The next window verifies the licensing agreement. Keep in mind that Optimization Service Center is a no-charge tool. If the terms are acceptable, select **I accept the terms of the license agreement** and then click **Next**, as shown in Figure 3-2.

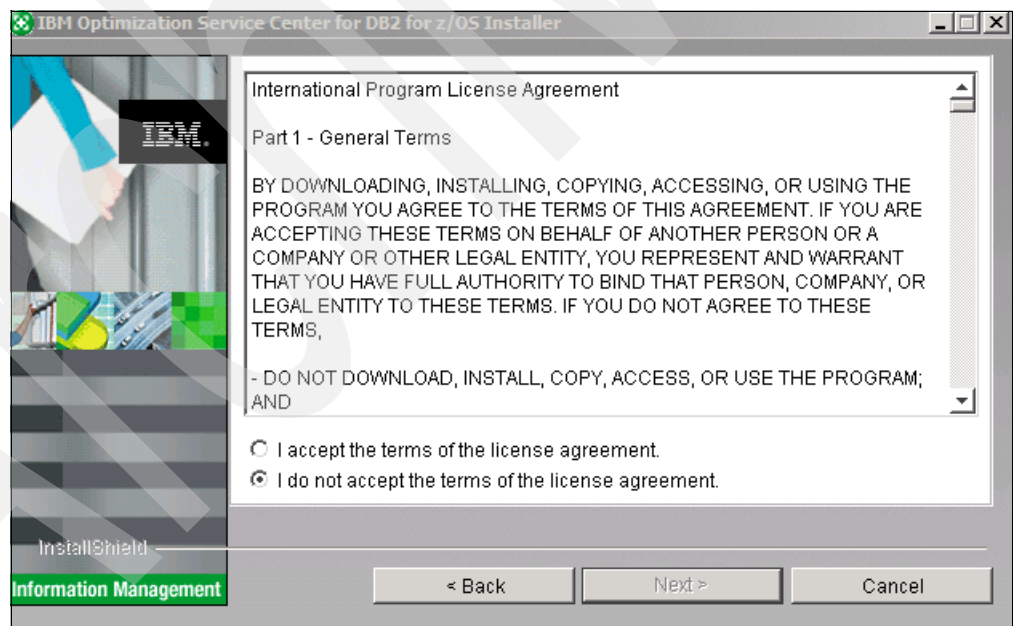


Figure 3-2 Optimization Service Center install - License agreement window

3. The next window, as shown in Figure 3-3 on page 90, asks if you want to install Optimization Service Center on the workstation you are working on or save the information for a future silent install for multiple workstations. If you prefer to continue the install on the workstation you are working on, select **Install the product on this computer** and click **Next**.

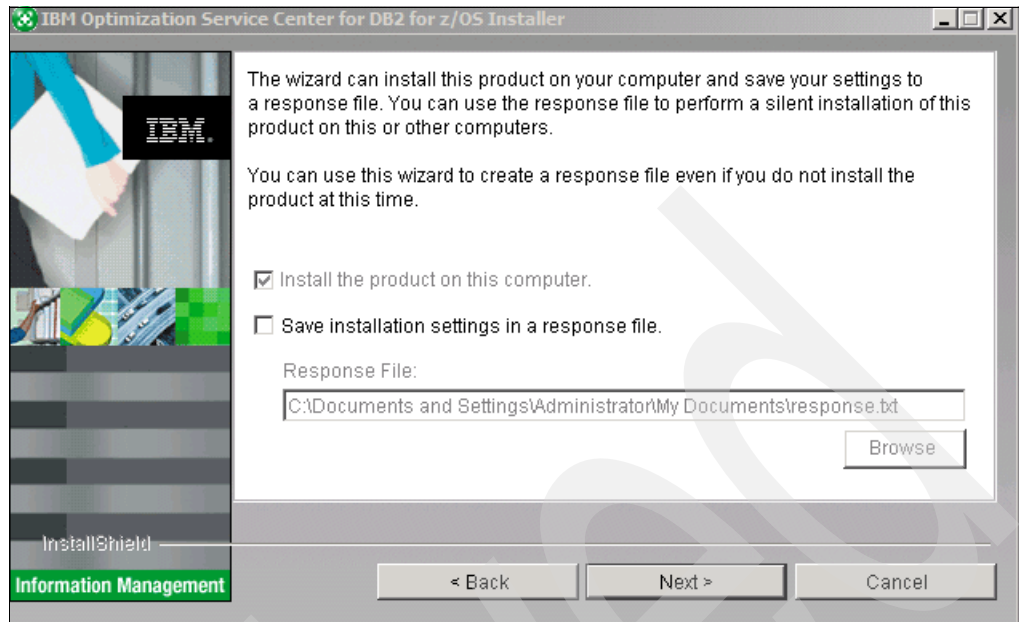


Figure 3-3 Optimization Service Center install - Save options window

4. The next window, as shown in Figure 3-4, displays the default directory to install Optimization Service Center, which you can change by clicking **Browse**. Space required and Space available are displayed. Choose your OSC install directory and click **Next**.

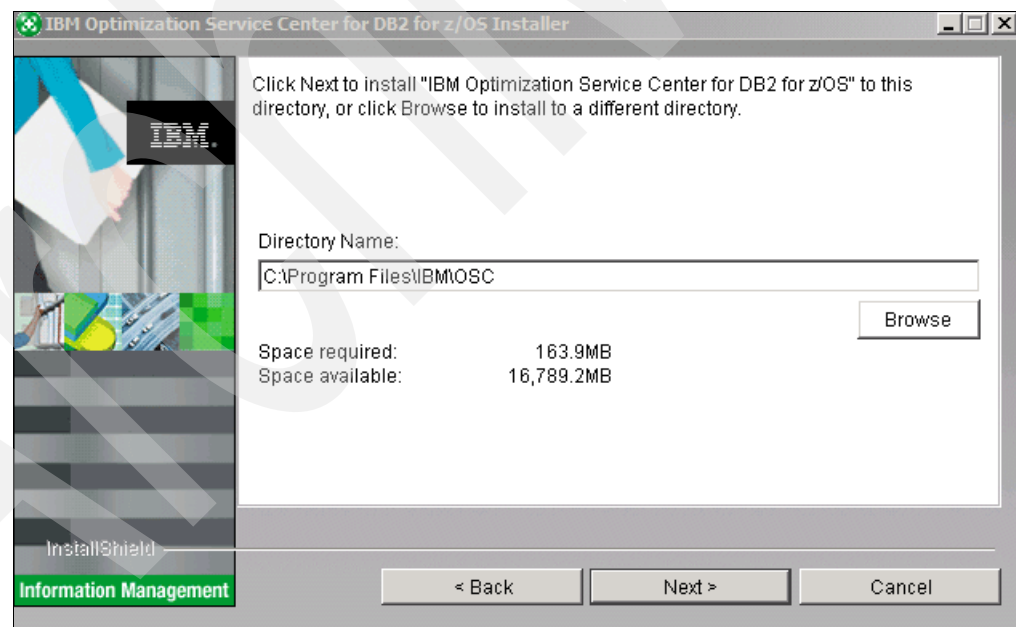


Figure 3-4 Optimization Service Center install - Directory window

5. The next window, as shown in Figure 3-5 on page 91, asks you to verify the install directory and the amount of space required. Click **Install** if the information displayed is correct.

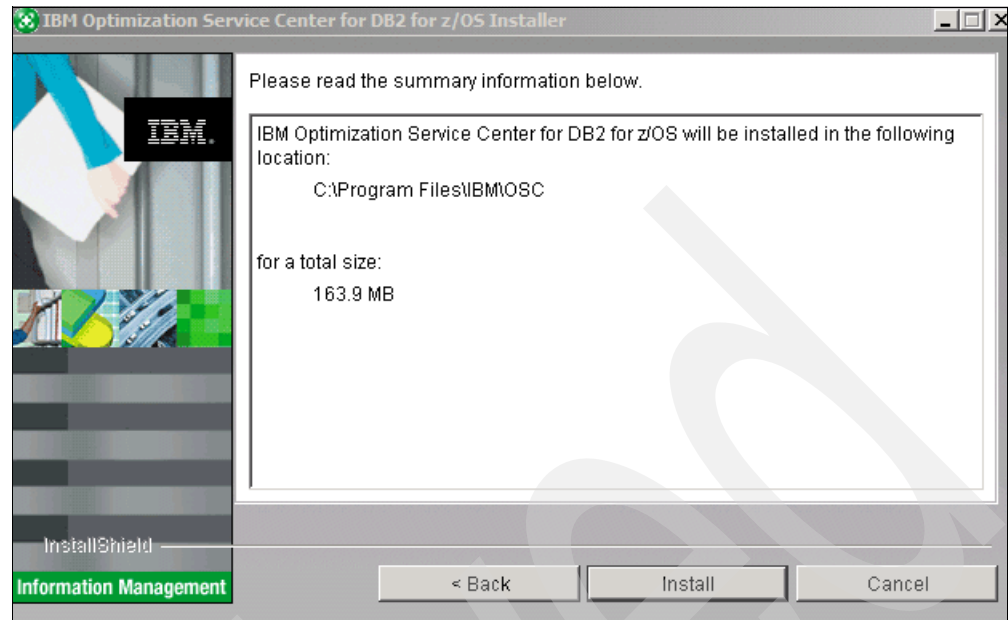


Figure 3-5 Optimization Service Center install - Directory and space verification window

6. The next window displays the install progress (Figure 3-6).

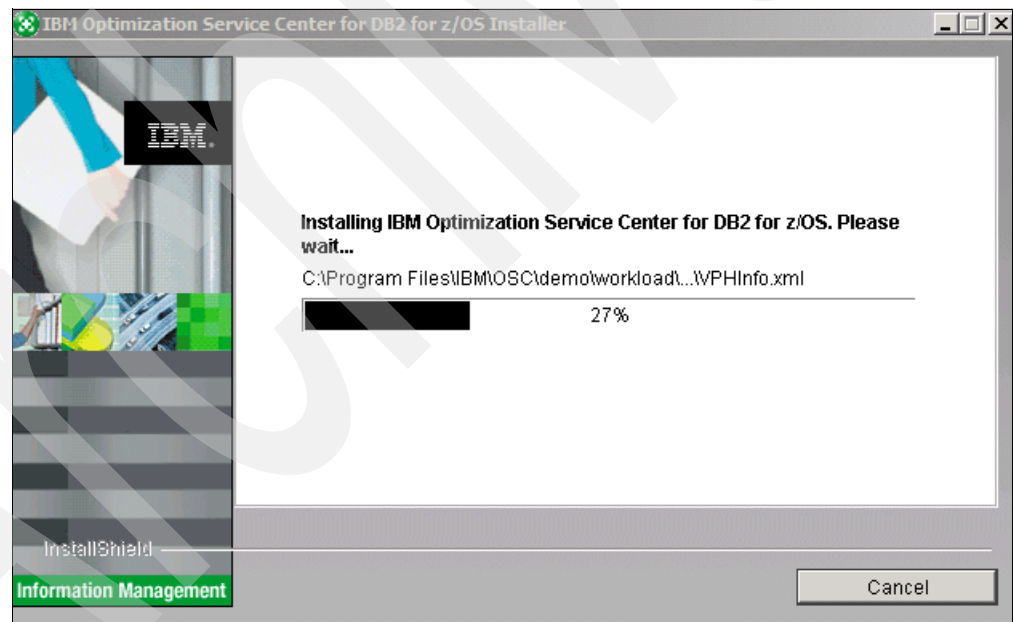


Figure 3-6 Optimization Service Center install - Progress window

7. If the install completed successfully, your last install window (Figure 3-7 on page 92) verifies this information. Click **Finish**. Otherwise, follow the subsequent windows.

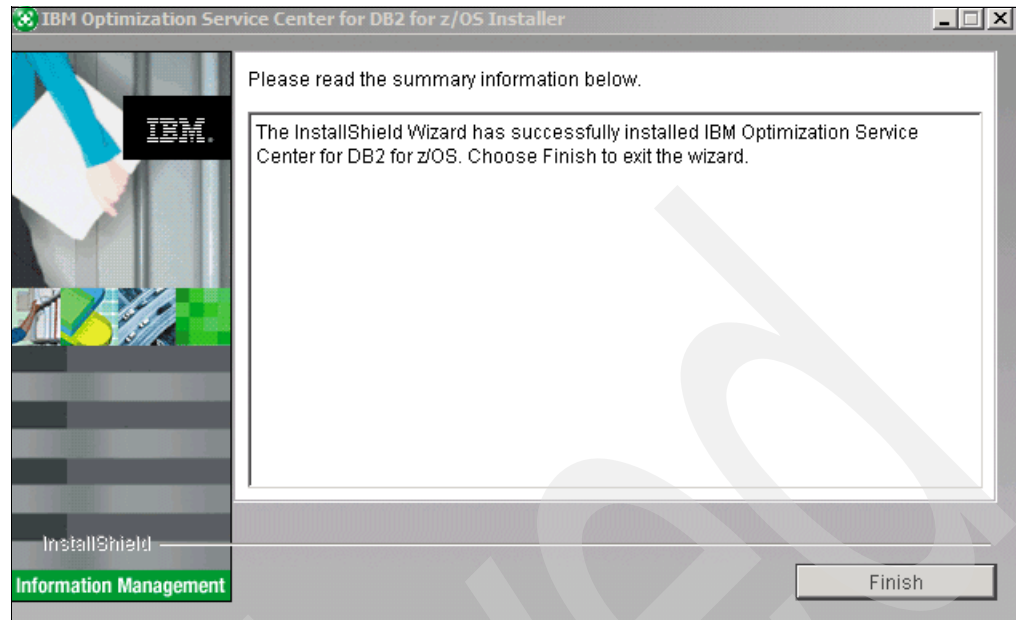


Figure 3-7 Optimization Service Center install - Successful install window

8. To verify the version installed, click the **Help** drop down and click **About OSC**. In this example, notice that Optimization Service Center with Fix Pack 1 is the current version installed (Figure 3-8).

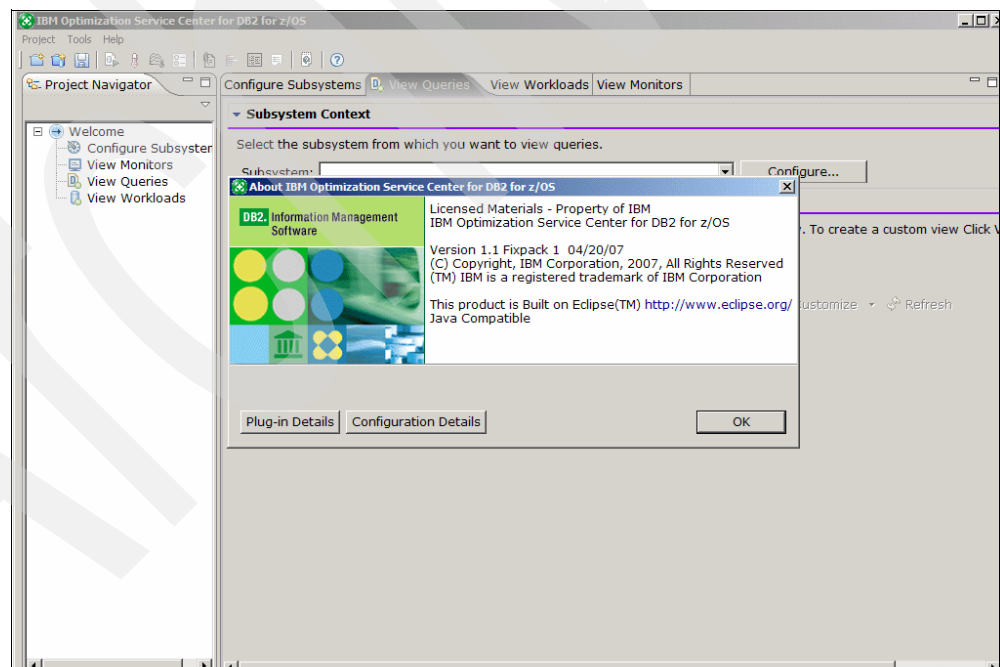


Figure 3-8 Optimization Service Center install - Successful install window

## 3.5 Performing a silent install

You can install Optimization Service Center on multiple workstations without being prompted for parameters on panels. This installation method called a *silent install*. The user can

download Optimization Service Center or push down to perform the silent install. As in a traditional install, hardware, software, and authority requirements must be in place prior to the install. This includes administrator privileges on each workstation.

To perform a silent install:

1. Download Optimization Service Center from one of the methods above to each workstation: from CD, post SMP/E, or Web download.
2. Extract the Optimization Service Center files into a temporary folder.
3. response.txt is one of the files used. You can edit this txt file to change the location of the installation package. The default installation directory is:

c:\Program Files\IBM\OSC\

As an example, you might want to change response.txt to point to a different path, in this case, MyOSC:

-P c:\MyOSC

4. On each workstation, execute silent.bat. A DOS command prompt opens to execute the install. The result is a successful install.
5. To launch Optimization Service Center from your Windows desktop, click **Start → All Programs → IBM Optimization Service Center for DB2 for z/OS → Optimization Service Center**. Optimization Service Center opens, and the Welcome page displays.

**Note:** Optimization Service Center requires that its software is downloaded and installed on the client's workstation. The install cannot use middleware for download and install purposes.

Archived

## Installing Optimization Expert

This chapter discusses how to install IBM DB2 Optimization Expert for DB2 for z/OS. You can install both DB2 Optimization Expert and DB2 Optimization Service Center on the same workstation. They can coexist.

You can install Optimization Expert on a Windows workstation, create a connection to DB2, and perform tuning tasks to improve the performance of SQL queries and workloads that run on that DB2.

This chapter describes the following topics:

- ▶ “Data server and DB2 requirements” on page 96
- ▶ “Workstation requirements” on page 100
- ▶ “Installing Optimization Expert on a workstation” on page 101
- ▶ “Step-by-step install instructions” on page 103
- ▶ “Performing a silent install” on page 107

## 4.1 Data server and DB2 requirements

Before you install Optimization Expert, ensure that each data server and DB2 system satisfy the hardware and software requirements.

For complete and detailed information about requirements for enabling a data server and DB2 system to work with Optimization Expert, see the *Program Directory for DB2 Optimization Expert*, GI10-8755.

Ensure that your data server and DB2 system meet the following prerequisites:

- ▶ Use DB2 Version 9.1 for z/OS.
- ▶ Enable Distributed Data Facility (DDF).
- ▶ Run two jobs, which are delivered with DB2 V9.1 for z/OS, to enable the Optimization Service Center portion of the Optimization Expert. The sample JCL jobs DSNTIJOS and DSNTIJRA are provided in the SDSNSAMP target library.

You can use the DSNTIJOS job to complete the following tasks on the DB2 system:

- Create objects that are used specifically by the Optimization Service Center, including tables, table spaces, and indexes.
- Bind Optimization Service Center packages.
- Create stored procedures that are used by the Optimization Service Center.
- Create aliases that are used by the Optimization Service Center.
- Grant the required execute privileges to packages and stored procedures.

**Note:** DSNTIJOS requires that buffer pools BP8K1 and BP16K1 have allocated memory before submitting.

You can use the DSNTIJRA job to create a RACF group that is required by the Optimization Service Center.

**Note:** If Optimization Service Center was installed prior to Optimization Expert, jobs DSNTIJOS and DSNTIJRA are not required.

- ▶ Enable DB2 Optimization Expert functions by running job AOCDDL, which you can find in the SMP/E library &AOCPREFIX.SAOCBASE. You can use AOCDDL to accomplish the following tasks:
  - Create objects that are used specifically by Optimization Expert, including tables, table spaces, and indexes.
  - Bind packages that are required specifically by Optimization Expert.

**Note:** AOCDDL requires that buffer pool BP32K1 has allocated memory before submitting.



- To use the Optimization Expert monitoring function, the Administrative Scheduler must be available. The Administrative Scheduler is a separate started task brought up by DB2; however, it is not shutdown by DB2. In fact, a DB2 crash will not shut down the Administrative Scheduler. The MODIFY command is required for all Administrative Scheduler started task changes. For more information about the Administrative Scheduler, see the following resources:
  - *DB2 9 Administration Guide SC18-9840*, Chapter 14, “Scheduling administrative tasks”
  - *DB2 9 Command Reference*, SC18-9844:
    - Chapter 43, “MODIFY admtproc,APPL=SHUTDOWN”
    - Chapter 44, “MODIFY admtproc,APPL=TRACE”
    - Chapter 72, “START admtproc”
    - Chapter 79, “STOP admtproc (z/OS)”
- Set up a WLM environment for running stored procedures.
- Install and enable the DB2-supplied stored procedures as shown in Table 4-1.

**Note:** As seen in the “Job that installs it” column, Optimization Expert requires that job DSNTJXP be run to create stored procedure DSN8.DSN8EXP. Run job DSNTIJCC to create stored procedure SYSPROC.DSNACCMD.

Table 4-1 Stored procedures required by Optimization Expert

Stored procedure name	What the procedure does	Job that installs it
SYSPROC.DSNWZP	Returns the current settings of the system parameters and allows you to view them in the Optimization Expert client.	DSNTIJSG
SYSPROC.DSNUTILU	Enables you to invoke utilities, such as the RUNSTATS utility directly from the Optimization Expert client.	DSNTIJSG
DSN8.DSN8EXP	Invokes the EXPLAIN statement, allowing you to run EXPLAIN from Optimization Expert.	DSNTEJXP
SYSPROC.OSC_RUNSQL	Uses a stored procedure to run EXPLAIN for the statement cache with SYSADM authority and captures workloads from the dynamic statement cache.	DSNTIJOS
SYSPROC.OSC_EXECUTE_TASK	Enables you to capture, run EXPLAIN for, and transform workloads and monitor profiles from Optimization Expert.	DSNTIJOS
SYSPROC.ADMIN_COMMAND_DB2	Stored procedure executes one or more DB2 commands on a connected DB2 system.	DSNTIJSG
SYSPROC.ADMIN_TASK_SCHEDULE	Schedules and updates workload control center tasks in the Administrative Scheduler.	DSNTIJSG
SYSPROC.ADMIN_TASK_REMOVE	Removes workload control center tasks from the Administrative Scheduler.	DSNTIJSG
SYSPROC.DSNACCMD	Executes DB2 commands (for instance, use it to enable cache trace).	DSNTIJCC

- Java environment:

For Optimization Expert to use a required Java Stored Procedure, you must enable the Java environment on the data server. For more information about how to set up the Java

environment, see *DB2 Version 9.1 for z/OS Application Programming Guide and Reference for JAVA*, SC18-9842, “Prerequisites and setup” in *DB2 for z/OS and OS/390: Ready for Java*, SG24-6435, and *DB2 for z/OS Stored Procedures: Through the CALL and Beyond*, SG24-7083.

The Java stored procedures, corresponding global table, and packages (DSNTIJMS) are listed in Table 4-2.

Table 4-2 DB2 objects for Optimization Expert Java procedures

Procedure	Global tables	Packages
SYSIBM.SQLCOLPRIVILEGES	SYSIBM.SQTCOLPRIVILEGES	DSNASPCC.DSNACPR8
SYSIBM.SQLCOLUMNS	SYSIBM.SQTCOLUMNS	DSNASPCC.DSNACOL8
SYSIBM.SQLFOREIGNKEYS	SYSIBM.SQTFORIGNKEYS	DSNASPCC.DSNAFNK8
SYSIBM.SQLPRIMARYKEYS	SYSIBM.SQTPRIMARYKEYS	DSNASPCC.DSNAPRK8
SYSIBM.SQLPROCEDURECOLS	SYSIBM.SQTSPECIALCOLUMNS	DSNASPCC.DSNAPCO8
SYSIBM.SQLPROCEDURES	SYSIBM.UINDEXES	DSNASPCC.DSNAPRC8
SYSIBM.SQLSPECIALCOLUMNS	SYSIBM.SQTPROCEDURECOLS	DSNASPCC.DSNASPC8
SYSIBM.SQLSTATISTICS	SYSIBM.SQTPROCEDURES	DSNASPCC.DSNASTA8
SYSIBM.SQLTABLEPRIVILEGES	SYSIBM.SQTSTATISTICS	DSNASPCC.DSNATBP8
SYSIBM.SQLTABLES	SYSIBM.SQTTABLEPRIVILEGES	DSNASPCC.DSNATBL8
SYSIBM.SQLGETTYPEINFO	SYSIBM.SQTTABLES	DSNASPCC.DSNATYP8
SYSIBM.SQLUUDTS	SYSIBM.SQLTABLETYPES	DSNASPCC.DSNAUDT8
SYSIBM.SQLCAMESSAGE	SYSIBM.SQLTYPEINFO	
	SYSIBM.SQTUDTS	

– JDK v1.4:

Ensure that version 1.4 JDK is installed under the folder in the UNIX Service System that is specified by the JAVA\_HOME environment variable.

– JDBC/SQLJ driver for z/OS:

Ensure that the driver file, db2jcc.jar, is installed under the folder in the UNIX Service System that is specified by CLASSPATH.

– JCC properties file:

Ensure that the SQLJ/JDBC runtime properties file is set up correctly. It contains settings for the JDBC/SQLJ driver for OS/390, and is specified by the DB2SQLJPROPERTIES environment variable.

DB2SQLJSSID: Specifies the name of the system to which a JDBC or an SQLJ application connects.

DB2SQLJPLANNAME: Specifies the name of the plan that is associated with a JDBC or an SQLJ application. The plan is created by the bind process.

For example:

```
# Any lines starting with the pound sign '#'
# are comments. Please see the DB2 for z/OS
# Application Programming Guide and Reference
# for Java for the description of these settings.
#
DB2SQLJSSID=V91C
DB2SQLJPLANNAME=DSNJDBC
DB2SQLJ_TRACE_FILENAME=/tmp/mytrc
DB2SQLJ_TRACE_BUFFERSIZE=1024
```

**Note:** UNIX Service System files and commands are case-sensitive.

- WLM-environment for Java stored procedures:

Ensure that the NUMTCB parameter of the WLM application environment for Java is not larger than 5.

Recommendation: Use a value of 1 to ensure that sufficient memory is available.

- Started-task job for Java stored procedures:

Ensure that a started-task job for Java stored procedures exists in the system PROCLIB. For example, the started task JCL for V91CWLMJ in SYS1.PROCLIB is shown in Example 4-1.

*Example 4-1 V91CWLMJ started task*

---

```
//V91CWLMJ PROC DB2SSN=V91C,NUMTCB=1,APPLENV=V91CWLMJ
//JPROC EXEC PGM=DSNX9WLM,TIME=1440,
// PARM='&db2ssn,&NUMTCB,&APPLENV',
// REGION=0M
//STEPLIB DD DISP=SHR,DSN=V91C.SDSNEXIT
// DD DISP=SHR,DSN=DSN910.NEW.SDSNLOAD
// DD DISP=SHR,DSN=DSN910.NEW.SDSNLOAD2
//JAVAENV DD DISP=SHR,DSN=V91C.JAVAENV
//JAVAOUT DD PATH='/V1R7/USR/db2910/JAVAOUT.TXT',
// PATHOPTS=(ORDWR,OCREAT,OAPPEND),
// PATHMODE=(SIRUSR,SIWUSR,SIRGRP,SIWGRP,SIROTH,SIWOTH)
//JAVAERR DD PATH='/V1R7/USR/db2910/JAVAERR.TXT',
// PATHOPTS=(ORDWR,OCREAT,OAPPEND),
// PATHMODE=(SIRUSR,SIWUSR,SIRGRP,SIWGRP,SIROTH,SIWOTH)
//JSPDEBUG DD SYSOUT=*
//SYSTSPRT DD SYSOUT=*
//CEEDUMP DD SYSOUT=*
//SYSPRINT DD SYSOUT=*
//OUT1 DD SYSOUT=*
//SYSABEND DD DUMMY
//DSNTRACE DD SYSOUT=*
```

---

- JDBC/SQLJ environment variables:

Ensure that JDBC/SQLJ environment variables are created under the UNIX Service System, such as V91C.JAVAENV shown in Example 4-1. In the sample below, dsnwccsp.jar is the Java stored procedure JAR file that was shipped as part of DB2 for z/OS. JCC\_HOME is the home directory of the JCC driver, and JAVA\_HOME is the home directory of JRE. You might need to change JCC\_HOME and JAVA\_HOME, depending on where your JCC driver and JRE are installed.

```
ENVAR("CLASSPATH=/usr/lpp/db2910_base/lib/IBM/dsnwccsp.jar",
"TZ=PST08",
"JCC_HOME=/usr/lpp/db2/devbase",
"JAVA_HOME=/usr/lpp/java140/J1.4"),
MSGFILE(JSPDEBUG,,,,ENQ),
XPLINK(ON)
```

**Note:** You must include all the environment variables (ENVAR) in this file. Ensure that the total length of all the entries put together, excluding blanks, does not exceed 245 bytes. Exceeding 245 bytes for ENVAR produces unpredictable results and errors. For entries that exceed the 245 byte limit, place most of the ENVAR entries into an HFS file, then point ENVAR to that file. The HFS file has no limitation of size. For more information, see *DB2 for z/OS Stored Procedures: Through the CALL and Beyond*, SG24-7083, section 17.2.6, “Setting up the JAVAENV data set for Java stored procedure execution”.

## 4.2 Workstation requirements

Ensure that your workstation meets the following hardware requirements:

- ▶ 500 MB of space on your hard disk drive
- ▶ 1 GB of memory
- ▶ A Pentium IV processor equivalent or above

Recommendation: For optimal performance, use faster processors with more memory.

Ensure that you have the following software installed on your workstation:

- ▶ One of the following Microsoft Windows operating systems:
  - Windows 2000
  - Windows XP
  - Windows 2003
- ▶ Microsoft Internet Explorer Version 5.5 or later
- ▶ Adobe SVG Viewer 3.0. Optimization Expert prompts you to download and install this component when you first access the visual plan hints function. You must download the SVG Viewer from the Adobe Web site.

**Note:** According to the Adobe Web site, Adobe will discontinue support for the SVG Viewer as of January 1, 2008. Optimization Service Center or Optimization Expert developers are working on an alternative solution.

- ▶ IBM DB2 Connect Application Server, Enterprise, Personal, or Unlimited Edition at Version 9.1 Fix Pack 1, Version 8.2 Fix Pack 6, or Version 8.1 Fix Pack 13 level or later.

**Note:** You can use a limited-use license for DB2 Connect Personal Edition V9, which is included with IBM DB2 V9.1 for z/OS, for connections from Optimization Expert to DB2. Customers requiring additional licenses need to inquire about an upgraded version of DB2 Connect.

- ▶ An IP network connection to DB2.

Recommendation: For faster access, do not use a dial-up connection. Use a broadband network connection.

## 4.3 Installing Optimization Expert on a workstation

You can install Optimization Expert on a workstation based on the following three options:

- ▶ The product CD
- ▶ SMP/E
- ▶ Purchasing and downloading DB2 Optimization Expert from the ShopzSeries Web page

**Note:** Optimization Expert is a purchased tool. Optimization Service Center is a no-cost tool.

Before installing Optimization Expert, ensure that your workstation meets the hardware and software requirements listed in 4.2, “Workstation requirements” on page 100.

Changes to DB2 might be required before you can install and use Optimization Expert. Such changes include enabling DDF and setting up a WLM environment. After you confirm that DB2 is configured properly, you can install Optimization Expert on your workstation. You must have administrator privileges on the workstation before you can install Optimization Expert.

Two versions of Optimization Expert were available while writing this book, the GA version and a fix pack version. For this version of Optimization Expert, you must install the GA version followed by installing Optimization Expert Fix Pack 1. Optimization Expert requires licensing information, which is housed in the GA version. If documentation does not exist for your install level to state whether to install the GA version prior to a fix pack version, contact support prior to the install to verify. In our situation, installing the fix pack version of Optimization Service Center without the base level was fully acceptable. However, this was not true for Optimization Expert, which had the same scenario.

**Note:** You can install Optimization Expert through the mainframe or workstation environments. If the install is done using the workstation approach, the Workload Control Center stored procedures will not be created. In this case, there are drawbacks to monitoring.

For customers requiring strong control over their environment, we recommend the mainframe approach using SMP/E.

### 4.3.1 Installing Optimization Expert from the product CD

If your workstation has a CD-ROM drive, you can install Optimization Expert from the product CD:

1. Insert the Optimization Expert product CD into your workstation's CD drive.
2. Navigate to the Windows location of your CD drive to view the contents of the product CD.
3. Double-click **setup.exe** and follow the instructions to complete the installation wizard. Upon successful completion, Optimization Expert is installed.
4. To launch Optimization Expert from your Windows desktop, click **Start → All Programs → IBM Optimization Expert for DB2 for z/OS → Optimization Expert**. Optimization Expert opens and the Welcome page displays.

### 4.3.2 Installing Optimization Expert from SMP/E

You can use SMP/E to transfer the Optimization Expert installation package to a workstation from the z/OS hierarchical file system (HFS).

1. To receive, apply, and accept the installation files to HFS using SMP/E, follow the instructions in the *IBM DB2 Optimization Expert for z/OS* program directory, GI10-8755.

**Note:** Review all SMP/E hold data before proceeding.

2. Transfer the installation package from the /usr/lpp/OE/IBM/ HFS directory to your workstation.
3. On your workstation, locate the installation files, double-click **setup.exe**, and follow the instructions to complete the installation wizard. Upon successful completion, Optimization Expert is installed.
4. To launch Optimization Expert from your Windows desktop, click **Start** → **All Programs** → **IBM Optimization Expert for DB2 for z/OS** → **Optimization Expert**. Optimization Expert opens and the Welcome page displays.

### 4.3.3 Purchasing and downloading DB2 Optimization Expert from the ShopzSeries Web page

You can purchase and download the DB2 Optimization Expert installation packages from the IBM ShopzSeries Web site. To learn more about how you can purchase it, follow the “Buy zSeries® software solutions online” link from the IBM e-business Transformation Web page at:

<http://www.ibm.com/software/os/zseries/transform/ordering.html>

## 4.4 Step-by-step install instructions

Perform the following install instructions:

1. After double clicking **setup.exe**, the installation wizard begins. The first prompted window shows the welcome window, as shown in Figure 4-1.

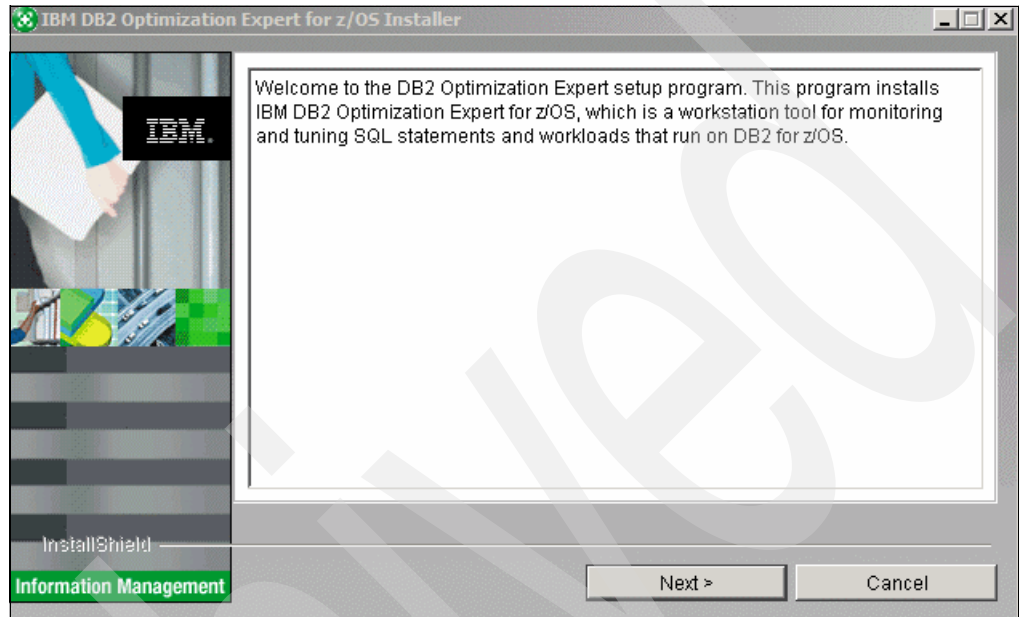


Figure 4-1 Optimization Expert Install - Welcome window

2. The next window verifies the licensing agreement. Keep in mind that Optimization Expert is a purchased tool. If the terms are acceptable, select "I accept the terms of the license agreement" and click **Next** (Figure 4-2).

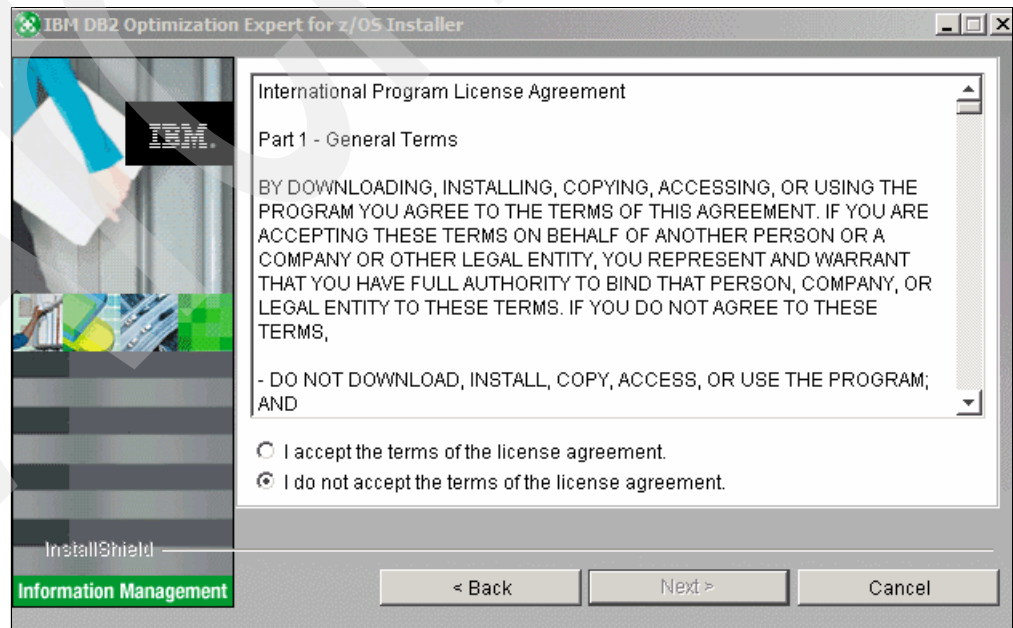


Figure 4-2 Optimization Expert install - License agreement window

3. The next window (Figure 4-3) asks if you would like to install Optimization Expert on the workstation you are working on or save the information for a future silent install for multiple workstations. If you prefer to continue the install on the workstation you are working on, select “Install the product on this computer” and click **Next**.

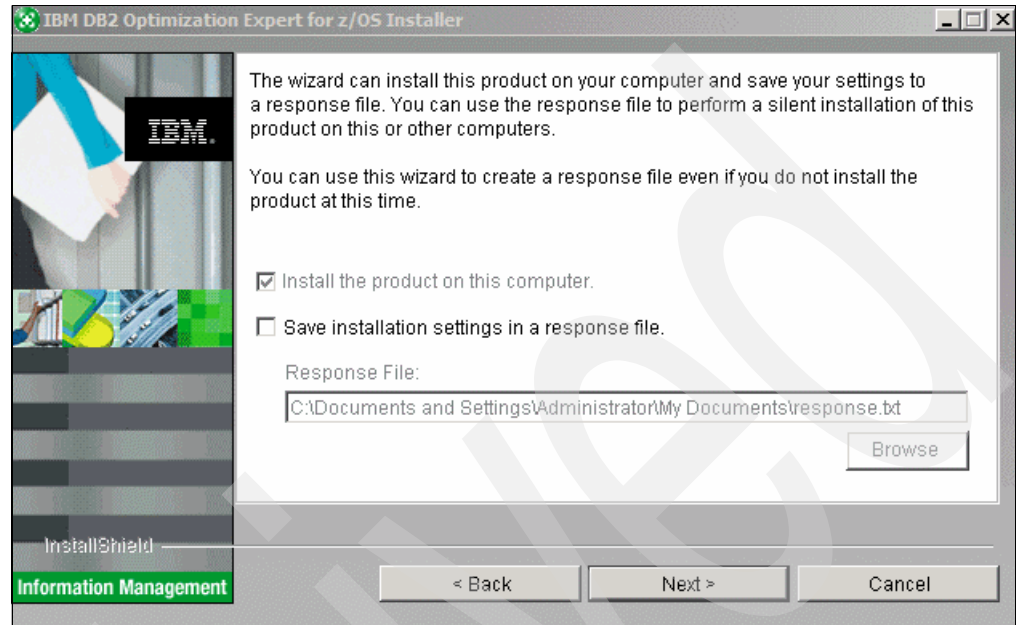


Figure 4-3 Optimization Expert install - Save options window

4. The next window (Figure 4-4) displays the default directory to install Optimization Expert, which you can change by clicking **Browse**. “Space required” and “Space available” are displayed as well. Choose your Optimization Expert install directory and click **Next**.

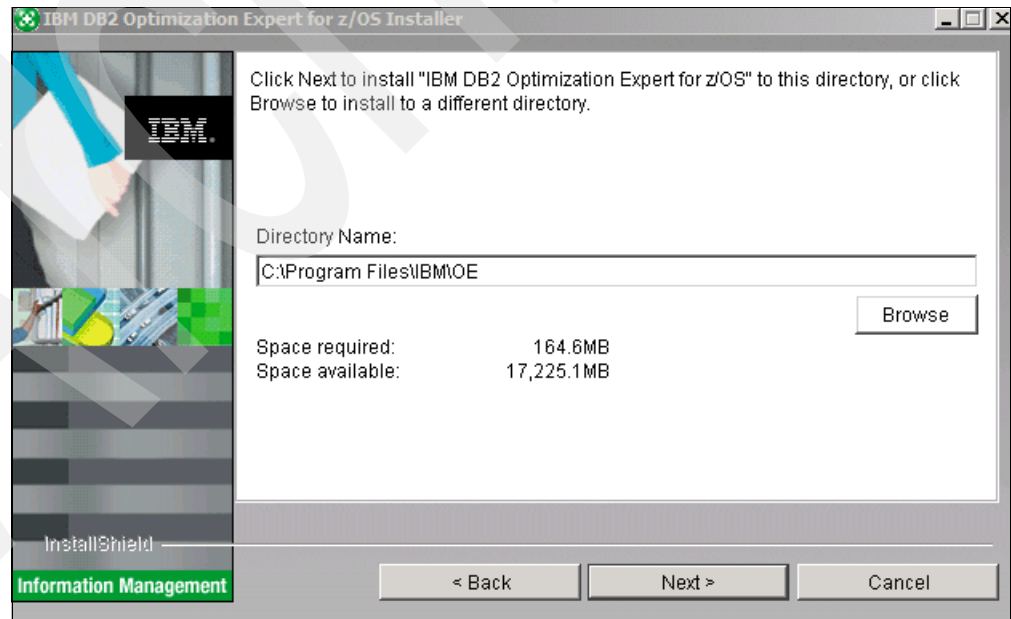


Figure 4-4 Optimization Expert install - Directory window



5. The next window (Figure 4-5) asks you to verify the install directory and the amount of space required. Click **Install** if the information displayed is correct.

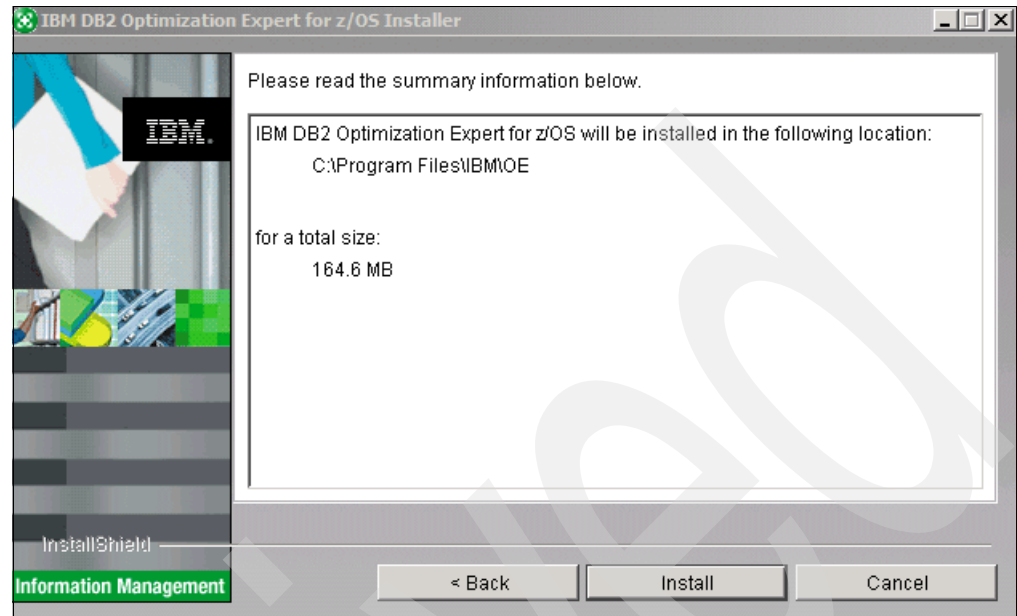


Figure 4-5 Optimization Expert install - Directory and space verification window

6. The next window (Figure 4-6) displays the install progress.

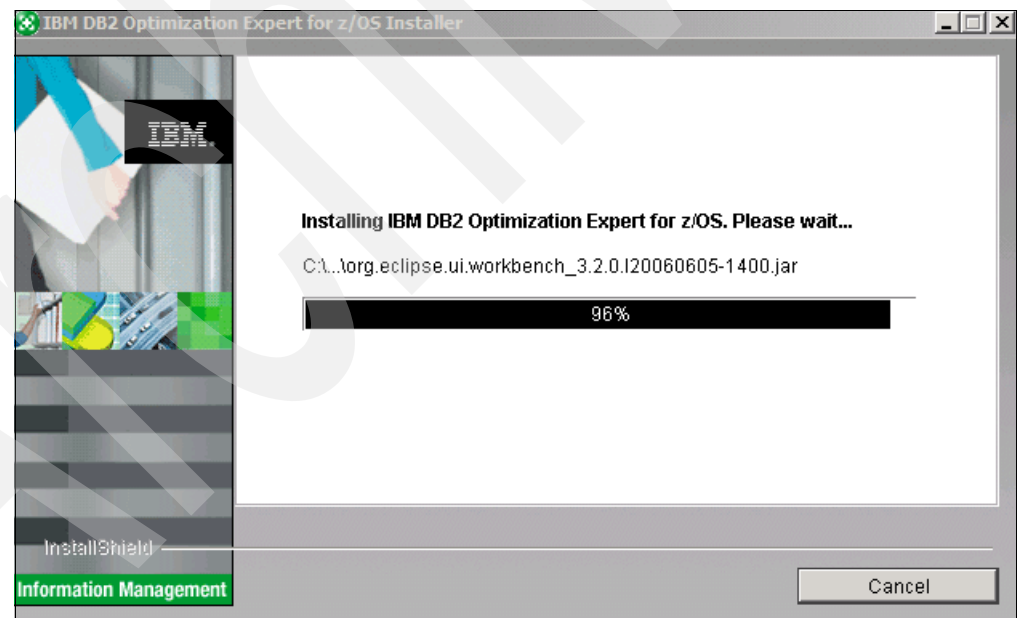


Figure 4-6 Optimization Expert install - Progress window

7. If the install completes properly, your last install window (Figure 4-7) verifies this information. Click **Finish**. Otherwise, follow the additional windows.

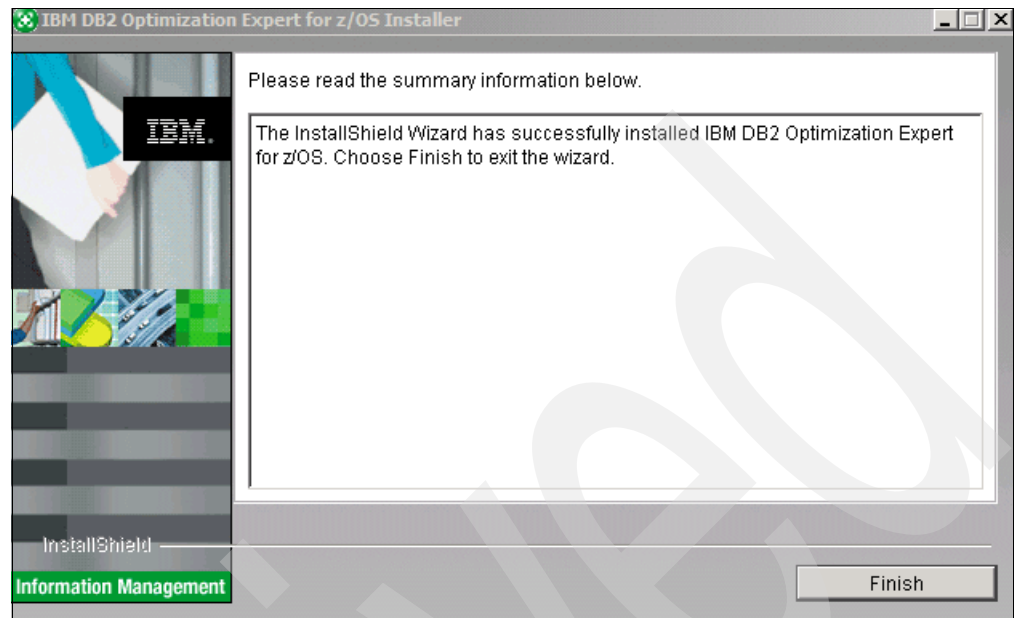


Figure 4-7 Optimization Expert install - Successful install window

8. To verify the version installed, click the **Help** drop down and click **About OE**. In this example, notice that Optimization Expert with Fix Pack 1 is the current version installed (Figure 4-8).

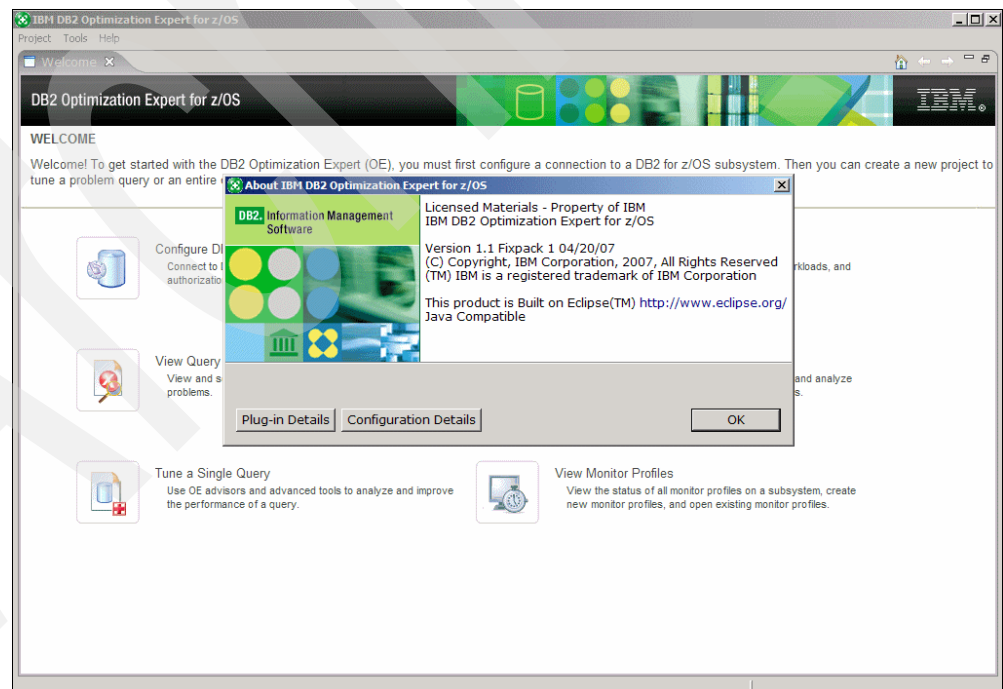


Figure 4-8 Optimization Expert - Verifying version installed

## 4.5 Performing a silent install

Customers can install Optimization Expert on multiple workstations without being prompted for parameters on panels. This method of install is called a *silent install*. You can download Optimization Expert or push down to perform the silent install. As in a traditional install, hardware, software, and authority requirements must be in place prior to the install. This includes administrator privileges on each workstation.

To perform a silent install:

1. Download Optimization Expert from one of the following methods to each workstation: from CD, post SMP/E, or Web download.
2. Extract the Optimization Service Center files into a temporary folder.
3. response.txt is one of the files used. You can edit this txt file to change the location of the installation package. The default installation directory is:  
c:\Program Files\IBM\OE\  
As an example, you might want to change response.txt to point to a different path, in this case MyOE:  
-P c:\MyOE
4. On each workstation, execute silent.bat. A DOS command prompt opens to execute the install. The result is a successful install.
5. To launch Optimization Expert from your Windows desktop, click **Start** → **All Programs** → **IBM Optimization Expert for DB2 for z/OS** → **Optimization Expert**. Optimization Expert opens, and the Welcome page displays.

**Note:** Optimization Expert requires that its software is downloaded and installed on the client's workstation. The install cannot use middleware for download and install purposes.

Archived



## Tools configuration and customization

This chapter includes considerations on configuring Optimization Service Center or Optimization Expert on the workstation for your environment. It also covers security consideration. Because the steps are the same for Optimization Service Center and Optimization Expert, the terms are interchangeable in this section.

## 5.1 Workstation configuration

Several steps are needed before you can use Optimization Service Center or Optimization Expert to perform monitoring and tuning tasks for a particular DB2 for z/OS system:

1. “Identify a DB2 system”
2. “Connect to a DB2 system” on page 113
3. “Bind packages on the DB2 system (optional)” on page 115
4. “Enable the EXPLAIN function” on page 119
5. “Create user tables for workload information” on page 127

### 5.1.1 Identify a DB2 system

Perform the following steps:

1. Launch Optimization Expert from your Windows desktop, and click **Start → All Programs → IBM Optimization Expert for DB2 for z/OS → Optimization Expert**. Optimization Expert opens and the Configure Subsystems window displays as shown in Figure 5-1.

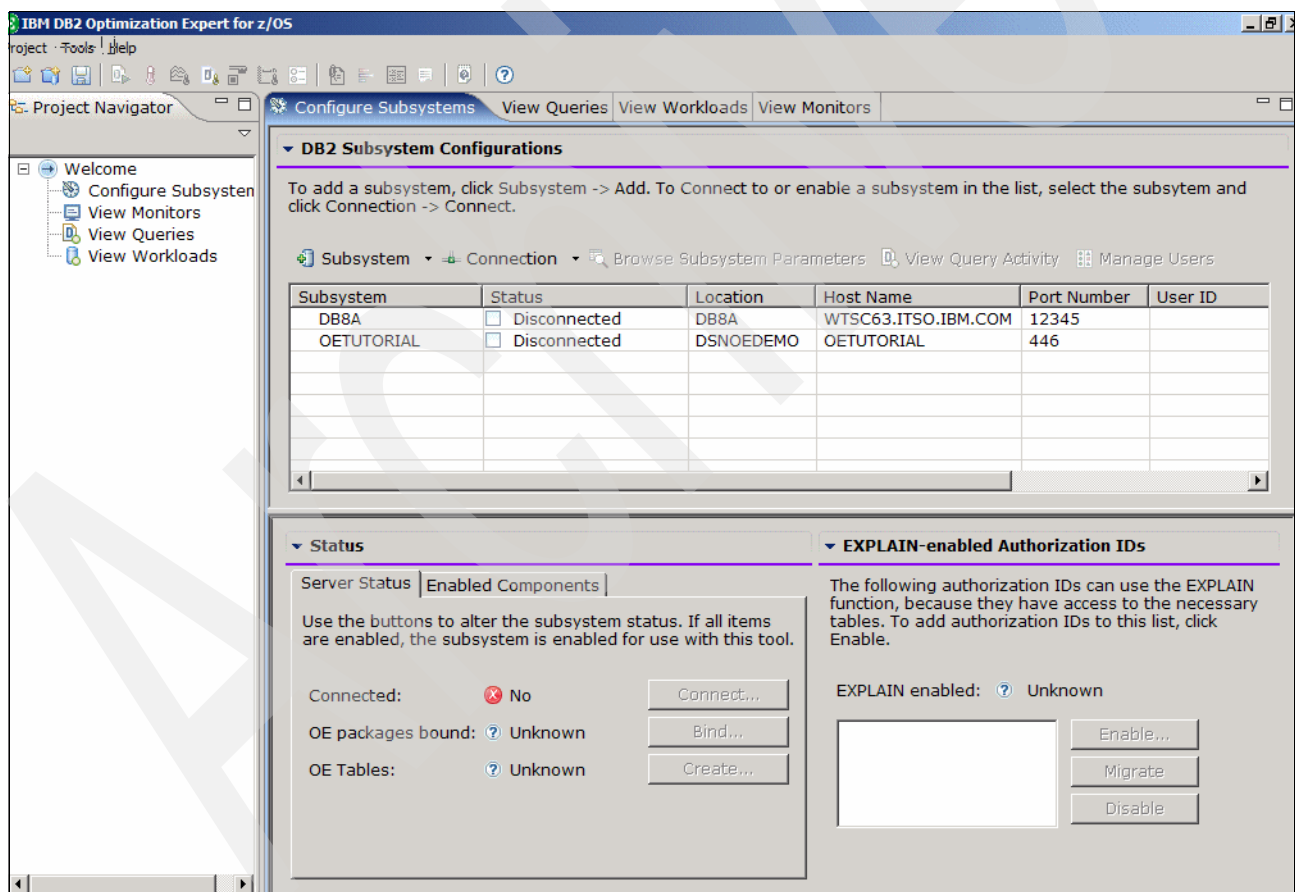


Figure 5-1 Configure Subsystem window

The Configure Subsystem window allows you to connect to a DB2 system previously configured and add or remove DB2 systems.

**Note:** Optimization Service Center or Optimization Expert often refer to the term “subsystem”. From a user perspective, you must view subsystem as a system, not if DB2 is part of a data sharing environment. Whether using DB2 data sharing with members or non-data sharing with subsystems, Optimization Service Center or Optimization Expert refers to all of these environments as subsystems.

In our scenario, we want to add a non-data sharing subsystem called DB9A. Click the **Subsystem** drop down button, and then click **Add**. The Subsystem Location window appears. Fill in the appropriate values, which you can find in these fields:

- Step DSNTLOG, which is the DDF portion of the original customized DSNZPARM member DSNTIJUZ.
- DDF portion of the DSNJU004 print log output.
- From the DB2 start up message DSNL004I. An example of the location output is found in Example 5-1.

*Example 5-1 DSNL004I display during DB2 start up*

---

```
DSNL004I  -DB9A DDF START COMPLETE
          LOCATION DB9A
          LU       USIBMSC.SCPDB9A
          GENERICLU -NONE
          DOMAIN   wtsc63.itso.ibm.com
          TCPPOINT 12347
          SECPOINT 12349
          RESPPOINT 12348
          IPNAME   -NONE
```

---

**Note:** The above information might not provide enough information to determine the host name for the sign-on configuration. In such cases, issue the DB2 command -DIS DDF and use IPADDR as the host name.

2. When the Subsystem Location window is filled in with your settings, as shown in Figure 5-2, click **Next**.

**Subsystem Properties**

**Subsystem Location**

Specify the following information about the subsystem that you want to configure and click Catalog.  
After the subsystem is cataloged, click Next.

**Steps**

- 1. Location
- 2. Connect

Subsystem alias: DB9A

Location: DB9A

Host name: wtsc63.itso.ibm.com

Port number: 12347

Comment:

Catalog

< Back   Next >   Finish   Cancel

Figure 5-2 Subsystem Location window



## 5.1.2 Connect to a DB2 system

To connect to a DB2 system:

1. Proceed to the Connect to subsystem window as shown in Figure 5-3. In this window, add your TSO user ID, password, and any special SQL ID. Click either **Finish**, or to test your connection, click **Connect**.

**Subsystem Properties**

**Connect to subsystem**

Specify the authorization information for the subsystem and click Connect. When the subsystem is connected, click Next or Finish.

**Steps**

- 1. Location
- 2. Connect

Complete the following fields and click Connect.

User ID: PAOLOR2

Password: .....

SQL ID: PAOLOR2

☒ Remember login ID

**Connect**

The following fields describe the OE status of this subsystem:

Connected: ✗ No

OE packages bound: ? Unknown

EXPLAIN enabled: ? Unknown

OE Tables: ? Unknown

< Back   Next >   Finish   Cancel

Figure 5-3 Connect to Subsystem window

2. In our scenario, we wanted to test the connection and clicked **Connect**. As you notice from Figure 5-4, our connection was successful. Access to the mainframe and Optimization Expert files are tested. You had already run the Optimization Expert setup jobs, so all the flags have already turned into green check marks. You can now click **Finish**.

**Subsystem Properties**

**Connect to subsystem**

Specify the authorization information for the subsystem and click Connect. When the subsystem is connected, click Next or Finish.

**Steps**

- 1. Location
- 2. Connect
- 3. Manage Users

Complete the following fields and click Connect.

User ID: PAOLOR2

Password: .....

SQL ID: PAOLOR2

☒ Remember login ID

Connect

The following fields describe the OE status of this subsystem:

Connected: ☒ Connected

OE packages bound: ☒ Bound

EXPLAIN enabled: ☒ PAOLOR2 is enabled

OE Tables: ☒ Available

< Back   Next >   Finish   Cancel

Figure 5-4 Testing the connection window

- After you click **Finish**, you are returned to the original Configure Subsystems window. The Status column now displays “OE Enabled”, and the Status area at the bottom of the window shows all green check marks (Figure 5-5). You are now ready to use Optimization Service Center or Optimization Expert.

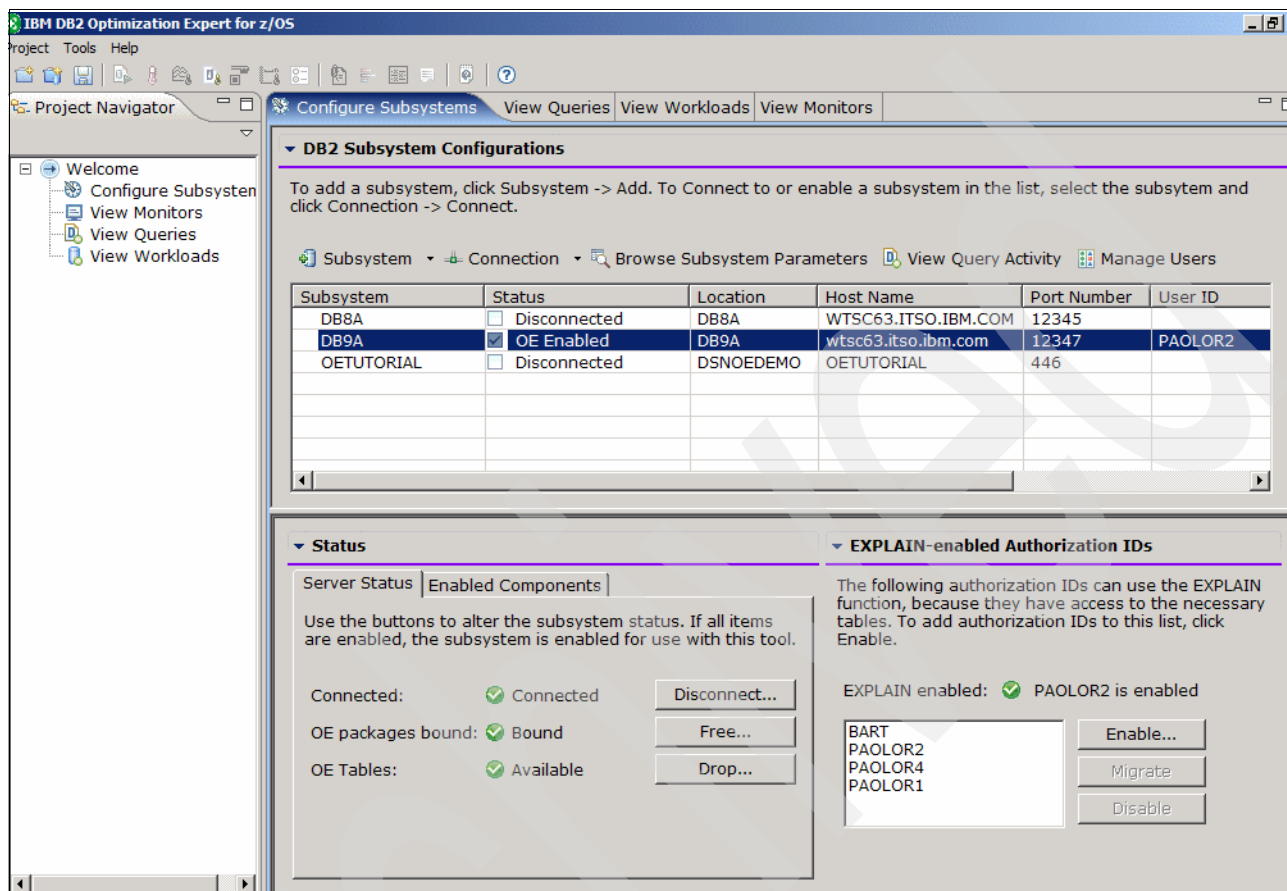


Figure 5-5 Configure Subsystems window with successful results

### 5.1.3 Bind packages on the DB2 system (optional)

Up to this point, there were no obstacles when trying to sign-on to the system. You needed to get the configuration information correct. You might be prompted for a response that asks if you want to BIND packages or CREATE the EXPLAIN tables. This can happen if the DSNTIJOS job was not successfully run, or if Optimization Service Center or Optimization Expert was installed using SMP/E at one level. However, your workstation install was at a different level. You encounter this scenario, for example, when SMP/E installs the GA version. However, you had downloaded the Fix Pack 1 version to your workstation. Upon launching the Optimization Service Center or Optimization Expert, you notice the red status lights for BIND and CREATE. With proper authority, you can have Optimization Service Center or Optimization Expert do the BIND and CREATE for you. In Figure 5-6 on page 116, you see a red “x” with the status of Package Not Found.

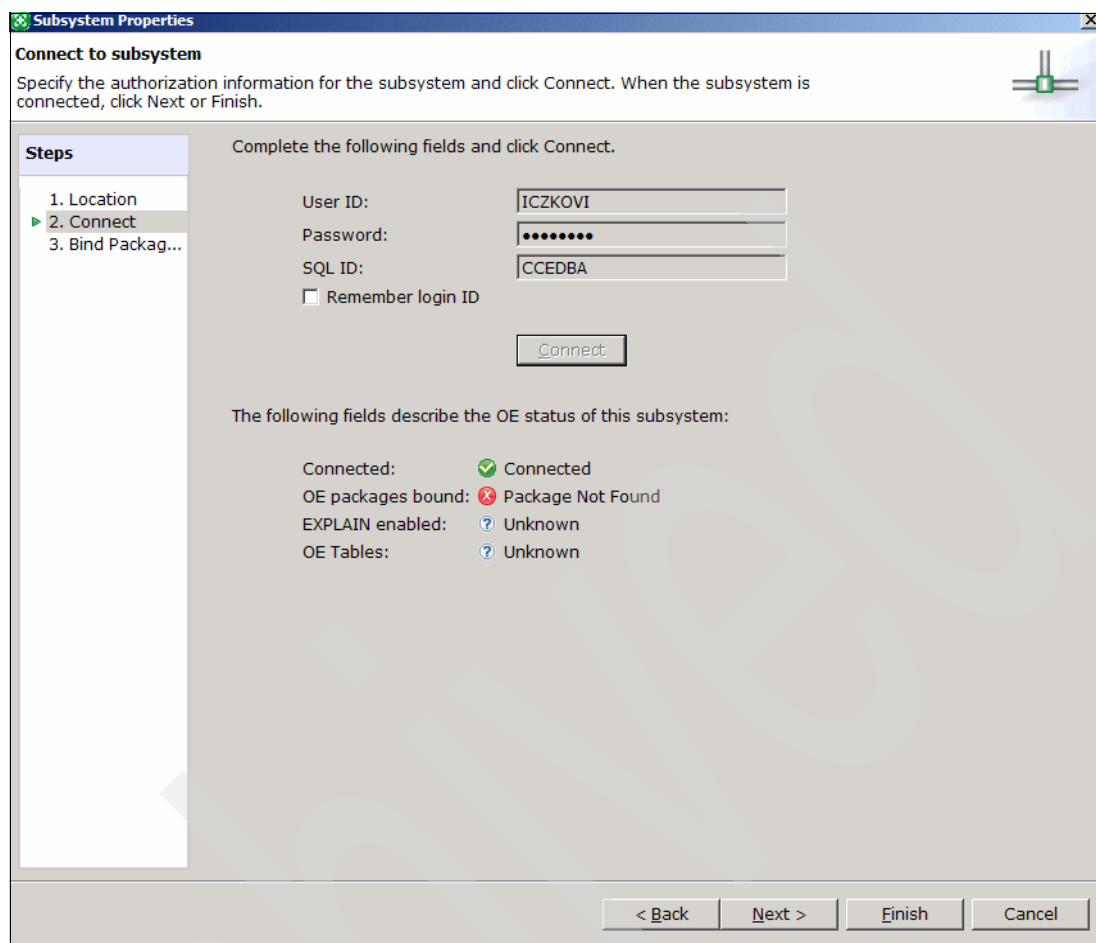


Figure 5-6 Connect - Package not found window

To bind the packages:

1. In Figure 5-7, a list of packages are displayed. Choose the packages you want to bind and click **Bind**.

**Note:** The stored procedures portion is optional. Stored procedures are not required. However, your processes will run faster if you enable them.

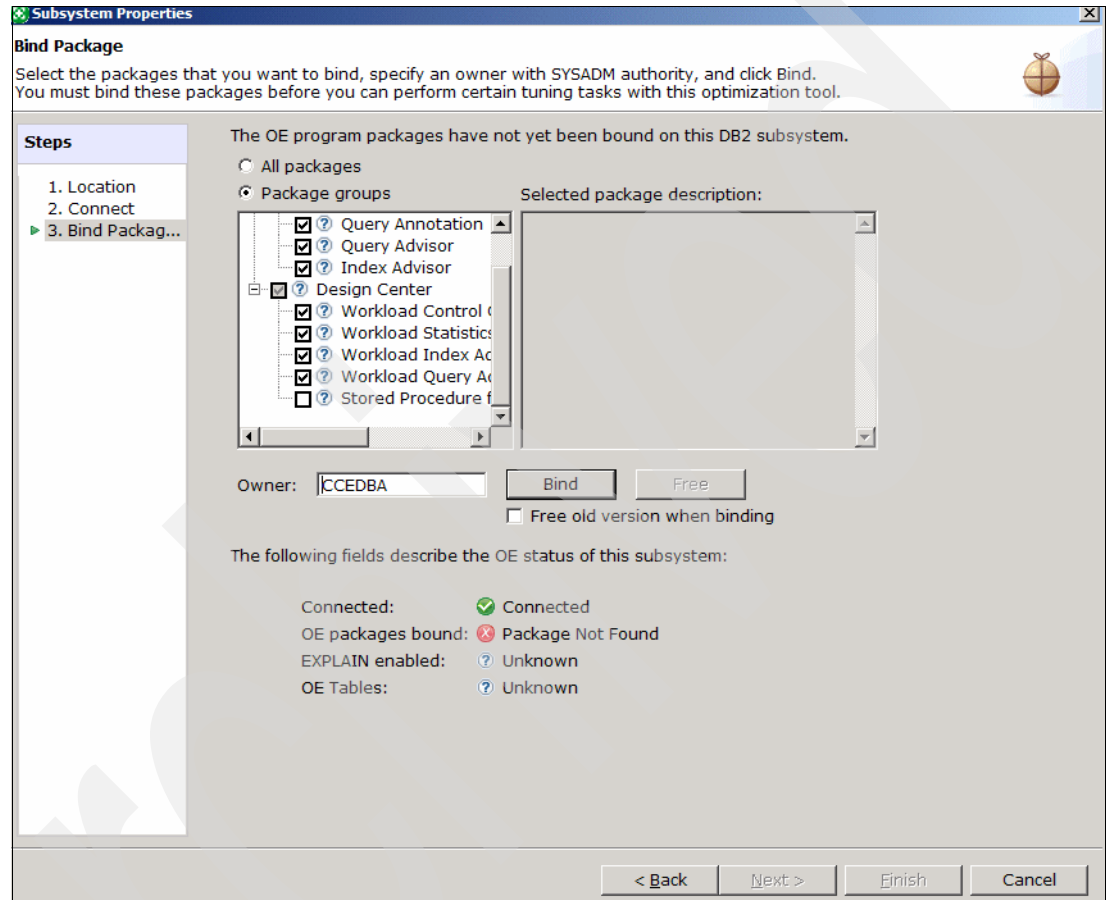


Figure 5-7 Connect - Bind window

2. Notice in Figure 5-8 that the BINDs were successful and the status changed to green check marks. Now the red “x” indicates that the EXPLAIN tables are not enabled. Click **Next** to continue.

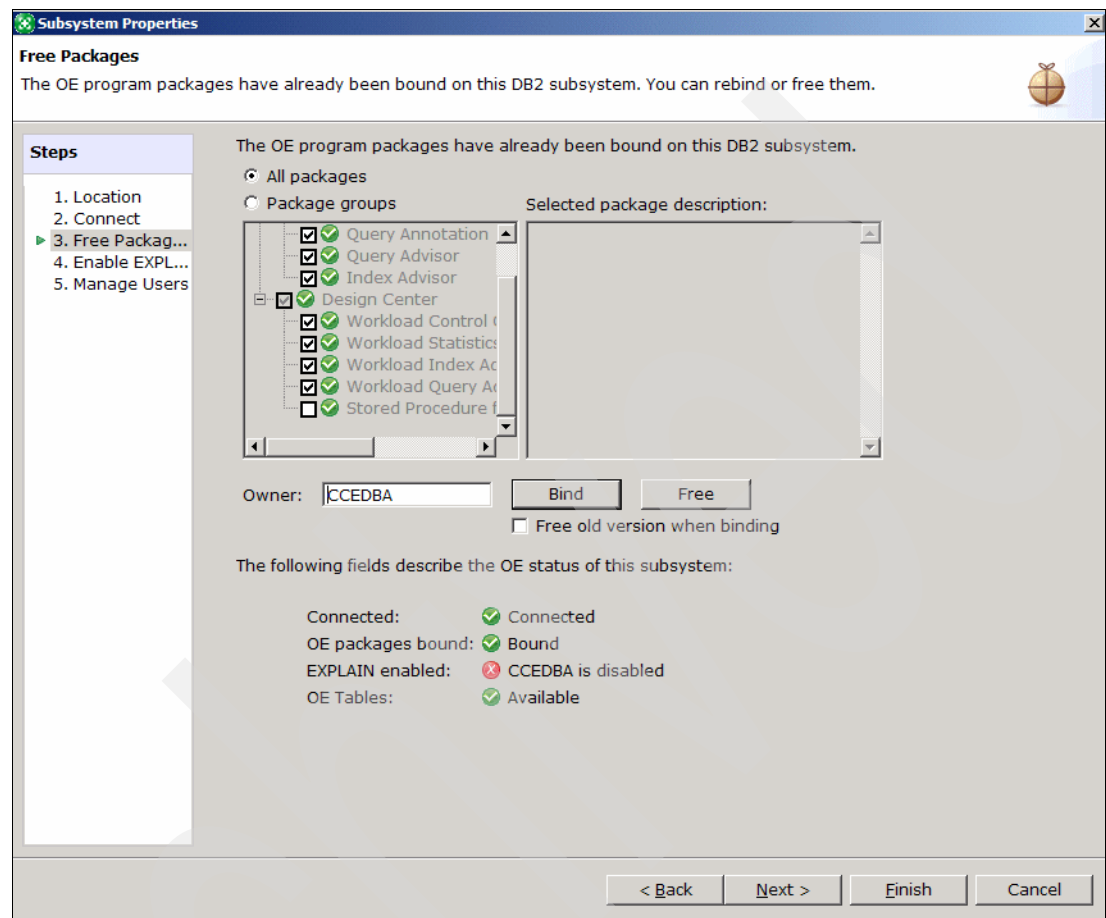


Figure 5-8 Connect - EXPLAIN tables are not enabled

## 5.1.4 Enable the EXPLAIN function

In Figure 5-9, notice Optimization Service Center or Optimization Expert allows you to create the EXPLAIN tables:

1. Provide the desired information and click **Next**.

**Subsystem Properties**

**Enable EXPLAIN**

Select an action to enable one or more SQL IDs to use the EXPLAIN function. Complete the related fields and click the appropriate button to complete the action. Then click Next or Finish.

**Steps**

1. Location
2. Connect
3. Free Packag...
- ▶ 4. Enable EXPL...
5. Manage Users

Action: Create new EXPLAIN tables

SQL ID: CCEDBA

Qualifier: CCEDBA

Database name: DSNDDB04 New...

4 KB table space: DSNSUMTS New...

8 KB table space: FXNTS New...

32 KB lob table space: LOBTS1 New...

32 KB lob table space: LOBTS2 New...

Create Tables

The following authorization IDs can use the EXPLAIN function on this subsystem, because the necessary EXPLAIN tables exist and they have access to them:

The following fields describe the OE status of this subsystem:

Connected: ✓ Connected

OE packages bound: ✓ Bound

EXPLAIN enabled: ✗ CCEDBA is disabled

OE Tables: ✓ Available

< Back Next > Finish Cancel

Figure 5-9 Connect - Enable EXPLAIN window

2. In Figure 5-10, the EXPLAIN table did not exist and Optimization Service Center or Optimization Expert provides a confirmation window that allows you to create it.

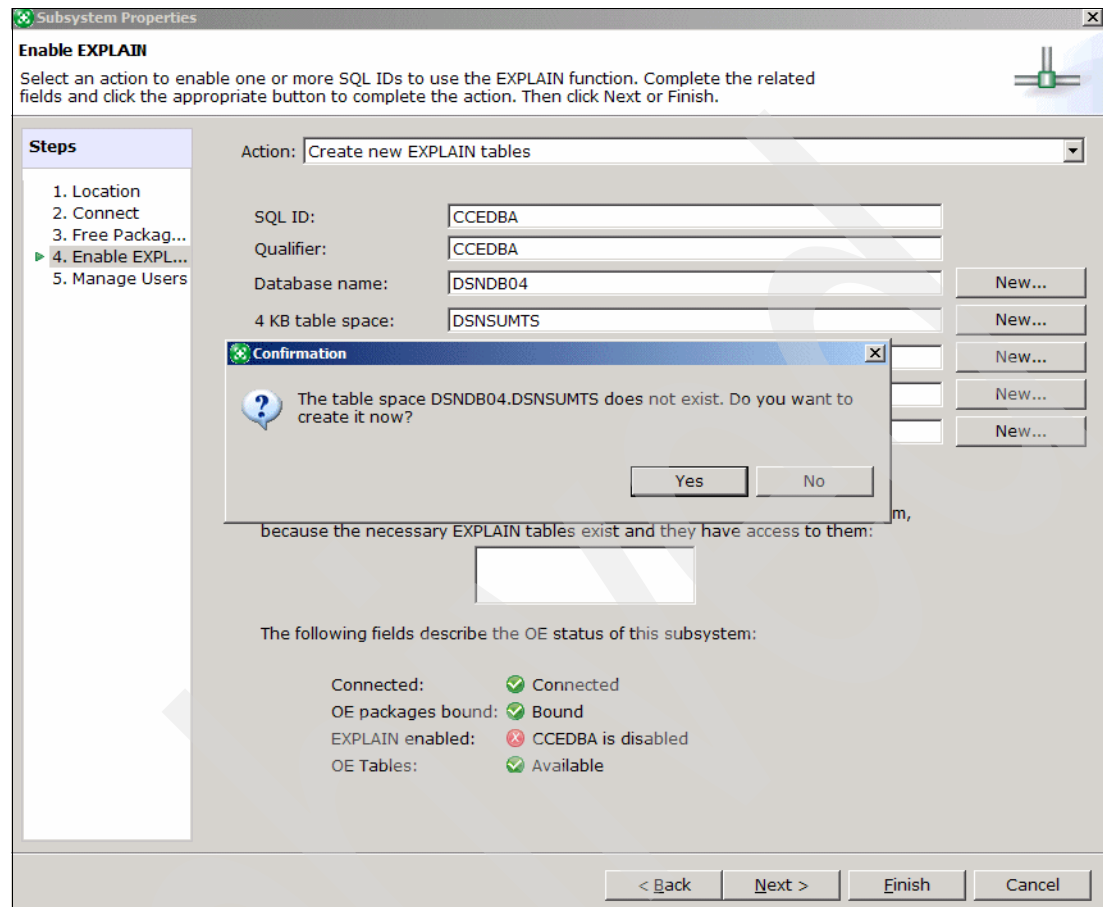


Figure 5-10 Connect - Create EXPLAIN table



3. You can override the CREATE options as seen in Figure 5-11. Click **Next** when you complete the options.

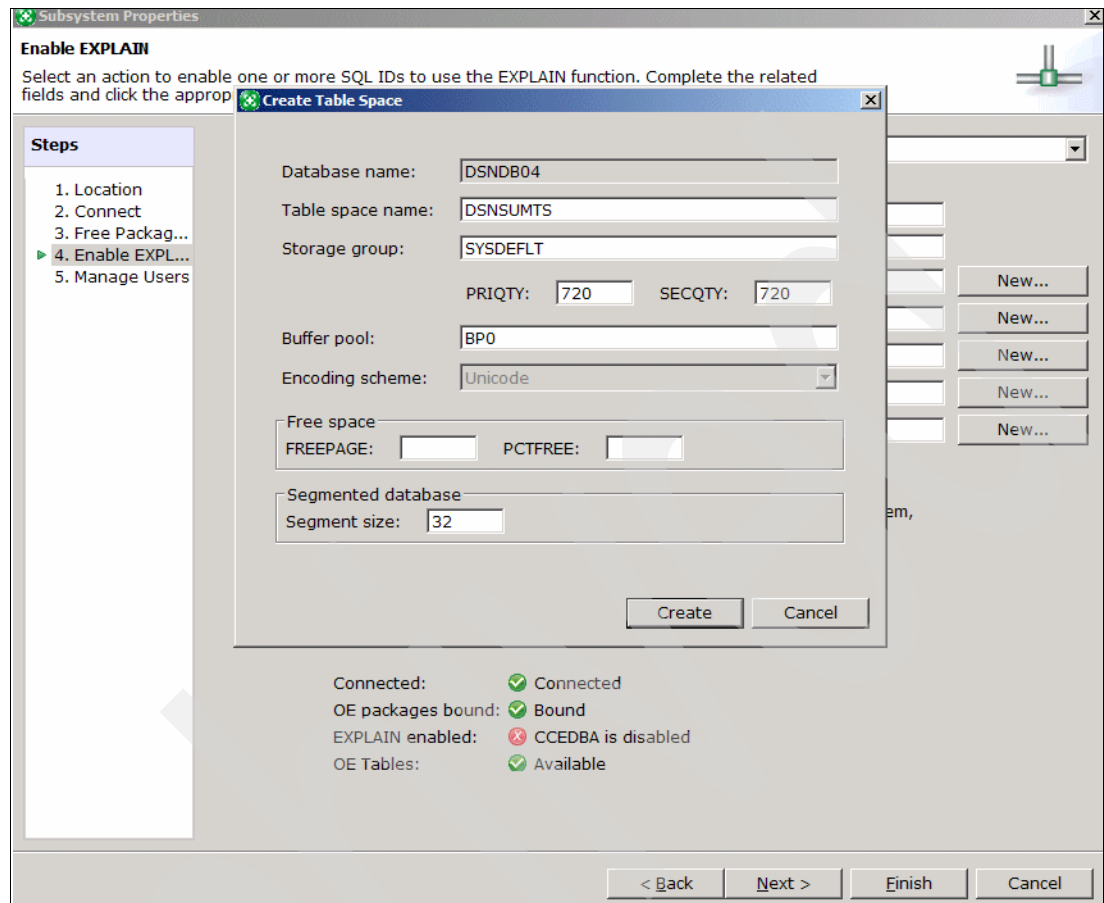


Figure 5-11 Connect - Override EXPLAIN options window

- Upon successful completion of the CREATE for the EXPLAIN tables, an information message window displays, as shown in Figure 5-12. Click **OK**, then **Next**.

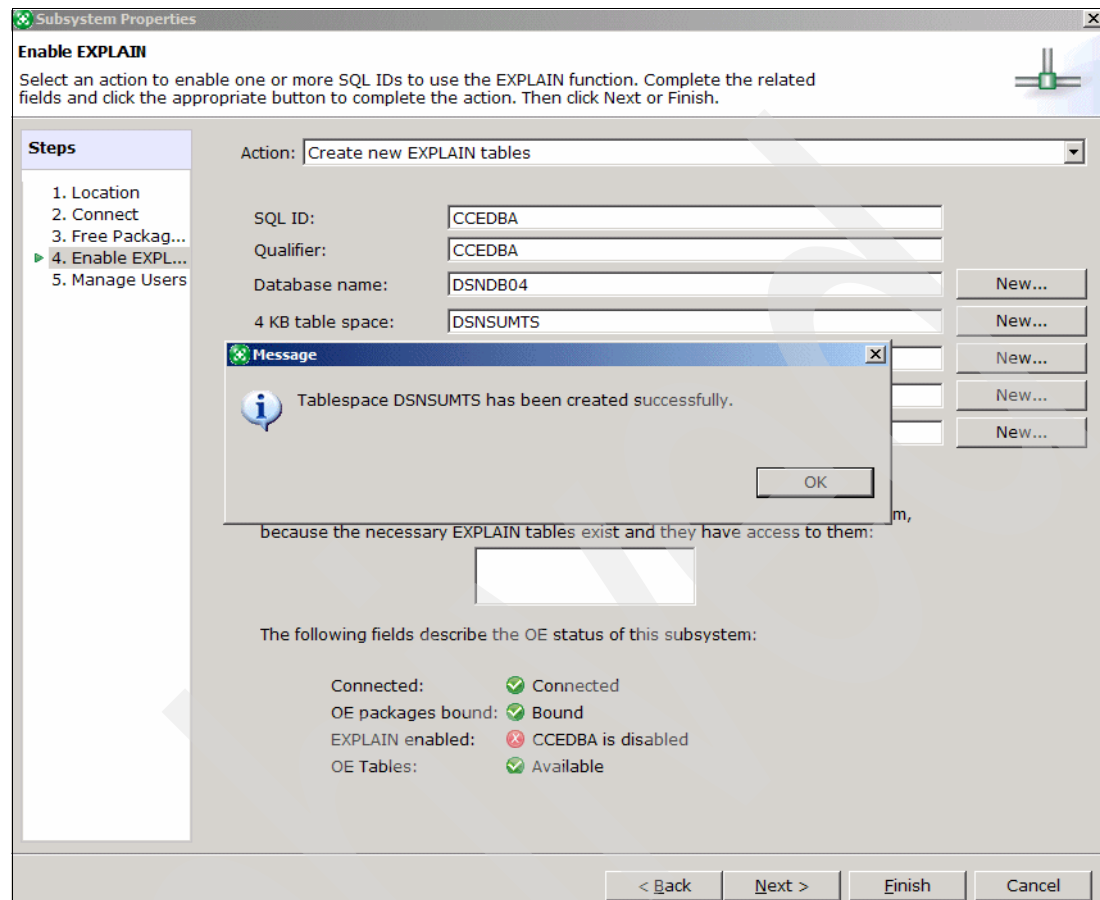


Figure 5-12 Connect - display successful CREATE of EXPLAIN tables

5. An error occurs when the EXPLAIN tables are not in the same format as expected, as shown in Figure 5-13. This happens in changes to code requiring a DROP of the old EXPLAIN tables and a CREATE of the new tables. You can manually DROP the tables displayed. Click **OK** and then **Next**.

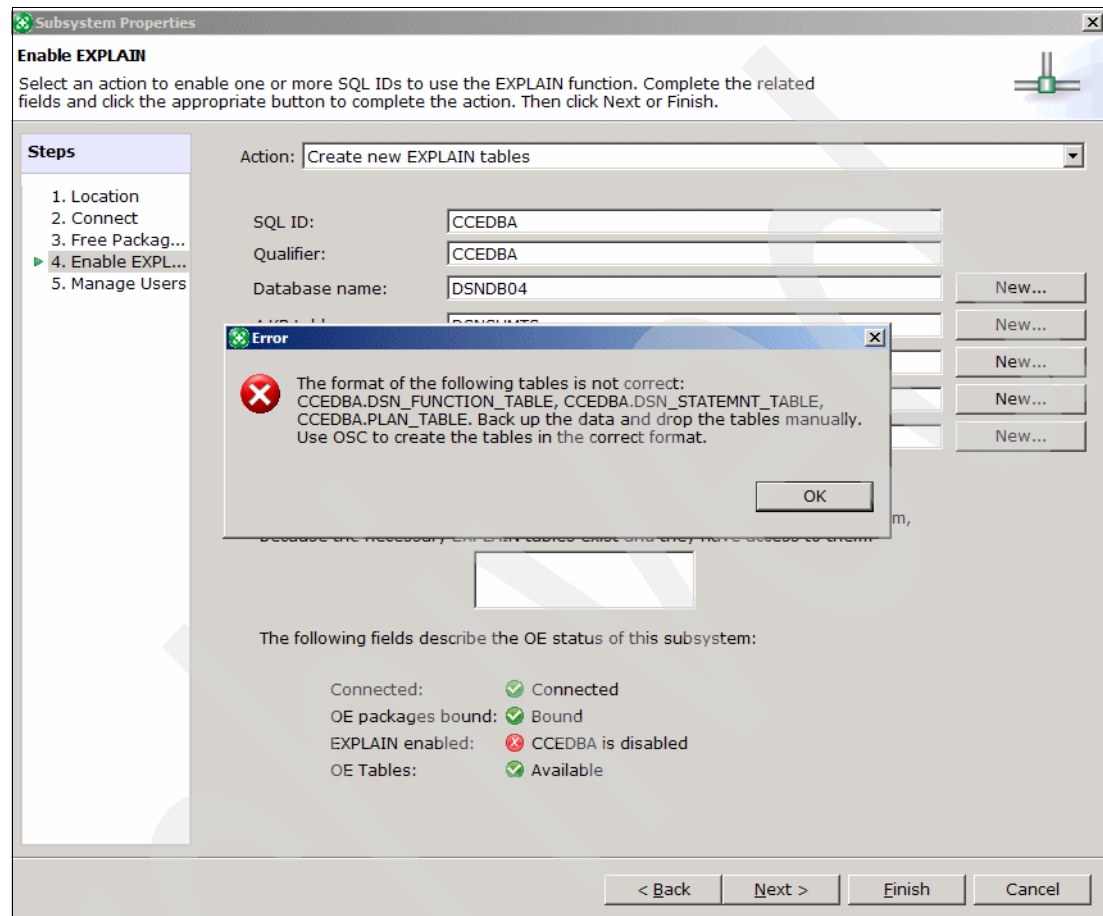


Figure 5-13 Connect - EXPLAIN version mismatch window

6. All lights are now green as shown in Figure 5-14. We are ready for Optimization Service Center or Optimization Expert to create the tables. Click **Next**.

**Subsystem Properties**

**Enable EXPLAIN**

Select an action to enable one or more SQL IDs to use the EXPLAIN function. Complete the related fields and click the appropriate button to complete the action. Then click Next or Finish.

**Steps**

1. Location
2. Connect
3. Free Packag...
4. Enable EXPL...
5. Manage Users

Action: Create new EXPLAIN tables

SQL ID: CCEDBA

Qualifier: CCEDBA

Database name: DSNDB04

4 KB table space: DSNSUMTS

8 KB table space: FXNTS

32 KB lob table space: LOBTS1

32 KB lob table space: LOBTS2

New... New... New... New... New...

Create Tables

The following authorization IDs can use the EXPLAIN function on this subsystem, because the necessary EXPLAIN tables exist and they have access to them:

CCEDBA

The following fields describe the OE status of this subsystem:

Connected: ☒ Connected

OE packages bound: ☒ Bound

EXPLAIN enabled: ☒ CCEDBA is enabled

OE Tables: ☒ Available

< Back Next > Finish Cancel

Figure 5-14 Connect - EXPLAIN tables are ready for next step

7. Existing EXPLAIN tables are displayed before proceeding with the CREATE, as shown in Figure 5-15. DROP is not necessary for these objects. Click **OK**, then **Next**.

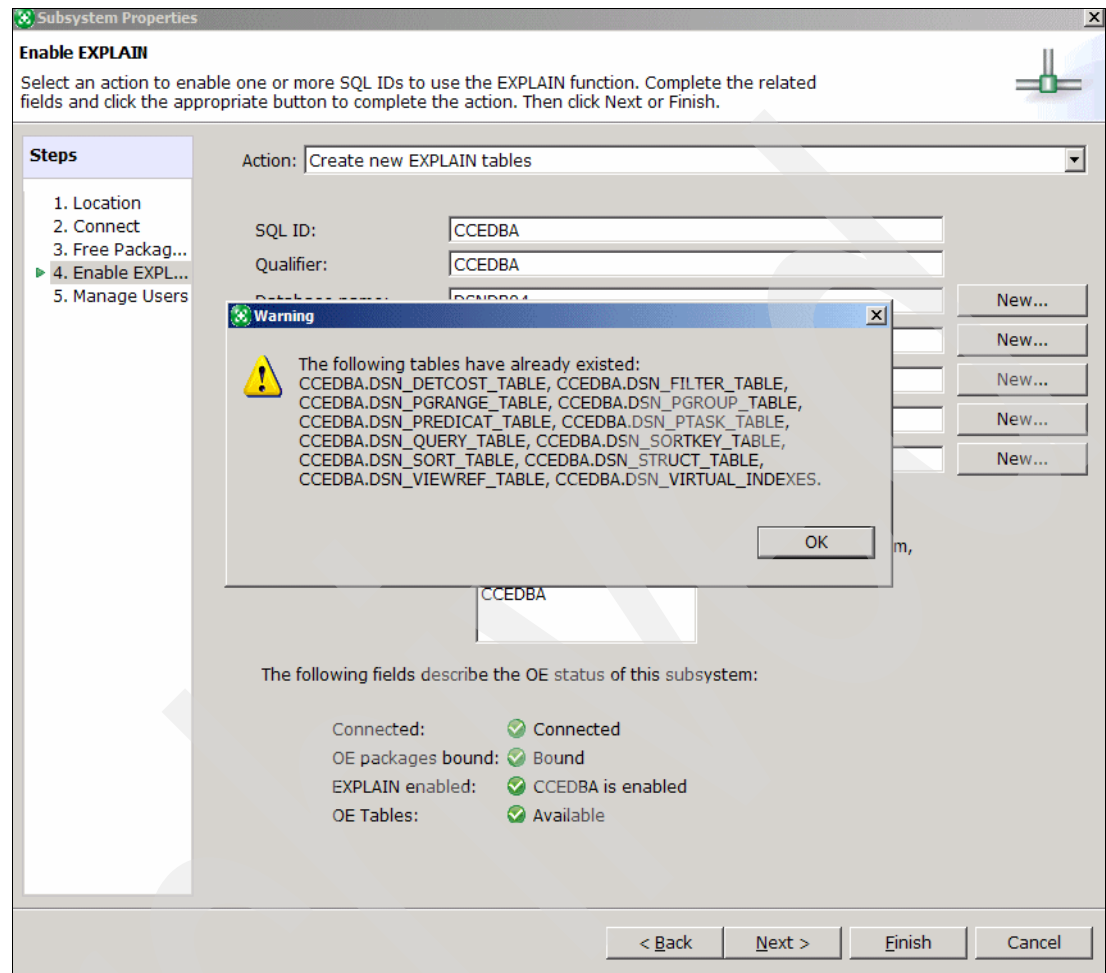


Figure 5-15 Connect - existing EXPLAIN tables

8. When all EXPLAIN tables have been created, you see a window similar to Figure 5-16. Click **Next**.

**Subsystem Properties**

**Enable EXPLAIN**

Select an action to enable one or more SQL IDs to use the EXPLAIN function. Complete the related fields and click the appropriate button to complete the action. Then click Next or Finish.

**Steps**

1. Location
2. Connect
3. Free Packag...
4. Enable EXPL...
5. Manage Users

Action: Create new EXPLAIN tables

SQL ID: CCEDBA

Qualifier: CCEDBA

Database name: DSND804 New...

4 KB table space: DSNSUMTS New...

8 KB table space: FXNTS New...

32 KB lob table space: LOBTS1 New...

32 KB lob table space: LOBTS2 New...

Create Tables

The following authorization IDs can use the EXPLAIN function on this subsystem, because the necessary EXPLAIN tables exist and they have access to them:

CCEDBA

The following fields describe the OE status of this subsystem:

Connected: ☒ Connected

OE packages bound: ☒ Bound

EXPLAIN enabled: ☒ CCEDBA is enabled

OE Tables: ☒ Available

< Back Next > Finish Cancel

Figure 5-16 Connect - EXPLAIN tables created window

### 5.1.5 Create user tables for workload information

The last step is to manage users as seen in Figure 5-17:

1. Provide users with the correct access and click **Finish**.

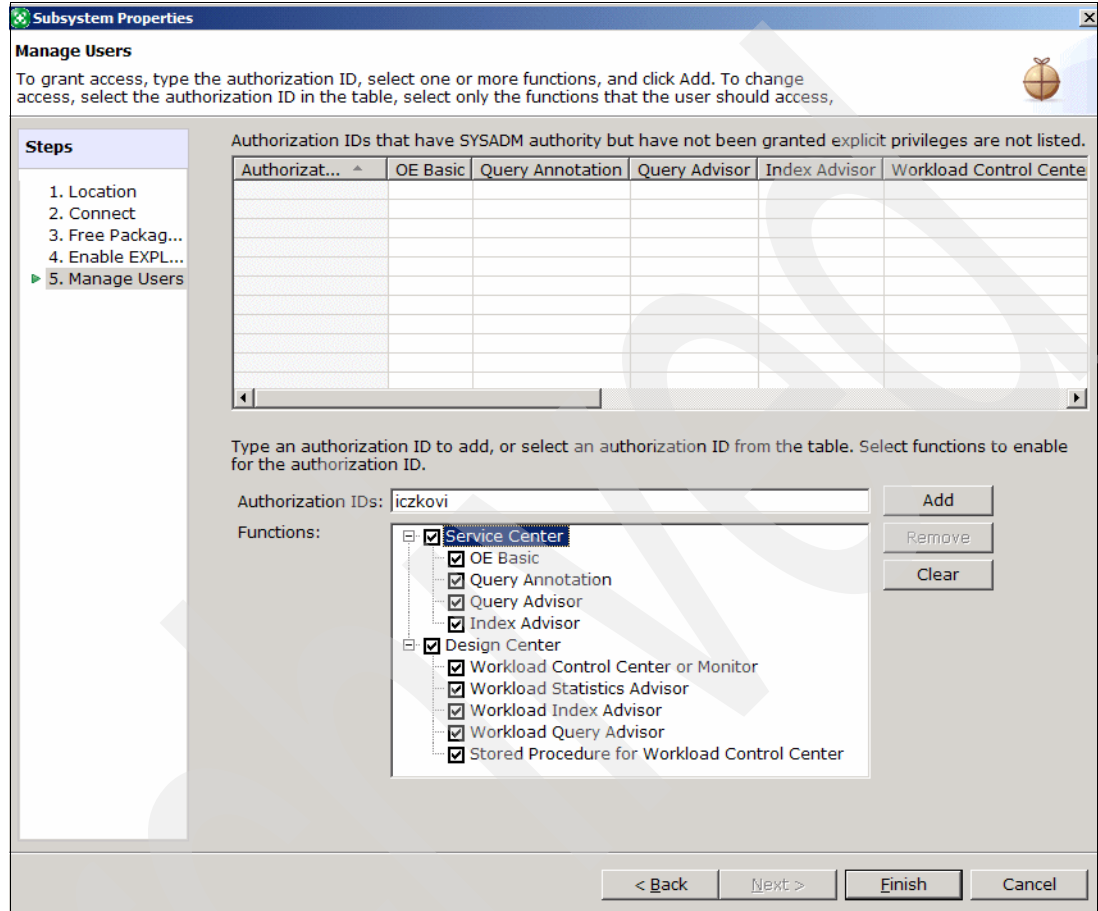


Figure 5-17 Connect - Manage Users window

- Upon successful completion, you are brought back to the Connect to Subsystem window with your system in enabled mode. You can view the users and access from that window by clicking the **Manage Users** tab. In our scenario, Figure 5-18 is the window that was displayed.

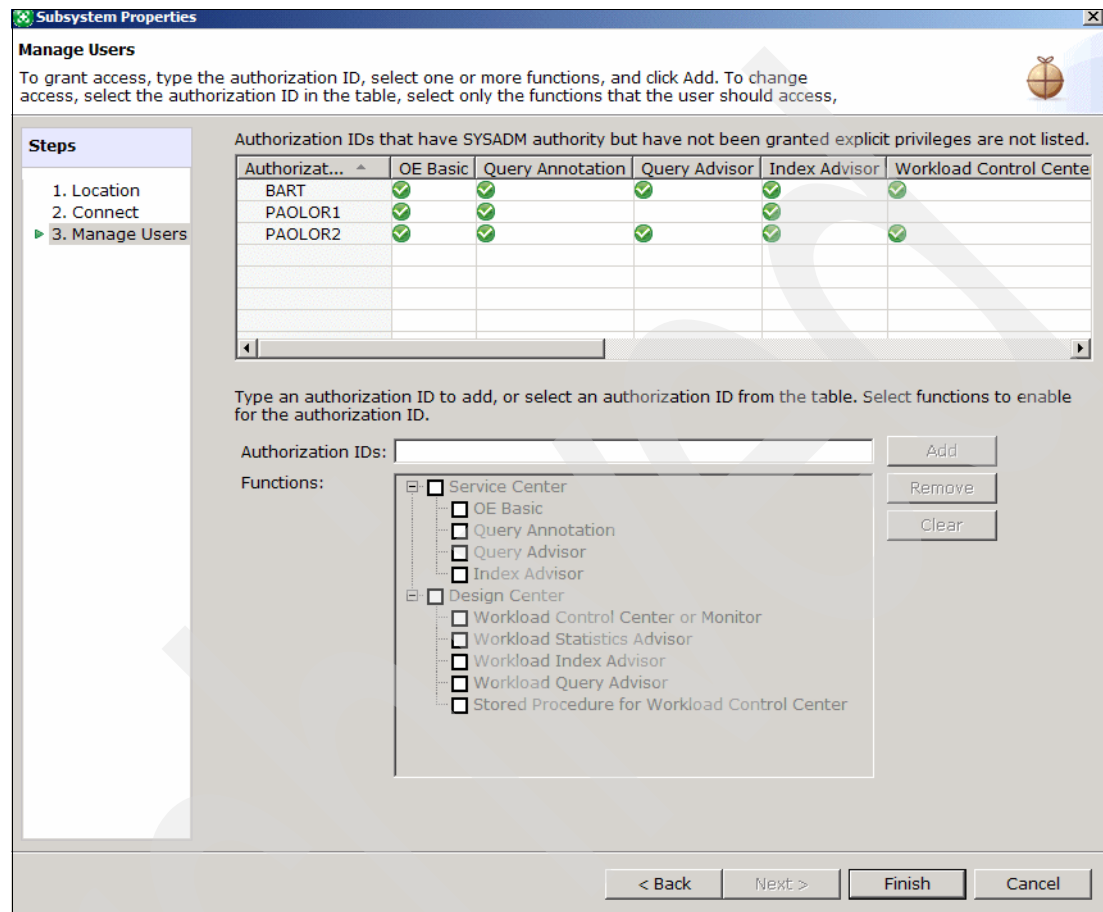


Figure 5-18 Manage Users window

## 5.2 Security considerations

This section describes a security scenario. Security levels and responsibilities vary in each environment. We show a sample configuration that takes into account the current security levels required by the tools. See also Appendix C.1, “Security requirements for Optimization Service Center or Optimization Expert” on page 410.

Customer security requirements vary widely. There are many different scenario that require consideration, such as what happens when a new fix pack version of Optimization Service Center or Optimization Expert becomes available? Are the DBAs allowed to directly download the new version to their workstations, or does a specific administrator approve and control the download process? Along with this comes other authorization issues, such as when an employee downloads a new version of Optimization Service Center or Optimization Expert, do they have the privileges to BIND the new packages to use the new version of the product? Other issues to consider include create and use authority for the explain tables. Can a user create or use their own tables, or use a common approved schema name? You must resolve these issues prior to using the products.





## Part 3

# Critical query detection

This part describes the setup for detecting and investigation trouble queries, both as individual ones or as part of a query workload. This part contains the following chapters:

- ▶ Chapter 6, “Profile monitoring” on page 131
- ▶ Chapter 7, “Workloads” on page 149

Archived

## Profile monitoring

The DB2 V9 execution “engine” contains a built-in SQL-level monitor called the *profile monitor*.

You can use the profile monitor to discover:

- ▶ Your normal execution profile; that is, which queries run in your environment, how often each runs, and what each costs in terms of CPU and elapsed time.
- ▶ Problem queries, specifically those with high CPUs.

This chapter shows how to easily isolate individual queries for analysis and tuning by the tools and advisors provided in the Optimization Expert.

Through the profile monitor, multiple queries are grouped into workloads and for index and statistics analysis. The recommendation results, when implemented, improve overall performance.

The profile monitor builds on and extends the capabilities provided for dynamic SQL previously in the dynamic statement cache (DSC) and adds support for static SQL.

This chapter contains the following sections:

- ▶ “Profile monitor capabilities” on page 132
- ▶ “Operating the profile monitor” on page 140
- ▶ “Profile monitor DB2 tables” on page 145

## 6.1 Profile monitor capabilities

Figure 6-1 illustrates the architecture of the profile monitor. The following sections describe the concepts and terms.

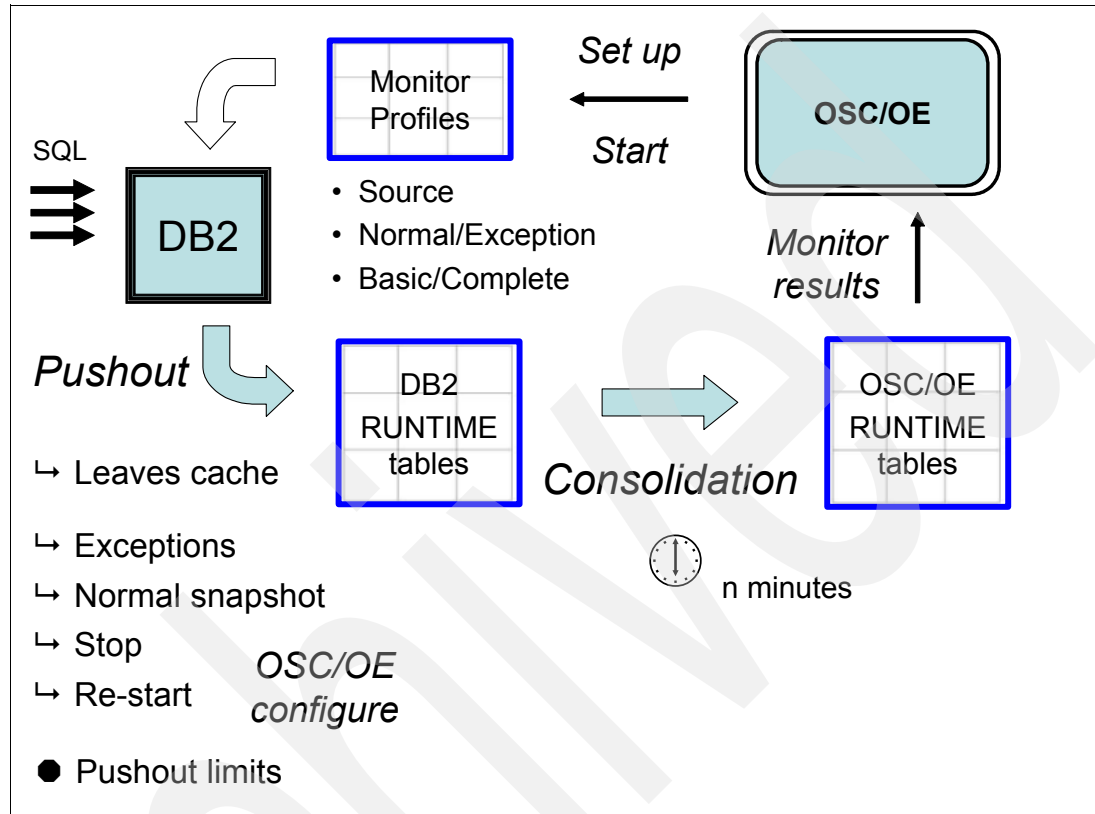


Figure 6-1 Profile monitor architecture

### 6.1.1 Monitor profiles

DB2 monitors (collect performance statistics for) SQL that runs in a *monitor profile*, a new DB2 construct utilized by Optimization Service Center or Optimization Expert with parameters that you set up with:

- ▶ A descriptive name (unique within a subsystem)
- ▶ One or more monitoring sources, for example, a list of PLANs
- ▶ A selection of monitor options and parameters

#### Monitor source

You can specify a monitor source by using any of the following scenarios:

##### All SQL that runs in a PLAN

Specify the PLAN name. For example, specify DISTSERV to monitor all distributed SQL.

##### All SQL that runs in a PLAN with a specified COLLID and PKNAME

Specify all three parameters.

### All SQL run by an AUTHID operating from a specific IP address

Specify both parameters.

### All SQL run by an AUTHID operating from the local subsystem

Use IP address 127.0.0.1 as a pseudo to mean the local DB2.

**Restriction:** Other scenarios are *not* supported. For example, you *cannot* monitor:

- ▶ All SQL from a COLLID; that is, without specifying the PLAN and the PKNAME.
- ▶ All SQL from an AUTHID running under a PLAN.
- ▶ Wild-carded PLAN names.

## SQL

An SQL statement is either:

- ▶ An individual SELECT INSERT UPDATE DELETE, or
- ▶ An entire cursor result set, that is, a single OPEN + all FETCHes + CLOSE considered as a single execution

You can monitor both dynamic and static SQL.

Statements are considered identical based on their unique SQL text.

## Cache

The term “cache” refers to the runtime structure that represents the execution of an individual SQL. When operating within the context of a monitor source, it contains monitoring information (at least #executions, CPU, elapsed time).

There are two caches, depending on the type of SQL being run:

- ▶ For dynamic SQL: The familiar dynamic statement cache
- ▶ For static SQL: New to V9, housed in the EDM pool

## Monitoring type: Normal or exception

There are two types of monitor:

### Normal

Aggregates all executions within a monitor source. Use it to get an idea of what SQL is being executed in your system. A normal monitor produces totals over all executions of a given SQL.

### Exception

Records individual executions that exceeds a CPU threshold. There are two thresholds, either or both can apply:

- Absolute: CPU greater than a specified threshold (in seconds).
- Spike: CPU greater or lower than the average CPU consumption for that SQL by a specified percentage. For a spike to be recognized, at least 10 executions must have occurred to form a meaningful average.

For a given monitor source (for instance, a PLAN) you can define both a normal and an exception monitor. This way, you can obtain totals and also collect individual CPU exceptions.

**Note:** The monitor source “PLAN” does not conflict with “PLAN COLLID PKNAME”.

## Monitoring granularity: Basic or complete

There are two levels of monitoring provided, a trade-off of the benefit provided by more complete information with the overhead needed to gather it. They are explained in more detail in section 13.3, “Defining a monitor profile” on page 344.

### Basic

The most important information - with only 1% overhead:

- # executions
- CPU time
- Elapsed time

### Complete

Much more information - but with 10% overhead:

- # executions
- CPU time
- Elapsed time
- Breakdown of wait time (Class 3)
- GETPAGES and other buffer pool data
- Internal values not externalized except with PERFM traces, for example, the number of rows that DB2 examined.

**Restriction:** Optimization Expert currently does not display the “Complete” information. The data is available only in the Optimization Service Center WCC tables. Do *not* use “Complete” unless you plan to read the Optimization Service Center WCC tables directly.

See section 6.3.2, “DB2 tables containing monitoring results” on page 146.

## Enabled or disabled

A monitor profile can have two statuses:

- ▶ **Enabled:** Eligible to be started. A subsystem monitor start serves to start all enabled profiles.
- ▶ **Disabled:** Not currently eligible to be started.

You can toggle between a given monitor profile, enabled or disabled at any time. This allows you to define a repertoire of monitor profiles to represent different workloads and selectively turn on only the ones you want to operate at a given time.

## Scheduling

A monitor profile can:

- ▶ Start immediately when first set up.
- ▶ Start manually at any point.
- ▶ Start and stop automatically at a later, designated time.

See Figure 6.1.4 on page 137 for scheduling considerations.

## Subsystem-wide monitor

From the DB2 subsystem point of view, there is a single monitor that operates globally within that subsystem (stand-alone or data sharing member). Optimization Expert groups all profile monitors and operates them together as the subsystem monitor.

This subsystem-wide monitor has parameters that control its activity. Its status is either ON (started, monitoring all enabled monitor profiles), or OFF. A DB2 command displays the status of the subsystem-wide monitor:

-DIS PROFILE

```
DSNT753I  -DB9A DSNT1DSP DISPLAY PROFILE REPORT FOLLOWS:
          STATUS = ON
          TIMESTAMP = 2007-07-25-21.21.57.407471
          PUSHOUTS = 3578 OUT OF 10000
          DISPLAY PROFILE REPORT COMPLETE.
DSN9022I  -DB9A DSNT1DSP 'DISPLAY PROFILE' NORMAL COMPLETION
```

### 6.1.2 Monitor pushout

The monitoring data is kept internally in DB2 storage (cache). It is written to DB2 tables that are consolidated to form the results, which Optimization Expert displays in its panels. Table 6-1 shows the three processes involved.

Table 6-1 Externalizing monitor data

Process	Actor	Process
<b>Pushout</b>	DB2 for z/OS internal	Inserts monitor data to the DB2 "RUNTIME" tables, according to pushout criteria (different for normal and exception monitors, see below).
<b>Snapshot™</b>	DB2 for z/OS, Optimization Service Center or Optimization Expert request	Externalizes, on demand, monitor data for normal monitors.
<b>Consolidation</b>	Optimization Service Center or Optimization Expert	Combines recent RUNTIME results with data previously recorded in its (WCC) tables.

Pushout and snapshot are described in section 6.1.3, "Consolidation" on page 137.

Because inserting rows (the only way monitor data is recorded) can itself become an overhead issue, a number of controls and limits (shown in *italic bold* below) have been established to cap this overhead and prevent runaway situations. You can choose the defaults established for these, or override them to obtain more or less information.

#### Exception monitors - pushout limits

There are two limits for exception monitors: per statement and overall.

##### *Limit for individual statement push*

For any particular statement, the monitor pushes out individual exceptions up to this limit. The default, 1, means that only the first exception for a given SQL is externalized and later

exceptions are discarded. Choosing 5 means that the first 5 exceptions for the same SQL are recorded, and so on.

### ***Limit for all statement pushes***

Provides a limit for ALL individual statement pushes for this monitor profile. A default limit of 200 means no more than 200 different SQL if the “Limit for individual statement push” is left at 1. Any exceptions after that limit are discarded (whatever SQL statement they apply to).

### **Normal monitors - pushout**

For a normal monitor, monitoring data is retained in DB2’s internal cache (one slot per executing statement) until one of the following happens:

- ▶ The statement leaves the cache, because it has not been executed recently and its cache slot is about to be reused. This happens unpredictably, and for SQL that gets executed over and over again, might never happen.
- ▶ When you restart the subsystem-wide monitor using the **Monitoring → Start Monitoring** function and the monitor profile, which was enabled previously, is now disabled (or deleted).
- ▶ When you choose **Monitoring → Stop Monitoring**.

Each individual normal monitor profile is capped as to the number of statements it can push out according to the following control:

### ***Limit for statement pushes (that leave the cache)***

By default, normal monitors record only 5000 individual statements. Any additional statements beyond this limit (inclusion criteria undetermined) are discarded.

**Note:** Consider reducing this pushout limit if Optimization Expert performance suffers.

### **Snapshot**

To capture the aggregated results of a normal monitor at any time, take a snapshot.

A snapshot takes the statistics collected in the cache for statements under the specified scope. It might include statements monitored for exception also, if they are under the scope. The same scope can cover both normal and exception monitor.

**Note:** Exception monitors do not require snapshots because their data, representing individual executions, is pushed out immediately, subject to pushout limits.

After a snapshot, the Optimization Expert monitor displays the totals representing all statements (within scope) executed since you started the monitor profile. If you do not take snapshots, you do not see the results until the monitor ends, or unpredictably as statements leave the cache.

**Note:** There is no restriction on the number of statements that are externalized by a snapshot. That is, none of the pushout limits apply.

### **Global pushout limit**

A global limit applies to a single execution of the subsystem-wide monitor, which caps the number of pushouts considered over all enabled monitor profiles.



### ***Limits for all monitor statement pushes - all monitor profiles***

By default, a single start of the subsystem monitor is limited to 10000 statements. Normal and exception monitors are considered together in this limit. Subsequent statements are not recorded until the monitor is restarted.

### **Restarting monitors to enable more recording**

The global pushout limit applies to a single start of the subsystem monitor. When the subsystem monitor is restarted, all counters are reset to 0, thereby enabling a new set of measurements. You can initiate restart through any of the following means:

- ▶ The **Monitoring** → **Start Monitoring** function from the monitor profiles list (starts all enabled monitors).
- ▶ The **Start Now** option when setting up a new monitor profile (restarts the subsystem-wide monitor).
- ▶ By having Optimization Service Center or Optimization Expert check on the progress of the subsystem monitor periodically and automate a restart before the limit of all monitor statement pushes is reached. The following threshold applies:

#### ***Restart all monitor profiles***

Set a threshold, for example 80%, of the “Limits for all monitor statement pushes - all monitor profiles”, which, when reached, triggers an automated monitor restart.

**Note:** We suggest the user to examine and analyze the report first. After analysis, the user is required to prune the report tables to prevent table space overflow.

## **6.1.3 Consolidation**

The information pushed out by DB2 is not available until it is consolidated, that is, combined into the Optimization Expert WCC tables to present a seamless record of SQL activity. It is your choice how often you want to do this and how this is to be done:

### ***Interval for consolidating statement pushes***

By default, data is consolidated every 30 minutes. That means, you might not see results until up to 30 minutes later. Reduce this interval if you want to see data refreshed more regularly.

There is only one consolidation process that applies to all monitor profiles together. When you change this value in a new monitor profile that you start immediately, it applies to all monitor profiles already in effect.

**Note:** You cannot see when a consolidation is scheduled next. However, you can see the history of previous consolidations in the workload history.

## **6.1.4 Scheduling: Client-side or administrative (server-side)**

During the time period when monitoring is active (see section 6.1.3, “Consolidation” on page 137), or to fulfill any workload scheduled function (such as a cache capture) that you request, Optimization Expert must schedule the processing to run at a future time. You can choose either your own PC (“client-side”) to execute the time-based event, or the z/OS Administrative Scheduler (“server-side”). See Figure 6-2 on page 138.

Figure 6-2 Client or Administrative Scheduling

Client-side scheduling means that your PC runs during the time when you are monitoring, so Optimization Service Center or Optimization Expert can run the consolidation directly from that PC.

**Recommendation:** For any long-running monitoring, we recommend that you use the server-side scheduler.

To enable the recommended “server” option (greyed out in Figure 6-2), make sure the “Use Java Stored Procedure for Workload Control Center” box is checked off (Figure 6-3).

Figure 6-3 Enable the Administrative Scheduler

Two separate prerequisites are needed:

- ▶ Install the WCC JAVA procedures. Make sure they are operational.
- ▶ Install and operate the Administrative Scheduler, a started task. See Appendix B, “Administrative Scheduler” on page 405.

**Note:** If the consolidation cannot run, for example, because client-side scheduling was chosen and the client PC was turned off, then monitoring will not operate normally.

### 6.1.5 Compared to other monitoring solutions

You need to assess the DB2 V9 profile monitor with other monitoring mechanisms already in place.

#### Compared to IFI ACCTG trace (as reported by DB2PM)

ACCTG data is reported on a thread and package level, aggregating information for all the SQL statements that run. It is often difficult to determine which SQL is causing a performance issue. The profile monitor tracks the performance of individual SQL statements, directly identifying the appropriate targets for analysis and correction.

### Compared to dynamic statement cache (IFCID 316)

For dynamic SQL, the profile monitor uses the same DB2 mechanisms previously implemented for the dynamic statement cache, that is, the IFCID 318 and related 317 and 316 trace to start/stop monitoring and results saved to the following table:

SYSIBM.DSN\_STATEMENT\_CACHE\_TABLE

Monitoring overhead depends on the granularity of data collection and reporting. For basic monitoring, the overhead is very small (< 1%), as already described. For complete monitoring, the overhead is a bit higher than IFCID 0318 because the monitor process collects more statistics than IFCID 0318, about < 10% as mentioned earlier.

**Restriction:** IFCID 0318 interferes with profile monitoring so do *not* start it if you run profile monitoring.

If IFCID 0318 is active when profile monitoring starts up:

- ▶ There is no indication of problems.
- ▶ Static SQL is monitored correctly.
- ▶ Dynamic SQL is monitored, but the statistics include the data collected since IFCID 0318 started.
- ▶ If IFCID 0318 is started after profile monitoring started, then a few collected statistics are lost because the start of IFCID 0318 resets the collected statistics to zero again. In this case, a warning message is generated as part of the report.

The advantages in using profile monitoring over dynamic statement cache performance statistics are:

- ▶ Can use basic monitoring, less expensive than a complete monitor, if only the # executions, CPU, and elapsed time are of interest.
- ▶ Can limit overhead to a specified source scenario, for example, an AUTHID accessing the system from a specified IP address.
- ▶ Have the advantages of the Optimization Expert Workload framework, especially for advisors.
- ▶ Can monitor static statements, in addition to dynamic statements.
- ▶ Can monitor a specific exception and generate a report at that moment.

A reason to choose dynamic statement cache performance statistics:

- ▶ Can collect all dynamic statements, not limited to the specific monitoring scenarios supported by the workload scenarios.

### Compared to IBM Query Monitor (or equivalent product)

An external tool, such as IBM Query Monitor, provides additional monitoring scenarios and more complete information.

## 6.1.6 Data sharing limitation

The monitor has local scope only. It operates only on the subsystem it was connected to when you selected **Start Monitor**.

**Note:** To run on a different member, shut down monitoring completely on the previously active member before attempting to start the monitor on another member.

### 6.1.7 Restarting monitoring after a DB2 restart

DB2 subsystems come up with no monitoring active whether or not monitoring was active before the subsystem was shut down.

When STOP DB2 is in progress, there is no implicit STOP PROFILE issued for push out and all that is cached will be lost. You need to stop monitoring with OSC before you stop DB2 to make sure the data is not lost.

**Note:** After IPL or DB2 subsystem restart, you must *restart* all monitors by using the Optimization Expert panels.

## 6.2 Operating the profile monitor

The Optimization Expert panels guide you through the process of defining monitors and examining monitor results. Before you start, make sure you understand the terms and facilities described in section 6.1, “Profile monitor capabilities” on page 132.

### 6.2.1 Viewing monitors

To access any profile monitor function, use **View Monitors**, which is available:

- ▶ As part of the “Welcome” project in the Project Navigator
- ▶ As one of the left most top tabs when operating in any project

Figure 6-4 on page 141 shows the View Monitors panel.

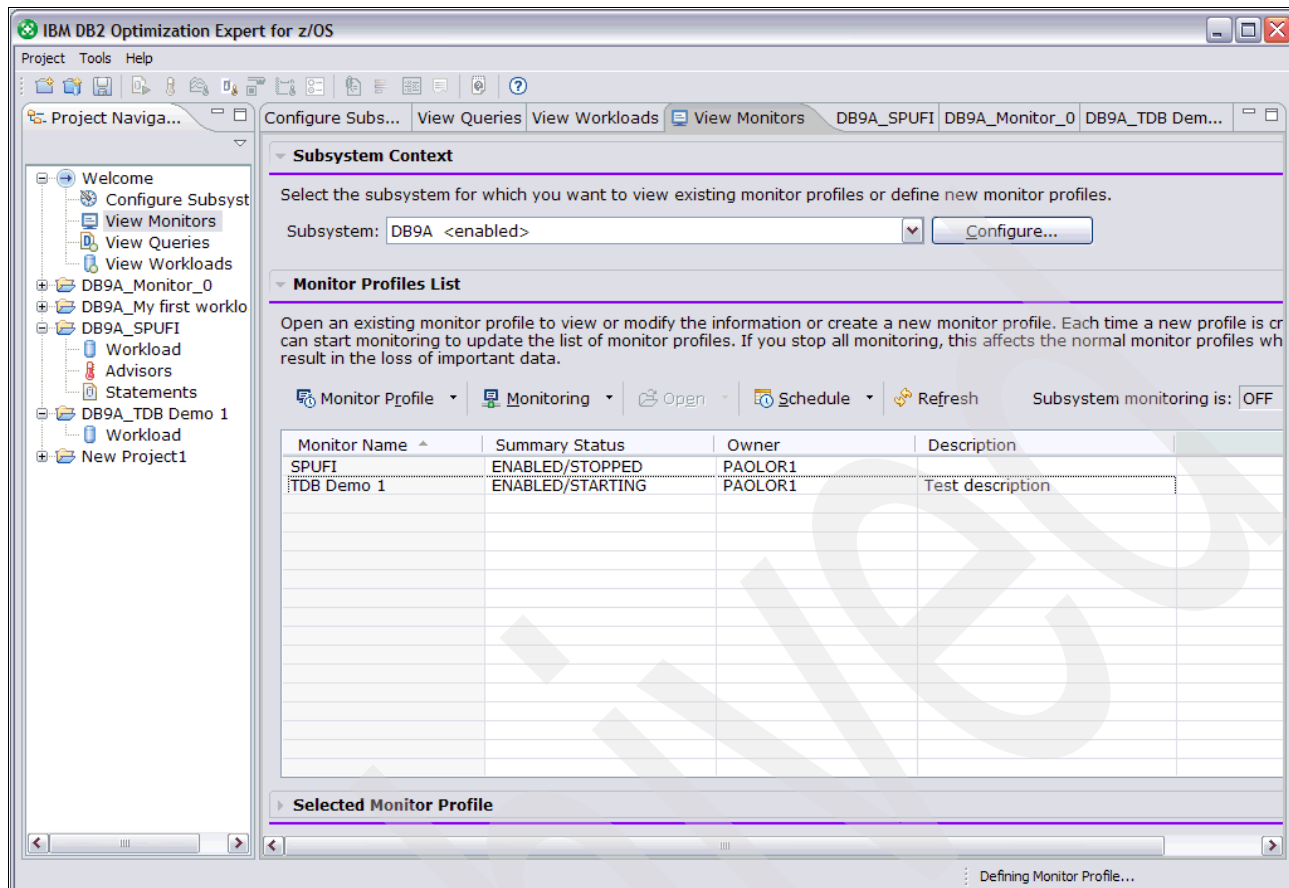


Figure 6-4 View monitors panel

This panel has two main functions:

- ▶ It displays the status of the monitoring currently in effect.
- ▶ It houses controls that allow you to:
  - Set up a new monitor profile, and optionally start it.
  - Start or stop all monitor profiles.
  - Snapshot normal profiles.
  - Display the statements gathered by a profile monitor.

**Note:** This panel does not refresh itself automatically. Use refresh frequently to update the list.

## 6.2.2 Displaying monitoring options currently in effect

The subsystem status on the top right lets you know that the subsystem-wide monitor is active (see Figure 6-5).

Subsystem monitoring is: **ON**

Figure 6-5 Subsystem monitoring

Each defined monitor profile is listed, along with its status:

- ▶ **ENABLED:** Available to be started when the subsystem monitor is started. You can disable it by using **Monitor Profile** → **Disable**.
- ▶ **DISABLED:** Defined, but not currently eligible to be started. You can enable it by using **Monitor Profile** → **Enable**.
- ▶ **START PENDING:** Defined, available to be started.
- ▶ **STARTED**
- ▶ **STOPPED**

All monitors are shown in the list, not only the ones you set up as **OWNER**.

### 6.2.3 Setting up a new profile monitor

The **Monitor Profile** → **New** action leads you through four windows (steps) that gather all monitor setup information. Complete each step before you proceed to the next step. At any point, you can go back to revisit previous steps. You cannot traverse by clicking on the **Steps** tab on the left.

**Note:** For information about the items that need to be supplied, see section 6.1, “Profile monitor capabilities” on page 132.

The steps are:

1. **Monitor Name:** Identify the monitor profile. Choose a name that can easily select from a list of all monitor names, which have global scope (different **OWNERS** cannot define the same monitor name). At this point, specify **Normal** or **Exception**.
2. **Monitor Source:** Supply one or more of the supported Monitor Source combinations, Dynamic or Embedded, adding each one individually to the list. You can also remove any previously added monitor sources.
3. **Filter:** This window prompts for thresholds and limits. These are different for Normal and Exception monitors.
4. **Start Monitor:** If you choose to **Start Now** or **Schedule**, your “Interval for consolidating statement” overrides the consolidation schedule currently in effect. Otherwise, the monitor goes into **START PENDING** status and you can start it later.

### 6.2.4 Starting all monitor profiles

The action of **Monitoring** → **Start Monitoring** starts all **ENABLED** monitor profiles.

**Note:** If you highlight a monitor profile, Start Monitoring still starts the group. Highlighting is ignored in this context.

You are prompted for a number of Monitor Options that apply to all monitor profiles in the subsystem (Figure 6-6).

Maximum number of exception pushes for an individual statement:	<input type="text" value="1"/>
Interval for consolidating statement pushes:	<input type="text" value="30"/>
Limits for monitor statement pushes:	
Normal monitor profiles:	<input type="text" value="5000"/>
Exception monitor profiles:	<input type="text" value="200"/>
All monitor profiles:	<input type="text" value="10000"/>

Figure 6-6 Monitor options

The parameters pointed to by the red arrows apply to the subsystem-wide monitor, the ones you are about to start. For more information, see section 6.1, “Profile monitor capabilities” on page 132.

Press **Start Options**. After monitoring starts, all “Enabled monitored profiles” are **STARTED**.

**Note:** Startup might not be instantaneous. Individual monitor profiles might go into **START PENDING** status. Click **Refresh** to update the status.

## 6.2.5 Monitor results


Monitor results are available by highlighting a monitor profile and clicking **Open** or **Open → Statements**.


There is one line for each SQL statement; that is, each unique SQL text string. The results are present in two tabs, **All** and **Monitor**.


Figure 6-7 shows sample monitor results. Columns have been removed (by dragging column separators) to show the main fields of interest.


▼ Workload Statements


Immediately capture statements or multiple sources to this workload, launch workload advisors, use tools to tune selected queries for schedule tasks for capture, consolidation, and analysis.


 Capture

 Workload Tools

 Schedule

 Remove

 Query Tools

 Refresh

All of the rows are displayed. The number of rows is 201.

AllMonitor

Execution Count	Source	Accumulated Elapse...	Accumulated CPU Ti...	Statement Text
1	MONITOR	150.93378	6.911325	SELECT lbrempln,lbritems,lbruprod,dd.itlwkce
1	MONITOR	147.9958	4.476378	SELECT lbrempln,lbritems,lbruprod,dd.itlwkce
1	MONITOR	145.87912	6.458591	SELECT lbrempln,lbritems,lbruprod,dd.itlwkce
1	MONITOR	142.50217	4.2139993	SELECT lbrempln,lbritems,lbruprod,dd.itlwkce
1	MONITOR	141.94307	4.3739142	SELECT lbrempln,lbritems,lbruprod,dd.itlwkce
1	MONITOR	138.19733	3.9911835	SELECT lbrempln,lbritems,lbruprod,dd.itlwkce
1	MONITOR	136.64264	4.072684	SELECT lbrempln,lbritems,lbruprod,dd.itlwkce
1	MONITOR	133.56398	3.9617426	SELECT lbrempln,lbritems,lbruprod,dd.itlwkce
1	MONITOR	132.78601	3.4327948	SELECT lbrempln,lbritems,lbruprod,dd.itlwkce

Figure 6-7 Sample monitor results

Table 6-2 shows the complete set of values provided under both tabs.

Table 6-2 Monitor results

Header and value	All tab	Monitor tab
Execution count	YES	YES
IP address	No	YES (Note 1)
Authorization ID	No	YES (Note 1)
Plan name	No	YES (Note 1)
Collection ID	No	YES (Note 1)
Package Name	No	YES (Note 1)
Total Elapsed	YES	YES
Average Elapsed	YES	YES
Total CPU	YES	YES
Average CPU	YES	YES
Exception count	No	YES (Note 2)
Statement Text	YES	YES
Note 1 - Appears only if selected as a monitor criterion. For example, AUTHID is shown only if AUTHID was a criterion. Note 2 - Valid only for Exception monitors.		

The order of the rows is initially undefined. Click any header to order the results in that order.

**Note:** Click twice on a header to view by descending CPU or elapsed.

Double-click any line to view the SQL text in a separate box. From this box, you can save the SQL or categorize it.

**Restriction:** You *cannot* save the monitor results as a group. You save individual SQL statements, one by one.

## 6.2.6 Monitoring workload projects

Each monitor profile creates a workload project, given the name `ssss_monitor`, where `ssss` is the name of the subsystem prefixing the monitor name.

The workload project shows the totals captured by the monitor (Figure 6-8).

Name ^	Summary Status	Execution Time
Monitor_0_CCE	ENABLED/STARTED	CPU time: 12459.82 seconds; Elapsed time: 142663.25 sec

Figure 6-8 Workload totals

You can view monitor results equivalently from:

- ▶ View Workloads, which brings up the Workload List.
- ▶ View Monitors, which brings up the list of monitors.



- ▶ Clicking directly within the Project Navigator.

The Open button operates similarly in both Workload and Monitor contexts, taking you to the statements (default), or to any other part of the project (choose it from the pull-down).

Controls provided within the Workload Project allow you to:

- ▶ Run workload advisors on the entire set of statements captured by the monitor.
- ▶ Within the workload statements, subset the SQL into smaller workloads for further analysis. Double-click the SQL to bring up the View SQL Statement box and Categorize (provide a keyword) that SQL. That category is the basis for a new workload.
- ▶ View the history of a monitor. The list shows you creation and all consolidations.

## 6.2.7 Why are you not seeing all the results?

For monitoring data to appear:

- ▶ DB2 must have pushed out the statement, see section 6.1.2, “Monitor pushout” on page 135. For a normal monitor, or to stop the profile to force the monitoring results to be reported, you need to run a snapshot. Use **Schedule** → **Snapshot** from the View Monitors window. You can specify any or all normal monitors when requesting a snapshot.
- ▶ Optimization Expert must have consolidated it, see section 6.1.3, “Consolidation” on page 137. Reduce your consolidation interval to run consolidation more often. Make sure you choose the right scheduling option (Figure 6.1.4 on page 137)
- ▶ The limits by monitor profile and the global monitor limit must not have exceeded. If they have been, monitor data is discarded. You might want to raise these to gather more information.

## 6.2.8 Important considerations

Important considerations are:

- ▶ It is not possible to clear the values associated with a monitor profile to collect a fresh series of measurements, other than by deleting and redefining the monitor profile.
- ▶ When you delete the monitor, the workload project stays around as a dangling orphan, attaching itself (that is, volens or nolens) to a different workload.

**Recommendation:** After deleting a monitor, make sure you delete the corresponding workload project. These are named, by default, as “ssss\_workload-name”.

- ▶ When you delete a monitor, all monitored statements (rows in DB2 tables) are deleted. This might take some time.
- ▶ It is not possible to collect a time series of different executions of a monitor source because Optimization Expert does not allow the same monitor source to be duplicated; that is, “Normal with another Normal” or “Exception with another Exception”.

## 6.3 Profile monitor DB2 tables

Workload center (WCC) tables and the SYSIBM.DSN% tables related to profile monitoring are described in Appendix E, *DB2 Version 9.1 for z/OS Performance Monitoring and Tuning Guide*, SC18-9851.

**Note:** Do not alter the contents of these tables. The contents of these tables can change without notice.

### 6.3.1 Profile monitor definitions

You can view the status of the individual profile monitors by querying the underlying tables (Example 6-1)

*Example 6-1 Profile monitor definitions*

---

```
SELECT
  W.WLID
  ,W.NAME
  ,W.OWNER
  ,S.LASTUPDATETS
  ,CASE
    WHEN (MONITOR_STATUS = 1) THEN 'START PENDING'
    WHEN (MONITOR_STATUS = 2) THEN 'STARTING'
    WHEN (MONITOR_STATUS = 3) THEN 'STARTED'
    WHEN (MONITOR_STATUS = 4) THEN 'TO BE STOPPED'
    WHEN (MONITOR_STATUS = 5) THEN 'STOPPING'
    WHEN (MONITOR_STATUS = 6) THEN 'STOPPED'
    ELSE      'STATUS ' || DIGITS(MONITOR_STATUS)
  END AS MONITOR_STATUS
  ,W.DESCRPTION
FROM   DB2OSC.DSN_WCC_WORKLOADS AS W
JOIN   DB2OSC.DSN_WCC_WL_SOURCES AS S
ON (W.WLID = S.WLID)
WHERE (TYPE = 9)      -- MONITOR
;
```

---

### 6.3.2 DB2 tables containing monitoring results

Complete monitoring gathers more values, but currently the Optimization Expert panels do not display them. Example 6-2 provides a sample SELECT for extracting complete monitoring results.

*Example 6-2 SQL to fetch Optimization Expert monitoring results*

---

```
SELECT
  INSTS.PLANNAME
  ,INSTS.COLLID
  ,INSTS.PKGNAME
  ,INSTS.VERSION
  ,INSTS.SECTNOI
  ,INSTS.LAST_UPDATE_TS
  ,RUNTM.*
  ,TEXTS.STMT_TEXT
FROM   DB2OSC.DSN_WCC_STMT_RUNTM AS RUNTM
JOIN   DB2OSC.DSN_WCC_STMT_INSTS AS INSTS
ON (RUNTM.INSTID = INSTS.INSTID)
JOIN   DB2OSC.DSN_WCC_STMT_TEXTS AS TEXTS
ON (INSTS.STMT_TEXT_ID = TEXTS.STMT_TEXT_ID)
```

WHERE (criteria on package, plan, time period, you name it)

---

See Table 6-2 on page 144 for a list of all the fields provided.

Archived

Archived

## Workloads

One of the advantages of Optimization Expert is allowing you to collect and operate on collections of SQL statements, not only single SQL queries that are analyzed one at a time.

These collections of SQL are called *workloads*. They can represent:

- ▶ Your actual operational SQL, as a snapshot from the profile monitor (static or dynamic SQL), or from the dynamic statement cache, along with their performance data.
- ▶ Static SQL text extracted from the DB2 catalog, which you can subset using various filters.
- ▶ The SQL in Query Management Facility (QMF) or QMF High Performance Option (HPO) queries or procedures.
- ▶ SQL as input from a file (created from any source).

Each workload is represented as a *workload project* within Optimization Expert and is managed using the Eclipse Framework.

This chapter first describes how to define workloads and how to do the analysis.

## 7.1 Defining a new workload

Before launching into the panel flows, it is helpful to understand what is available and the terminology that is applied to workloads.

### 7.1.1 Workload sources

There are several possible *sources* for workload SQL. Most are qualified by *source filters*. Each is associated with a *capture task* that collects the SQL into the workload.

#### Profile monitor

The profile monitor collects execution statistics for statements that run in DB2 based on parameters, such as monitor source (for example, PLAN) and time period. See Chapter 6, “Profile monitoring” on page 131 for details.

Real performance information is fundamental in directing attention to the SQL that actually executes, and especially to the CPU-intensive SQL. These are only a small subset of the total, so it is important to identify costly statements. The Workload Analysis tools use performance data to prioritize recommendations to suit the most important SQL in a workload.

#### Dynamic statement cache

The dynamic statement cache captures all dynamic SQL, and if IFCID 0318 is enabled, performance data as well.

You can take slices (snaps) of dynamic SQL according to filters:

- ▶ AUTHID or CURSQLID (that did the initial PREPARE).
- ▶ BIND options for the statement.
- ▶ Any of the STAT columns, for example, statements that ran longer than 1 second.

All filters qualify columns in DSN\_STATEMENT\_CACHE\_TABLE. See *DB2 Version 9.1 for z/OS Performance Monitoring and Tuning Guide*, SC18-9851, Appendix E.

You can capture information once, several times, or at intervals. The options are:

- ▶ Immediately: Once
- ▶ One-time: Once, at a specified time (in the future)
- ▶ Time period: Start now or at a future time, repeat every n minutes, stop at a specified time
- ▶ Periodic sampling: Start now or at a future time, repeat every n minutes, open-ended
- ▶ Customize: More advanced options

#### DB2 catalog

An important source is the static SQL text in the catalog (SYSPACKSTMT or SYSSTMT), which you can subset using either or both of the filters:

- ▶ Packages: Filter by COLLID NAME OWNER BINDTIME or other SYSPACKAGE columns, and select individual QUERYNO or lists of QUERYNO
- ▶ Plans: By NAME CREATOR or other SYSPLAN columns

If your BINDs have been done with EXPLAIN(YES), you can further filter this SQL according to facts externalized in PLAN\_TABLE and DSN\_STATEMENT\_TABLE:

- ▶ Cost: PROCSU or PROCMS (from DSN\_STATEMENT\_TABLE)
- ▶ Object: That is, the table or index referenced by the SQL (from PLAN\_TABLE)

**Note:** A workload with all SQLs that accesses a given table is a good basis for complete Statistics Advisor and Index Advisor recommendations for that table.

- ▶ PLAN\_TABLE indicators: Table space scans? Sorts? Any or all of these can filter.

A catalog source is always captured immediately. No performance data (weighting) exists (each statement has count 1), but you can add weighting later by using “Edit Statement Runtime Information”.

### QMF or QMF HPO

Gather SQL from QMF or QMF HPO queries. You can collect all, or filter the QMF repository Q.OBJECT\_DIRECTORY, typically by filters on TYPE and NAME:

- ▶ TYPE=QUERY: All queries, or ones qualified by NAME.
- ▶ TYPE=PRO: The queries that are contained, directly or indirectly, in the named PROCs.

A QMF or QMF HPO source is always captured immediately. No performance data (weighting) exists, but you can add this later by using “Edit Statement Runtime Information”.

### Text file

A text file allows the entry of any SQL for workload analysis. This is an ideal way to capture SQL from any source that supports an SQL save, or any SQL that you want to input.

**Note:** There is no direct text entry window for SQL text for workloads, unlike the query-based tool. You must create a plain text file.

For each statement, simply provide the SQL text and terminate it with a semicolon (“;”). Repeat for all other SQL statements you want to include.

You can qualify the table, alias, and view references in the SQL by schema, or not qualify them. If the latter, make sure to supply it in the qualifier input field.

**Note:** You must enter the qualifier at definition time. You cannot provide it later.

### Categories

A “category” is an arbitrary keyword you can apply to individual SQL statements to group them together for a reason, for example, “Important CURSORS”. A category, in turn, can be a source of a workload. That way, a workload can target “Important CURSORS” specifically.

Categorization is done either through:

- ▶ The Categorize action in the View SQL Statement panel. Bring up the SQL of interest in that panel by double-clicking its line in Workload Statements.
- ▶ Within a Query Project, Query Text window, and select **Query** → **Categorize**.

### Other workloads

This workload source lets you copy statements from one workload to another.

### Multiple sources

A workload is typically created from a single source, but it might contain several sources. Each is identified by a source name that defaults to a meaningless unique value, such as “Source\_0”. Provide a meaningful name if you plan to have several.

Each source is identified in a separate tab in the List Statements panel. An “All” tab is always present at the left. It combines statements from all sources.

## 7.1.2 Tasks

Tasks are separately scheduled operations, which run asynchronously for any operation that can take a long time to complete:

- ▶ Capture: Create a workload from a source
- ▶ Explain: Explain SQL, statement by statement
- ▶ Consolidate: See discussion in section 7.2.4, “Workload statements” on page 158
- ▶ Advisor: Statistics Advisor, Index Advisor, Query Advisor, singly, or in combination

Depending on the context, tasks might run immediately (for example, capture a catalog source), or schedule them later (for example, schedule a cache snapshot).

When a long-running task is running in the foreground for a workload (for example, a large capture) and you do not want to wait for it, you have the option to “Run in Background”. The task minimizes and shows up as a in-progress animation in the bottom right of a panel. Any workload with background tasks is “Locked”. You cannot do anything with it. The advantage of running in the background is that you can operate on other workloads.

A history function shows you all the tasks that have run for a workload.

## 7.1.3 Workloads and workload projects

Workloads are managed by workload projects. Figure 7-1 shows their relationship:

	WORKLOAD	WORKLOAD PROJECT
Held in	Mainframe tables, one set per DB2	PC files
Shareable	Global, per subsystem (per Users options)	Specific to that PC
Related?	Linked, both necessary to operate OSC	
Create	New Workload OSC creates project when needed	New Workload Project, creates Workload too
See list	View Workloads	Project Navigator, Project Tabs

Figure 7-1 Workloads and workload projects

## 7.1.4 Defining a new workload

You can start by defining either a workload or a workload project. See section 7.1.1, “Workload sources” on page 150.



## New workload

From **View Workloads**, which displays the Workload List, choose **New Workload**. You can choose:

- ▶ The **Workload Wizard**, see below.
- ▶ **Statement Cache, Catalog, QMF, QMF HPO, File, Category, Other workload**: These are different ways of invoking the Workload Wizard with the source pre-populated.

When you open the new workload, a workload project is created for it. Its name is `ssss_workload_name`, where `ssss` is the DB2 subsystem. You can change this name.

## New Workload Project

Right-click anywhere in the Project Navigator, choose **New Workload Project** → **Create a new workload**. You are immediately launched into the Workload Wizard.

Make sure you give the workload project a meaningful name. The workload name defaults to the workload project name. If you rename the workload in the Workload Wizard, then you cannot locate the workload in the project navigator.

## Workload Wizard

A series of panels (“steps”) prompts you for setup information. Click **Next** to advance to the next step or **Prev** to redo. You cannot click directly on the Steps pane on the left.

Step 1: Workload names the workload. This name has a global scope (unique across the subsystem.) The owner is preset to the current SQLID provided on the Subsystem Configure panel.

**Suggestion:** Use a meaningful name because it is not possible to change the workload name later. There is also no facility to show the parameters you supplied to the Wizard later.

Step 2: Supply the Source (see section 7.1.1, “Workload sources” on page 150).

Step 3: Filter is different for each Source. Figure 7-2 shows an example of a “Column Operator Value” filter, here for Statement Cache. The “Operator” has a drop-down box, useful for providing multiple values (IN) or LIKE matches.

Column Name	Operator	Value	Comment
PRIMAUTH	=		The primary authorization ID that did the initial PREPAR
CURSQLID	=		The CURRENT SQLID that did the initial PREPARE of the
SCHEMA	<		The value of the CURRENT SCHEMA special register.
BIND_QUALIFIER	<=		The BIND qualifier. For unqualified table names, this is t
BIND_ISO	IN		The value of the ISOLATION BIND option that is in effec
BIND_CDATA	LIKE		The value of the CURRENTDATA BIND option that is in e

Figure 7-2 Workload Source Filter panel

Step 4: Capture is always “Immediately”, except for the Statement Cache, where you can choose deferred or periodic captures. “Customize” brings up “Step 5. Advanced”, allowing you to change explain capture (see section 7.1.5, “Gather explain information” on page 154).

## Monitor workloads

You set up a monitor workload by operating the monitor, not by creating a New Workload or New Workload Project (as you do for all other workload sources).

When you double-click or “Open a monitor”, a new Workload Project is automatically created. Its name is ssss\_monitor\_name, where ssss is the DB2 subsystem.

### 7.1.5 Gather explain information

When you define a workload, Optimization Expert attempts to gather Explain information for the sources:

- ▶ Dynamic statement cache - from the dynamic statement cache Explain tables
- ▶ Catalog - from the user explain tables (qualified by the package/plan owner)

Collecting existing explain information saves time by re-explaining statements later.

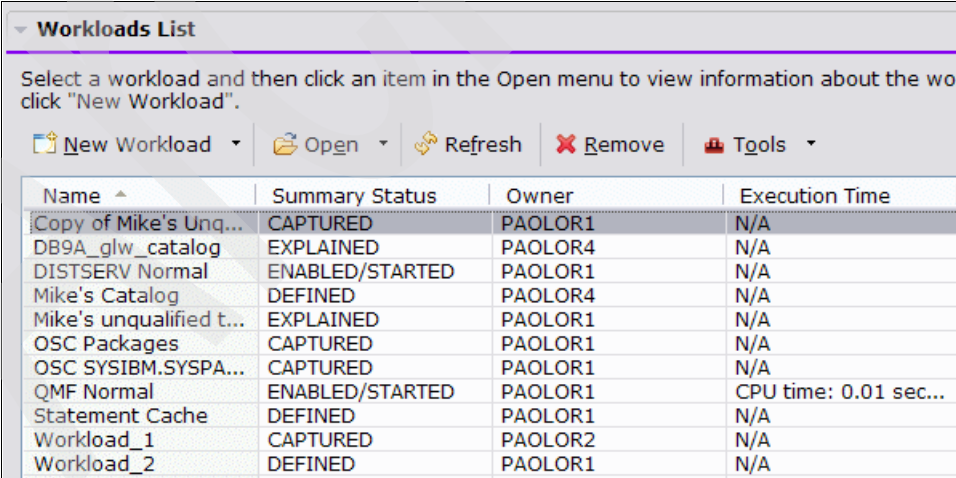
## 7.2 Working with workloads

When workloads are defined (section 7.1, “Defining a new workload” on page 150), you can examine workload statements, run any or all of the analyses, add to the workloads, consolidate, or take other actions.

### 7.2.1 Workload list

The Workload List panel (Figure 7-3) is the focal point for all operations having to do with workloads. Use it to find all workloads that are currently defined.

To bring up the list (Figure 7-3), use **View Workloads**, available at the top of the Project Navigator or as a top left tab.



Name ^	Summary Status	Owner	Execution Time
Copy of Mike's Unq...	CAPTURED	PAOLOR1	N/A
DB9A_glw_catalog	EXPLAINED	PAOLOR4	N/A
DISTSERV Normal	ENABLED/STARTED	PAOLOR1	N/A
Mike's Catalog	DEFINED	PAOLOR4	N/A
Mike's unqualified t...	EXPLAINED	PAOLOR1	N/A
OSC Packages	CAPTURED	PAOLOR1	N/A
OSC SYSIBM.SYSPA...	CAPTURED	PAOLOR1	N/A
QMF Normal	ENABLED/STARTED	PAOLOR1	CPU time: 0.01 sec...
Statement Cache	DEFINED	PAOLOR1	N/A
Workload_1	CAPTURED	PAOLOR2	N/A
Workload_2	DEFINED	PAOLOR1	N/A

Figure 7-3 Workloads List

### Workload list

The list of workloads is initially presented in alphabetical name order.

Table 7-1 is a list of a workload's summary status, an important indicator:

Table 7-1 Workload Summary Status

Workload status	Meaning	Notes
DEFINED	Workload is created, but no SQL captured.	Note 1
CAPTURED	At least 1 SQL captured, no EXPLAIN so far.	
EXPLAINED	EXPLAIN complete.	Note 2
<b>Workload In-progress indicators</b>		
UPDATING	Workload definition is being updated.	
CAPTURING	SQL capture is in progress.	
EXPLAINING	EXPLAIN is in progress.	
ANALYZING	ANALYZE in progress.	Note 2
IN PROCESSING	Another task, such as consolidation in-progress.	
<b>Monitor indicators</b>		
ENABLED or DISABLED / a START status		Note 3
<p>Note 1 - CAPTURE failed to find a single SQL. The workload is inoperative and needs to be removed.</p> <p>Note 2 - There is no ANALYZED status. To see if any analyses have been run, examine the information in the workload project.</p> <p>Note 3 - See Chapter 6, "Profile monitoring" on page 131.</p>		

## Owner

You see all workloads that are contributed by all users operating Optimization Service Center on this subsystem. You can click the **Owner** header to sort by owner name. Currently, there is no way to limit the display to your own workloads.

## 7.2.2 Workloads - operations

Controls in Chapter 7.2.1, "Workload list" on page 154 allow you to initiate the following actions:

### New workload

See section 7.1.4, "Defining a new workload" on page 152.

### Open

Clicking **Open** opens the highlighted Workload Project (if not already open) and positions you on the Statements tab in the Workload Project. Double-clicking the workload has the same effect.

The Open pull-down allows direct access to any tab in the Workload Project. Choose **Project** to see the main workload project panel.

### Refresh

Use **Refresh** to update the list with the most recent information. The list does not auto-refresh itself.

## Remove

Deletes the highlighted workload. There is no right-click menu for highlighted workloads. You can also delete the project from the Project Navigator.

Remove also deletes all information in the persistent mainframe WCC tables. This might take a long time if the workload has many statements.

**Warning:** If you remove a workload, any workload project linked to that workload is not removed. Instead, it attaches itself to a different workload. This is not ideal. After removing a workload, always remove any linked workload projects by right-clicking them in the Project Navigator and choosing **Delete Project**.

## Tools

There are two tools for special purposes:

**Tools → Maintain → Recover Tasks** - to recover from a situation where any of the workload tasks (such as Capture or Explain) runs unacceptably long.

**Tools → Maintain → Restore Status** - of any workload that is getting errors. The workload is restored to a previous consistent point.

### 7.2.3 Workload project

The workload project is the Optimization Service Center desktop's way of operating (displaying, running Advisors on, and so on) with workloads.

Bring up the project (Figure 7-4 on page 157) in one of two ways:

- ▶ From the Worklist List panel, see Chapter 7.2.1, "Workload list" on page 154, the **Open** action.
- ▶ By double-clicking any project in the Project Navigator.

Project: DB9A\_QMF Normal Project Rename Project...

Subsystem: DB9A <enabled> Configure...

Workload Name: QMF Normal

Workload Owner: PAOLOR1

Summary Status: ENABLED/STARTED

Description: Normal QMF consumption Change...

**Workload Statements**  
Immediately capture workload statements, get tuning recommendations from the workload advisors, and use tools to tune an individual query.

**Users**  
Grant or revoke owner, update, and read-only access to the workload. You can create authorization IDs in the workload control center only if you have the proper authority.

**Run Advisors** ▼  
Get recommendations for workloads that could improve workload performance. Schedule workload analysis for a later time.

**History**  
View the history of this workload, including when it was created, each time that statements were captured or consolidated, and each time EXPLAIN information was gathered.

**Schedule Tasks**  
Schedule when to capture statements, consolidate statements, and gather EXPLAIN information.

Workload Statements History Advisors

Figure 7-4 Workload Project

At the top, you see the same information as the Workload List, see Chapter 7.2.1, “Workload list” on page 154. Note the Summary Status (Table 7-1 on page 155).

If any background task is operating for the workload, it is shown at the bottom right.

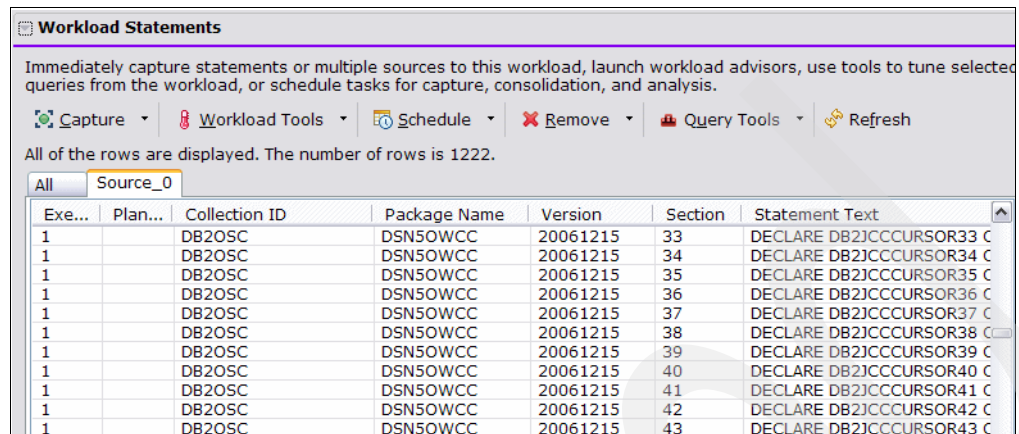
The five action buttons (Workload Statements, and so on) apply to the entire workload. If any of these have been run before, they create project items, which show up equivalently:

- In the Project Navigator “tree” in the left.
- As a bottom tab in the Workload panel. To close any of these tabs down, move your cursor over the tab and choose **X Close**.

These operations are discussed in the following sections.

## 7.2.4 Workload statements

Figure 7-5 shows a typical Workload Statements list.



The screenshot shows the 'Workload Statements' window. It has a toolbar with buttons for Capture, Workload Tools, Schedule, Remove, Query Tools, and Refresh. Below the toolbar, it says 'All of the rows are displayed. The number of rows is 1222.' There are two tabs: 'All' and 'Source\_0'. The 'All' tab is selected, showing a table with the following columns: Exe..., Plan..., Collection ID, Package Name, Version, Section, and Statement Text. The table contains 13 rows of data, all with 'DB2OSC' as the Collection ID and 'DSN5OWCC' as the Package Name. The Statement Text for all rows is 'DECLARE DB2JCCCURSOR33 C'.

Exe...	Plan...	Collection ID	Package Name	Version	Section	Statement Text
1		DB2OSC	DSN5OWCC	20061215	33	DECLARE DB2JCCCURSOR33 C
1		DB2OSC	DSN5OWCC	20061215	34	DECLARE DB2JCCCURSOR34 C
1		DB2OSC	DSN5OWCC	20061215	35	DECLARE DB2JCCCURSOR35 C
1		DB2OSC	DSN5OWCC	20061215	36	DECLARE DB2JCCCURSOR36 C
1		DB2OSC	DSN5OWCC	20061215	37	DECLARE DB2JCCCURSOR37 C
1		DB2OSC	DSN5OWCC	20061215	38	DECLARE DB2JCCCURSOR38 C
1		DB2OSC	DSN5OWCC	20061215	39	DECLARE DB2JCCCURSOR39 C
1		DB2OSC	DSN5OWCC	20061215	40	DECLARE DB2JCCCURSOR40 C
1		DB2OSC	DSN5OWCC	20061215	41	DECLARE DB2JCCCURSOR41 C
1		DB2OSC	DSN5OWCC	20061215	42	DECLARE DB2JCCCURSOR42 C
1		DB2OSC	DSN5OWCC	20061215	43	DECLARE DB2JCCCURSOR43 C

Figure 7-5 Workload Statements

The **All** tab on the left gathers all sources and presents it with execution data:

- ▶ Execution Count - actual values if Cache or Monitor, the number of times the SQL appears in the source.
- ▶ Source - simply says CATALOG or MONITOR, and so on.

**Note:** You cannot tell which source contributed if you have more than one of the same type.

- ▶ Elapsed Time - values only if cache or monitor, otherwise 0.
- ▶ CPU Time - values only if cache or monitor, otherwise 0.
- ▶ SQL Text

Each source appears in a different tab. It has information specific to the source:

- ▶ Execution Count - actual values if Cache or Monitor, the number of times the SQL appears in the source.
- ▶ Values specific to the source, for example, Query Name if QMF, Package Name if Catalog, and so on.
- ▶ SQL Text

### SQL operations

You can see the SQL text in its entirety in a separate panel (Figure 7-6), by double-clicking any SQL. You can save the SQL or categorize it.

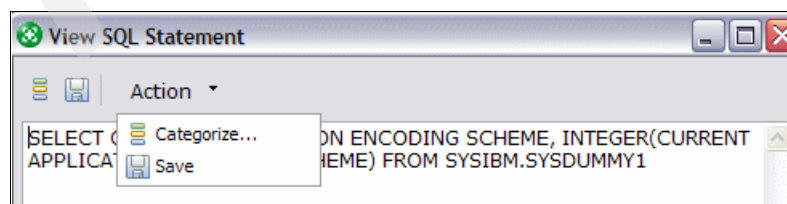


Figure 7-6 View SQL Statement

By highlighting a single SQL and choosing Query Tool, you can:

- ▶ Run any Query Tool or Advisor, such as Access Path Graph or Statistics Advisor. Each creates a separate query project.
- ▶ Edit statement runtime information, allowing you to alter the query priority in the workload by manipulating the execution count or CPU or elapsed time.

### Workload advisors

You can run the workload advisors from the Workload Statements panel. This is equivalent to the “Run Advisors” action available on the workload main panel.

To run these at a later time, use **Schedule** → **Advisors** instead.

### Capture

Capture adds new sources to your workload or replaces existing sources. You can run it:

- ▶ Directly from the **Capture** button.
- ▶ From **Schedule** → **Capture** - only for statement cache because this is the only capture that can be scheduled to run in future.

In both cases, the Capture Wizard is launched. The choices are similar to those of the Workload Wizard described in 7.1.4, “Defining a new workload” on page 152.

By choosing **New source**, you add an additional source, whose statements appear in a separate Source tab and also are combined into the All tab.

By choosing **Existing source**, and one of the source names, you can refresh that source’s contents, for example, a Catalog source based on COLLID to pick up new package versions. All “2. Filter options” are greyed out (as the existing source.) The default action is to replace all statements. A customized choice in the “3. Capture” panel lets you change this by selecting **Keep old statements** in the “4. Advanced” panel.

### Consolidation

Execute **Schedule** → **Consolidate** periodically to reduce the volume of explain information kept by OE. SQL might be explained several times and rows with different explain times will exist in the Explain tables.

When you run Consolidate, the default action is to replace all existing explain rows. There is no benefit in choosing to “Keep old rows” because Optimization Expert operates on only the most current rows. Another choice is **Consolidate**, which means to keep only different explains and delete duplicates.

The default action is to “Consolidate literal values to parameter markers” by, for example, combining C=7 and C=6 to a common C=?. To keep these variations (which might produce different access paths), choose the bottom check box, “Consolidate access path information”, which does not consolidate literal values.

## 7.2.5 Workload schedule tasks and history

The Schedule Tasks button on the Workload Project panel:

- ▶ Lists all scheduled tasks - ones completed in the past and others currently running.
- ▶ Contains a button that allows you to schedule a new task, an equivalent operation to choosing Schedule within the Workloads Statements panel.

The History button on the Workload Project panel lists all completed events, which contain scheduled tasks and other operations, together with their Start/Stop time (can see duration) and Status (FINISHED).





## Part 4

# Solution components

This part examines in more detail the main functional components of the tools. It contains the following chapters:

- ▶ Chapter 8, “Query tools” on page 163
- ▶ Chapter 9, “Statistics Advisor” on page 223
- ▶ Chapter 10, “Index Advisor” on page 243
- ▶ Chapter 11, “Access Path Advisor and Query Advisor” on page 283

Archived

## Query tools

In addition to dealing with workloads and monitors, Optimization Service Center or Optimization Expert also provides the following two major sets of functions:

- ▶ Advisors
- ▶ Tools

This chapter focuses on the available tools. All tools are available in both the Optimization Service Center and Optimization Expert products. The available tools are:

- ▶ “Access Plan Graph” on page 171
- ▶ “Query Annotation tool” on page 180
- ▶ “Query reports” on page 191
- ▶ “Gather service information” on page 198
- ▶ “Visual Plan Hint” on page 204

## 8.1 Providing to a query to analyze

Before looking into the different tools that are available, we first look at how to obtain a query to analyze. Unlike the “advisors”, the “tools” work on an individual query basis. For example, you cannot obtain an access path graph for a *set* of queries in a single operation.

### 8.1.1 Selecting a query from a workload

If you open up a workload that contains a number of captured SQL statements, it is probably easiest if you use the scroll bar to scroll to the right to see the SQL statement text of the different SQL statements that were captured, as shown in Figure 8-1.

Before you can invoke any of the tools, you need to select a statement. Then you can click **Query Tools** and the list of available tools opens up, allowing you to select a tool from the list.

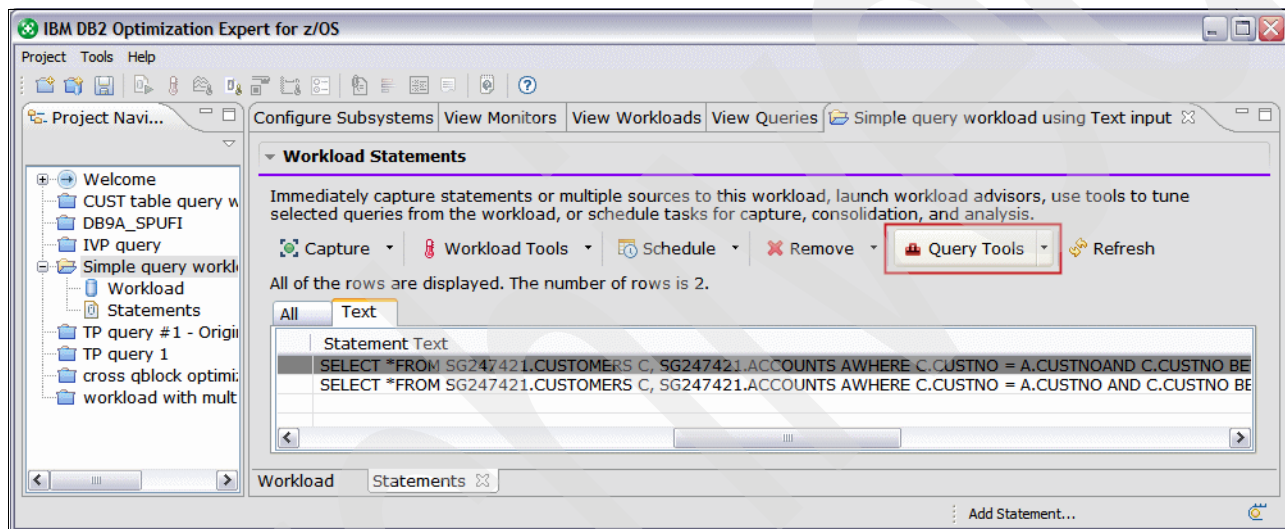


Figure 8-1 Selecting a Query Tool from a workload

When you invoke one of the query tools from a workload project, this automatically triggers the creation of a new query project, copies the SQL statement into the newly created query project, and invokes the selected query tool. The results of the invocation of a query tool from a workload are always stored in a new query project, never in the workload project they originated from. This is currently a restriction of the Optimization Service Center or Optimization Expert product, which is being considered for removal.

### 8.1.2 Selecting a query using a query project

When you are using a query project, clicking the Query tab at the bottom, or the Query icon in the project navigator pane on the left shows the SQL statement text, as shown in Figure 8-2 on page 165. Clicking **Tools** opens up the list of available tools you can select.

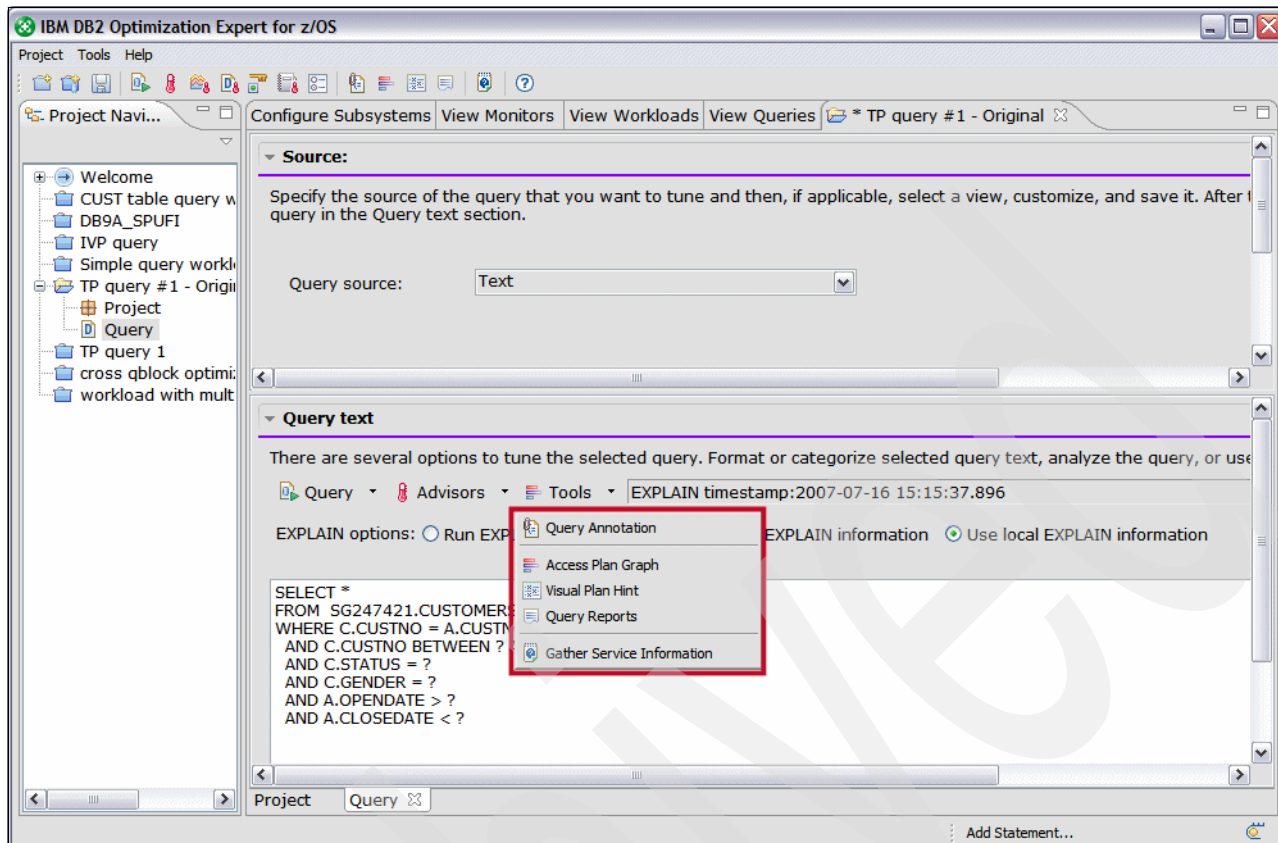


Figure 8-2 Selecting a Query Tool in a query project

When you are using a query project, you can also invoke the query tools from the main project window, either by selecting the query tool from the tool bar at the top, or by using one of the options in the “Tune a Single Query” window, as shown in Figure 8-3 on page 166.

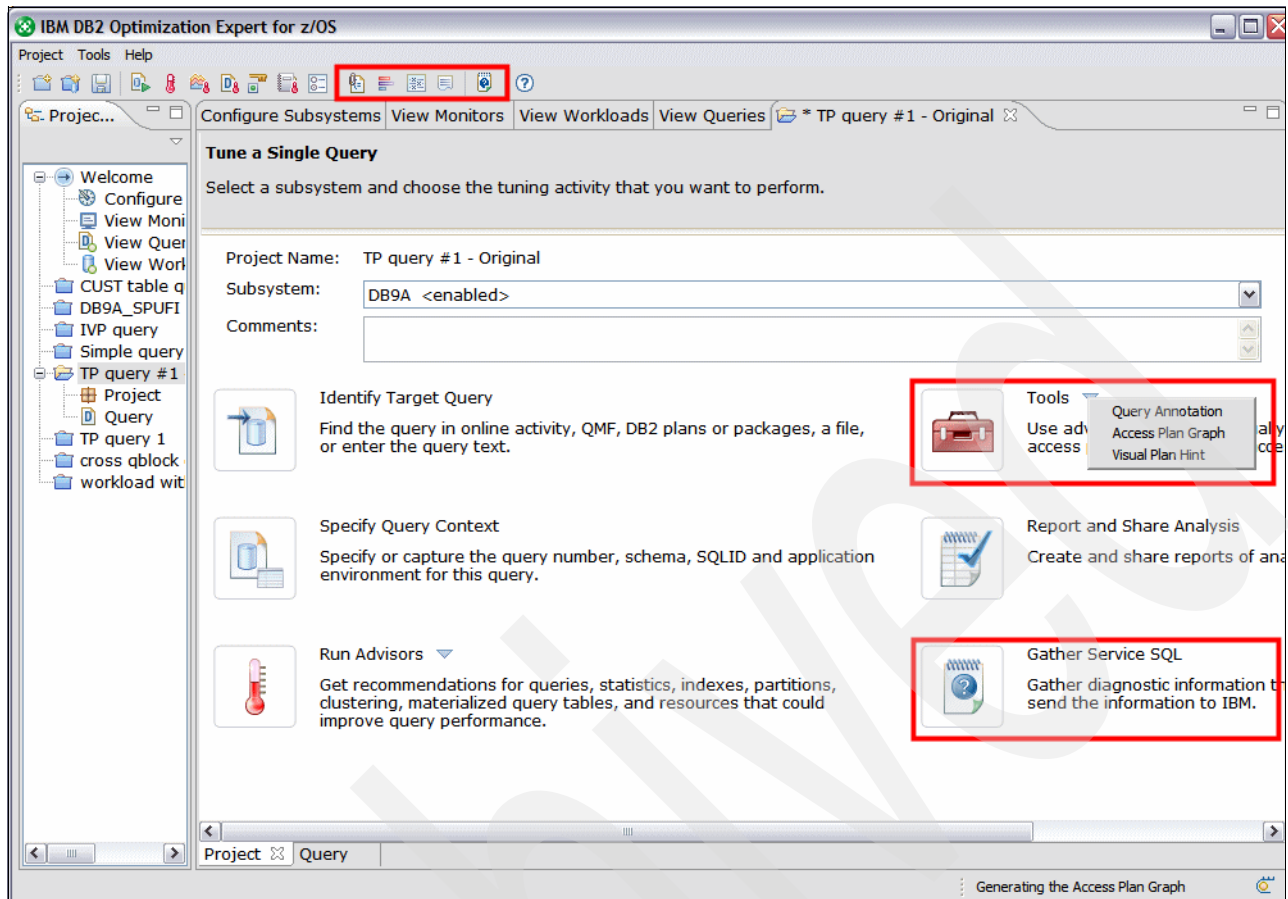


Figure 8-3 Query tools invocation from project window

### 8.1.3 Context setting

The query context specifies the application environment to be used by the Advisors and Query tools. The context window is invoked from the query project's main "Project" window (see Figure 8-4 on page 167).

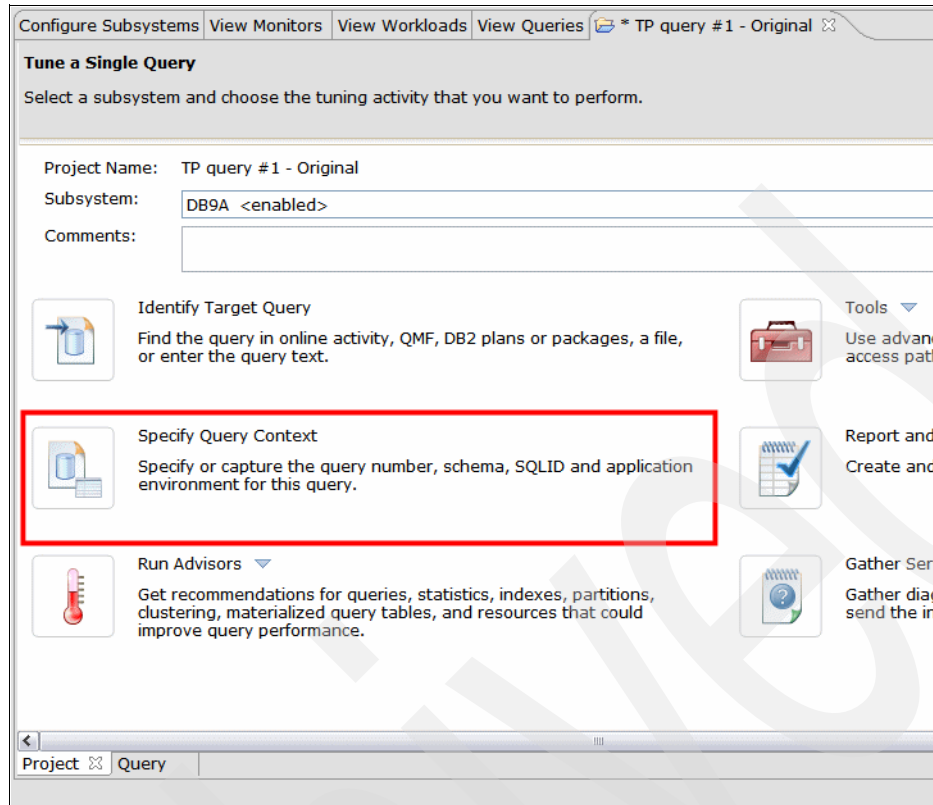


Figure 8-4 Specify Query Context invocation

The query context (see Figure 8-5 on page 168) specifies the settings that are to be used when the tools (or Advisors) are invoked from the query project. You can also indicate whether you want to use the EXPLAIN stored procedure instead of running the explain functions under the user ID that is currently connected to the DB2 system. This is a powerful function because it lets users, such as database administrators, to invoke the EXPLAIN function without having the authority to actually perform the actual operation in the SQL statement that is being explained. Without this stored procedure, when the user is explaining a SELECT statement against table TAB#1, the user must have SELECT authority on TAB#1 to be able to EXPLAIN that SQL statement. When using the stored procedure, this is not required. The user only needs to invoke the stored procedure and does not need SELECT authority on TAB#1.

Using the stored procedure requires that it has been installed. Installing the EXPLAIN stored procedure is part of the normal DB2 installation process because everyone can use the stored procedure, not just Optimization Service Center or Optimization Expert. Table 3-1 on page 83 of Chapter 3, "Installing Optimization Service Center" on page 81 lists the stored procedures used by the tools and the jobs to create them.

**EXPLAIN Options**

Specify the options to use for the EXPLAIN function for the specified query.

Query number:

SQLID:

Optimization hint:  [Update Local Cache Information](#)

☐ Use the following stored procedure to issue the EXPLAIN statement:

Table qualifier:

Procedure name:

**Application Environment**

Specify the application environment for current SQL statement.

Schema:

Current degree:

Current refresh age:

Current maintained table types:

Project Query Context

Figure 8-5 Query context options

Note that the “Schema” option specifies the value for the CURRENT SCHEMA special register that is used to qualify unqualified names in dynamic SQL statements. Unlike the CURRENT SQLID, this option is not associated with any authorization.

Do not forget to fill in the “Current degree” field when working with query parallelism, or the “Current refresh age” and “Current maintained table types” field when working with materialized query tables (MQTs).

## 8.1.4 Explaining an SQL statement

All the Query tools require that the SQL statement is explained. Selecting any of the tools will trigger DB2 to explain the SQL statement. To avoid explaining the statement over and over again, you can reuse the explain information that was produced by a previous invocation of any of the other tools.

To control whether or not a statement is explained again, and if not, where the information from a previous explain is retrieved from, use **Tools** → **Preferences** → **Re-EXPLAIN** as shown in Figure 8-6 on page 169.



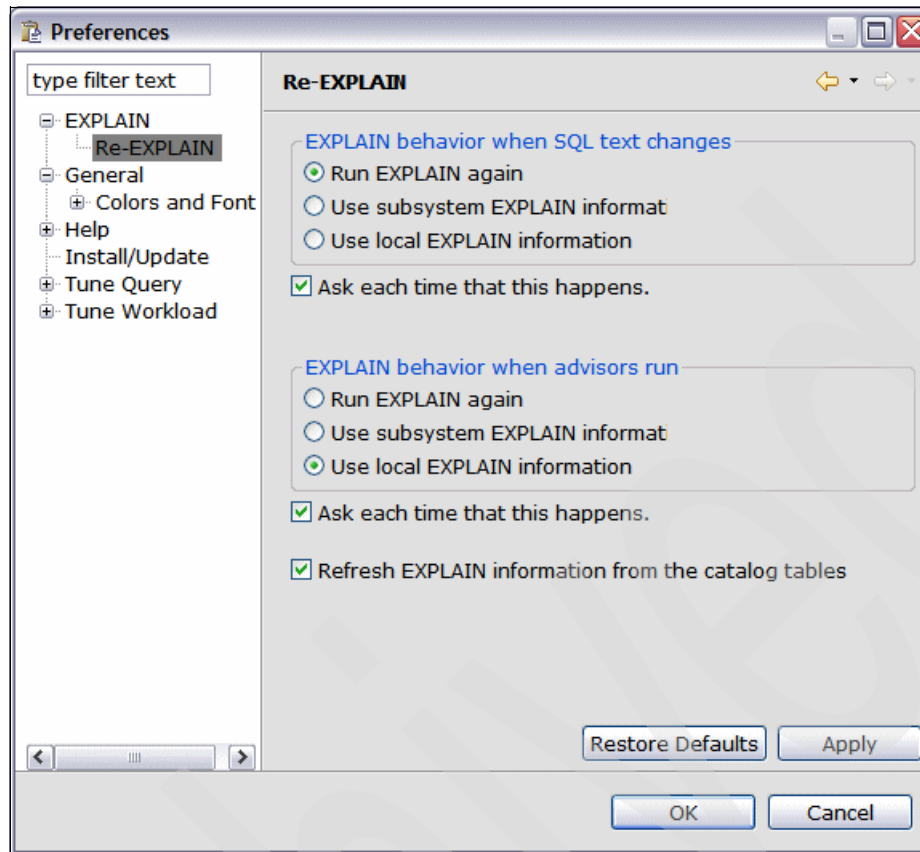


Figure 8-6 Preferences Re-EXPLAIN window

The set of options at the top govern the EXPLAIN behavior when Optimization Service Center or Optimization Expert notices that the SQL text has changed.

The bottom set governs what EXPLAIN behavior is used when an Advisor or Query tool is used.

The different options are:

#### **Use local EXPLAIN information**

Specifies that Optimization Service Center or Optimization Expert uses the EXPLAIN information that is cached on your local machine. This option is available only when the EXPLAIN time stamp field has a value, which means that the EXPLAIN information for this query exists in your local cache. If this information exists, but is not up-to-date, consider selecting one of the other two options.

#### **Use subsystem EXPLAIN information**

Specifies that Optimization Service Center or Optimization Expert collects existing EXPLAIN information for this query from the EXPLAIN tables on the selected subsystem. If this information exists, but is not up-to-date, consider selecting the “Run EXPLAIN again” option.

#### **Run EXPLAIN again**

Specifies that Optimization Service Center or Optimization Expert issues the EXPLAIN statement for the specified query, and then collects the resulting information from the EXPLAIN tables on the subsystem.

**Tip:** We suggest that you leave the “Ask each time that this happens” box checked. This way, you are always prompted when a function is invoked and you do not run into any implicit behavior triggered by an inadequate setting for a certain action.

When the “Ask each time that this happens” box is checked, the following window (Figure 8-7) is presented each time an Advisor or Query tool is invoked.

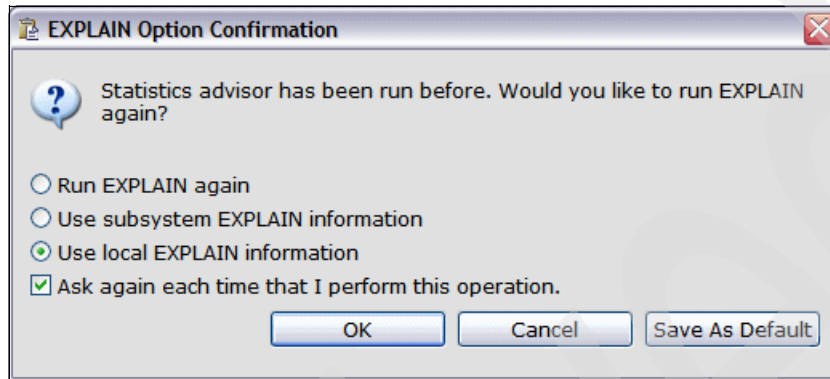


Figure 8-7 Explain Option Confirmation window

If you want to change the default setting, you can use the preferences option discussed above. You can also change the option on this window and click **Save As Default** to make it the default from now on.

Another way to indicate how the explain information is to be obtained, is by using the buttons on the “Query text” window, as shown in Figure 8-8 on page 171. This option is only useful when the “Ask again each time” box is not checked. Otherwise, the Explain Option confirmation window (Figure 8-7) appears each time to allow an override. Because we recommend to check the “Ask again each time” box, using the option on the SQL query window is not useful.

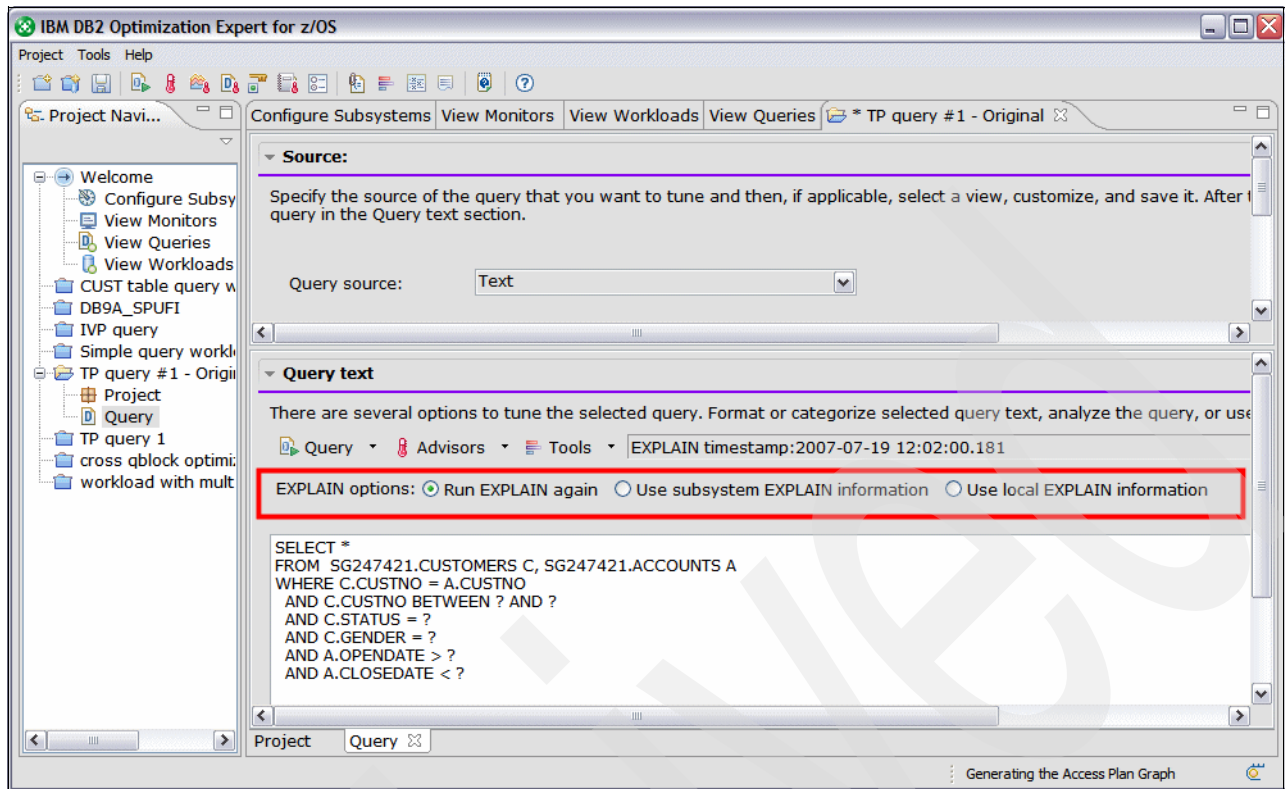

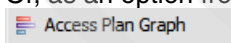


Figure 8-8 Specifying Explain options using the Query text window

## 8.2 Access Plan Graph

The first query tool that we look at is the *Access Plan Graph*. It provides a visual representation of the access path of a query that was picked by the optimizer, as Visual Explain did. To invoke the Access Plan Graph, you use the **Access Plan Graph** button:

- ▶ From the tool bar at the top:  .
- ▶ Or, as an option from the **Tools** button (same icon) on the Query text window:  .

To illustrate, we use the same query as shown in Figure 8-8. The Access Plan Graph is shown in Figure 8-9 on page 172.

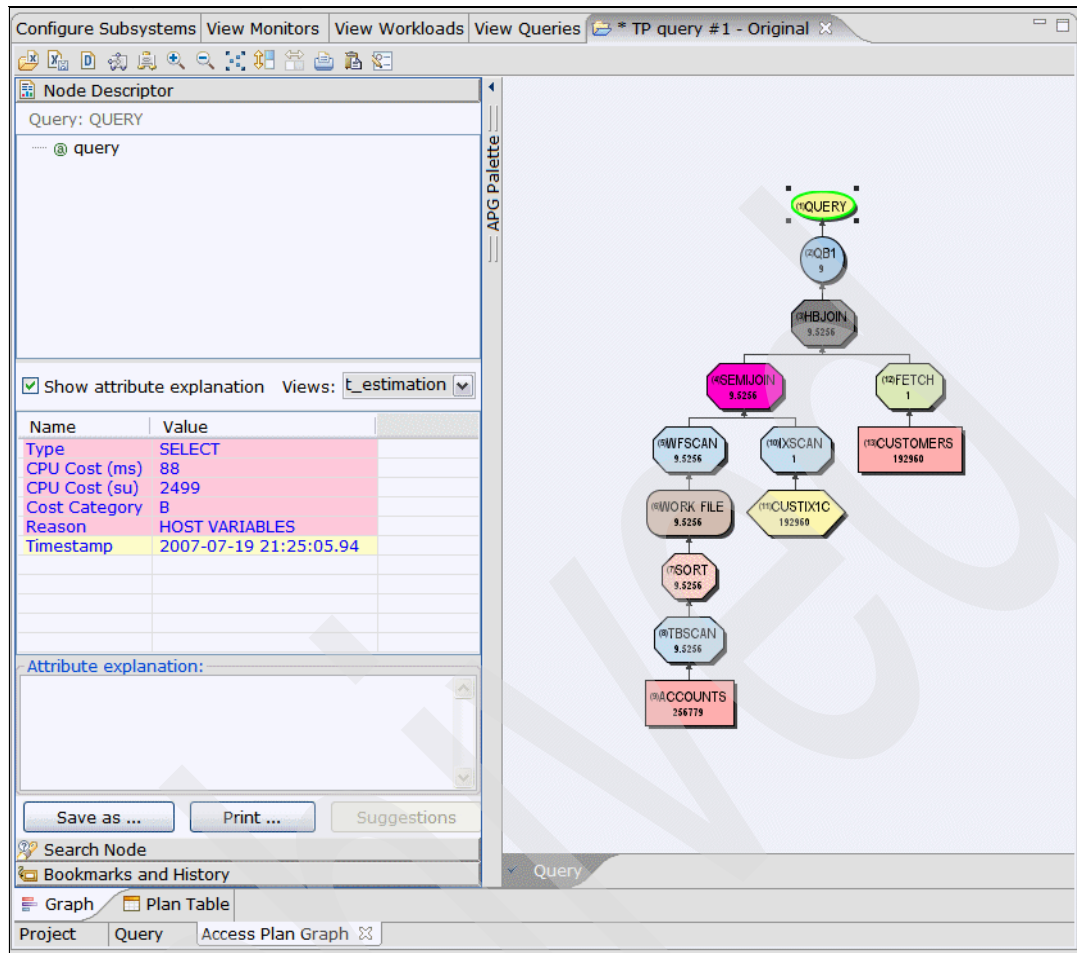


Figure 8-9 Access Plan Graph

By default, the graph is shown. However, notice that next to the Graph tab there is also a Plan Table tab. This displays the rows as they appear in the PLAN\_TABLE.

## Access Path Graph and Node Descriptor

An introduction to the *Access Plan Graph* and the *Node Descriptor* is provided in section 2.3, “Investigating a query by typing the text” on page 38. This section describes the “advanced” features of the Graph tab.

At the bottom of the Node Descriptor window are a number of buttons that allow you to:

- ▶ Save the current information of the Node descriptor to a file. You can specify different output formats, such as text, XML, HTML, and CSV.
- ▶ Print the current Node Descriptor information.
- ▶ Obtain help information for the current Node Descriptor.

## Search Node

When the access path graph is initially displayed, the graph itself is on the right and the Node Descriptor is on the left. Below the Node Descriptor is the *Search Node* window. Initially it is minimized, but when you click it, it opens up (and minimizes the Node Descriptor window), as shown in Figure 8-10 on page 173.

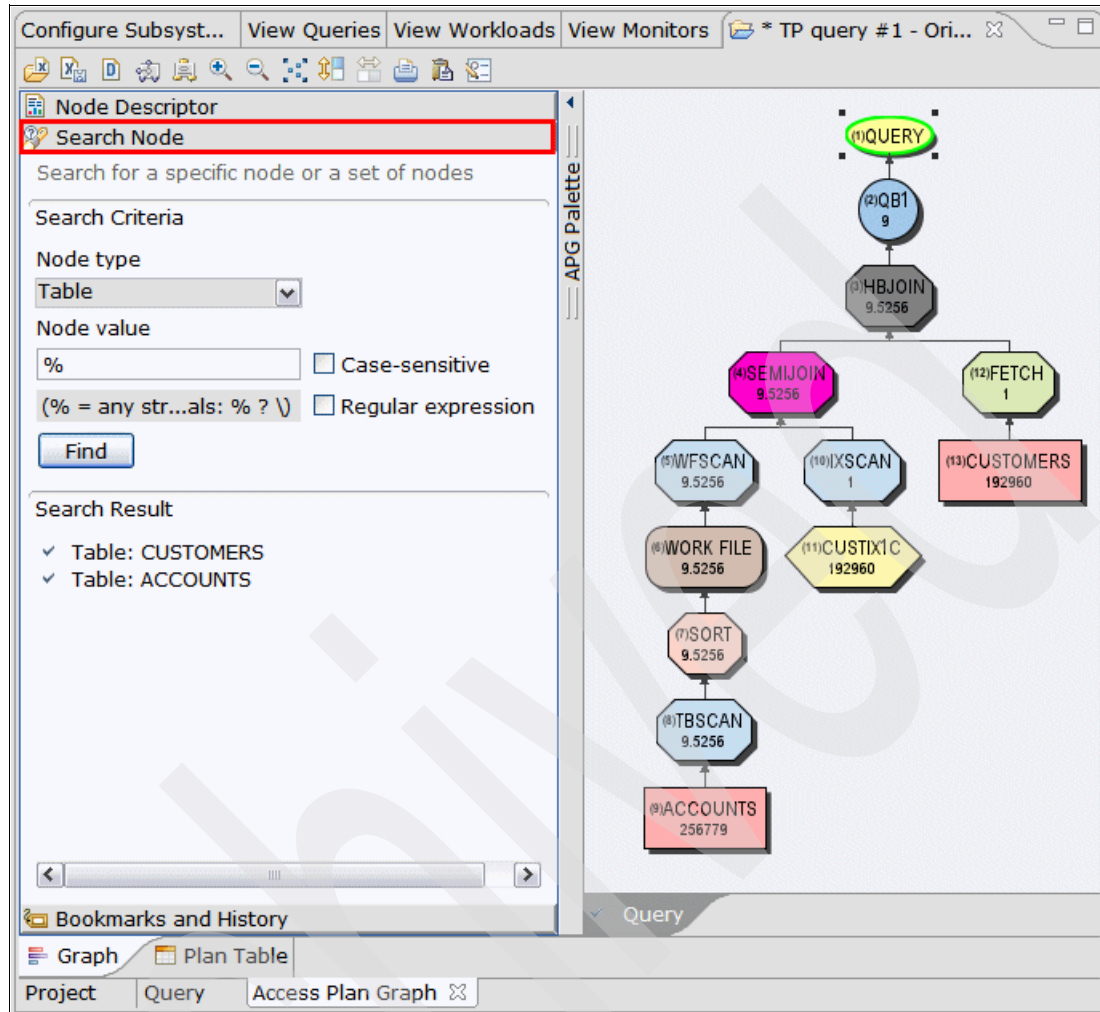


Figure 8-10 Search Node

As indicated by the name, the Search Node window searches for certain node types, matching the search value that you can provide as a Node value. If you want to find all nodes of a certain node type, specify the “%” wildcard character. Click **Find** when ready and the search result is displayed at the bottom of the Search Node window. In Figure 8-10, we searched for all table nodes in the current access path graph. You must have explained the SQL statement prior to using this option.

The Search Node feature is useful when dealing with large queries with many tables and many different node types, and queries where the access path graph does not easily fit on the window, such as our simple example.

## Bookmarks and History

Below the Search Node window, we have the *Bookmarks and History* window at the bottom left. As the Node Search window, it is initially minimized, but when you click it, it opens up. The result is shown in Figure 8-11 on page 174.



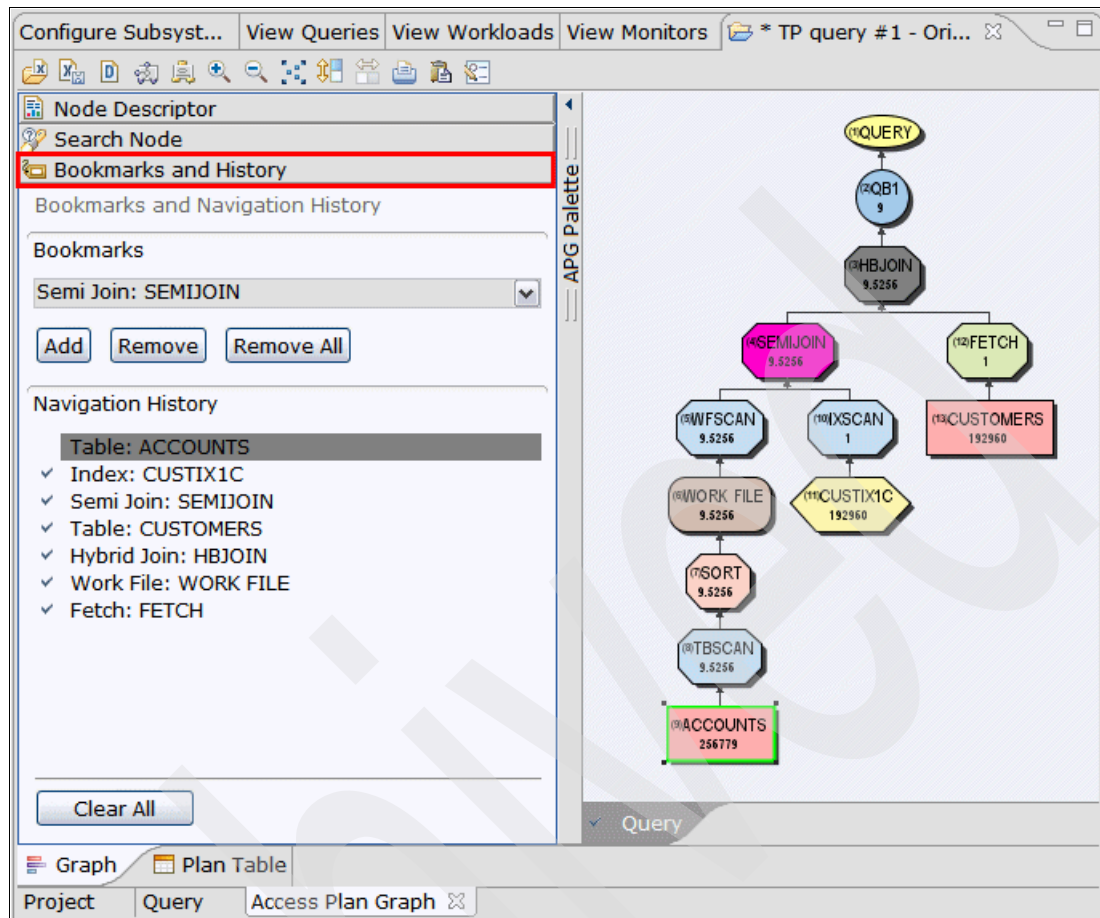


Figure 8-11 Bookmarks and History

The bottom part of the window contains the history of the nodes you clicked in the access path graph on the right. In this example, we first looked at the Table: ACCOUNTS node, next at the Index node CUSTIX1C, and so on.

The top part bookmarks certain nodes so you can easily go back to them. By clicking **Add**, the current node in the graph (indicated by the green dots) is added to the bookmarks drop-down list. You can also remove a single node from your bookmarks, or remove all bookmarks. When you want to go directly to a book marked node, select it from the drop-down list. As with the Search Node option, bookmarks are most useful when dealing with complex queries that have many nodes and subqueries.

## Access Plan Graph buttons

You can use the buttons at the top of the Access Plan Graph tab (Figure 8-12) for a wide range of things.

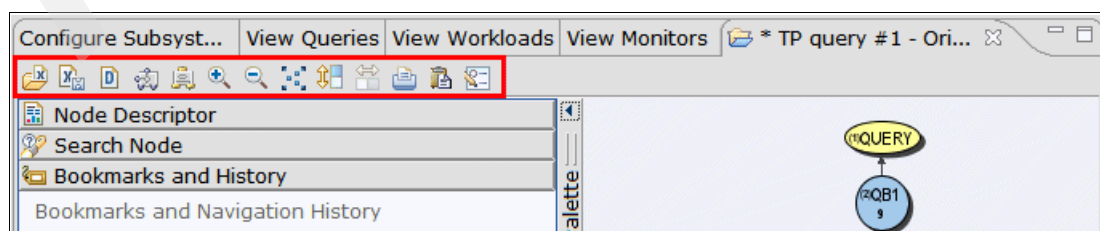





Figure 8-12 Access Plan Graph buttons

In this section, we look at each individual button in more detail:

- ▶  is used to open a saved XML file (that contains the access path graph information). The file is created using save XML file option.
- ▶  is used to save the access path graph as an XML file.

**Tip:** This option is useful to save all access path-related information into a file (XML) that another person can look at, without the need or authority to actually explain the SQL statement. The XML file contains all relevant information about the SQL statement and its access path.

- ▶  is used to look at the SQL statement text. You can use the Query tab at the bottom of the query project to look at the SQL statement, but this button has the advantage in that it saves the SQL statement text to a text file by using the **Save** button (Figure 8-13). This is useful when the statement was extracted from a monitor or the dynamic statement cache.

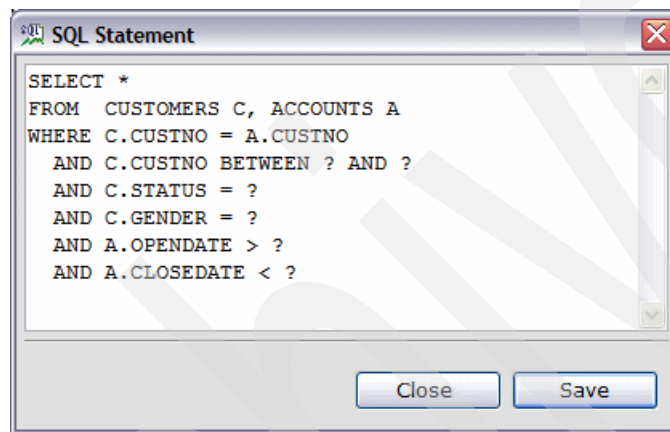



Figure 8-13 View SQL statement

- ▶ The  button provides you with an overview of the access path graph. This is useful when looking at a complex query. An example is shown in Figure 8-14 on page 176. The light area is the part that is currently displayed as part of the access path graph in the graph window. To illustrate, we purposely reduced the size of the graph window so only a few nodes can fit.

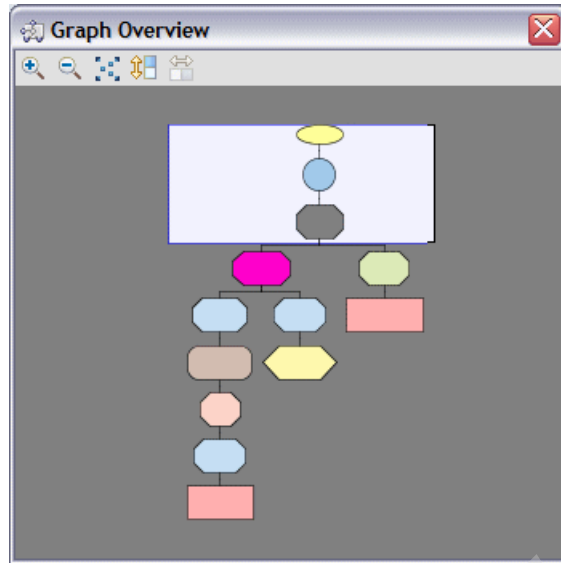





Figure 8-14 Graph Overview window

- ▶ The  buttons are used to zoom in and zoom out, or to resize the access path graph so its size fits within the current size of the access path graph window.
- ▶ The  buttons allow you to reverse the graph, either vertically or horizontally. Reversing the access path graph vertically provides a more logical view of how the query is executed, as the operations are from left to right, top to bottom. The function of reversing horizontally is enabled only if the graph is reversed (not all graphs are reversed). Otherwise, it is disabled. The button indicates a parent/child relationship where, for example, QB1 is the parent of QB2, while QB1 still has another child WFSCAN, so that QB2 is put at the left or right of QB1. To reverse horizontally means changing the position of QB2 from left to right or right to left.
- ▶ The print  button prints the SQL statement, the PLAN\_TABLE information, and the access plan graph. When you click **Print**, the print options window appears (Figure 8-15 on page 177). There is **Print Preview** button that gives you an idea what the printed output looks like without actually sending it to the printer. The print option menu also prints a selected query block, or an area that you previously marked in the access path graph. You can mark an area by clicking the left mouse button where you want to start and drag the mouse to mark the area to be printed.



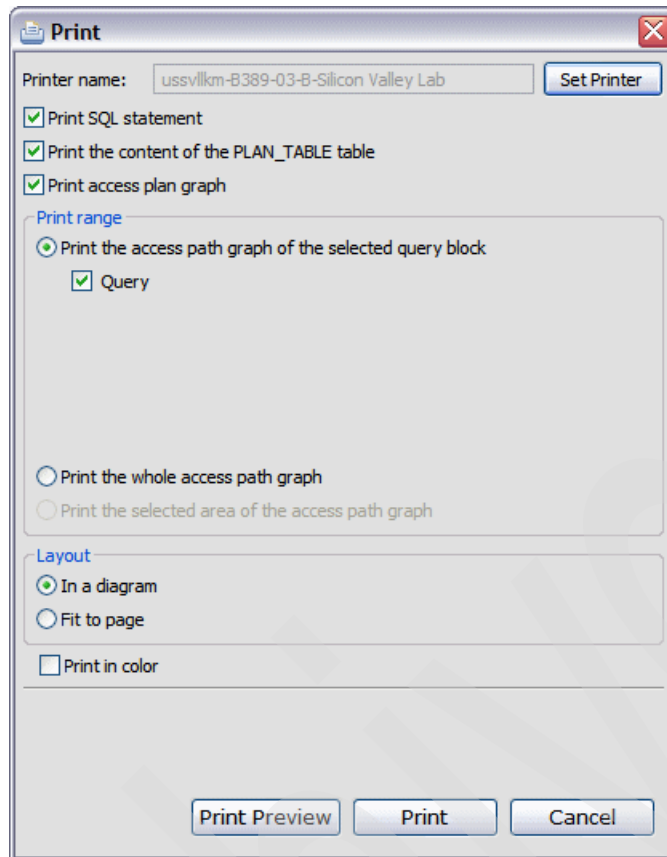



Figure 8-15 Print options window

- The reports  button is the one that you will use most. After pressing the button, the Report Options window displays (Figure 8-16 on page 178):
  - At the top, you can select the report format; text or HTML.
  - In the left pane, you can indicate which items to include in the report. You can include the same information that is displayed inside the node descriptors.
  - As with the node descriptors, you can indicate whether you want to include space, cost or all information, by selecting from the “Attributes view” drop-down list on the right.
  - You can also limit the report to a specific query block if there is no need to print the data of the entire access path. In our simple example, the query only consists of a single query block.
  - At the bottom, you can indicate where the reports are to be saved. We recommend that you use a separate directory for each query project.

When done, click **Generate Report**.

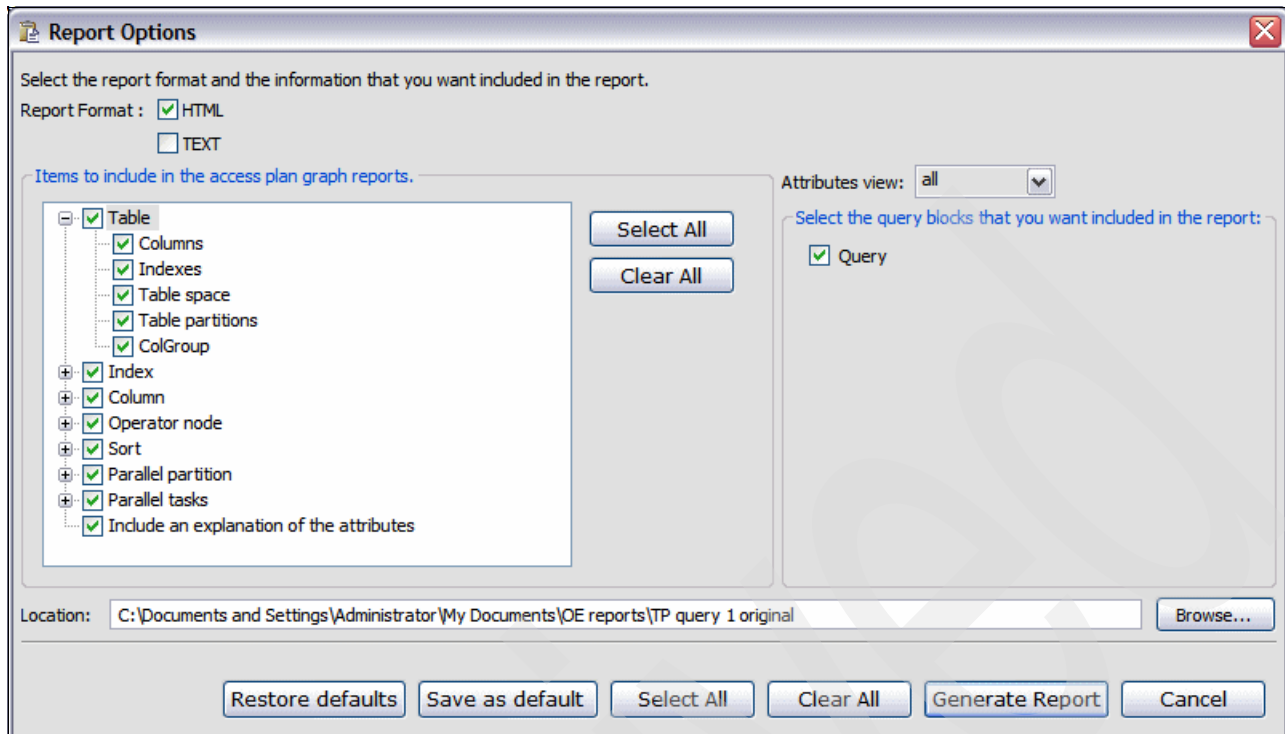


Figure 8-16 Report Options window

As we selected the HTML option, the report is opened automatically in our default Web browser. The result is shown in Figure 8-17 on page 179.

The complete report is not a single HTML file, but a number of files. Using the left pane, you can navigate between the different parts of the report.

When you use the TEXT report format option, the output is a single .txt file.

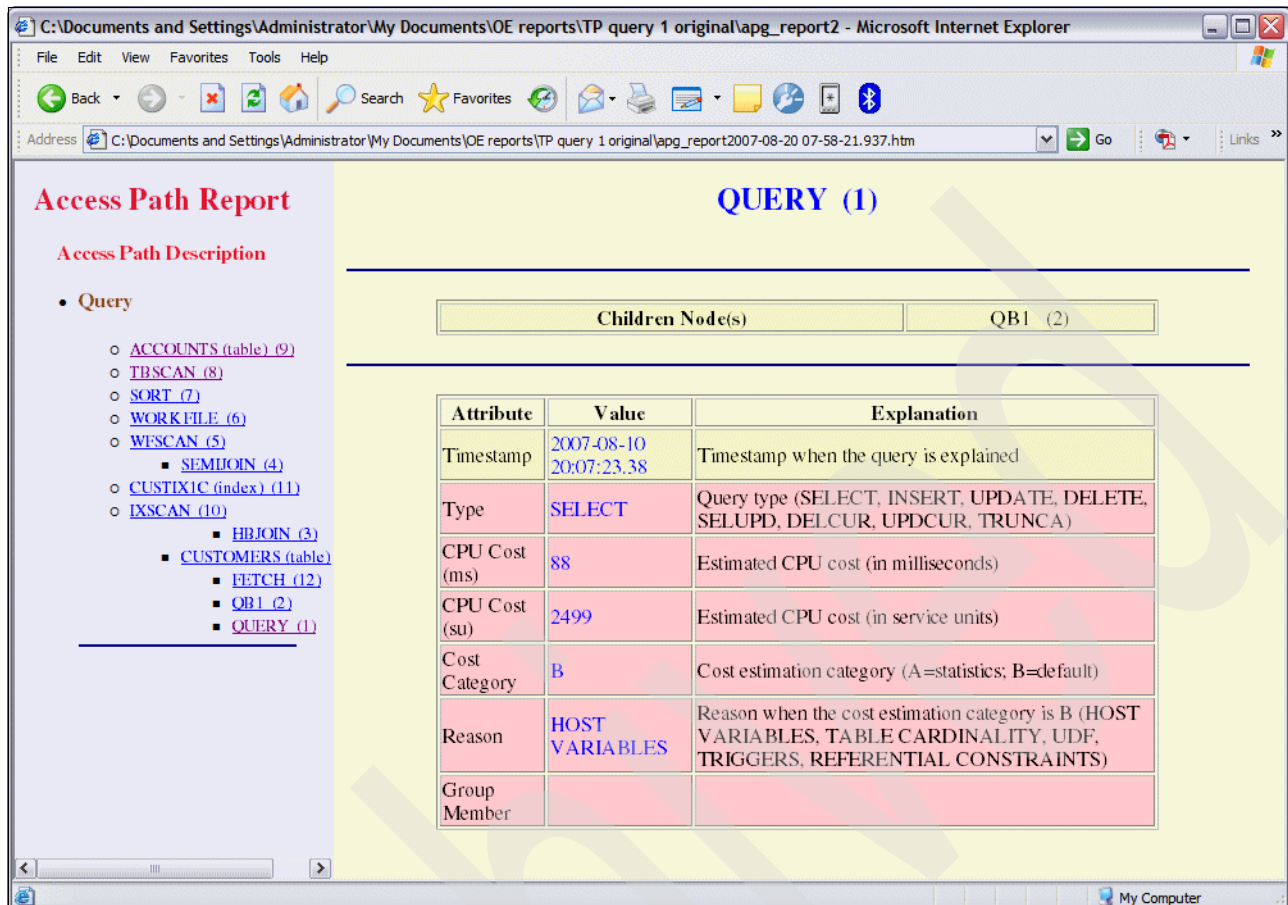



Figure 8-17 Access Path Report in HTML format

- The last button, , changes the settings used by the access path graph feature. You can change the colors and shape of all the nodes, but there is normally not many reasons to do so. The last tab, "Label", is more interesting, as shown in Figure 8-18 on page 180. It selects which type of information you want to see displayed on the table, index, and operator node in the access path graph.

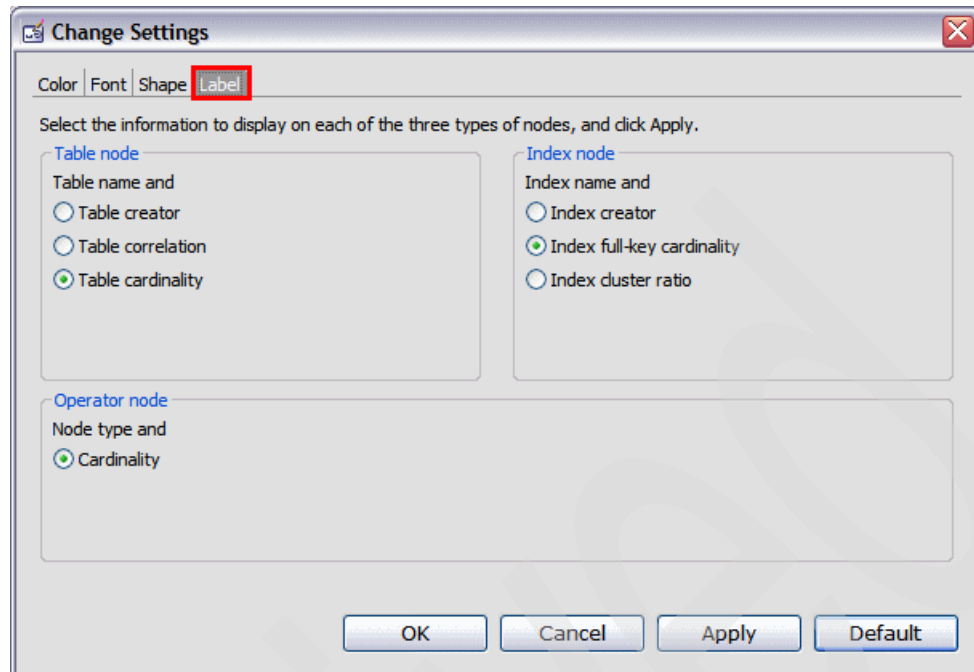




Figure 8-18 Change Settings window

## 8.3 Query Annotation tool

The *Query Annotation* tool, as the name suggests, annotates the query that you are analyzing. It nicely formats the query and also adds additional statistical information. You can invoke the Query Annotation feature:

- From the toolbar at the top of the Optimization Service Center or Optimization Expert window:  .
- Or, as an option from the **Tools** button (same icon) on the Query text window:  Query Annotation .

The result of running the query annotation is shown in Figure 8-19 on page 181.

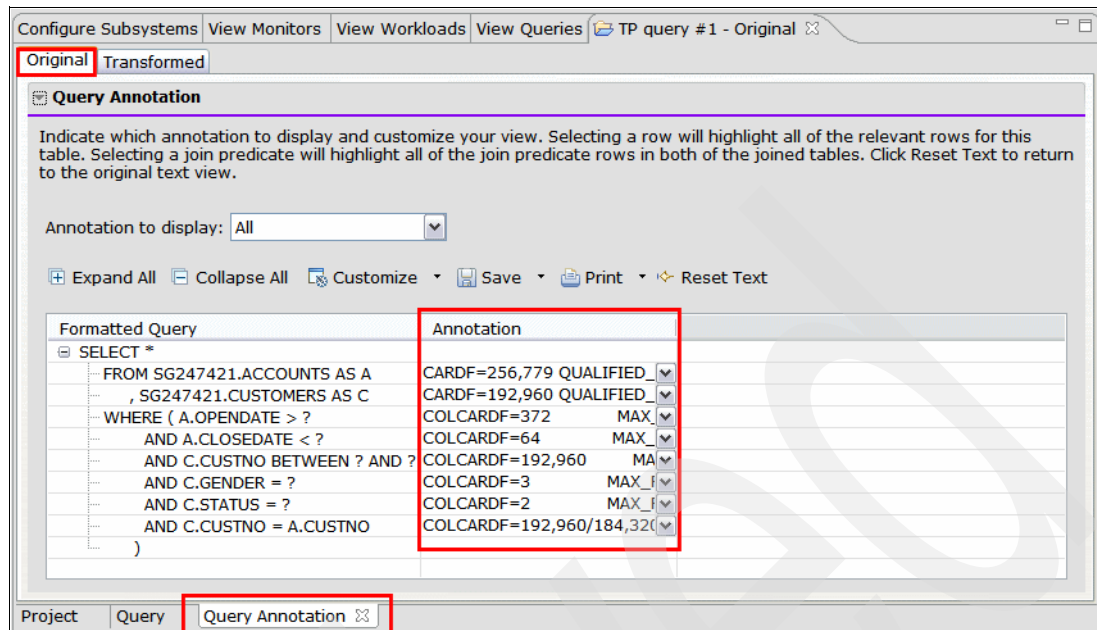


Figure 8-19 Original query in Query Annotation

Running query annotation adds a new “Query Annotation” tab to the project. As you can see, the query itself has been nicely formatted which is a huge time saver in its own, and a formatted query is the first step to understanding the query. Also note that an “Annotation” column has been added. It contains relevant statistics related to the table, column, and predicates.

After invoking a query annotation, the original (formatted) query is shown. However, in addition to the original query, Optimization Service Center or Optimization Expert also lets you look at the query after the optimizer’s query transformation component has analyzed and potentially rewritten the query.

This is useful and powerful. Prior to Optimization Service Center or Optimization Expert query annotation, there was no way to see the query after query transformation. By looking at the explain output from the `PLAN_TABLE`, you can conclude that the query transformation had occurred. For example, the explain output shows a join, whereas the query itself is using a subselect. Now you see the actual resulting query after query transformation.

To look at the transformed query, click the **Transformed** tab as shown in Figure 8-20.

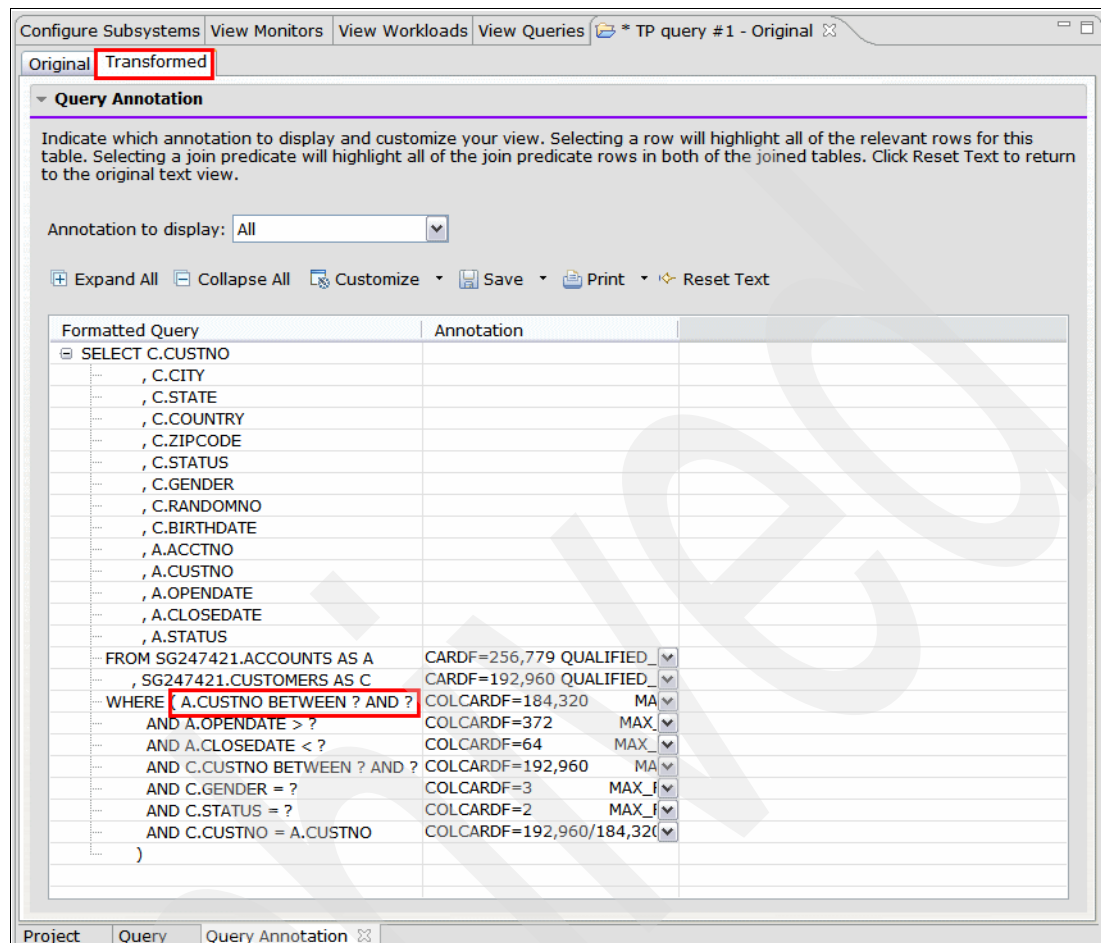


Figure 8-20 Transformed query in Query Annotation

Our current SQL statement is a simple example, and the original and the transformed query do not look different. Although when you look carefully, you notice that they are not identical. DB2 predicate transitive closure has added the following predicate:

A.CUSTNO BETWEEN ? AND ?

The original query specifies:

C.CUSTNO BETWEEN ? AND ?

AND C.CUSTNO = A.CUSTNO

This lets DB2 derive that predicate A.CUSTNO BETWEEN ? AND ? must also be true, so the DB2 query transformation added that predicate to the query before it is presented to the optimizer. This gives the optimizer more access path choices, because it can now also consider using an index on A.CUSTNO, for example.

### 8.3.1 Annotation

As mentioned above, the annotation feature adds relevant statistics to the query to make it easier to understand the query. Figure 8-21 shows an example. An “Annotation” column is added to the window that shows the SQL statement.

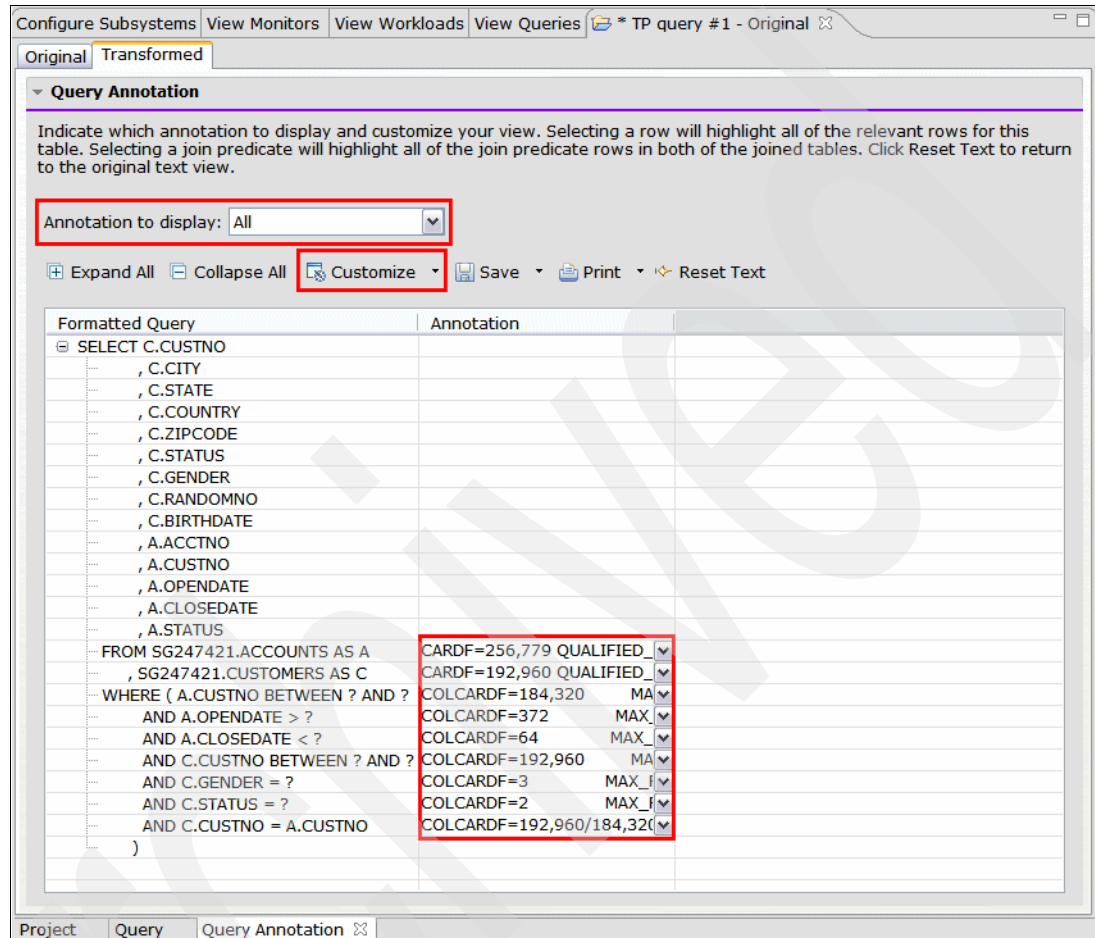


Figure 8-21 Annotated query

You can adjust the size of the “Annotation” column to show more of the fields. If all the added statistics do not fit in the column, you can use the drop down box to list them all. You can also control which data is shown in the annotation column via the “Annotation to display” drop down list, as well as the **Customize** button.

The “Annotation to display” drop down list lets you choose between:

- ▶ All
- ▶ Catalog Statistics
- ▶ Cost Estimation

The **Customize** → **Customize Annotations...** takes you to the window shown in Figure 8-22 on page 184.

Here, the type of annotations is split between table and predicate annotations. You can move the fields between the left (available) and the right (displayed) box to indicate which annotations you like to show in the annotated query.

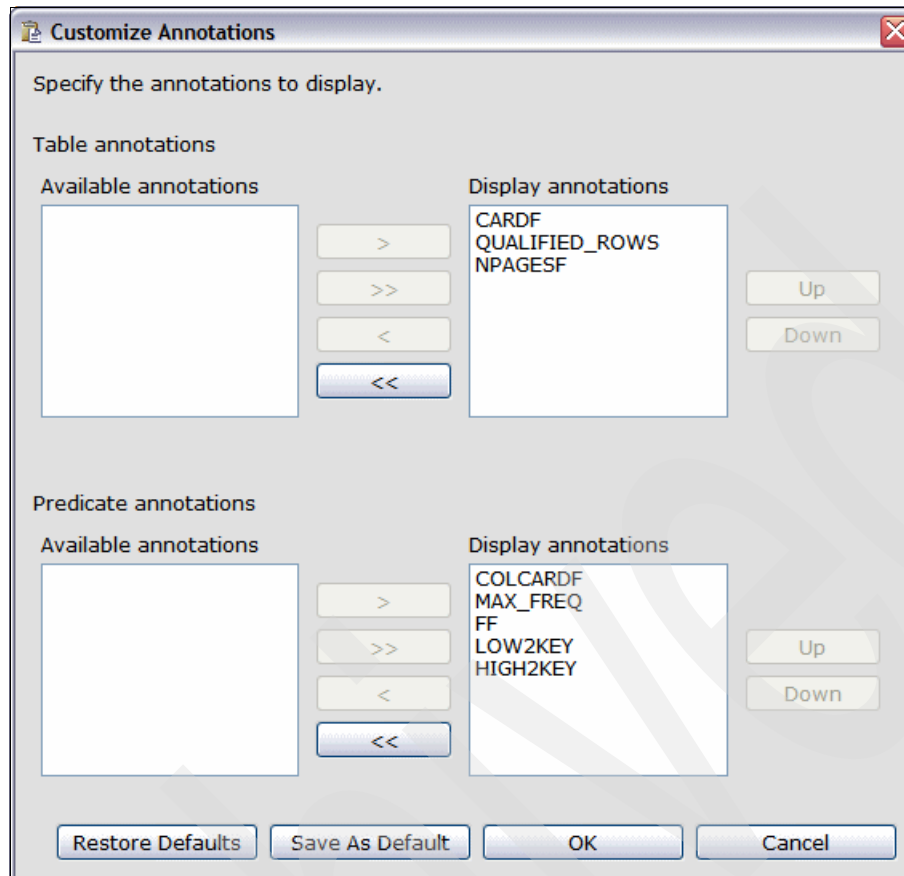


Figure 8-22 Customize Annotations window

The table/predicate annotations and catalog statistics/cost estimation interact as shown in Table 8-1.

Table 8-1 Annotation option interactions

	Catalog statistics	Cost estimation
<b>Table annotation</b>	CARDF (cardinality) NPAGESF (number of pages)	QUALIFIED_ROWS (number of qualified rows)
<b>Predicate annotation</b>	COLCARDF (cardinality) MAX_FREQ (highest collected frequency value) LOW2KEY (second lowest value) HIGH2KEY (second highest value)	FF (filter factor)



You can also change the order in which predicates are presented in the query using **Customize** → **Customize Sort Keys**, as shown in Figure 8-23.

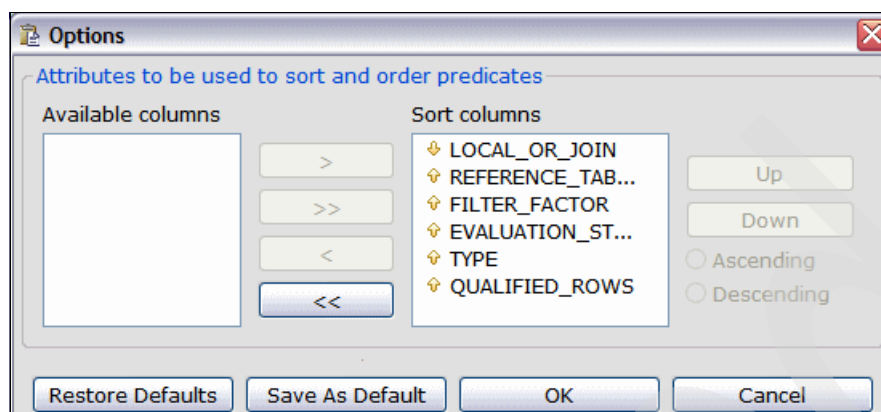


Figure 8-23 Annotation sort and order predicate options

Using the options in Figure 8-23 present the local predicates first, in the order of the tables referenced. Per table, the predicates are presented in filter factor order, the stage at which the predicate is evaluated (stage 1 or stage 2), the predicate type (for example equal, between, and so on), and the number of qualifying rows. You can remove the sort columns from the list using the arrows (<, <<), or change the sort column sequence by using the up and down button. You can also change the sort order (Ascending, Descending).

Another feature of the annotation tool is that when you click a column in the select list, a table, or a predicate, an annotation immediately highlights all the related columns and local predicates that belong to the same table, as shown in Figure 8-24 on page 186. If the predicate is a join predicate, for example, `t1.c1=t2.c2`, then only the join predicates between table `t1` and `t2` are highlighted.

As you can see, clicking the `CUSTOMERS` table highlights all the columns in the `SELECT` list and all the local and join predicates referencing the `CUSTOMERS` table. This is useful when you notice, for example, a high number of `QUALIFIED_ROWS` in the annotation for a certain table. By clicking the table, you immediately see the predicates that can help to filter down the number of rows and have a quick look at the filter factors (FF) for each of the predicates to see how much filtering the optimizer expects them to provide.

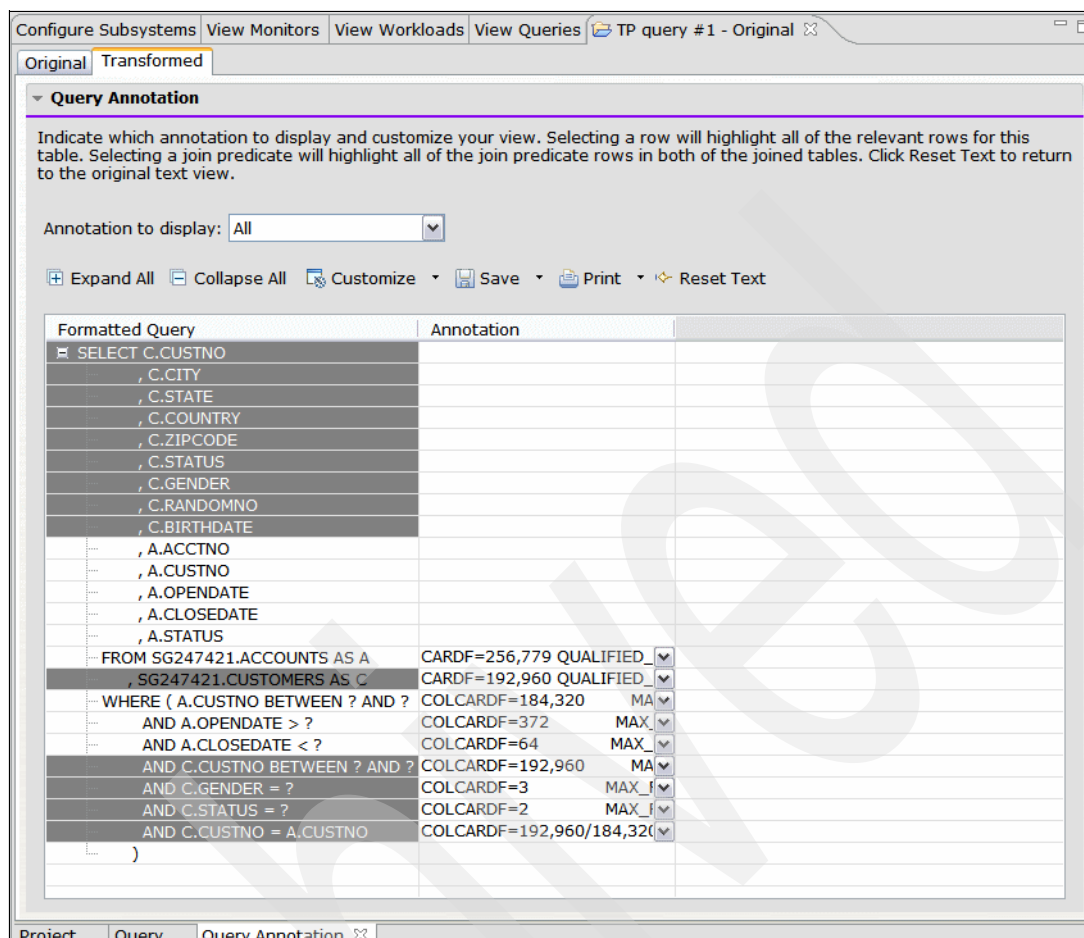


Figure 8-24 Annotation highlighting

When you are looking at a complex query, you can use the collapse and expand buttons to collapse part or all of the different parts from the query. In this simple example that we are currently using, there is no need to do this, but for large queries it lets you focus on certain parts of the query more easily. Clicking **Reset Text** will undo the highlighting.

If you are a pencil and paper person, you can also save the query to a text file or print it, using the **Save** or **Print** button. Both buttons provide the option to save or print the query, or save or print the query and the annotation data. Saving and printing is available for both the original and the transformed version of the query.

Example 8-1 shows the result of a saved query with annotation.

**Example 8-1** Saved query with annotation

---

SELECT C.CUSTNO				
, C.CITY				
, C.STATE				
, C.COUNTRY				
, C.ZIPCODE				
, C.STATUS				
, C.GENDER				
, C.RANDOMNO				
, C.BIRTHDATE				
, A.ACCTNO				
, A.CUSTNO				
, A.OPENDATE				
, A.CLOSEDATE				
, A.STATUS				
FROM SG247421.ACCOUNTS AS A	CARDF=256,779	QUALIFIED_ROWS=9.525602	NPAGESF=2,470	
, SG247421.CUSTOMERS AS C	CARDF=192,960	QUALIFIED_ROWS=32.992493	NPAGESF=3,860	
WHERE ( A.CUSTNO BETWEEN ? AND ?	COLCARDF=184,320	MAX_FREQ=(missing)	FF=9.999999310821295E-4	LOW2KEY=2 HIGH2KEY=192959
AND A.OPENDATE > ?	COLCARDF=372	MAX_FREQ=(missing)	FF=0.09999996423721313	LOW2KEY=1976-02-25 HIGH2KEY=2015-06-30
AND A.CLOSEDATE < ?	COLCARDF=64	MAX_FREQ=(missing)	FF=0.3333333134651184	LOW2KEY=1994-05-09 HIGH2KEY=9999-12-31
AND C.CUSTNO BETWEEN ? AND ?	COLCARDF=192,960	MAX_FREQ=(missing)	FF=9.999999310821295E-4	LOW2KEY=2 HIGH2KEY=192959
AND C.GENDER = ?	COLCARDF=3	MAX_FREQ=(missing)	FF=0.3333333134651184	
AND C.STATUS = ?	COLCARDF=2	MAX_FREQ=95.053%	FF=0.5	LOW2KEY=N HIGH2KEY=Y
AND C.CUSTNO = A.CUSTNO	COLCARDF=192,960/184,320	MAX_FREQ=(missing)/(missing)	FF=5.182420864002779E-6	
)				

---

Using the transformed and annotated version of the query provides a well-organized overview of the query that is presented to the optimizer. By looking at the fields added by annotation, you can immediately find the big tables (CARDF), how many rows the optimizer is expecting after applying the local predicates (QUALIFIED\_ROWS), and how much filtering the optimizer expects to be done by each of the predicates (FF). For range predicates, HIGH2KEY and LOW2KEY are also provided.

You can also change the default value of the option “always output h2k and l2k” to show HIGH2KEY and LOW2KEY for predicates other than range predicates.

Annotation also shows whether frequency statistics are present or missing (MAX\_FREQ). When present, only the highest frequency value collected is shown in the annotation (which is not necessarily the value associated with the value present in the predicate). Its intention is not to provide the actual frequency statistics for all collected column values, but it is usually enough to give you an idea of whether the data skew is present or not. You can use the report options or query graph to obtain those.

For example, Example 8-1, it is immediately obvious that frequency statistics have not been gathered for most columns. Running Statistics Advisor also points out that these statistics are missing.

One of the most common causes of poor performance is that the optimizer cannot accurately estimate the cost of the query. The root cause is often an under or overestimation of the number of rows qualifying in the individual tables, which then leads to a poor choice of the sequence in which tables are best joined together.

The QUALIFIED\_ROWS field provides you the optimizer’s estimate of the number of qualifying rows of each table after applying local predicates. Often, the developer or DBA has a good idea about how many rows are likely to qualify, so you can contrast that number with DB2’s estimate. If DB2’s estimate is different, you can click the table and the annotation highlights all the predicates related to that table. You can analyze the individual predicates in more detail.

## Annotation usage with predicate pushdown

Lets look at another example that shows the value of query annotation. Looking at the query in Example 8-2, it seems to be a simple query.

### Example 8-2 SQL statement

```
SELECT DISCOUNT
      , SUM(QUANTITY) AS SUMQ
      , AVG(QUANTITY * PRICE) AS AVGP
FROM LITEM_UIV_NOGBY
WHERE QUANTITY <= 10
      AND DISCOUNT > 0.08
      AND ORDERKEY > 30000000
GROUP BY DISCOUNT
```

The access path resulting from the explain (Figure 8-25) on the other hand, looks complex.

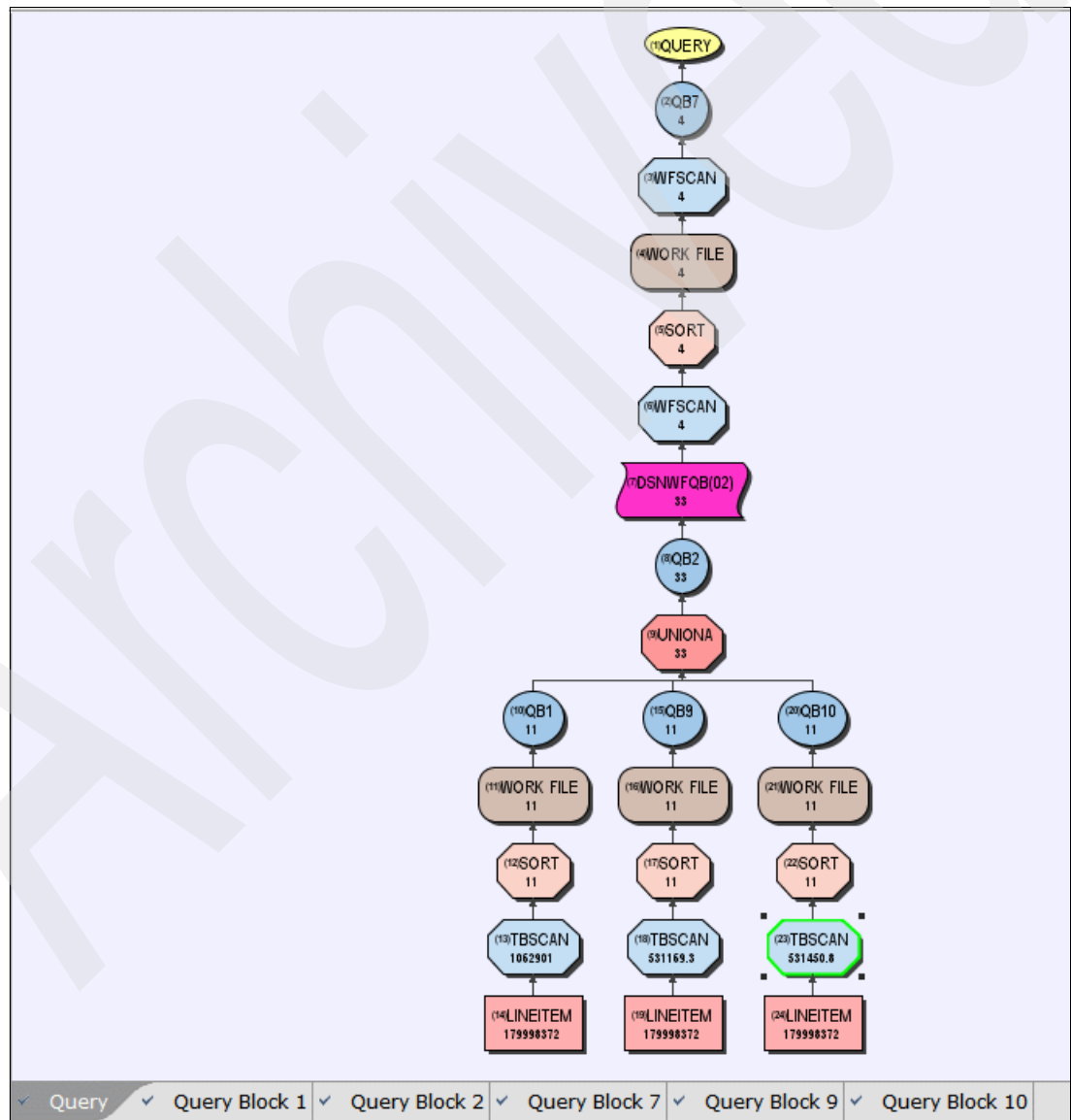


Figure 8-25 Access path graph

What the query itself does not show is that LITEM\_UIV\_NOGBY is a view made up of a number of UNION ALL statements (see Example 8-3).

*Example 8-3 CREATE VIEW statement*

---

```
CREATE VIEW LITEM_UIV_NOGBY (ORDERKEY, QUANTITY, PRICE, DISCOUNT)
AS
  SELECT L_ORDERKEY, L_QUANTITY, L_EXTENDEDPRICE, L_DISCOUNT
  FROM LINEITEM
  WHERE L_ORDERKEY BETWEEN 0000001 AND 1000000
UNION ALL
  SELECT L_ORDERKEY, L_QUANTITY, L_EXTENDEDPRICE, L_DISCOUNT
  FROM LINEITEM
  WHERE L_ORDERKEY BETWEEN 6000001 AND 9000000
UNION ALL
  SELECT L_ORDERKEY, L_QUANTITY, L_EXTENDEDPRICE, L_DISCOUNT
  FROM LINEITEM
  WHERE L_ORDERKEY BETWEEN 12000001 AND 18000000
UNION ALL
  SELECT L_ORDERKEY, L_QUANTITY, L_EXTENDEDPRICE, L_DISCOUNT
  FROM LINEITEM
  WHERE L_ORDERKEY BETWEEN 30000001 AND 45000000
UNION ALL
  SELECT L_ORDERKEY, L_QUANTITY, L_EXTENDEDPRICE, L_DISCOUNT
  FROM LINEITEM
  WHERE L_ORDERKEY BETWEEN 60000001 AND 75000000
UNION ALL
  SELECT L_ORDERKEY, L_QUANTITY, L_EXTENDEDPRICE, L_DISCOUNT
  FROM LINEITEM
  WHERE L_ORDERKEY BETWEEN 120000001 AND 150000000;
```

---

Another feature of query annotation is that it can put the view's or MQT's detail information, after the first appearance of a view, in the before-transformation query annotation. Therefore, by checking the before-transformation result, you can read the view's or MQT's definition.

When you suspect that something is wrong with the access path of this query, or want to analyze whether the access path chosen by the optimizer for this query is efficient, using the transformed annotated query shows its full power.

The transformed, annotated query is shown in Figure 8-26 on page 190. You can expand each leg of the union in view by clicking the + button. To reduce the size of the figure, only the first leg is expanded.

Notice that:

- The columns in the SELECT clause of the query, the predicates in the WHERE clause, and the GROUP BY have been pushed down into the view.

- There are only three UNION ALL legs in the transformed query. The original view definition has five. Two legs have been pruned by the query transformation process. Looking at the query in more detail, the ORDERKEY > 30 000 000 predicate eliminates the first two legs:

```

...
WHERE L_ORDERKEY BETWEEN 0000001 AND 1000000
UNION ALL
...
WHERE L_ORDERKEY BETWEEN 6000001 AND 9000000

```

The transformed query shows the query as it is presented to the optimizer for access path selection. This way, you no longer have to guess which transformations took place.

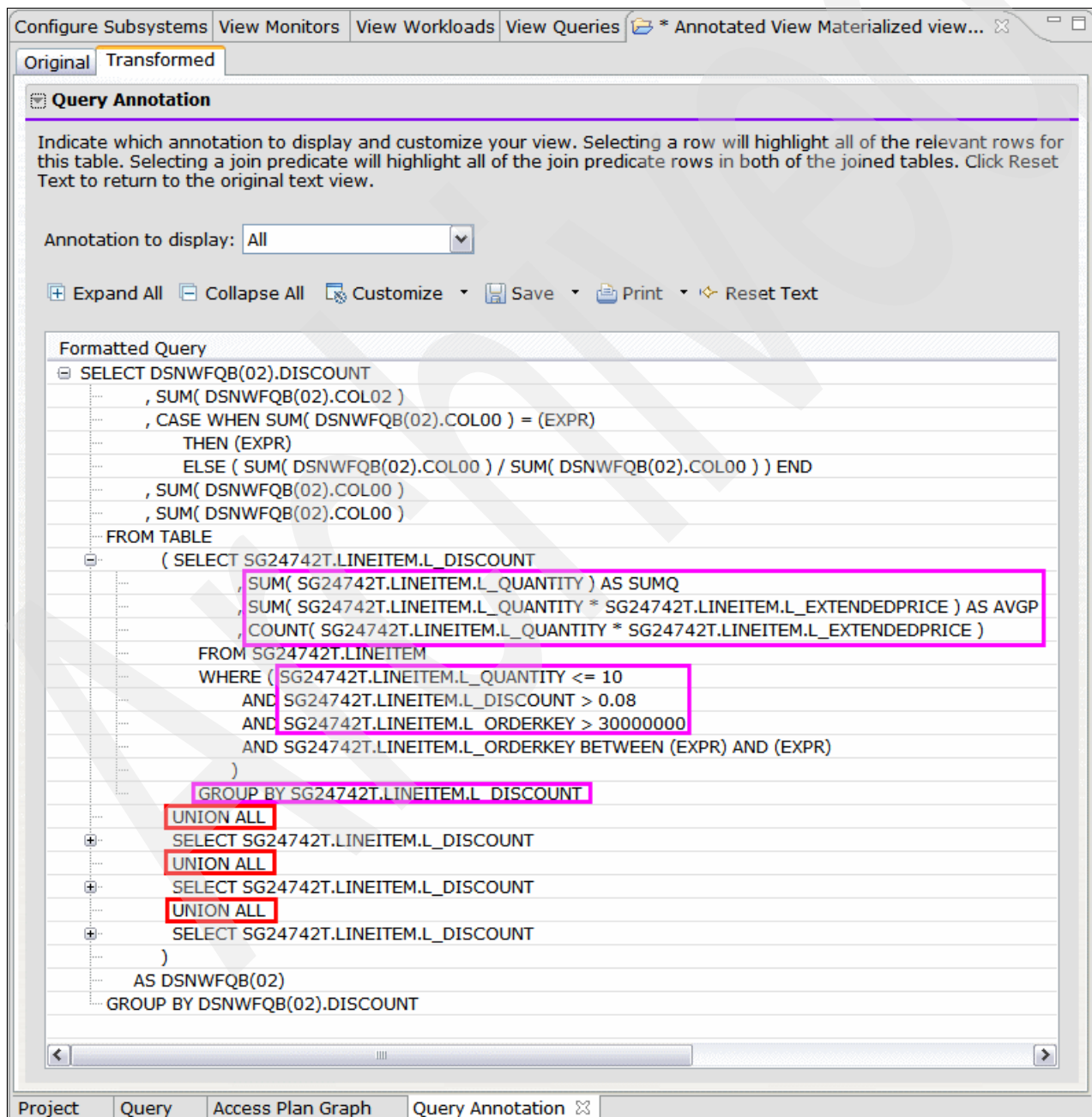

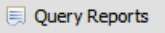


Figure 8-26 Transformed annotated query

## 8.4 Query reports

Analyzing queries in more detail, for example to diagnose an access problem, Optimization Service Center or Optimization Expert will allow you to create a number of query reports. To create them, use the **Query Reports** button:

- ▶ From the tool bar at the top:  .
- ▶ Or, as an option from the **Tools** button (same icon) on the Query text window:  
 .

To illustrate, we use the same query as shown in Figure 8-8 on page 171. After selecting the Query Reports option, the Report tab opens (see Figure 8-27).

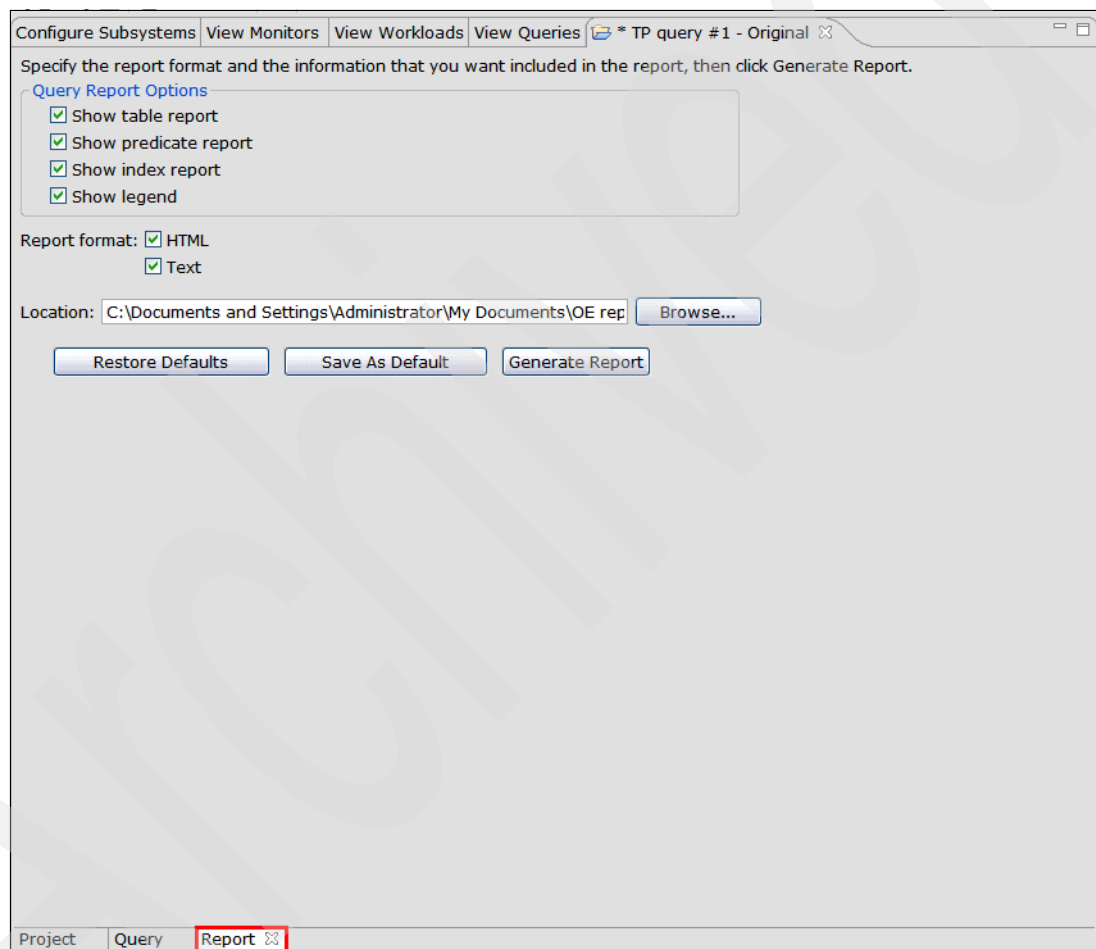


Figure 8-27 Query reports tab

The Report tab selects which reports you want to produce (table, predicate, index) and the output format that is to be used (HTML, Text, or both). The “Show legend” option generates a heading on each report that make it easier to interpret the data in the reports. This is useful when you start using the tool, but when you are used to the report format and understand the meaning of each of the columns, you can uncheck this box.

The generated reports are not stored as part of the project, but are saved in the location that you specify in the Location: path. We recommend that you use a separate directory for each of the projects. This way, it is easy to correlate the query reports with your project information

that is stored inside the tool. The location path is stored with the project so you only need to specify it the first time you generate a query report.

After the reports are generated, you are asked whether you want to display the generated reports or not (see Figure 8-28).

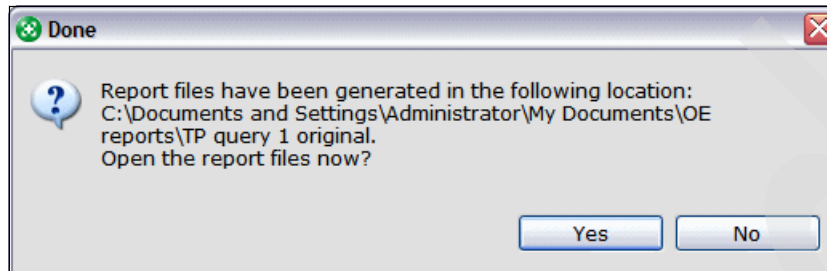


Figure 8-28 Open report files

After selecting **Yes**, a new window is opened for each of the reports you requested, either in Notepad for Text reports, or in your favorite (default) browser for the HTML reports. The data presented in the HTML and the text reports is identical.



## 8.4.1 Table report

Lets first look at the HTML version of the table report that is shown in Figure 8-29.

**Table Report**

Legend for column names that have been truncated											
PARTS	-- Number of partitions in tablespace										
QUALROWS	-- Optimizer's estimate for how many rows qualify if this table were the outer table										
CLU	-- Is the index the clustering index?										
UR	-- Unique rule										
CR	-- Clusterratiof										

TABLE_SPACE	NACTIVEF	PARTS	SEGSIZE	PG_SIZE							
TPTEST.TPTSTTS2	2700.0	1	4	4							
		TABLE	CARDF	NPAGESF	TABNO	QUALROWS					
		SG247421.ACCOUNTS	256779	2470	2	9.525602					
		INDEX	CLU	UR	NLEAF	NLEVEL	CR	KEYCOLNAME	COLCARD	MCARD	
		SG247421.ACCTIX1C	Y	U	702	3	0.999	ACCTNO	256779	256779	
		COLUMN_GROUP	MCARD								
TABLE_SPACE	NACTIVEF	PARTS	SEGSIZE	PG_SIZE							
TPTEST.TPTSTTS1	5040.0	1	4	4							
		TABLE	CARDF	NPAGESF	TABNO	QUALROWS					
		SG247421.CUSTOMERS	192960	3860	1	32.992493					
		INDEX	CLU	UR	NLEAF	NLEVEL	CR	KEYCOLNAME	COLCARD	MCARD	
		SG247421.CUSTIX1C	Y	U	528	3	1.0	CUSTNO	192960	192960	
		SG247421.CUSTIX2	N	D	251	3	0.998	STATUS	2	2	
								CITY	26	52	
		SG247421.CUSTIX3	N	D	1710	3	0.998	STATUS	2	2	
								CITY	26	52	
								CUSTNO	192960	192960	
		COLUMN_GROUP	MCARD								
		GENDER.STATUS	6.0								
		STATUS.CITY	52.0								

Figure 8-29 HTML table report

Example 8-4 shows the text version of the report.

Example 8-4 Text table report

### Table Report

```
+-----+
+ Legend for column names that have been truncated +
+ PARTS -- Number of partitions in tablespace +
+ QUALROWS -- Optimizer's estimate for how many rows +
+ -- qualify if this table were the outer table +
+ CLU -- Is the index the clustering index? +
+ UR -- Unique rule +
+ CR -- Clusterratiof +
```

+-----+

== 1 ==

TABLE_SPACE	NACTIVEF	PARTS	SEGSIZE	PG_SIZE				
TPTEST.TPTSTTS2	2700.0	1	4	4				
TABLE	CARDF	NPAGESF	TABNO	QUALROWS				
SG247421.ACCOUNTS	256779	2470	2	9.525602				
INDEX	CLU	UR	NLEAF	NLEVEL	CR	KEYCOLNAME	COLCARDF	MCARDF
SG247421.ACCTIX1C	Y	U	702	3	0.999	ACCTNO	256779	256779
COLUMN_GROUP	MCARDF							

== 2 ==

TABLE_SPACE	NACTIVEF	PARTS	SEGSIZE	PG_SIZE				
TPTEST.TPTSTTS1	5040.0	1	4	4				
TABLE	CARDF	NPAGESF	TABNO	QUALROWS				
SG247421.CUSTOMERS	192960	3860	1	32.992493				
INDEX	CLU	UR	NLEAF	NLEVEL	CR	KEYCOLNAME	COLCARDF	MCARDF
SG247421.CUSTIX1C	Y	U	528	3	1.0	CUSTNO	192960	192960
SG247421.CUSTIX2	N	D	251	3	0.998	STATUS	2	2
SG247421.CUSTIX3	N	D	1710	3	0.998	CITY	26	52
						STATUS	2	2
						CITY	26	52
						CUSTNO	192960	192960
COLUMN_GROUP	MCARDF							
(GENDER,STATUS)	6.0							
(STATUS,CITY)	52.0							

Note the legend at the top of both reports. It clarifies the more cryptic column abbreviations.

The following information is presented in the table report for each table. A new table is indicated by “== x ==”.

- ▶ The *table space* that contains the table and some of its characteristics, such as:
  - The number of active (formatted) pages in the table space (NACTIVEF)
  - The number of partitions in the table space (PARTS)
  - The SEGSIZE for segmented or universal table space
  - The page size (PG\_SIZE) that is used for the table space
- ▶ The *table* related information in the report is:
  - The number of rows (CARDF)
  - The number of pages used by the table (NPAGESF)
  - The number of qualifying rows after applying local predicates (QUALROWS)
- ▶ The *index* related information consists of:
  - Whether or not the index is the clustering index (CLU)
  - Whether the index is unique or allows duplicates, the unique rule (UR)
  - The number of index leaf pages (NLEAF)

- The number of index levels (NLEVEL)
- The clusterratio (CR)

Note that index information is gathered for all available indexes, not just for the indexes that are used by the current access path. This lets you evaluate whether other indexes can provide a better access path than the current one.

- For each *index column*:
  - The column cardinality (COLCARDF)
  - The multi-column cardinality or correlation statistics (MCARDF):
    - If index statistics have been collected, MCARDF for the first (FIRSTKEYCARDF) and the last column (FULLKEYCARDF is present).
    - If KEYCARD was specified when running RUNSTATS, you also see values for the other index key combinations. If not, -1 is displayed. For example, the CITY column in the CUSTIX3 index has an MCARD of 52. This means that the number of distinct values of the STATUS and CITY columns combined is 52. If KEYCARD had not been used on the RUNSTATS statement, CITY would show -1.
    - Note that there is no correlation between STATUS and CITY. There are two different STATUSes, and there are 26 CITYs and the number of distinct values of the columns combined is 52, which means that each status occurs in each city, which is to be expected.
- For each column group that the statistics were gathered for, you also see the column combination and the MCARDF value. In Example 8-4 on page 193, you can see that the ACCOUNTS table had no COLGROUP statistics, but CUSTOMERS has COLGROUP statistics on:
  - GENDER,STATUS and
  - STATUS,CITY

## 8.4.2 Index reports

A sample index report (in text format) is shown in Example 8-5. There is an equivalent HTML version with identical information, so it is not included here.

*Example 8-5 Text index report*

```

                                Index Report
+-----+-----+-----+-----+-----+
+ Legend for column names that have been truncated                                     +
+ INDEX_ONLY    --The access path does not require any data pages because the access information is available in the index. +
+ ONE_FETCH     --Retrieving only one row.                                           +
+ EQUAL_UNIQUE  --An index that is fully matched and unique, and in which all matching predicates are equal-predicates.  +
+ GB_OB_DIST    --Avoiding sort for GROUP BY, ORDER BY and DISTINCT.                 +
+-----+-----+-----+-----+-----+

== QB:1, PLAN: 1 ==
=====
TABLE          CORR_NAME
SG247421.ACCOUNTS  A
=====
INDEX          INDEX_ONLY  ONE_FETCH  EQUAL_UNIQUE  GB_OB_DISTINCT
SG247421.ACCTIX1C  N          N          N              N
=====
KEYCOL         ORDER        COLUMN_CARD  MULTI_COL_CARD  PRED
ACCTNO         ASCENDING   256779.0     256779.0
=====

== QB:1, PLAN: 2 ==
=====

```

**TABLE** CORR\_NAME  
SG247421.CUSTOMERS C

INDEX	INDEX_ONLY	ONE_FETCH	EQUAL_UNIQUE	GB_OB_DISTINCT
SG247421.CUSTIX1C	N	N	Y	N
<b>KEYCOL</b>	<b>ORDER</b>	<b>COLUMN_CARD</b>	<b>MULTI_COL_CARD</b>	<b>PRED</b>
CUSTNO	ASCENDING	192960.0	192960.0	C.CUSTNO BETWEEN (EXPR) AND (EXPR) (FF:9.999999310821295E-4) C.CUSTNO=A.CUSTNO (FF:5.182420864002779E-6)
INDEX	INDEX_ONLY	ONE_FETCH	EQUAL_UNIQUE	GB_OB_DISTINCT
SG247421.CUSTIX2	N	N	N	N
<b>KEYCOL</b>	<b>ORDER</b>	<b>COLUMN_CARD</b>	<b>MULTI_COL_CARD</b>	<b>PRED</b>
STATUS	ASCENDING	2.0	2.0	C.STATUS=(EXPR) (FF:0.5)
CITY	ASCENDING	26.0	52.0	
INDEX	INDEX_ONLY	ONE_FETCH	EQUAL_UNIQUE	GB_OB_DISTINCT
SG247421.CUSTIX3	N	N	Y	N
<b>KEYCOL</b>	<b>ORDER</b>	<b>COLUMN_CARD</b>	<b>MULTI_COL_CARD</b>	<b>PRED</b>
STATUS	ASCENDING	2.0	2.0	C.STATUS=(EXPR) (FF:0.5)
CITY	ASCENDING	26.0	52.0	
CUSTNO	ASCENDING	192960.0	192960.0	C.CUSTNO BETWEEN (EXPR) AND (EXPR) (FF:9.999999310821295E-4) C.CUSTNO=A.CUSTNO (FF:5.182420864002779E-6)

The index report also starts with a legend (if requested) to explain the less obvious abbreviations in the report.

The index report is organized by query block, and plan number (as indicated in the PLAN\_TABLE). The report has an entry for:

- ▶ Each *table* accessed

And for each table, the available indexes are listed. For each *index* the reports lists if:

- ▶ Index only access is used (INDEX\_ONLY).
- ▶ One fetch access is used (ONE\_FETCH).
- ▶ The index is fully matched and unique (and all matching predicates are equal predicates - EQUAL\_UNIQUE).
- ▶ A group by, order by, or distinct sort was avoided (GB\_OB\_DISTINCT).

For each index column, the report shows:

- ▶ The index order, ascending or descending (ORDER).
- ▶ The column cardinality of the index column (COLUMN\_CARD).
- ▶ The multi-column cardinalities (MULTI\_COL\_CARD) when present.
- ▶ The predicates that can be satisfied using this index column (PRED) and the filter factor (FF) for each of these predicates.

Frequency statistics are not shown in the index report. They are included in the table report.

### 8.4.3 Predicate reports

The last report to discuss is the predicate report. An example is shown in Example 8-6 on page 197. The report has been slightly changed so it can fit into the book. The parts in *italic* are repeated to improve readability of the report.

### Example 8-6 Text predicate report (adapted)

```
+-----+
+ Legend for column names that have been truncated +
+ FF --Filter factor +
+ BT --Whether this predicate is a boolean term predicate +
+ S1 --Whether the predicate is a stage 1 predicate +
+ JN --Whether this predicate is a join predicate +
+ AJ --Whether the predicate evaluation phase is after the join +
+ PTC --Whether the predicate is generated by transitive closure,which +
+ means that DB2 generates additional predicates to provide more +
+ information for access path selection when the set of +
+ predicates that belong to a query logically imply other predicates +
+ MARKER --Whether this predicate includes host variables, parameter +
+ markers, or special registers +
+ PREDNO --The predicate number +
+-----+
```

TABLE	TABNO	COLNAME	COLNO	COLCARDF	HIGH2KEY	LOW2KEY	MAX_FREQ
SG247421.CUSTOMERS	1	CUSTNO	1	192960	192959	2	
		STATUS	6	2	Y	N	0.950533789~13
		GENDER	7	3	M	M	
SG247421.ACCOUNTS	2	CUSTNO	2	184320	192959	2	
		OPENDATE	3	372	2015-06-30	1976-02-25	
		CLOSEDATE	4	64	9999-12-31	1994-05-09	

TABLE	TABNO	COLNAME	COLNO	TYPE	OTH_TABLE	OTH_TABNO	OTH_COLNAME	OTH_COLNO	OTH_COLCARDF	OTH_HIGH2KEY	OTH_LOW2KEY
SG247421.CUSTOMERS	1	CUSTNO	1	EQUAL	SG247421.ACCOUNTS	2	CUSTNO	2	184320	192959	2
		STATUS	6	BET~EN		0	VALUE				
		GENDER	7	EQUAL		0	VALUE				
SG247421.ACCOUNTS	2	CUSTNO	2	EQUAL	SG247421.CUSTOMERS	1	CUSTNO	1	192960	192959	2
		OPENDATE	3	BET~EN		0	VALUE				
		CLOSEDATE	4	RANGE		0	VALUE				

TABLE	TABNO	COLNAME	COLNO	FF	BT	S1	JN	AJ	PTC	MARKER	PREDNO
SG247421.CUSTOMERS	1	CUSTNO	1	5.182420864002779E-6	Y	Y	Y	N	N	N	2
				9.999999310821295E-4	Y	Y	N	N	N	Y	3
		STATUS	6	0.5	Y	Y	N	N	N	Y	4
		GENDER	7	0.3333333134651184	Y	Y	N	N	N	Y	5
SG247421.ACCOUNTS	2	CUSTNO	2	5.182420864002779E-6	Y	Y	Y	N	N	N	2
				9.999999310821295E-4	Y	Y	N	N	Y	Y	8
		OPENDATE	3	0.09999996423721313	Y	Y	N	N	N	Y	6
		CLOSEDATE	4	0.3333333134651184	Y	Y	N	N	N	Y	7

The predicate report lists for each table the columns that have predicates against them in the query that is being analyzed. Remember that all these reports are on a per query basis.

As before, the report starts out with a legend (if requested). For each column, the following information is presented in the report:

- ▶ The name of the column (COLNAME)
- ▶ The number of the column in the table (COLNO)
- ▶ The column cardinality (COLCARDF)
- ▶ The second highest and lowest value in that column (HIGH2KEY, LOW2KEY)

- ▶ The highest column frequency that was collected for the column (MAX\_FREQ)
- ▶ The predicate type (equal, range... - TYPE)
- ▶ Whether the predicate references another table, such as a join predicate (OTH\_TABLE):
  - The other table's table number in the query (OTH\_TABNO)
  - The other table's column name (OTH\_COLNAME)
  - The other table's column number (OTH\_COLNO)
  - The other table's column cardinality (OTH\_COLCARDF)
  - The other table's second highest and lowest value (OTH\_HIGH2KEY, OTH\_LOW2KEY)
- ▶ The expected filtering done by the predicate (FF)
- ▶ Whether the predicate is a boolean term (BT)
- ▶ Whether the predicate is a stage 1 predicate (S1)
- ▶ Whether the predicate is a join predicate (JN)
- ▶ Whether the predicate is evaluated after the join (AJ)
- ▶ Whether the predicate is generated by predicate transitive closure (PTC)
- ▶ Whether the right hand side of the predicate is a parameter marker (MARKER)
- ▶ The predicate number (PREDNO)

As you can see, the combination of the table, index, and predicate reports contains a wealth of information related to the query you are analyzing. We suggest that you print these text or HTML reports and put them next to the information about the current access path used by the query when starting your query analysis.


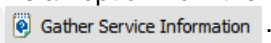
**Note:** Access Plan Graph, annotation, and query reports are recommended for SQL performance review. The access path chosen is in easy graphic or text format. Tables, indexes, and indexed columns appear automatically in the report, which avoids manual errors and interpretation.

## 8.5 Gather service information

If after analyzing the query you expect that the optimizer is mistaken about the best access path and you contact your IBM service representative, he or she might ask you to provide additional information for the service team. In the past, it often required several iterations before all the necessary documentation was provided, which was frustrating for both the customers and the IBM service staff.

The “Gather Service Information” option collects all necessary information by pressing a few buttons. This function is also known as “Service SQL”.

To gather the service information, use one of the following buttons:

- ▶ The tool bar at the top:  .
- ▶ As an option from the **Tools** button (same icon) on the Query text window:  .

### 8.5.1 Specifying SQL statement and tables to be used

The Service SQL window opens, and the current SQL statement from the query project is propagated to the SQL statement portion of the window, as shown in Figure 8-30 on page 199. You can also import a file containing an SQL statement from the file system of your workstation by using the **From File...** button.

Unfortunately, you cannot import a list of tables from a file as you can for SQL statements. This is easily circumvented by coding an SQL statement that is a cartesian join of all tables that you want to include in the Service SQL stream, and to import that SQL statement. For example, if you are interested in the Service SQL info for TAB1, TAB2, TAB3, and TAB4, simply code:

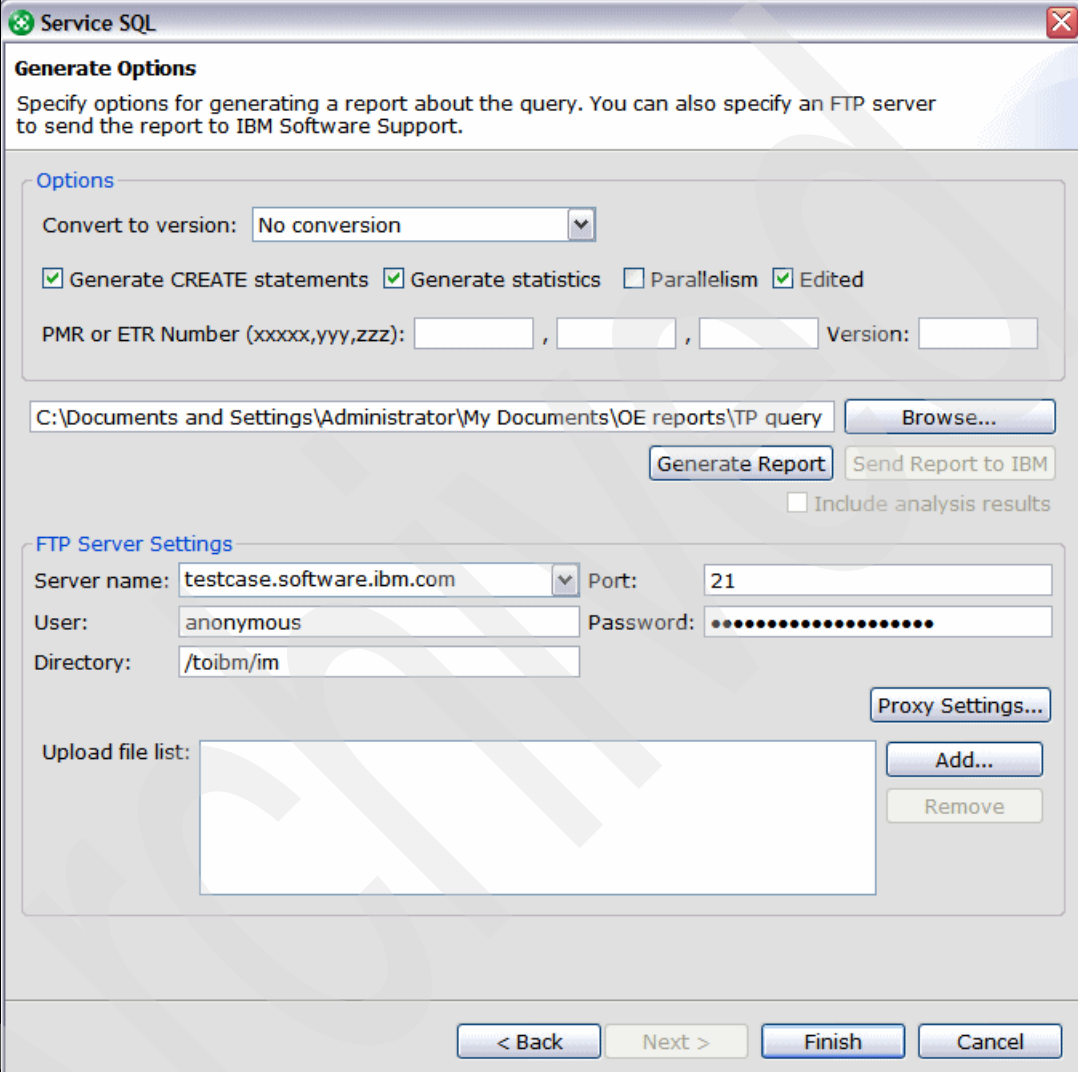
Then import that SQL statement from the file system.

*Figure 8-30 Specify objects*

Chapter 8. Query tools **199**

## 8.5.2 Specifying which data to generate

When you are satisfied with the objects listed, click **Next**. The top part of the next window, see Figure 8-31, allows you to specify which service data is to be generated. The bottom part lets you send the information directly to IBM using FTP.



The dialog box is titled "Service SQL" and contains the following sections:

- Generate Options**: A header section with a description: "Specify options for generating a report about the query. You can also specify an FTP server to send the report to IBM Software Support."
- Options**: A section containing:
  - A "Convert to version:" dropdown menu set to "No conversion".
  - Four checkboxes: "Generate CREATE statements" (checked), "Generate statistics" (checked), "Parallelism" (unchecked), and "Edited" (checked).
  - Fields for "PMR or ETR Number (xxxxx,yyy,zzz):" and "Version:".
  - A file path field showing "C:\Documents and Settings\Administrator\My Documents\OE reports\TP query" with a "Browse..." button.
  - "Generate Report" and "Send Report to IBM" buttons.
  - An unchecked checkbox for "Include analysis results".
- FTP Server Settings**: A section containing:
  - Fields for "Server name:" (testcase.software.ibm.com), "Port:" (21), "User:" (anonymous), and "Password:" (masked).
  - A "Directory:" field showing "/toibm/im".
  - A "Proxy Settings..." button.
  - An "Upload file list:" area with an "Add..." button and a "Remove" button.
- Navigation**: Buttons at the bottom for "< Back", "Next >", "Finish", and "Cancel".

Figure 8-31 Generate options

Lets have a closer look at the different options first (the top part of the window).

- *Convert to version:* This drop down list allows you to specify which DB2 version and which DB2 mode you want the data to be generated. Normally, you use the default of "No conversion". However, it might be useful to have the output generated in a way that allows it to run against a different DB2 version. For example, assume that you have an access path that changed after migrating to DB2 V9, and you want to see whether the access path is the same if you are still running against a V8 system. In that case, you can set the Convert to version to "Version 8 new-function mode". The result is a DDL and statistics file that you can execute successfully against a V8 new-function mode catalog. This option is mainly used by IBM service personnel, but it can come in handy when you want to propagate catalog statistics to a DB2 system that is at a different release.



- ▶ *Generate CREATE statements*: Checking this box triggers the service SQL function to create a file with all the DDL, related to all objects specified in the previous panel. The DDL includes the tables and indexes and it also generates the DDL for database, table space, auxiliary tables, clone tables, MQTs, synonyms, views, and aliases, if defined.
- ▶ *Generate statistics*: This triggers the creation of the statistics file that contains the catalog statistics for all objects referenced in the previous window (SQL statement, Table information).
- ▶ *Parallelism*: Checking this box indicates that Service SQL also needs to gather the additional catalog statistics that are used during access path selection when SQL query parallelism is used. Remember that information from the CARD, HIGHKEY, LOWKEY, HIGH2KEY, and LOW2KEY columns in SYSBIM.SYSCOLSTATS is used to determine the degree of parallelism. If the *Generate statistics* above is not selected, *parallelism* is ignored.
- ▶ *Edited*: The value is YES or NO. If YES, it indicates that the DDL is edited to allow the IBM service personnel to easily run the DDL on their system. To analyze the access path, most of the time the actual data is not required. For create database, create table space and create index statements, Service SQL uses SYSDEFLT as the storage group name. For PRIQTY and SECQTY, Service SQL uses a fixed value.
- ▶ *Problem management record (PMR) or electronic service request (ESR)*: If you have an open problem ticket with IBM, you can provide that information here. This way, the generated files have the correct names that you can send directly to IBM by using FTP.

**Note:** The “Version” that you specify has *nothing* to do with the DB2 version you are working with. It generates unique file names, in case you go through multiple iterations of documentation gathering. For example, after you run additional RUNSTATS to gather statistics that were previously missing, you want to generate a new set of documentation to send to IBM for analysis.

The default access path is currently the default install directory for Optimization Service Center or Optimization Expert.

When you are done, click **Generate Report**. Optimization Service Center or Optimization Expert connects to the DB2 system and extracts the requested information. When this process is finished, the generated files are presented in the bottom part of the window (Figure 8-32 on page 202), ready to be sent.

**Note:** The “Include analysis results” check box is always be grayed out because this option is currently not used by Optimization Service Center or Optimization Expert. It is reserved for future use.

**Service SQL**

**Generate Options**

Specify options for generating a report about the query. You can also specify an FTP server to send the report to IBM Software Support.

**Options**

Convert to version: No conversion

☒ Generate CREATE statements ☒ Generate statistics ☐ Parallelism ☒ Edited

PMR or ETR Number (xxxxx,yyy,zzz): 12128 , 180 , 000 Version: 1

C:\Documents and Settings\Administrator\My Documents\OE reports\TP query Browse...

Generate Report Send Report to IBM

☐ Include analysis results

**FTP Server Settings**

Server name: testcase.software.ibm.com Port: 21

User: anonymous Password: .....

Directory: /toibm/im Proxy Settings...

Upload file list:

- rts\TP query 1 original\12128.180.000.D070805.V91.FB1024.expl
- rts\TP query 1 original\12128.180.000.D070805.V91.FB1024.log
- rts\TP query 1 original\12128.180.000.D070805.V91.FB80.stats
- rts\TP query 1 original\12128.180.000.D070805.V91.FB80.ddl
- rts\TP query 1 original\12128.180.000.D070805.V91.FB1024.parm
- rts\TP query 1 original\12128.180.000.D070805.V91.FB80.sql
- rts\TP query 1 original\epInfo\_2007\_08\_05\_14\_17\_15\_052.xml
- rts\TP query 1 original\ParseInfo1186348688146.XML

Add... Remove

< Back Next > Finish Cancel

Figure 8-32 Generate Report result

The following files are generated by Service SQL:

expl	Contains the PLAN_TABLE and other explain table output for analysis by the IBM Service staff. This file is only created when an SQL statement is used as part of the input for Service SQL.
log	Contains a log of the actions performed by Service SQL.
stats	Contains the catalog statistics file. This file is generated when the “Generate statistics” box is checked.
ddl	Contains the DDL of all the objects involved. This file is generated when the “Generate CREATE statements” box is checked.
parm	Contains an extract of the DSNZPARMs of the running system and additional information required for the IBM service team to reproduce your access path. Service SQL uses the DSNWZP stored procedure to retrieve the DSNZPARM information.

sql	Contains the SQL statement that is being analyzed. This file is only created when an SQL statement is used as part of the input for Service SQL.
colst	Contains the SYSIBM.SYSCOLSTATS catalog information used by the optimizer when query parallelism is used. This file is only produced when the “Parallelism” box is checked.
.xml	The XML files are internal representations of the access path graph and related information. These files are always generated and are mostly used to debug problems with Optimization Service Center or Optimization Expert. These files are not the same as the files generated by the SAVE function.

### 8.5.3 Sending Service SQL information to IBM

The list of files that Service SQL generates is normally intended for analysis by the IBM service team. It needs to be sent to one of the FTP servers set up by IBM for this purpose. Optimization Service Center or Optimization Expert allows you to send these files directly to IBM. The most commonly used servers that upload documentation for software problems are available from the drop-down list. They are:

- ▶ testcase.software.ibm.com
- ▶ ftp.emea.ibm.com

However, you can change the names if you want to send the information elsewhere. Also, the correct directory to upload DB2 documentation is pre-coded, but you can change it. These servers allow anonymous FTP, and the password that is provided is not important. We highly recommend that you put your complete e-mail address in that field to indicate who uploaded this information in case a failure occurs. You can also provide proxy settings if your installation requires it by using the **Proxy Settings...** button.

You can also easily add or delete files from the list that was created when the reports were generated, by clicking **Add...** or **Remove**. Note that you can only remove files that you added manually. You cannot remove files generated by Optimization Service Center or Optimization Expert.

When the settings are complete you can click **Send Report to IBM** and the specified files are sent to IBM.

Service SQL makes communication with the IBM Service team much easier. The purpose is to gather all required documentation in a single, consistent way.

**Tip:** You can also use Service SQL to copy production statistics to a development or quality assurance environment, where objects are often much smaller. Having the production statistics available gives you a better chance that the SQL statements will have the same access path in the development and QA environment than in your production environment.

However, there are other factors besides statistics that affect the access path selected by the optimizer, such as the buffer pool sizes, the processing speed of the machine, and the DB2 software level.

**Note:** You get an error message if you upload files with the same names more than once. Upload files with different names or rename or delete the existing files before uploading them again.

## 8.6 Visual Plan Hint

When using parameter marker or host variables, the optimizer has to make assumptions about predicate filter factors. Because the actual values are unknown at bind or prepare time, chances are that the optimizer estimates incorrectly and picks a non-optimal access path.

There are several ways to “fix” such a bad access path. Which option is best depends on the situation. For static SQL, the use of REOPT(ALWAYS) is usually the recommended way to resolve this problem. However, there are occasions where re-optimization at runtime using the actual values is too expensive. For example, when the query itself uses little CPU time, re-optimization can add a substantial percentage of CPU time to the transaction. For such cases, the use of optimization hints is a better solution, and Visual Plan Hint (VPH) provides an easy way to implement optimization hints.

### 8.6.1 Introducing optimization hints

Optimization hints were introduced in DB2 Version 6. Optimization hints allow you to direct the optimizer towards using a predefined access path. The predefined access path must be specified through rows in the PLAN\_TABLE. The design objective allows for easy fallback to a previously existing access path in case the optimizer’s access path selection algorithm picks an access path that does not perform as well as the older one after a REBIND.

Before you can start using optimization hints, enable the functionality by specifying OPTHINTS=YES DSNZPARM. You also need a PLAN\_TABLE that has at least the 49-column format (which means it includes the OPTHINTS, HINT\_USED, PRIMARY\_ACCESSTYPE columns). If you have Optimization Service Center or Optimization Expert create the PLAN\_TABLE for you, your PLAN\_TABLE will be at the latest level, which is highly recommended.


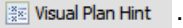
### 8.6.2 Introducing Visual Plan Hint

Optimization hints rely on information in the PLAN\_TABLE to direct the optimizer to use information in these PLAN\_TABLE rows during access path selection, instead of using the normal access path selection process.

Updating the PLAN\_TABLE or inserting correct rows into the PLAN\_TABLE so they are used as an optimization hint is not a trivial exercise in many situations. Visual Plan Hint is a GUI interface to generate optimization hints and is designed to make the process of implementing optimization hints much easier. It lets you focus on the access path change rather than on the mechanics of implementing the change. VPH lets you easily verify that the hint is correct; verification is mandatory. It provides basic consistency checking of the hint and highlights the differences between both access paths.

### 8.6.3 Invoking Visual Plan Hint

To invoke the visual plan hint, use the Visual Plan Hint button:

- ▶ From the tool bar at the top:  .
- ▶ Or, as an option from the **Tools** button (same icon) on the Query text window:  .

This invokes a VPH session inside your query project and adds a new tab to the query project, as shown in Figure 8-33 on page 206.

Note the query has slightly changed from the query that was used in the previous sections of this chapter to better illustrate the functionality provided by VPH. The new statement is shown in Example 8-7.

*Example 8-7 SQL statement*

---

```
SELECT *  
FROM ACCOUNTS A,CUSTOMERS C  
WHERE C.CUSTNO = A.CUSTNO  
      AND C.STATUS = ?  
      AND C.GENDER = ?  
      AND A.OPENDATE BETWEEN ? AND ?  
      AND A.CLOSEDATE < ?  
      AND C.BIRTHDATE > ?
```

---

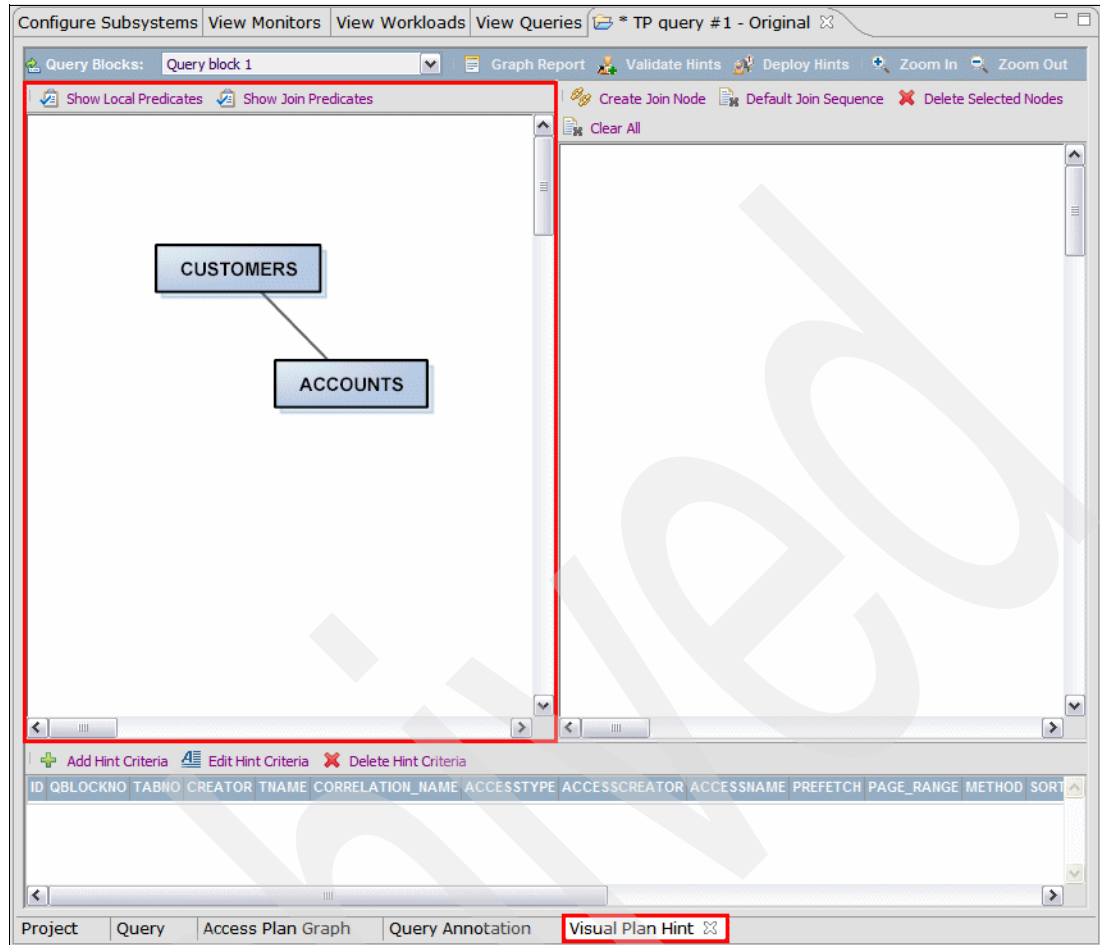




Figure 8-33 Initial VPH tab

VPH uses the free Adobe SVG viewer. If it is not installed, you are prompted to download and install it.

## 8.6.4 Visual Plan Hint join graph

When Visual Plan Hint is invoked from a query project, the current query's *join graph* is presented in the left pane (see the red box in Figure 8-33). It is a useful access path analysis display of what tables are joined to each other via join predicates and to see what local predicates are on each table.

- ▶ The different boxes represent the different tables in the query.
- ▶ The connecting lines represent the join relationships.
- ▶ By clicking the **Show Local Predicates** button, , the local predicates for each table are added to the figure.
- ▶ By clicking the **Show Join Predicates** button, , the join predicates are added to the figure.

The result is shown in Figure 8-34 on page 207.



Figure 8-34 Join graph with added predicates

### 8.6.5 Visual Plan Hint general toolbar

The tool bar at the top of the VPH tab (Figure 8-35) lets you:

- ▶ Select a query block. Remember that optimization hints operate at the query block level.
- ▶ Generate a report. This report contains the same information as the three visual windows, but in a tabular format. We will discuss the report in more detail in section 8.6.11, “Visual Plan Hint graph reports” on page 220.
- ▶ Validate the hint when it is created.
- ▶ Deploy the hint in the PLAN\_TABLE.
- ▶ Zoom in and out.

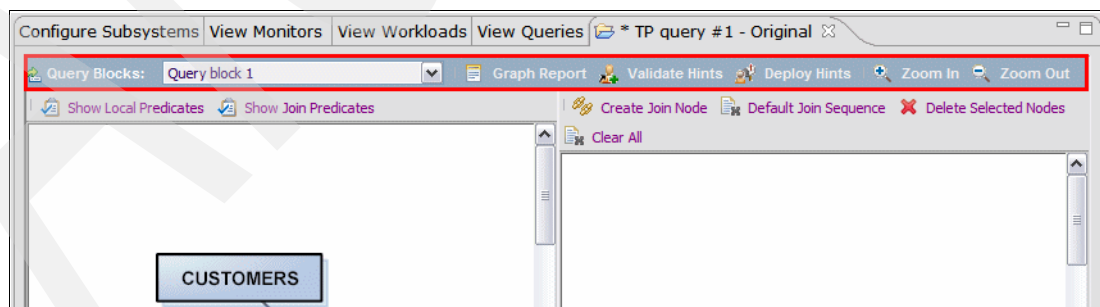


Figure 8-35 VPH top toolbar

## 8.6.6 VPH hint area

Now it is time to create a hint. The hint is created in the *hint area* window, inside the red box in Figure 8-36.

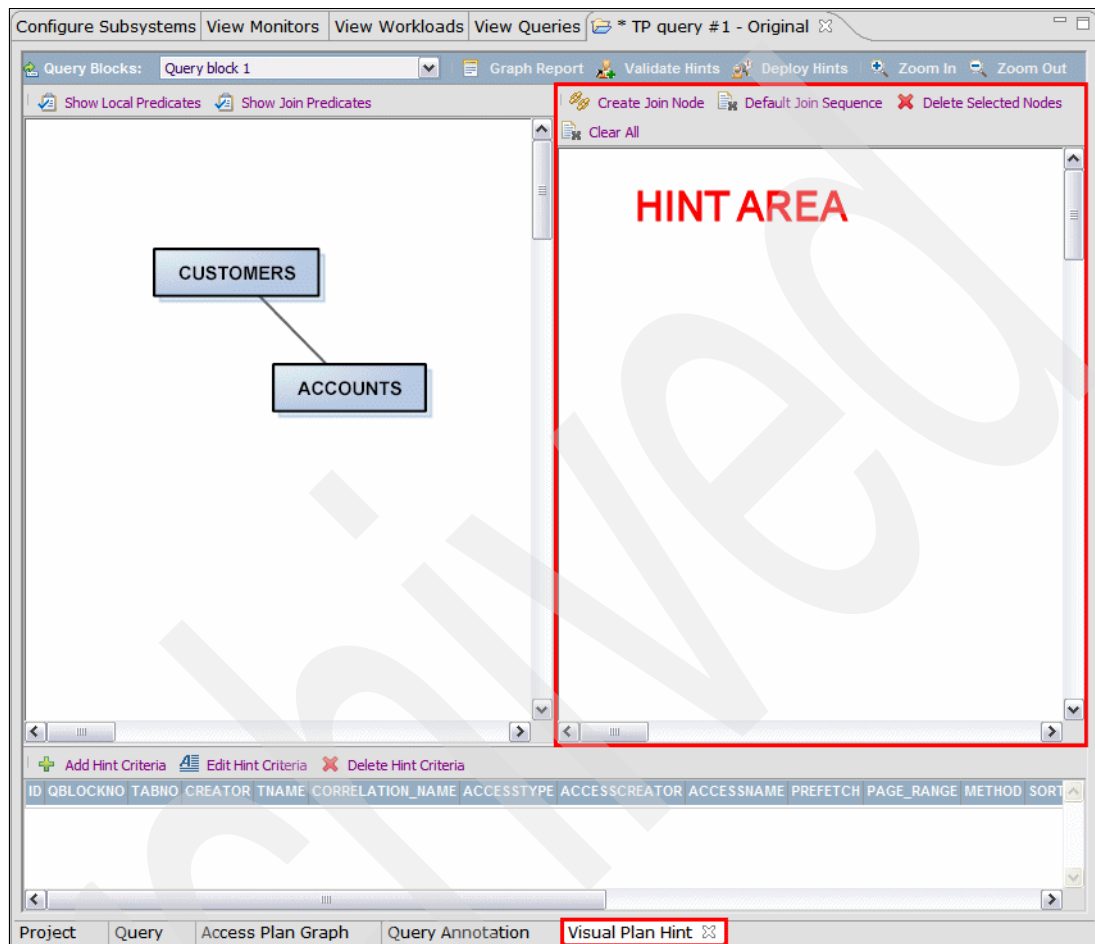



Figure 8-36 VPH Hint area

You start by clicking the **Default Join Sequence** button,  **Default Join Sequence**. This copies the join sequence of the current access path into the hint area. The result is shown in Figure 8-37.

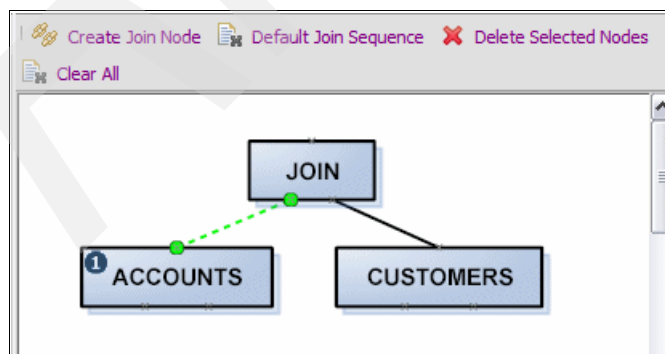


Figure 8-37 Default join sequence



Note that the ACCOUNTS table has a little number 1. This means that it is the leading table in the join. By clicking a box, you can select the table. By clicking the line between the join and a table box, you can select the link. A table or join link turns green when selected as shown in Figure 8-37 on page 208.

By double clicking a box, you open the Hint Customization Rule window, which contains the “properties” for the box. An example is shown in Figure 8-38. You can change the properties by clicking the drop down list on the right hand side. We will discuss this in more detail later in this section.

CREATOR	SG247421
TNAME	ACCOUNTS
CORRELATION_NAME	A
ACESSTYPE	RSCAN
ACCESSCREATOR	NULL
ACCESSNAME	NULL
PREFETCH	S
PAGE_RANGE	NULL
METHOD	NULL
SORTN_JOIN	N
SORTC_JOIN	N
PARALLELISM_MODE	NULL
ACCESS_DEGREE	NULL
JOIN_DEGREE	NULL
PRIMARY_ACESSTYPE	NULL
WHEN_OPTIMIZE	NULL


☒ Leading table

OK Cancel

Figure 8-38 Hint Customization Rule window

### 8.6.7 Implementing an optimization hint using Visual Plan Hint

Lets assume that the optimizer underestimated the number of qualifying rows for one of the predicates and chose this table incorrectly as the outer table of the join. Monitoring the actual filtering of this predicate, we know that reversing the join order provides better performance. To reverse the join order using VPH, you must:

1. Remove the current join relationship, either by:
  - Selecting (single-click) and deleting the lines (relationships) between the tables and the join node, or
  - Selecting and deleting the join node itself.
2. Then, you switch the ACCOUNTS and CUSTOMERS box by dragging them to the opposite side. Now the CUSTOMERS node is on the left, ACCOUNTS on the right.
3. Put back the join relationship. How to do this depends on what you deleted in Step 1.
  - If you deleted the join node, you need to insert it back using the **Create Join Node** button,  **Create Join Node**, and drag it into position.

- Add the links between the tables and join node back in. If you only deleted the relationships (lines) in Step 1, you only need to add them back now. When you move over a node, a green circle appears. You can use it as an anchor for a link to another node. When you are over a green circle, you can hold the cursor button and drag a line (link) to a green circle in another box.

If you try to connect two nodes incorrectly, you see the message as shown in Figure 8-39.

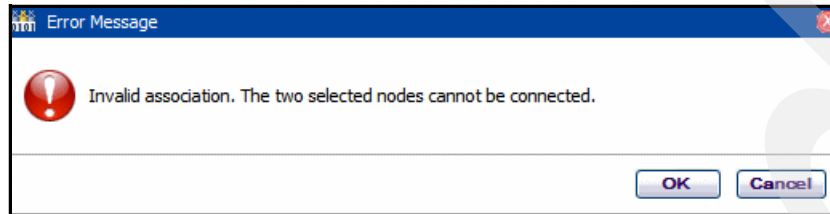


Figure 8-39 Invalid association message

When you connect the ACCOUNTS node to the join node, you receive the message shown in Figure 8-40. It is a warning message that the ACCOUNTS table can no longer be the leading table in the join (as previously indicated by the number 1 in the ACCOUNTS node).

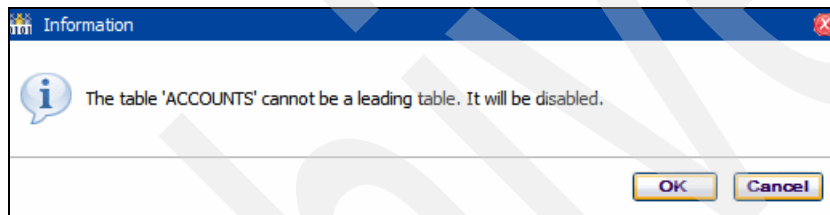


Figure 8-40 Leading table disabled message

After the switch, the hint area looks like Figure 8-41 on page 211.

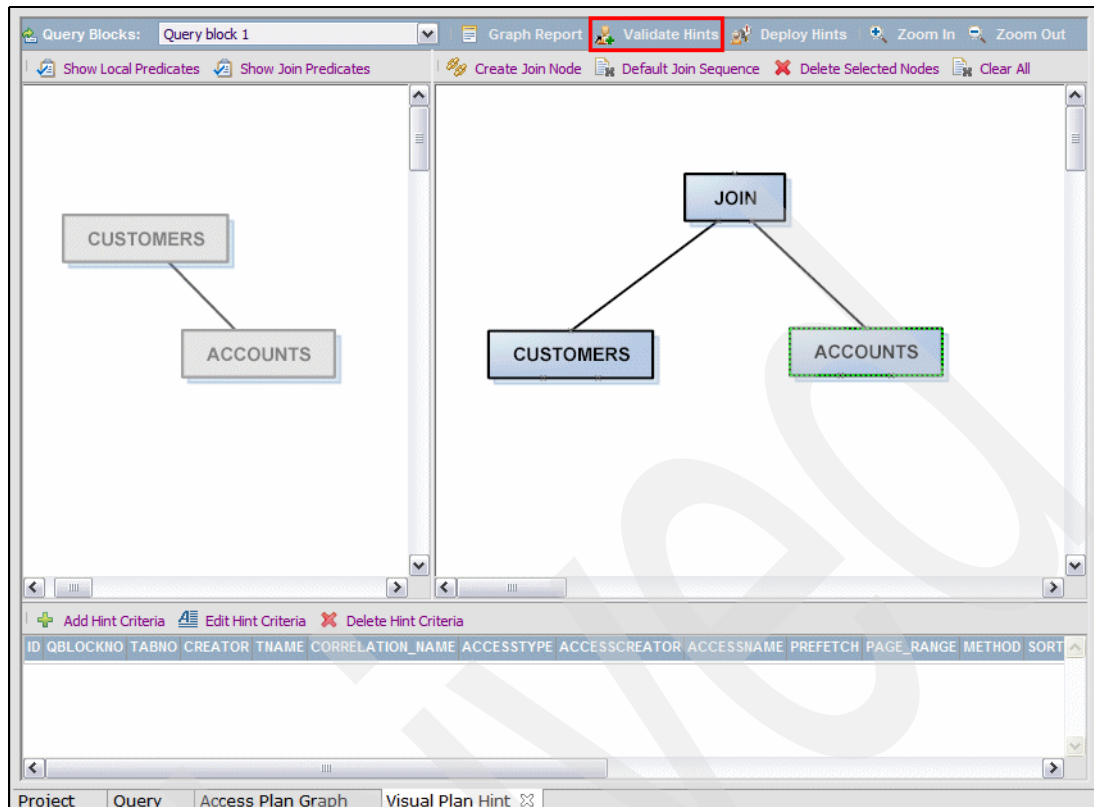


Figure 8-41 Hint area after join sequence change

## 8.6.8 Validating the hint

To validate the hint, use the **Validate Hints** button,  **Validate Hints**, located in the red box in Figure 8-41.

The result is shown in Figure 8-42 on page 212. The report is divided into a number of sections:

- ▶ The *Summary* section shows whether the hint can be successfully implemented, the name of the hint, and the SQLCODE.
- ▶ The *Generated Full Hint* section shows the access path that is derived from the hints section.
- ▶ The *Plan table without using plan hint* section shows the original access path.
- ▶ The *Plan using plan hint* section shows the result of implementing the access path. The access path is similar to the access path in the Generated Full Hint section when the hint was successfully implemented. However, for a number of things, the optimizer is always in charge. For example, the optimizer has the last say regarding the number of matching columns in the index for example, irrespective of the value specified in the hint.

The report also highlights the differences between the access path in red. This allows you to spot the differences quickly between the original access path, the hint, and the implemented hint.

http://127.0.0.1:52852 - Plan Hint Summary - Microsoft Internet Explorer

## Plan Hint Summary

Summary											
Hint Type	Full Hint										
Hint Name	VPHTest										
Hint Used	Hint is used , Hint name is:VPHTest										
SQL Code	394										
Reason Code											
SQL Statement	SELECT * FROM CUSTOMERS C, ACCOUNTS A WHERE C.CUSTNO = A.CUSTNO AND C.STATUS = ? AND C.GENDER = ? AND A.OPENDATE BETWEEN ? AND ? AND A.CLOSEDATE < ? AND C.BIRTHDATE > ?										
Generated Full Hint											
QUERYNO	QBLOCKNO	APPLNAME	PROGNAME	PLANNO	METHOD	CREATOR	TNAME	TABNO	ACCESSTYPE	MATCHCOLS	A
1129	1		SYSLH200	1	0	SG247421	CUSTOMERS	1	I	1	SC
1129	1		SYSLH200	2	1	SG247421	ACCOUNTS	2	R	0	
Plan table without using plan hint											
QUERYNO	QBLOCKNO	APPLNAME	PROGNAME	PLANNO	METHOD	CREATOR	TNAME	TABNO	ACCESSTYPE	MATCHCOLS	A
112	1		SYSLH200	1	0	SG247421	ACCOUNTS	2	R	0	
112	1		SYSLH200	2	1	SG247421	CUSTOMERS	1	I	1	SC
Plan table using plan hint											
QUERYNO	QBLOCKNO	APPLNAME	PROGNAME	PLANNO	METHOD	CREATOR	TNAME	TABNO	ACCESSTYPE	MATCHCOLS	A
1129	1		SYSLH200	1	0	SG247421	CUSTOMERS	1	I	0	SC
1129	1		SYSLH200	2	1	SG247421	ACCOUNTS	2	R	0	

Figure 8-42 Plant Hint Summary window

Looking at the report above, you can see that the join order has been switched. In the original query (plan table without using plan hint), the join order is ACCOUNTS → CUSTOMERS. The join sequence in the hint (plan table using plan hint) is CUSTOMERS → ACCOUNTS.

The join order is reversed as requested, but when you take a closer look, you see that we use a table space scan (relational scan, ACCESSTYPE = 'R') on the inner table. Even with a better join sequence, doing a table space scan on the inner table most likely does not improve performance.

The validation of the optimization hint shows the +394 / +395 SQL code and it also drives the user to actually check the plan table. In fact, it is possible to get a +394 and DB2 adds a mandatory sort. Looking at the SQLCODE, you might not realize this. VPH allows an easier full access path comparison.

### 8.6.9 Refining the hint

To make the access path (using the optimization hint) perform properly, you need to access the inner table using an index. To make this change, double-click the **ACCOUNTS** node in the hints area. The Accounts table window opens up as shown in Figure 8-43 on page 213.

Parameter	Original Value	Customized Value
CREATOR	SG247421	SG247421
TNAME	ACCOUNTS	ACCOUNTS
CORRELATION_NAME	A	A
ACCESSTYPE	RSCAN	INDEX
ACCESSCREATOR	NULL	SG247421
ACCESSNAME	NULL	ACCTIX2C
PREFETCH	S	NULL
PAGE_RANGE		NULL
METHOD	NULL	NLJ
SORTN_JOIN	N	NULL
SORTC_JOIN	N	NULL
PARALLELISM_MODE	NULL	NULL
ACCESS_DEGREE	NULL	NULL
JOIN_DEGREE	NULL	NULL
PRIMARY_ACCESTYPE		NULL
WHEN_OPTIMIZE		NULL

☐ Leading table

OK Cancel

Figure 8-43 ACCOUNTS table hints customization rule

The left side shows the original settings, the right side initially has all drop-down lists set to “null”. This means that they use the original setting of the left side. To make a change, open a drop-down list and select the proper value. For example, for the index name ACCESSNAME, the available indexes of the ACCOUNTS table are listed. We know that ACCTIX2C is on CUSTNO, the join column, so we select it. We also select NLJ as the join method. The fields that were changed shows a “push pin” in the left margin so you can quickly find the modified fields. When you are done, click **OK**.

The drop down lists allows for easy verification of index names. Often manual comparison produce typos, which prevents a successful optimization hint.

After the customization rule window closes, note that the *hints criteria* area, the bottom area of the VPH window, has changed as shown in Figure 8-44 on page 214.

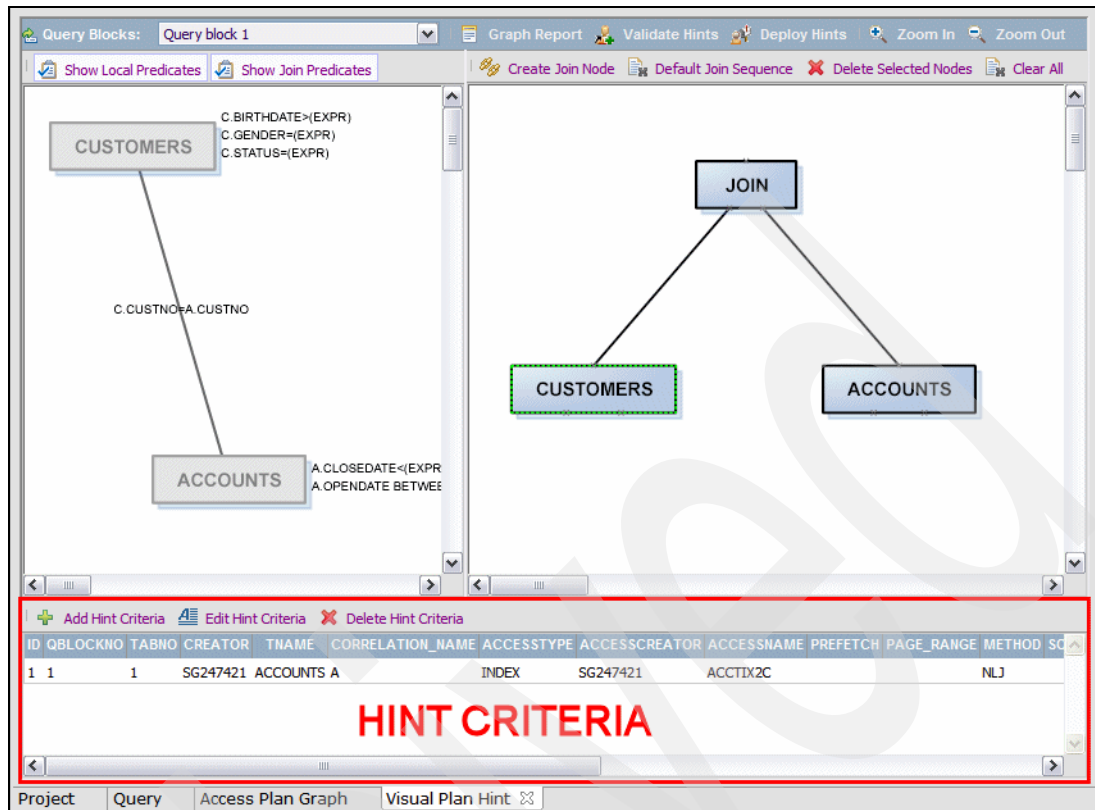


Figure 8-44 VPH Hints Criteria area

To verify the access path hint, we validate the hint again. The result is shown in Figure 8-45 on page 215. The report has been manipulated to show all the columns of interest that have changed by the optimization hint. In the actual report, these columns are further apart and require scrolling to the right.

The report shows that:

- ▶ The hint is valid and used:
  - SQLCODE +394.
  - HINT\_USED column (last column of report, and partially cut off) is non-blank.
- ▶ We are using the desired join sequence (CUSTOMERS → ACCOUNTS).
- ▶ The hint uses the ACCTIX2C index on ACCOUNTS (inner) table.

http://127.0.0.1:52852 - Plan Hint Summary - Microsoft Internet Explorer

Plan Hint Summary

Summary

Hint Type	Full Hint
Hint Name	VPHTest
Hint Used	Hint is used , Hint name is:VPHTest
SQL Code	394
Reason Code	
SQL Statement	SELECT * FROM CUSTOMERS C, ACCOUNTS A WHERE C.CUSTNO = A.CUSTNO AND C.STATUS = ? AND C.GENDER = ? AND A.OPENDATE BETWEEN ? AND ? AND A.CLOSEDATE < ? AND C.BIRTHDATE > ?

Generated Full Hint

QUERYNO	QBLOCKNO	PLANNO	METHOD	TNAME	TABNO	ACCESSTYPE	ACCESSNAME	ACCESSTYPE	PREFETCH	OPTHINT	HINT
1377	1	1	0	CUSTOMERS	1	I	CUSTIX1C	I			VPHTest
1377	1	2	1	ACCOUNTS	2	I	ACCTIX2C	I	S		VPHTest

Plan table without using plan hint

QUERYNO	QBLOCKNO	PLANNO	METHOD	TNAME	TABNO	ACCESSTYPE	ACCESSNAME	ACCESSTYPE	PREFETCH	OPTHINT	HINT
112	1	1	0	ACCOUNTS	2	R		R	S		
112	1	2	1	CUSTOMERS	1	I	CUSTIX1C	I			

Plan table using plan hint

QUERYNO	QBLOCKNO	PLANNO	METHOD	TNAME	TABNO	ACCESSTYPE	ACCESSNAME	ACCESSTYPE	PREFETCH	OPTHINT	HINT
1377	1	1	0	CUSTOMERS	1	I	CUSTIX1C	I	D		VPHTest
1377	1	2	1	ACCOUNTS	2	I	ACCTIX2C	I	S		VPHTest

Figure 8-45 hint summary after index change

However, note that we are using sequential prefetch to access the inner table. This is a left-over from the initial access path, where the ACCOUNTS table was accessed using a table space scan using sequential prefetch. Now that we are using a matching index, this is no longer required.

To change it, click the **ACCOUNTS** node and change the setting, or select the row in the hint criteria area and click the **Edit Hint Criteria** button, [Edit Hint Criteria](#).

Either way, the Hint Customization Rule window opens, and you can change the Prefetch drop-down list from null (the original value - sequential prefetch) to blank (unknown or no prefetch), as shown in Figure 8-46 on page 216.

Hint Customization Rule

QBLOCKNO: 1

CREATOR: SG247421

TNAME: ACCOUNTS

CORRELATION\_NAME: A

ACCESSTYPE: INDEX

ACCESSCREATOR: SG247421

ACCESSNAME: ACCTIX2C

PREFETCH: Blank(Unknown or no prefetch)

PAGE\_RANGE: NULL

METHOD: Blank(Unknown or no prefetch)

SORTN\_JOIN: NULL

SORTC\_JOIN: NULL

PARALLELISM\_MODE: NULL

ACCESS\_DEGREE:

JOIN\_DEGREE:

PRIMARY\_ACCESTYPE: NULL

WHEN\_OPTIMIZE: NULL

OK Cancel

Figure 8-46 Customizing prefetch

When done, click **OK**. The window closes and the hint criteria is updated as shown in Figure 8-47.

ID	QBLOCKNO	TABNO	CREATOR	TNAME	CORRELATION_NAME	ACCESSTYPE	ACCESSCREATOR	ACCESSNAME	PREFETCH	PAGE_RANGE	METHOD	SC
1	1	1	SG247421	ACCOUNTS	A	INDEX	SG247421	ACCTIX2C	Blank		NLJ	

Figure 8-47 Changed hint criteria

When looking at the hint in Figure 8-45 on page 215, the outer table is using the CUSTIX1C index. However, CUSTOMERS is now the outer table and there is no local predicate on CUSTNO (the indexed column of CUSTIX1C). Instead, we want to use CUSTIX2 on STATUS to provide filtering in the outer table.

The easiest way to make this change is to double-click the **CUSTOMERS** table node in the hint area and change the ACCESSNAME to CUSTIX2. However, we use this opportunity to demonstrate the use of the **Add Hint Criteria** button, . Clicking this button opens up an “empty” hint customization rule window. Because it is completely empty (“null”), there is more opportunity for mistakes. Therefore, we recommend to double-click a node in the hint area.

After filling in the correct values (see Figure 8-48 on page 217), click **OK**.



QBLOCKNO	1
CREATOR	SG247421
TNAME	CUSTOMERS
CORRELATION_NAME	C
ACESSTYPE	INDEX
ACCESSCREATOR	SG247421
ACCESSNAME	CUSTIX2
PREFETCH	NULL
PAGE_RANGE	NULL
METHOD	NULL
SORTN_JOIN	NULL
SORTC_JOIN	NULL
PARALLELISM_MODE	NULL
ACCESS_DEGREE	NULL
JOIN_DEGREE	NULL
PRIMARY_ACESSTYPE	NULL
WHEN_OPTIMIZE	NULL

Figure 8-48 Hint customization rule for CUSTOMERS

A new rule is added to the Hint Criteria area as shown in Figure 8-49.

ID	QBLOCKNO	TABNO	CREATOR	TNAME	CORRELATION_NAME	ACESSTYPE	ACCESSCREATOR	ACCESSNAME	PREFETCH
1	1	1	SG247421	ACCOUNTS	A	INDEX	SG247421	ACCTIX2C	Blank
2	1	1	SG247421	CUSTOMERS	C	INDEX	SG247421	CUSTIX2	

Figure 8-49 Adding a new rule

To check whether all is correct, you can validate the hint again and take a look at the report.

**Note:** When you clear the hint area by clicking **Clear All**, this does not clear the entries in the hint criteria area. You can delete those by selecting them one at a time and clicking **Delete Hint Criteria**.

### 8.6.10 Implementing the hint

Now that all changes have been made and verified, we are ready to implement the optimization hint. To deploy the hint, you can use the **Deploy Hints** button, . This opens the Hint deployment Options window (see Figure 8-50 on page 218).

OPTHINT	
QUERYNO	
APPLNAME	
PROGNAME	SYSLH200
COLLID	NULLID
VERSION	

OK Cancel

Figure 8-50 Hint Deployment Options

You need to provide the key fields that allows DB2 to identify the hint. Assume that we want to deploy this hint using the dynamic SQL sample program DSNTEP2. To identify the hint, we specify:

OPTHINT	Hint-id to be used (CUSTHINT1).
QUERYNO	The query number used by the query (1001). For static SQL, the QUERYNO is the statement number in the program for the SQL statement. To avoid an invalid because the statement line in the program has changed, it is best to use the QUERYNO clause in your SQL statements.
APPLNAME	Leave blank because DSNTEP2 is bound into a package.
PROGNAME	The program or package name (DSNTEP2).
COLLID	The collection-id that contains the DSNTEP2 package. In this case, the collection-id is the same as the package name (DSNTEP2).
VERSION	DSNTEP2 uses a blank version string.

The result is shown in Figure 8-51.

OPTHINT	CUSTHINT1
QUERYNO	1001
APPLNAME	
PROGNAME	DSNTEP2
COLLID	DSNTEP2
VERSION	

OK Cancel

Figure 8-51 Hint deployment for DSNTEP2

Click **OK** when done. This inserts the hint rows into the PLAN\_TABLE. When successful, you receive the following message shown in Figure 8-52 on page 219.

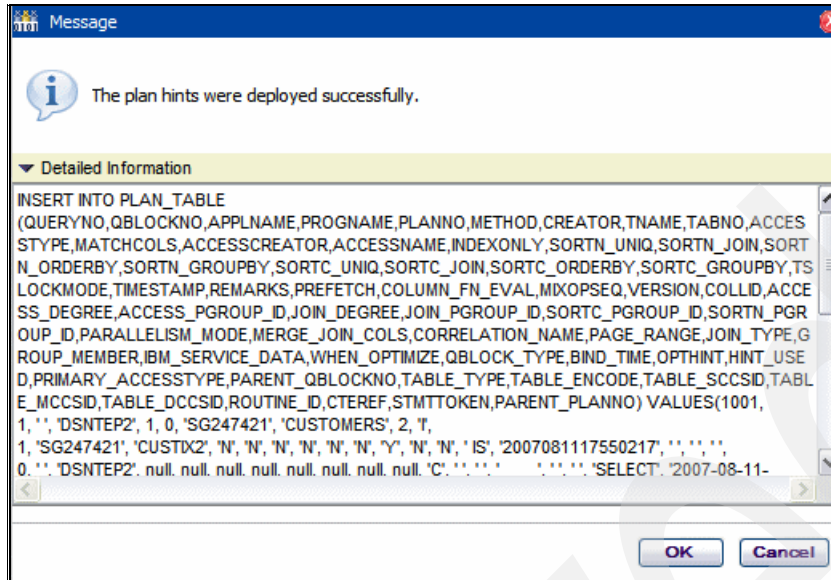


Figure 8-52 Hint successfully deployed

If a hint with the same qualifications already exists, the following window is shown (Figure 8-53).

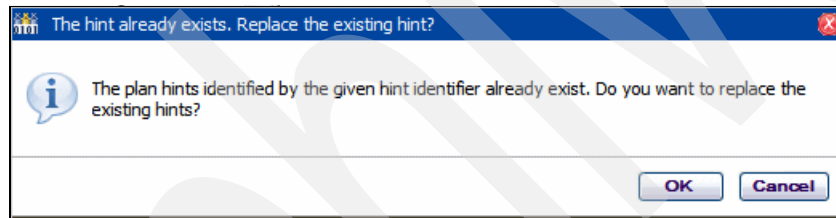


Figure 8-53 Hint already exists

To validate whether the hint works in the batch job, run the DSNTPE2 query specifying the hint, as shown in Example 8-8.

Example 8-8 Hint successfully used by DSNTPE2 at execution time


```
--#SET PREPWARN YES
***INPUT STATEMENT:
  SET CURRENT OPTIMIZATION HINT = 'CUSTHINT1';
RESULT OF SQL STATEMENT:
DSNT400I SQLCODE = 000, SUCCESSFUL EXECUTION
***INPUT STATEMENT:
  SELECT *
  FROM SG247421.CUSTOMERS C,
       SG247421.ACCOUNTS A
  ...
  QUERYNO 1001;
SQLWARNING ON SELECT  COMMAND, PREPARE  FUNCTION
RESULT OF SQL STATEMENT:
DSNT404I SQLCODE = 394, WARNING:  USER SPECIFIED OPTIMIZATION HINTS USED
                                   DURING ACCESS PATH SELECTION
```

Note that we used the “#SET PREPWARN YES” setting. This lets DSNTPE2 show the true SQLCODE at prepare time. Without this setting, DSNTPE2 returns SQLCODE 0 even though the hint is used.

As illustrated in the example above, manually constructing hints is challenging. You must understand the DB2 for z/OS access path selection to avoid making the access path worse, or to provide the optimal access path. However, in many cases the hint is only intended to make a minor change to the access path, for example using a different index. In that case, using VPH makes implementing optimization hints extremely simple. You can:

1. Explain the problem query and verify the current bad access path.
2. Invoke VPH.
3. Click **Default Join Sequence** to create a graph of the current join sequence.
4. Double-click the table node to change the index, and change to the new index.
5. Validate the hint to make sure the new index is used.
6. Deploy the newly generated hint.
7. Test whether the hint is used in real life.

## 8.6.11 Visual Plan Hint graph reports

As we are trying to provide a complete list of the functionality provided by VPH, we also mention the option to create a VPH report. The **Graph Report** button,  **Graph Report**, generates a report that is a tabular version of what is graphically presented in the Visual Plan Hint window. Depending on the actions you performed, the report has fewer or more sections.

As an example, we show the report from all the actions we took on our sample SQL statement. After all the manipulations we explained in the previous sections, the VPH tab looks like Figure 8-54.

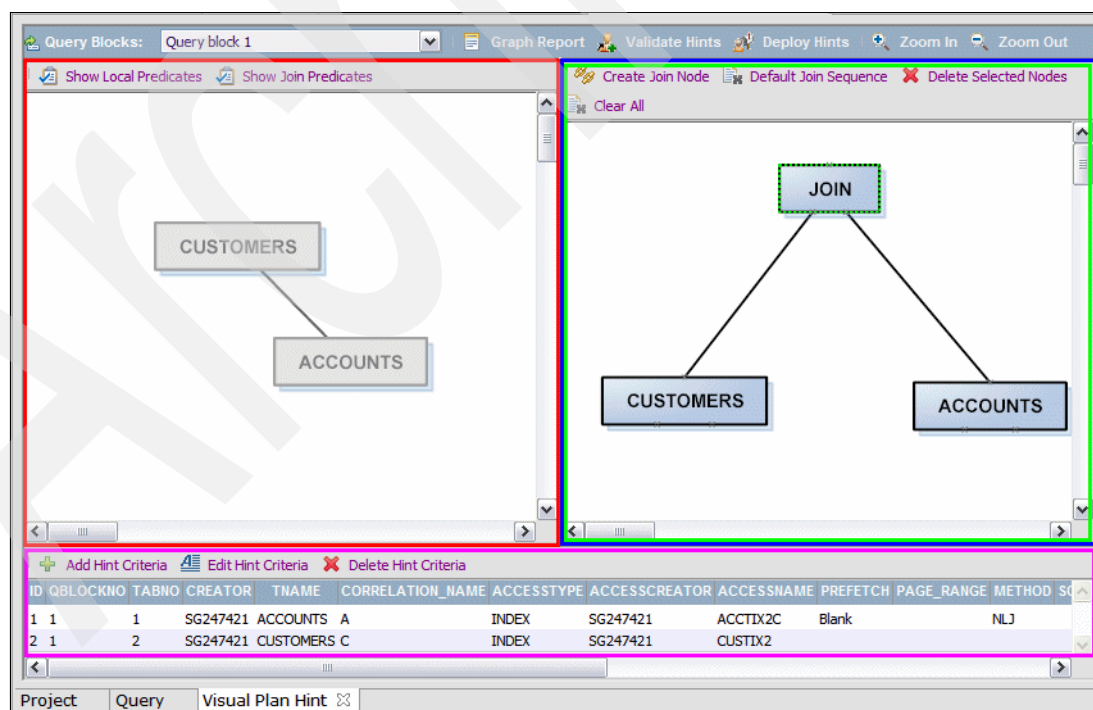


Figure 8-54 Visual Plan Hint window after implementing the hint

The resulting HTML report (only HTML format is supported for the VPH report) is shown in Figure 8-55 on page 221.

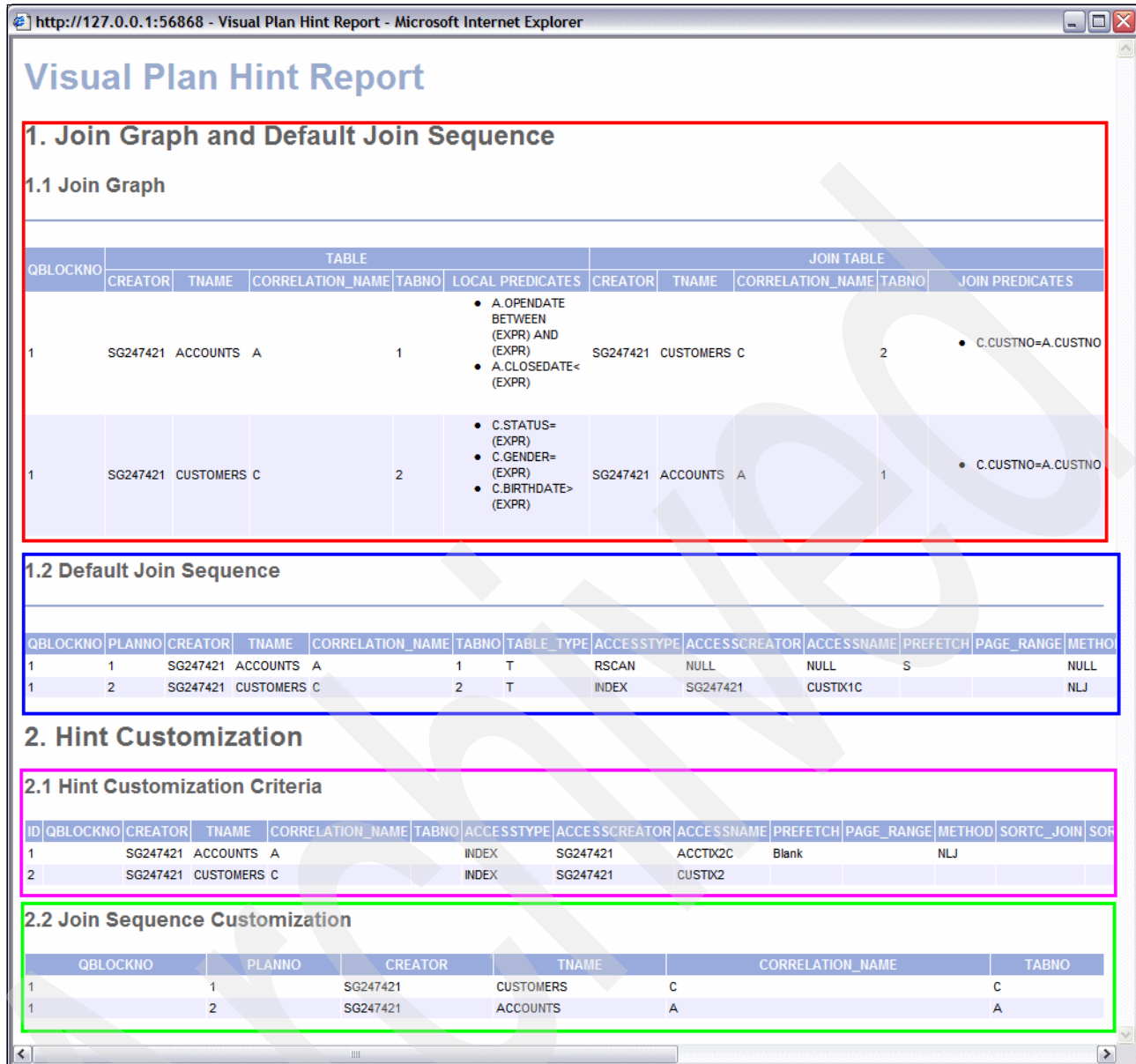


Figure 8-55 Visual Plan Hint Report

The first part of the report is called *Join Graph and Default Join Sequence*. The first section, called *Join Graph*, contains the same information that you also have in the join graph area of the VPH tab (top-left). The second section called *Default Join Sequence* lists the default join sequence. This is basically the initial (current) access path used by the optimizer. This is the same information that you see when you initially clicked **Default Join Sequence** in the top bar of the Hint area (top-right).

The second part of the report is called *Hint Customization*. It lists the customizations you performed to the access path. The first section, called *Hint Customization Criteria*, contains the same information as the hint criteria section (bottom part) of the VPH window. The *Join Sequence Customization* section describes the changes that you made to the join sequence in the hint area, dragging nodes around and reconnecting them in a different order to the join node. The report does not contain any new information. It is another way to look at the information used by the Visual Plan Hint feature.



## Statistics Advisor

The DB2 optimizer seeks out the least costly way of executing SQL. Its cost formulas are dependent on statistics gathered by the RUNSTATS utility (because DB2 does not sample the data directly). It is the installation's task to execute RUNSTATS in a timely and complete fashion and to provide the optimizer with the best information about the characteristics of the data to be accessed.

About half the access path issues reported to IBM DB2 Support are the result of incorrect statistics that do not match the data or statistics that had not been gathered at all. "Garbage in, garbage out"; in other words, this results in pervasive DB2 performance degradations that execute sub-optimal access paths optimized to the wrong set of statistics.

The problem of Access Path Statistics is a mix of practical and theoretical concerns. There are several types of statistics available, with many possible permutations. Furthermore, executing the required RUNSTATS within the utility windows can itself become an issue because RUNSTATS variations are costly to gather.

The Statistics Advisor provides a targeted solution, recommending statistics needed to optimize the SQL submitted for its analysis. It formats RUNSTATS statements, ready to run. The Statistics Advisor tool was introduced in Visual Explain ("Analyze").

Optimization Service Center includes the Statistics Advisor and extends its scope to encompass entire workloads (collections of SQL statements, culled from various sources), not only individual SQL statements. It also takes advantage of the increased usability provided by the Project framework of the Optimization Expert for z/OS.

## 9.1 Access Path statistics introduction

This section summarizes:

- ▶ The various types of statistics used for access path selection. See *DB2 Version 9.1 for z/OS Performance Monitoring and Tuning Guide*, SC18-9851.
- ▶ The RUNSTATS syntax and overhead to collect each type. See *DB2 Version 9.1 for z/OS Utility Guide and Reference*, SC18-9855.
- ▶ The types of SQL that benefit from these statistics.

Figure 9-1 shows the statistic types.

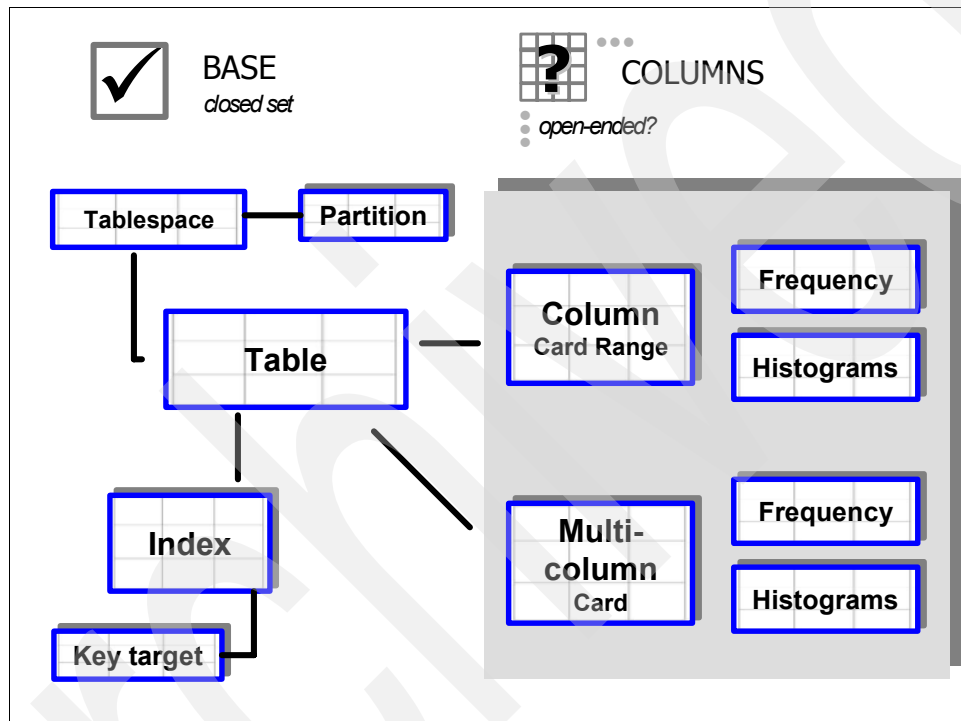


Figure 9-1 Type of statistics

DB2 can also keep a history of each of these statistics in a separate set of tables, but they are not used by the optimizer.

### 9.1.1 BASE Statistics

These statistics provide basic information about the tables and indexes used in the SQL:

- ▶ Tables (cardinality, that is # rows, # pages, compression percentage)
- ▶ Table spaces (# active pages)
- ▶ Partitions (cardinality)
- ▶ Indexes (cardinalities, clusterratio, # leaf pages, # levels)

These statistics are gathered by the basic RUNSTATS syntax:

```
RUNSTATS TABLESPACE
RUNSTATS TABLESPACE TABLE
RUNSTATS INDEX
```



Overhead is minimal, mostly I/O. We recommend to gather these statistics for all objects on a regular basis.

You can use the Statistics Advisor to format these RUNSTATS statements, then save and rerun them regularly.

## 9.1.2 COLUMN statistics

The most important factor in optimization is estimating the number of rows processed at each step in the execution of an SQL statement. For external predicates, that is, columns coded in the WHERE clause, this means the filtering provided by conditions on those columns and column groups (also called multi-column, column combinations.)

COLUMN statistics provide those filter factors and are, therefore, important to optimization. There are several types of column statistics (cardinality, range, histogram, frequency) that apply to single columns and column groups.

Considering all columns singly and all possible combinations of 2, 3, and so on, mean a large number of possible RUNSTATS syntax variations. Furthermore, many of these are expensive to gather (the ones that involve SORT especially), so assessing cost and benefit of each statistic becomes difficult. This presumes that the RUNSTATS syntax (greatly enhanced in DB2 V8) is understood in the first place.

The Statistics Advisor *solves* this problem by analyzing the predicates in the SQL of interest and *provides* all statistics and only those statistics that are needed to optimize that SQL.

In the tables below, the term “marker” refers to:

- ▶ A host variable in static SQL
- ▶ A parameter marker in dynamic SQL
- ▶ Any special register, such as CURRENT DATE

The opposite, a “literal”, is a constant value or an expression made up completely of constant values.

Table 9-1 lists all single-column statistics with examples of the types of predicates they help optimize, RUNSTATS syntax, and SYSIBM catalog tables that contain the statistics.

Table 9-1 Single-column statistics

Type of statistic	Details	Example
Single-column cardinality	Optimizes	COL = marker
	RUNSTATS	TABLE(table/ALL) - defaults to COLUMN(ALL)
		COLUMN(COLx, ...)- collects enumerated columns individually
	Catalog	SYSCOLUMNS COLCARDF
Single-column range	Optimizes	COL > literal, other RANGE operators
	RUNSTATS	TABLE(table/ALL) - defaults to COLUMN(ALL)
		COLUMN(COLx ...) - collects enumerated columns individually
	Catalog	SYSCOLUMNS HIGH2KEY, LOW2KEY

Type of statistic	Details	Example
Single-column frequency	Optimizes	COL = literal
	RUNSTATS	INDEX(index) FREQVAL - collects leading column in index
		COLGROUP(COLa) FREQVAL
		COUNT of values to collect
		MOST LEAST or BOTH
	Catalog	SYSCOLDIST TYPE='F' or 'N', COLVALUE, FREQUENCYF
Single-column histogram	Optimizes	COL BETWEEN literal1 AND literal2, other RANGE operators
	RUNSTATS	INDEX(index) - histogram statistics can only be collected on the prefix columns with the same order
		COLGROUP(COLa) HISTOGRAM
		NUMQUANTILES count
	Catalog	SYSCOLDIST and SYSCOLDISTATS TYPE='H', QUANTILENO, LOWVALUE, HIGHVALUE, CARDF

Table 9-2 lists all multi-column statistics with examples of the types of predicates they help optimize, RUNSTATS syntax, and the SYSIBM catalog tables that contain the statistics.

Table 9-2 Multi-column statistics

Type of statistic	Details	Example
Multi-column cardinality	Optimizes	COLa = marker AND COLb = marker
	RUNSTATS	INDEX(index) KEYCARD - collects leading columns in index: 2 up to (#index columns-1)
		COLGROUP(COLa,COLb, ...) - for any column combination (2,3, ... columns)
	Catalog	SYSCOLDIST TYPE='C', NUMCOLUMNS, COLGROUPCOLNO, CARDF
Multi-column frequency	Optimizes	COLa = literal AND COLb = literal
	RUNSTATS	INDEX(index) FREQVAL NUMCOLS n - collects values for leading n columns in index
		COLGROUP(COLa,COLb, ...) FREQVAL
		COUNT of values to collect
		MOST LEAST or BOTH
	Catalog	SYSCOLDIST TYPE='F' or 'N', NUMCOLUMNS, COLVALUE, FREQUENCYF

Type of statistic	Details	Example
Multi-column histogram	Optimizes	COLa BETWEEN literal1 AND literal2 AND COLb BETWEEN literal3 and literal4, other RANGE operators
	RUNSTATS	INDEX(index) - histogram statistics can only be collected on the prefix columns with the same order
		COLGROUP(COLa, COLb, ...) HISTOGRAM
		NUMQUANTILES count
	Catalog	SYSCOLDIST and SYSCOLDISTATS, TYPE='H', NUMCOLUMNS, COLGROUPCOLNO, QUANTILENO, LOWVALUE, HIGHVALUE, CARDF

## 9.2 Statistics Advisor analysis

You can invoke the Statistics Advisor for a single query or for a workload by:

- Query Statistics Advisor: Analyzes a single query.
- Workload Statistics Advisor: Analyzes all queries separately, then aggregates the results into a single set of recommendations and reports.

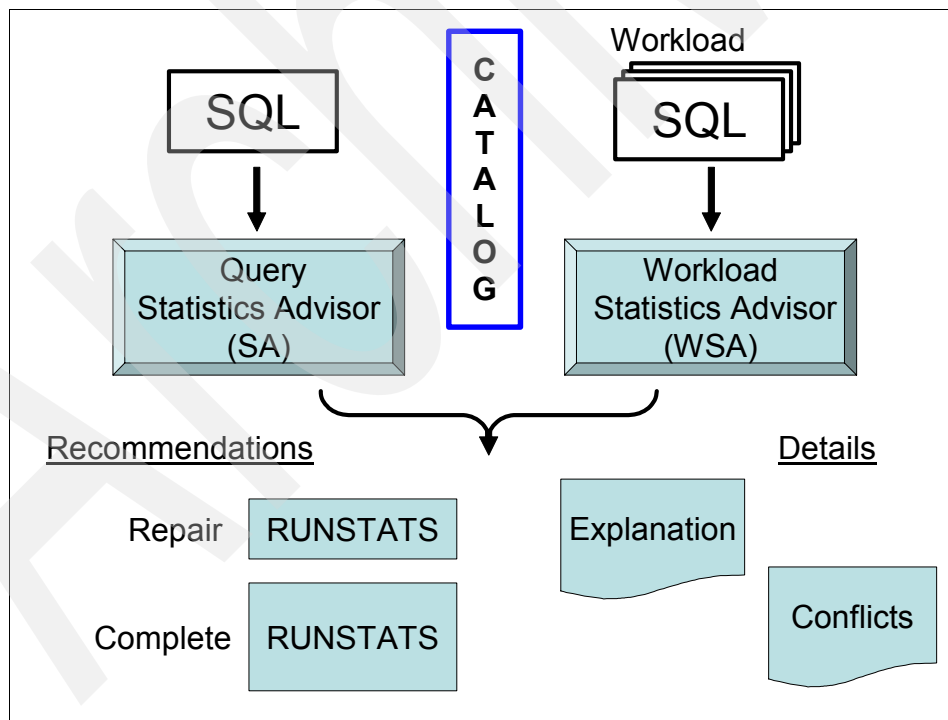


Figure 9-2 Statistics Advisor

### Goals of Statistics Advisor and Workload Statistics Advisor

Both variations of the Statistics Advisor operate similarly, but not identically, because their goals are different.

Query Statistics Advisor tunes a single query, presumably one that is under analysis because it is experiencing a problem. This makes REPAIR STATISTICS its high priority.

Workload Statistics Advisor, on the other hand, aims to produce good-health statistics for the entire workload. This makes COMPLETE STATISTICS its high priority.

**Note:** Workload Statistics Advisor always recommends COMPLETE STATISTICS as a high priority, whether or not pre-existing statistics are correct and complete.

There are other minor differences, pointed out in the discussion below. Throughout, Statistics Advisor refers to the common functionality. The terms Query Statistics Advisor or Workload Statistics Advisor are used to note specific behavior of the query and the workload variants.

### Identify interesting objects

The Statistics Advisor identifies *interesting* objects; that is, interesting to optimization, meaning objects whose statistics influence optimization. RUNSTATS syntax is generated to gather these statistics. The list of interesting objects with their statistics details appears in the Detail report.

From the predicate breakdown produced by EXPLAIN:

left-hand-side operator right-hand-side

*Interesting columns* are columns that appear in predicates, excluding a few predicates, such as  $C1+C2>100$ , where C1's or C2's statistics are not used for optimization.

*Interesting column groups* are combinations of columns used in predicates for which multi-column statistics might be helpful. These are identified based on a set of rules, which include whether the columns appear in an index.

### Missing statistics?

By examining the catalog tables for the interesting columns and column groups, the Statistics Advisor discovers statistics that have not been gathered, and it assumes the default values in the cost formulas. The actual statistics, when gathered, provide the optimizer with a better cost estimate of the query.

Missing example 1: Interesting column, EQUAL :host-variable or ? marker:

SYSCOLUMNS.COLCARDF = -1

Missing example 2: Interesting column, EQUAL literal

SYSCOLDIST no TYPE='F' row

Missing example 3: Interesting column group, EQUAL :host-variable or ? marker:

SYSCOLDIST no TYPE='C' row

The first execution within DB2 V9 points out missing statistics that were added to DB2 V9 (see 9.4.1, "Conversion to DB2 V9" on page 236).

### Conflict statistics?

The various access path statistics for a single table along with its indexes and columns and column groups form a set, interrelated by rules that verify the correctness of one statistic with another.

Conflicting example 1 - Table and UNIQUE index:

SYSTABLES.CARDF <> SYSINDEXES.FULLKEYCARDF

Conflicting example 2 - Frequent values cannot exceed 100% of the population:

```
SUM(SYSCOLDIST.FREQUENCYF) > 1
```

Conflicting example 3 - Multi-column cannot exceed product of column cardinality:

```
SYSCOLDIST.CARDF > product(all SYSCOLUMNS.COLCARDF)
```

There are over 25 such rules in effect. If any of these rule tests fail, then there is a problem - procedural or technical - in the way statistics are being gathered. When you change the collection strategy, you get better health statistics.

These rules have thresholds that you can manipulate in the Statistics Options. Minor inconsistencies occur naturally and cause no problem. Default thresholds allow for these minor variations and are acceptable in most situations.

**Note:** Statistics vary in their accuracy. Statistics that come directly from indexes and tables, such as CARDF, need to be exact. Others, such as column statistics, rely on algorithms and are less precise.

You can choose the time delta between the statistics collection time (STATSTIME) for objects involved in a conflict as a criterion. The Statistics Option “Tolerate conflicts within a short collection period” sets a count of days (default 7) where conflicts will not produce targeted RUNSTATS recommendations, but they will still be reported.

### Obsolete statistics?

The Statistics Advisor attempts to re-gather statistics that it considers obsolete. There are two cases:

- ▶ An empty table (CARDF=0) is considered “possibly obsolete - contains no rows” and is always recommended for RUNSTATS REPAIR, in case rows have been inserted since that point. If not, the RUNSTATS takes no time to execute.
- ▶ Statistics gathered a long time ago, specifically  $\text{STATSTIME} \leq \text{CURRENT}$  minus a “maximum number of days for statistics to be considered old”, a user option defaulting to 180 days. This option is enabled only for Query Statistics Advisor, because Workload Statistics Advisor refreshes all relevant statistics for the workload, regardless of their currency.

### Predicates needing REOPT

When a column is compared to a parameter marker, that is, not a literal value, DB2 must assume uniform distribution, that is, a filter factor of  $1/\text{cardinality}$ :

```
column op ?    or    column IN (?,?,...)
```

? is a host variable, dynamic parameter marker or special register

The Statistics Advisor recommends REOPT(ALWAYS) or REOPT(ONCE) to take advantage of the filter factors for the values actually provided at runtime. This is important when point data skew exists, which is true to a greater or lesser extent in most real world data.

### Apply heuristic rules for skew

It is not possible to know ahead of time all the statistics that are needed. Heuristic rules help estimate potential data skews and motivates the collection of frequency values and histograms to better cost out matches against literal values.

Heuristic rules in use:

- ▶ Have a much lower COLCARD, as compared with the table CARD, which is a likely indication that the column's data is skewed.
- ▶ Match a DEFAULT value that typically filters a different number of rows as compared with the average. DEFAULTs are inferred from two sources:
  - The column DEFAULT, from SYSIBM.SYSCOLUMNS
  - Query Statistics Advisor only: Consider also a list of values, supplied by user option. Pre-supplied values include, for example, 0 -1 9999-12-31 Y N.
- ▶ Workload Statistics Advisor only: If the same literal value occurs frequently in the query population, then we recommend a collection of the frequency statistics of that column.

Two types of skew are being analyzed:

Point skew

Compare against a single value or list of values

Equal predicates: = IN or IS NULL

Range skew

Compare against a range of values

Range predicates: > >= < <= LIKE BETWEEN

### Workload Statistics Advisor consolidates workloads

Workload Statistics Advisor analyzes each SQL in the workload, then combines its results into a single set of recommendations, reports, and RUNSTATS statements.

This consolidation is controlled by a weighting factor that assigns a priority to each SQL statement. This priority is based on:

- ▶ If Monitor or Statement Cache - use the actual performance weighting:  
#Executions, Elapsed time, CPU time
- ▶ If Catalog, QMF, File input:  
The number of times this SQL statement appears in the source

A facility in the Workload Project allows you to Edit Statement Runtime Information, that is, to change the #Executions, Elapsed time, and CPU time for weighting.

Workload are weighted, by default, to elapsed time (Figure 9-3):

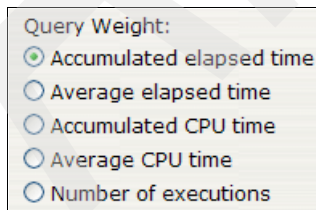


Figure 9-3 Workload Weighting Options

## Formats RUNSTATS and prioritizes their recommendations

The Statistics Advisor produces two kinds of RUNSTATS:

### COMPLETE RUNSTATS

Gathers a complete RUNSTATS for the query or workload. The complete syntax is intended to be saved and rerun periodically as a good health measure.

### REPAIR RUNSTATS (if necessary)

Repairs the immediate statistics problems, mainly to gather the *missing* statistics, resolve the *conflicting* statistics, or re-gather *obsolete* statistics.

The RUNSTATS syntax is optimized to collect the information in the most efficient way (where there are alternatives.)

**Note:** The Statistics Advisor is aware that RUNSTATS can be costly and so it attempts to not collect unnecessary statistics.

Statistics Advisor produces one or more recommendations, ranked as high, medium, or low.

Recommendations can include:

- ▶ Run Complete Runstats - ranked high if Workload Statistics Advisor.
- ▶ Run Repair Runstats - ranked high if Query Statistics Advisor.
- ▶ REOPT - if parameter markers are used in the SQL.

## Creates detail reports

Statistics Advisor produces detail reports for the user to consult:

### Statistics detail

Lists all objects relevant to the analysis and their statistics details and status (OK, missing, conflict, or obsolete.)

### Conflict report

Lists the particular conflicts found in the statistics. Conflicting statistics are a sign of procedural or technical issues and generate a REPAIR RUNSTATS recommendation.

## 9.3 Invoking Statistics Advisor

The following examples show Workload Statistics Advisor. Similar controls are available for Query Projects.

You can run Workload Statistics Advisor from the Workload Project main panel (top most in the project organizer, left most in the bottom tabs), as shown in Figure 9-4 on page 232.

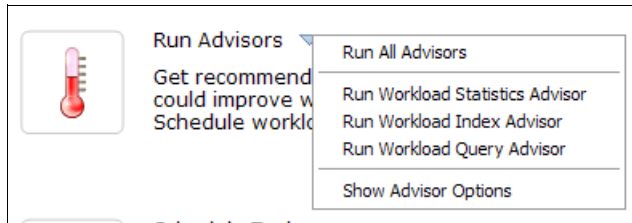


Figure 9-4 Workload Project Run Advisors

Or, from the drop down when viewing the Query or the Workload Statements (Figure 9-5).

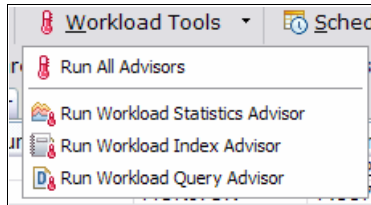


Figure 9-5 Workload Tools - Statistics Advisor

You can also schedule the Statistics Advisor to run at a later time (Figure 9-6)

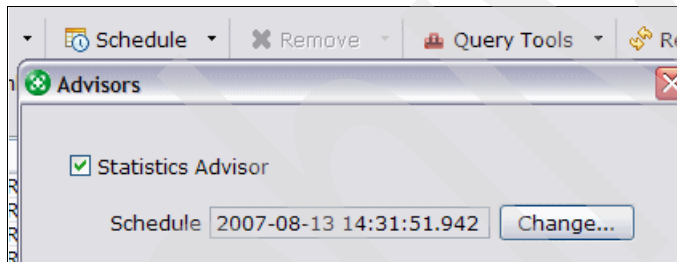


Figure 9-6 Workload Statistics Advisor - Schedule

### 9.3.1 EXPLAIN as a pre-requisite

The Statistics Advisor uses Explain information to base its analysis. If Explain has not been run, it is required now (Figure 9-7):

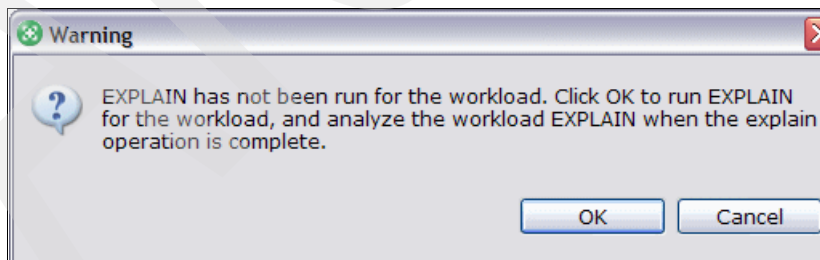


Figure 9-7 Workload EXPLAIN required

For large workloads, this statement-by-statement EXPLAIN may take time. You might want to run Explain in the background, or schedule it to run at a different time.

The resulting information is saved in DB2 WCC tables on the mainframe and is available for all analyses.



**Note:** Access path statistics are captured at the same time as the EXPLAIN information proper. If you update the access path statistics, you must rerun EXPLAIN.

### 9.3.2 Statistics Advisor options

You can change the options by selecting **Tool → Preferences** or **Show Advisor Options → Options**.

The following default is an issue when you migrate to DB2 V9:

- ▶ **Disable Data Repetition Factor (DRF) statistics check - Default OFF**  
Check ON if you have just converted to DB2 V9 and did not run a complete RUNSTATS. Otherwise, Statistics Advisor will note “Missing” DRF for all indexes because SYSINDEXES DATAPEATFACTORF is still -1.

The following defaults manage the trade-off between RUNSTATS cost versus the benefit of extra statistics. The defaults suit the needs of most installations, but consider changing these to a more “aggressive” posture, if the extra RUNSTATS cost is not an issue for you:

- ▶ **Collect histogram for RANGE predicate - Default OFF**  
Better to set ON unless the RUNSTATS cost is an overriding issue.
- ▶ **Collect COLGROUP statistics information aggressively - Default OFF**  
Workload Statistics Advisor recommends the most significant COLGROUPs of combinations of EQUAL columns plus all RANGE/IN columns. It might be worth experimenting with ON.
- ▶ **Collection parameters for RUNSTATS**  
If the RUNSTATS cost is not an overriding issue, you can increase accuracy by increasing the Table Sampling Threshold and the Table Sampling Rate.
- ▶ **Thresholds (Workload Statistics Advisor only)**  
These thresholds, Uniform and Non-Uniform, deal with the way queries are weighted in workload statistics recommendations. Reduce these numbers to gather more statistics.

### 9.3.3 Statistics Advisor output panel

When the Statistics Advisor completes its analysis, it displays its results in a Statistics Advisor project tab (item), as shown in Figure 9-8 on page 234.

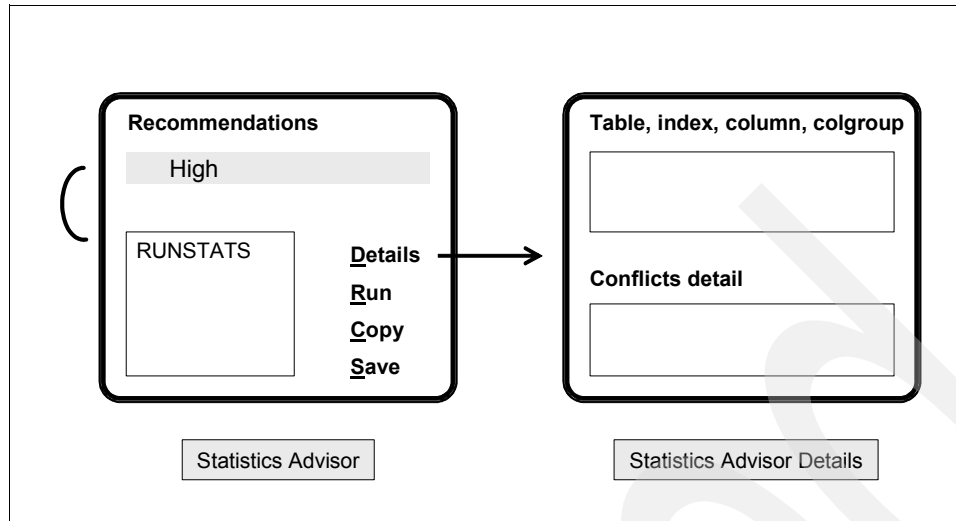


Figure 9-8 Statistics Advisor result panels

Outputs include the following information:

### Recommendations

Listed at the top, prioritized in order, high to medium to low:

- ▶ Run the REPAIR RUNSTATS to resolve the immediate statistic issues: missing, conflicting, or obsolete status.
- ▶ Run the COMPLETE RUNSTATS for the query or workload.
- ▶ Run REOPT to provide run-time information for parameter markers (data skew).

Query Statistics Advisor prioritizes REPAIR RUNSTATS High because a quick response to a problem query is likely required. Workload Statistics Advisor prioritizes COMPLETE RUNSTATS High, since good health status of the entire workload is the goal. Also, executing REPAIR RUNSTATS alone for workloads might introduce other discrepancies.

Moving your cursor over any Recommendation brings up the recommended RUNSTATS in the bottom panel.

### The recommended RUNSTATS

This is the complete RUNSTATS syntax (REPAIR or COMPLETE), ready to run.

For small tables or the REPAIR of single queries:

The Run button executes RUNSTATS immediately, using the DSNUTILU stored procedure. The RUNSTATS result appears in a lower pane in the window.

For larger tables and COMPLETE RUNSTATS for workloads:

The better option is to Save to a file, then upload that file to a z/OS SYSIN to run. In the case of COMPLETE RUNSTATS, you need to do so periodically.

### Detail report

Use the Details button to access this report. It appears as a separate project tab, Statistics Advisor Details.

Figure 9-9 shows the format of the Detail Report.

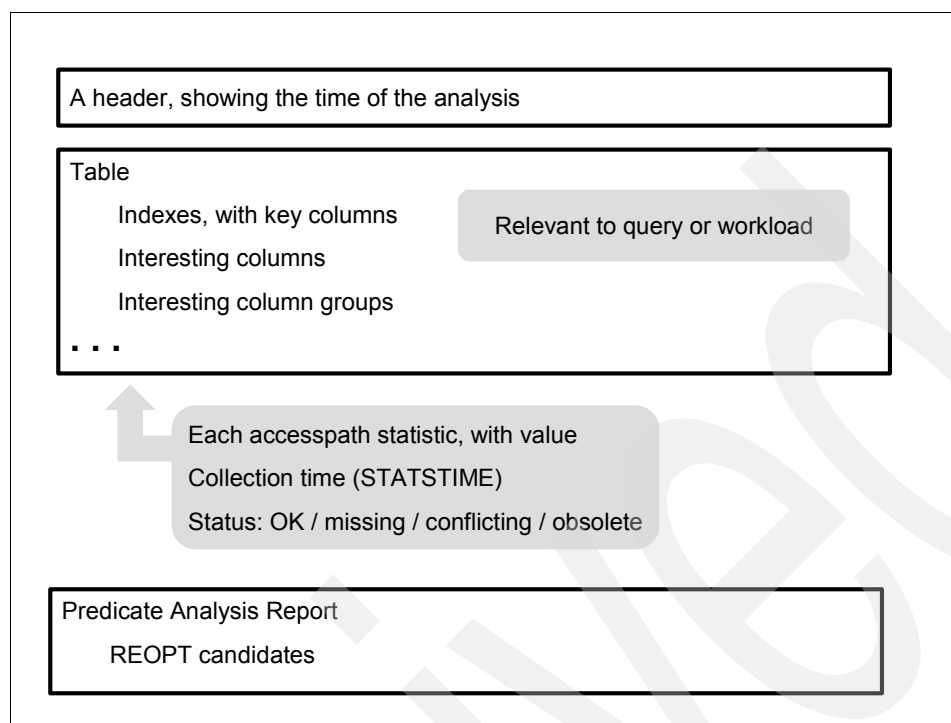


Figure 9-9 Statistics Advisor Detail Report

There is no Save button, but you can save the report by selecting all text with Ctrl-A, Ctrl-C to copy, and Ctrl-V to paste to a .txt file.

Take a look at the following in the report:

- ▶ The list of interesting columns and column groups.
- ▶ All status - lines reporting missing, conflicting, or obsolete. All of them prompt REPAIR statistics.
- ▶ Collection time - note that the time is much earlier than now.
- ▶ A summary of relevant access path statistics.

Examples of these reports are in section 9.4, “Sample Statistics Advisor scenarios” on page 236.

**Note:** The detail report does not list the full statistics of the objects involved in the query or workload. For individual queries, use the facilities in the Access Path Graph.

## Conflict report

Use the Details button to access this report. It appears as a separate project tab, Statistics Advisor Details.

The results of the various tests appear, along with thresholds, which you can change by manipulating the Conflict Checking Thresholds in Statistics Advisor Preferences.

If any of these rule tests fail, then there is a problem - procedural or technical - in the way statistics are being gathered:

- ▶ Statistics being gathered at different times, for example RUNSTATS TABLE runs frequently, but RUNSTATS INDEX runs much less frequently.
- ▶ Manual statistics update for new tables.

The solution is to save the COMPLETE statistics and run them regularly. Then, take periodic health checks on statistics using the Statistics Advisor.

See section 9.4.3, “Checking for consistent statistics” on page 239 for an example of conflict statistics.

## 9.4 Sample Statistics Advisor scenarios

The scenarios in this section are examples of the benefits provided by the Statistics Advisor.

For illustration purposes, these scenarios use a schema familiar to many: the DB2 catalog, DSNDB06.SYSDBASE, containing the tables SYSIBM.SYSTABLES and SYSIBM.SYSCOLUMNS.

### 9.4.1 Conversion to DB2 V9

DB2 V9 implements two new kinds of statistics for pre-V9 constructs:

- ▶ Histogram statistics - more precise estimates of range data skew.
- ▶ Index DATAREPEATFACTORF (DRF), a better measure of data clustering (if DSNZPARM STATCLUS is enabled).

When you first migrate to DB2 V9, run the Statistics Advisor to collect these new statistics.

For example, see the query in Example 9-1.

*Example 9-1 BETWEEN predicate SQL*

---

```
select count(*) AS “2006 Columns”
from sysibm.syscolumns
where createdts between '2006-1-1-00.00.00' and '2006-12-31-24.00.00'
```

---

It generates the HISTOGRAM specification in Example 9-2.

*Example 9-2 BETWEEN predicate RUNSTATS*

---

```
RUNSTATS TABLESPACE DSNDB06.SYSDBASE
          TABLE(SYSIBM.SYSCOLUMNS)
          COLUMN(CREATEDTS)
          COLGROUP(CREATEDTS) HISTOGRAM NUMQUANTILES 20
          ...
```

---

### 9.4.2 Typical RUNSTATS - is it sufficient?

Example 9-3 on page 237 shows a typical RUNSTATS statement, executed more often than any other variant in practice.

---

*Example 9-3 Statistics Advisor Sample Scenario - RUNSTATS*

---

```
RUNSTATS TABLESPACE DSND06.SYSDBASE
TABLE(ALL) INDEX(ALL)
SHRLEVEL CHANGE
```

---

Is this set of statistics sufficient to optimize the query (Example 9-4)?

---

*Example 9-4 Statistics Advisor Sample Scenario - SQL*

---

```
-- Count all DECIMAL columns in all tables under a given schema
SELECT
  COUNT(*) AS N_DECIMAL_COL
FROM SYSIBM.SYSTABLES AS TB
JOIN SYSIBM.SYSCOLUMNS AS CO
  ON (TB.CREATOR = CO.TBCREATOR
      AND TB.NAME = CO.TBNAME)
WHERE TB.CREATOR = ?
      AND CO.COLTYPE = 'DECIMAL'
```

---

The access path is shown in Example 9-5.

---

*Example 9-5 Statistics Advisor Sample Scenario - EXPLAIN*

---

```
Outer table SYSIBM.SYSTABLES
  index SYSIBM.DSNDTX01(CREATOR,NAME), matchcols 1
Nested-loop join to SYSIBM.SYSCOLUMNS
  index SYSIBM.DSNDX01(TBCREATOR,TBNAME,NAME), matchcols 2
```

---

We now look at what the Statistics Advisor suggests for improvements.

### **Statistics Advisor output**

Figure 9-10 on page 238 shows the high priority recommendation: Run the REPAIR RUNSTATS as formatted.

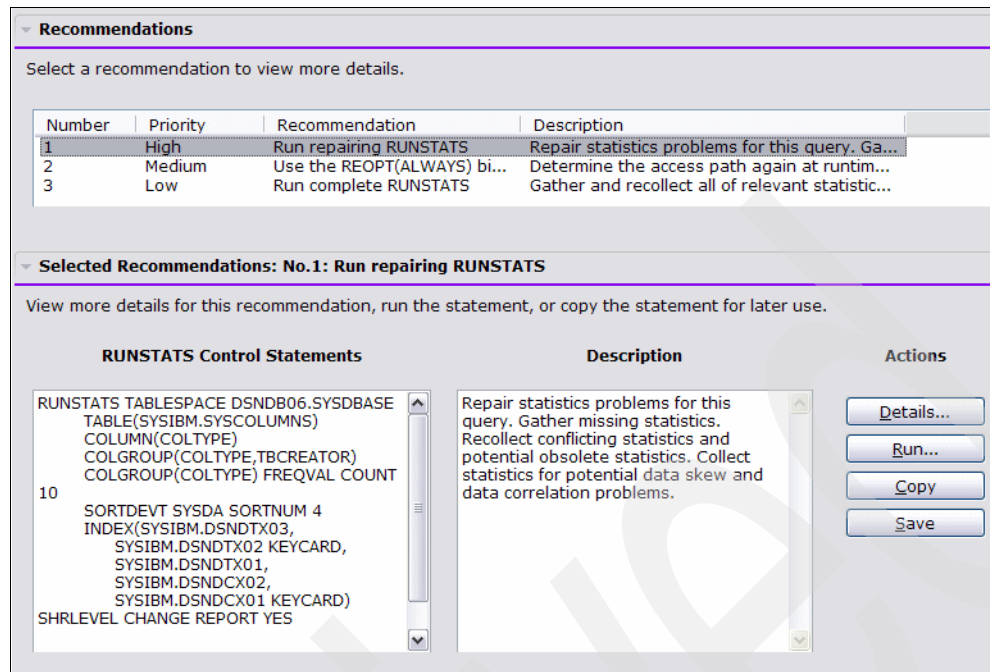


Figure 9-10 High priority - Run repair RUNSTATS

Query Statistics Advisor prioritizes the REPAIR RUNSTATS as 1 High. Selecting it shows the REPAIR RUNSTATS statements to use.

## The REPAIR RUNSTATS

REPAIR RUNSTATS asks for the base statistics for the tables and indexes to be run and adds the following statistics:

For the external predicate TB.CREATOR = ?

- SYSIBM.SYSTABLES index SYSIBM.DSNDTX02 KEYCARD  
To capture multi-column cardinality for (DBID,OBID,CREATOR)

For the external predicate C0.COLTYPE = 'DECIMAL'

- SYSIBM.SYSCOLUMNS COLUMN(COLTYPE) - For COLTYPE cardinality
- SYSIBM.SYSCOLUMNS COLGROUP(COLTYPE) FREQVAL - For COLTYPE value frequencies

For the join columns (TB.CREATOR=C0.TBCREATOR AND TB.NAME=C0.TBNAME)

- SYSIBM.SYSCOLUMNS index SYSIBM.DSNDX01 KEYCARD - To capture multi-column cardinality for (TBCREATOR,TBNAME), the join cardinality

## Detail report

For corroboration, consult the details of each table, index, column, column group: their access path statistics, collection time, and statistics status.

Figure 9-11 on page 239 shows the most interesting column, COLTYPE, the column that needs frequency statistics.

```

Interesting Columns:

COLTYPE
  Cardinality:                14.0
  Uniform Statistics Collection Time: 2007-07-15 05:00:18.490091
  Uniform Statistics Status:    OK
  Frequency Statistics Collection Time: null
  Frequency Statistics Status:  missing
  Histogram Statistics Collection Time: null
  Histogram Statistics Status:  missing
  Possibly Point Skewed:       Yes
  Possibly Range Skewed:       No

```

Figure 9-11 Statistics Advisor Sample Scenario - Detail report extract

## Conclusion

This example demonstrates that executing the “default RUNSTATS” does not necessarily give you the best possible access path for all your queries.

In this case, as in most cases of “DECIMAL” data types, the assumption by the optimizer of uniform data distribution in the absence of frequency statistics is not a good assumption for this predicate. This is why STATS ADVISOR recommends FREQVAL on the COLTYPE column.

In the next section, we create inconsistent statistics to verify the Conflict report.

### 9.4.3 Checking for consistent statistics

For demonstration purposes, FULLKEYCARDF was purposely altered as shown in Example 9-6.

#### Example 9-6 Conflict statistics demonstration - Manipulation

```

UPDATE SYSIBM.SYSINDEXES
SET FULLKEYCARDF = 10  -- correct value: 86808
WHERE CREATOR = 'SYSIBM'
AND NAME = 'DSNDTX01' -- unique index of SYSIBM.SYSTABALES

```

The Conflict Report (Example 9-7) found three discrepancies.

#### Example 9-7 Conflict statistics demonstration - Conflict Report

```

Cardinality (86808.0) of Table SYSIBM.SYSTABLES <> fullkey cardinality (10.0) of
unique Index SYSIBM.DSNDTX01
  Tolerance: 0.0010

```

```

  Firstkey cardinality (323.0) of Index SYSIBM.DSNDTX01 > fullkey cardinality
  (10.0)
  Tolerance: 0.0

```

```

  Fullkey cardinality (10.0) of Index SYSIBM.DSNDTX01 < cardinality (323.0) of its
  key (CREATOR)
  Tolerance: 0.0010

```

## 9.4.4 Workload Statistics Advisor sample

Figure 9-12 shows a test workload as monitored by a Normal Profile Monitor. One statement's SQL text is shown.

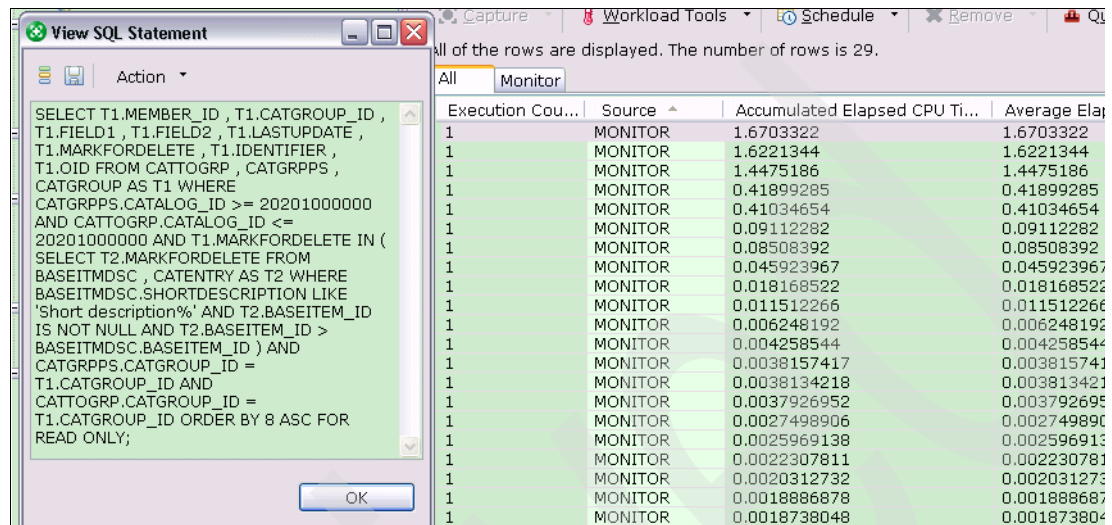


Figure 9-12 Workload Statistics Advisor - Monitor sample

These tables had statistics produced by the familiar sample RUNSTATS:

RUNSTATS TABLE(ALL) INDEX(ALL)

Workload Statistics Advisor recommended to run a COMPLETE statistics to optimize this complete workload (Figure 9-13).

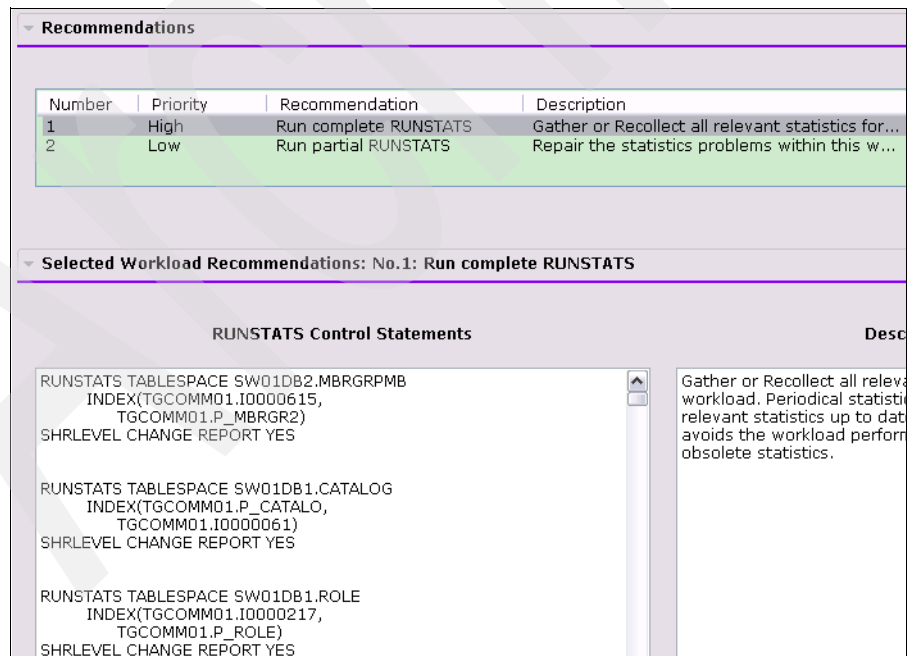


Figure 9-13 Workload Statistics Advisor - Recommendations

The RUNSTATS statements were saved and uploaded to z/OS to run as //SYSIN (Figure 9-14 on page 241).



```

RUNSTATS TABLESPACE SW01DB2.CATTGRP
TABLE(TGCOMM01.CATTGRP)
COLUMN(CATGROUP_ID,CATALOG_ID)
INDEX(TGCOMM01.I0000532,
TGCOMM01.I0000373,
TGCOMM01.P_CATTGP FREQVAL NUMCOLS 1 COUNT 10
HISTOGRAM NUMCOLS 1 NUMQUANTILES 20,
TGCOMM01.TX_CATTGRP)
SHRLEVEL CHANGE REPORT YES

RUNSTATS TABLESPACE SW01DB2.CATGRPPS
TABLE(TGCOMM01.CATGRPPS)
COLUMN(PRODUCTSET_ID,CATALOG_ID,CATGROUP_ID)
INDEX(TGCOMM01.TX_CATGRPPS KEYCARD,
TGCOMM01.I0000291 KEYCARD
FREQVAL NUMCOLS 1 COUNT 10
HISTOGRAM NUMCOLS 1 NUMQUANTILES 20,
TGCOMM01.I0000526,
TGCOMM01.P_CATGR1 KEYCARD,
TGCOMM01.I0000292)
SHRLEVEL CHANGE REPORT YES

```

Figure 9-14 Workload Statistics Advisor - Sample COMPLETE RUNSTATS

This SQL shows the additional statistics needed to optimize the SQL statement, which is shown in Figure 9-12 on page 240.

#### 9.4.5 Statistics first-cut recommendations for very large workloads

When the scale of the workload (thousands or tens of thousands of SQL statements) prevents convenient processing by Workload Statistics Advisor, consider the procedure described in Appendix A.2, “Automated RUNSTATS column lists” on page 401.

Archived

## Index Advisor

This chapter describes the functions of the Index Advisor component of the Optimization Expert for z/OS.

This chapter contains the following topics:

- ▶ “Index Advisor overview” on page 244
- ▶ “Query Index Advisor” on page 244
- ▶ “Workload Index Advisor” on page 257

## 10.1 Index Advisor overview

The Index Advisor, which is part of Optimization Expert for z/OS, makes recommendations for new indexes that might improve the performance of either a single SQL statement or a workload containing many SQL statements. Only basic capabilities are available when the Advisor is applied to a single SQL statement. The Advisor becomes more powerful and flexible when it is used against a workload containing many SQL statements.

## 10.2 Query Index Advisor

We use a simple example to demonstrate the basic Index Advisor facilities for a single SQL statement.

### 10.2.1 Running Query Index Advisor

We have defined a Query project called Text01 with the SQL source defined as Text. We have typed in a simple Select statement with a subquery, as shown in Figure 10-1. The text of the SQL statement is shown in Figure 10-2.

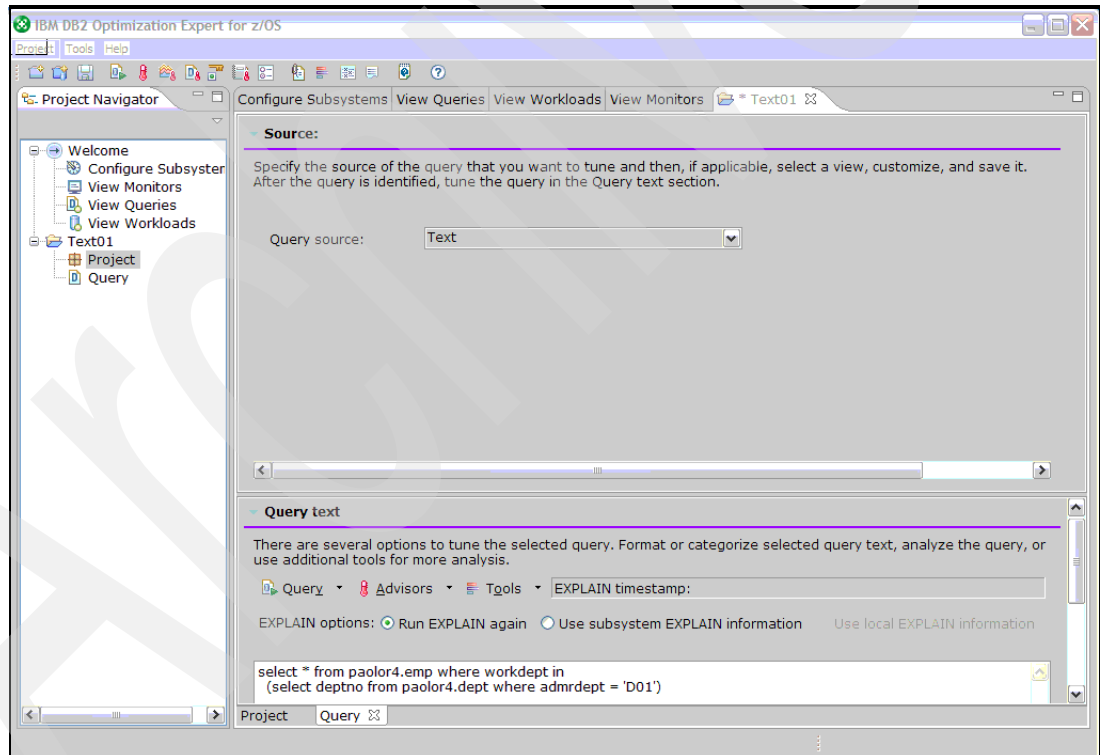


Figure 10-1 Specifying a single SQL statement

```
select * from paolor4.emp where workdept in
(select deptno from paolor4.dept where admrdept = 'D01')
```

Figure 10-2 The SQL statement

When we produce the access plan graph, the subquery has been transformed into a join, as shown in Figure 10-3.

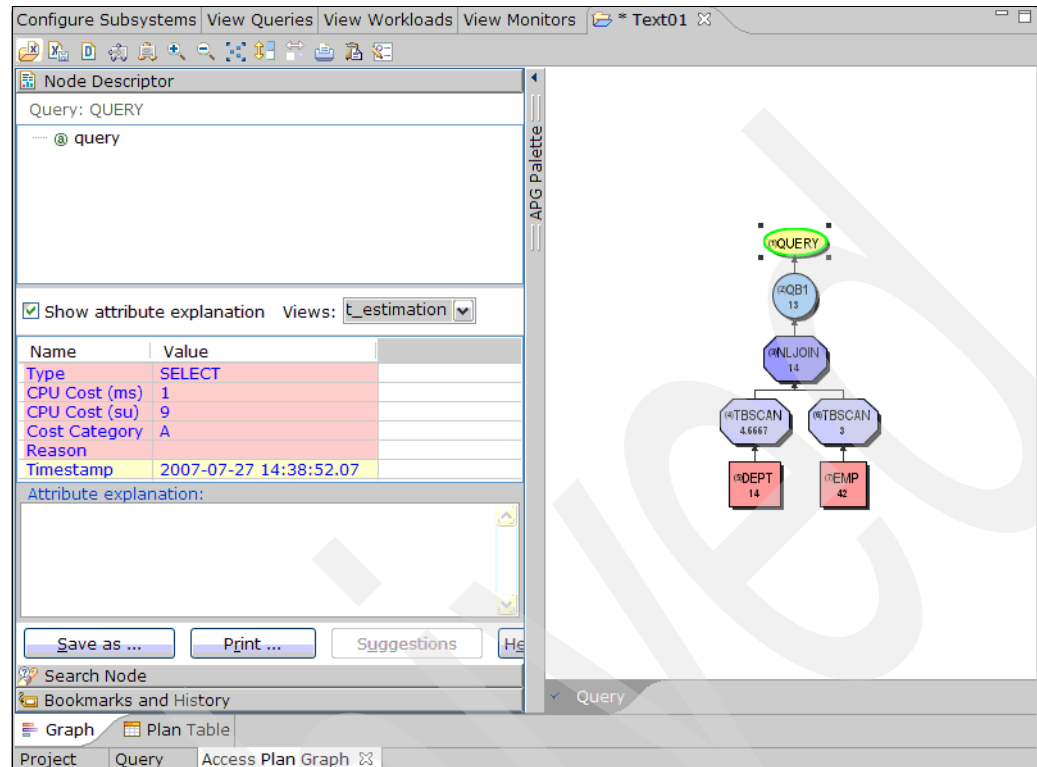


Figure 10-3 The access plan graph

If we switch back to the Query tab, we can run the Index Advisor for this SQL statement. We do this by clicking **Advisors** above the SQL statement text box and selecting **Run Index Advisor** from the drop-down menu. This is shown in Figure 10-4 on page 246.

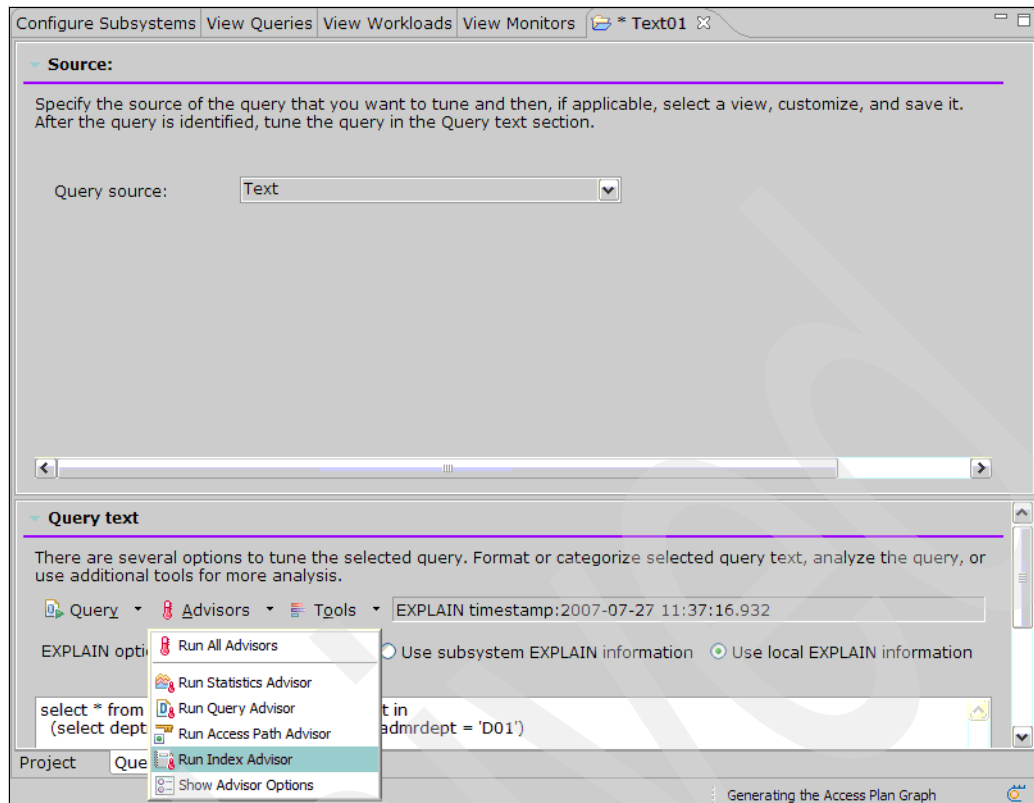


Figure 10-4 Run the Index Advisor

Before the Index Advisor runs, it asks for the source of Explain information (Figure 10-5).

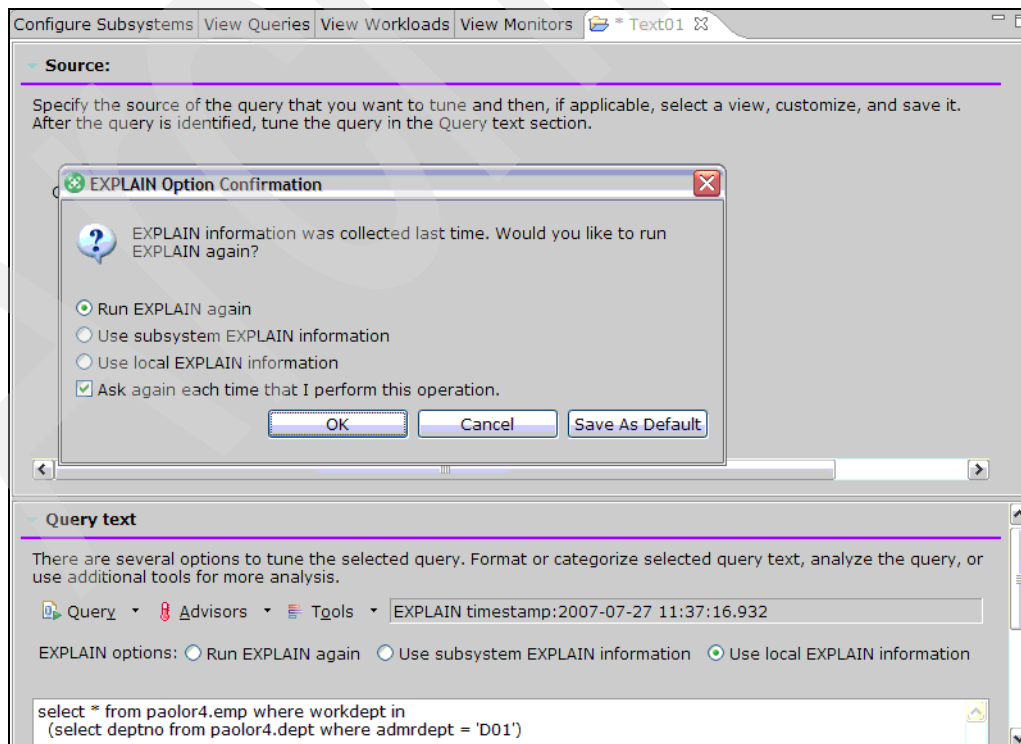


Figure 10-5 Asking for the source of Explain information

If you select the third option (Use local EXPLAIN information), the Index Advisor might complain. The explain done to generate an access plan graph does not generate all the information required by the Index Advisor (Figure 10-6).

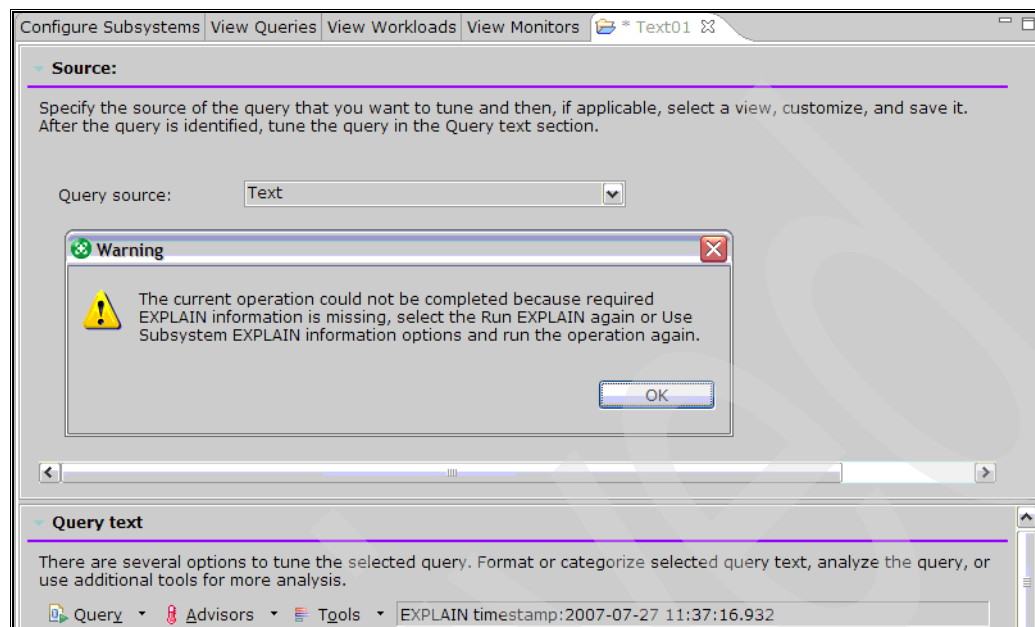


Figure 10-6 Local EXPLAIN information not complete

Either the “Run EXPLAIN again” or “Use subsystem EXPLAIN information” options can provide enough information. We choose the first option, Run EXPLAIN again. You see a progress indicator while the Advisor is running.

## 10.2.2 Query Index Advisor recommendations

When the Index Advisor completes, it opens a new tab named “Index Advisor”, which contains the recommendations as shown in Figure 10-7 on page 248.

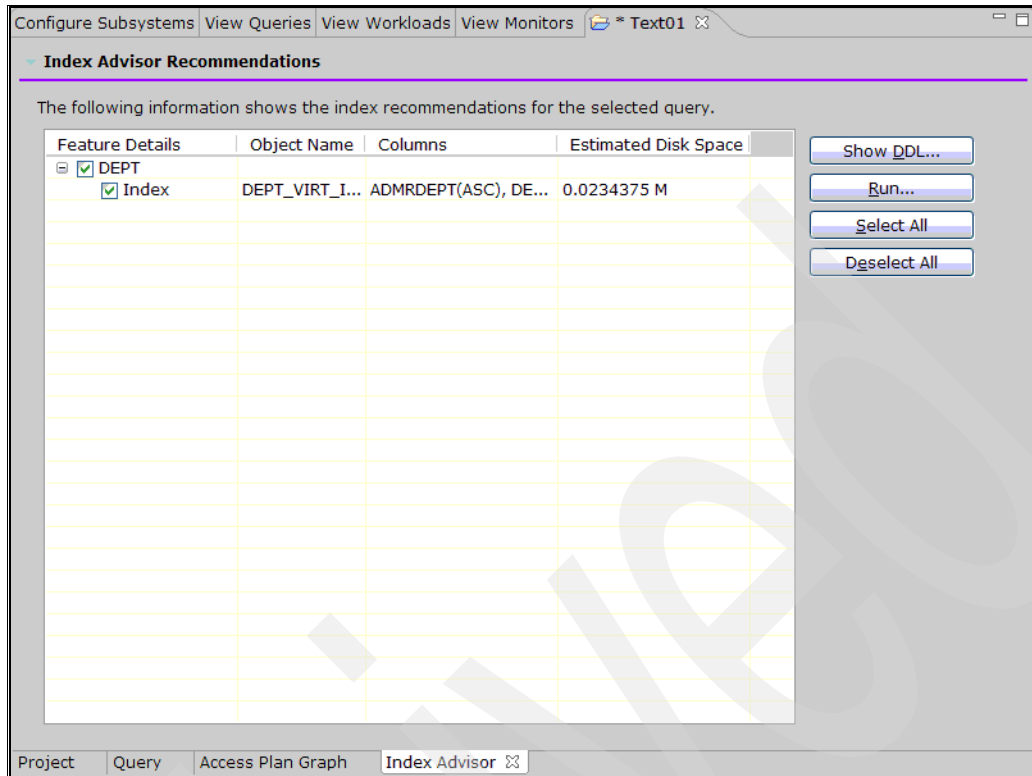


Figure 10-7 Index Advisor recommendations

The Advisor has recommended the creation of one new index on the DEPT table. You cannot see all the columns that are part of the index key, but if you hover the cursor over the Columns information, you see all the columns.

You can also expand the Columns so that all the index keys are shown.

You can also click **Show DDL** at the top right. This opens a panel that shows the Create Index statement (see Figure 10-8 on page 249).



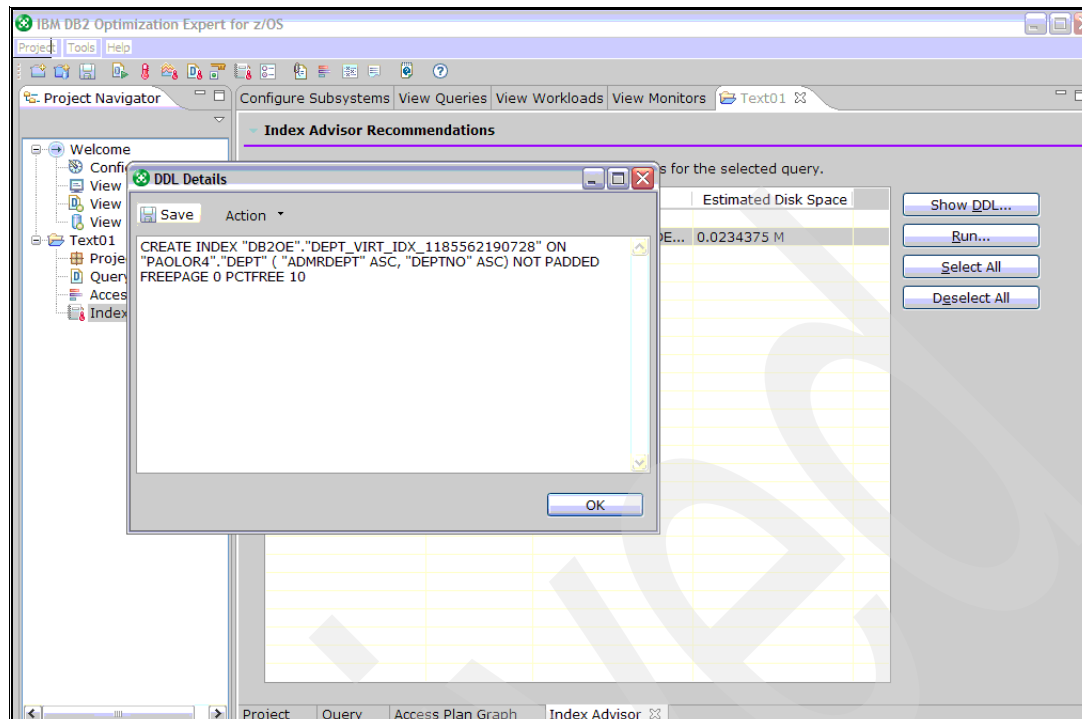


Figure 10-8 Viewing the Create Index DDL

We can see that the index key is derived from two DEPT table columns: ADMRDEPT and DEPTNO, which are both stored in ascending sequence. If we look back at the SQL statement that we provided to the Advisor (Figure 10-2 on page 244), we can assume that the Advisor expects the new index to allow index-only access to the DEPT table. To test the assumption, create the index, run Runstats, and then re-Explain the SQL statement.

The Query Index Advisor might recommend indexes for the following reasons:

- ▶ Foreign keys that do not have indexes defined.
- ▶ Indexes that will provide index filtering for the SQL statement.
- ▶ Indexes that will provide index filtering and screening for the SQL statement.
- ▶ Indexes that will provide index-only access for the SQL statement.

To understand the difference between index filtering and index screening, see Chapter 7, “Tuning your queries,” in *DB2 Version 9.1 for z/OS Performance Monitoring and Tuning Guide*, SC18-9851, or consider the example shown in Figure 10-9 on page 250. This shows an example of both Index Filtering and Index Screening. An index is created on the table CUSTOMER, on the columns TOWN and CUST\_NO. The picture shows the leaf page entries with key values between TOWN = ‘LEEDS’ and TOWN = ‘LONDON’. Each entry contains a value for TOWN and a value for CUST\_NO followed by the RID (Rnn) of the row containing those values.

Given the Select statement shown in the picture, DB2 can use the index XTC on TOWN and CUST\_NO to select the required rows using a Matching Index Scan with one matching column. It can use Index Filtering to apply the predicate TOWN BETWEEN ‘LEEDS’ AND ‘LONDON’ and restrict access to those index leaf entries having values of TOWN between ‘LEEDS’ and ‘LONDON’. Use The index tree structure to avoid processing any index leaf entries, and any rows that those entries point to, outside that range. This eliminates index leaf entries and rows that fall outside the range from processing. The column TOWN is the one matching column in the Matching Index Scan. A matching column is one that provides Index Filtering.

You cannot apply the second predicate, `CUST_NO > 200`, using Index Filtering because you must scan all leaf page entries for the range of TOWN values. A `CUST_NO` value  $> 200$  can occur for any value of TOWN. However, you can apply the predicate, `CUST_NO > 200`, using Index Screening. Each leaf page entry for TOWN values in the range “LEEDS” to “LONDON” is scanned, but only when the `CUST_NO` value in the index key is  $> 200$  is the RID used to retrieve the row from the table space. When the `CUST_NO` value is  $\leq 200$ , the row access is avoided.

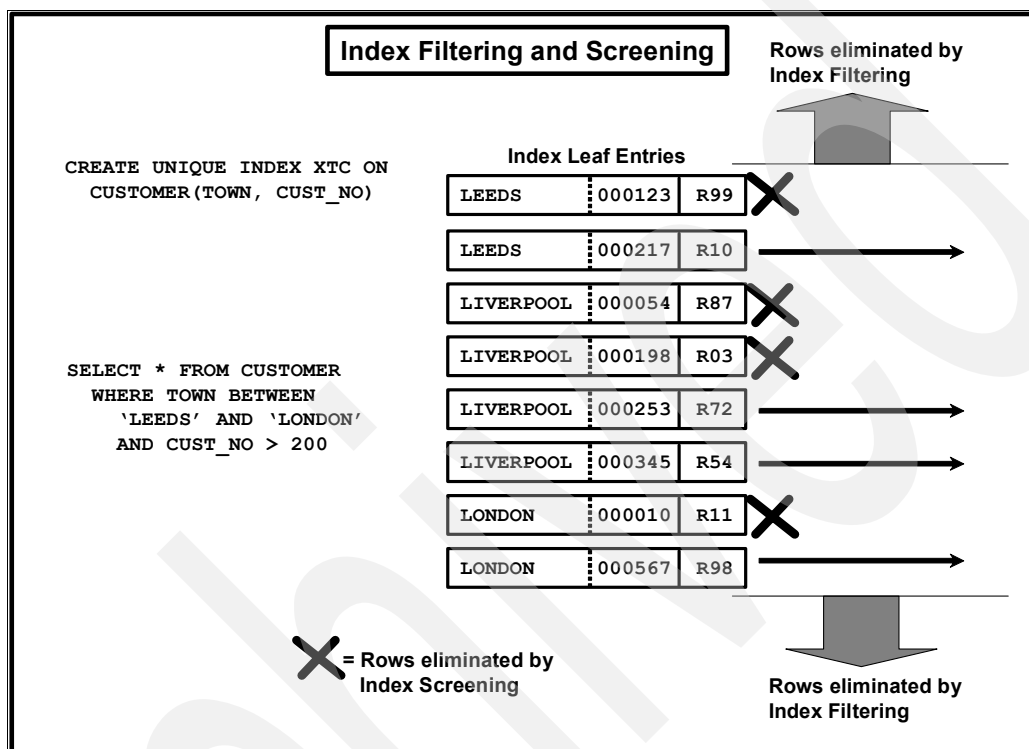


Figure 10-9 Index filtering and screening

### 10.2.3 Acting on the recommendations

You can run the Create Index statement directly from the tool if you are happy with the DDL as generated. Just click **Run**. It is likely, however, that you want to modify the DDL to conform to your installation naming standards. To do this, first save the DDL as a text file. From the DDL details panel (see Figure 10-8 on page 249), you can click **Save**, which allows you to specify a file name for the text file. You can then modify the DDL and run it using any method available to you for running DDL, such as SPUFI, QMF, and so on.

**Note:** You can change the default Creator ID that the Query Index Advisor uses when generating the Create Index DDL. See section 10.2.4, “Setting Query Index Advisor options for the current project” on page 253 and section 10.2.5, “Setting Query Index Advisor option defaults for new projects” on page 256.

We can use a trick to modify and run the DDL within Optimization Expert. In Figure 10-10 on page 251, we have created a new Query project (named Temp) and specified the query source as Text. We then copy the DDL into the text box and edit it to conform to our naming standards. We can run the Create Index by selecting Run from the Query pull-down menu above the text box. In our example, we received a warning SQLCODE = +20002 because of

the NOT PADDED specification in the DDL. Because the two columns in the index key are not VARCHAR, DB2 ignores the NOT PADDED specification.

**Note:** You must have Create DB2 privileges to create the index.

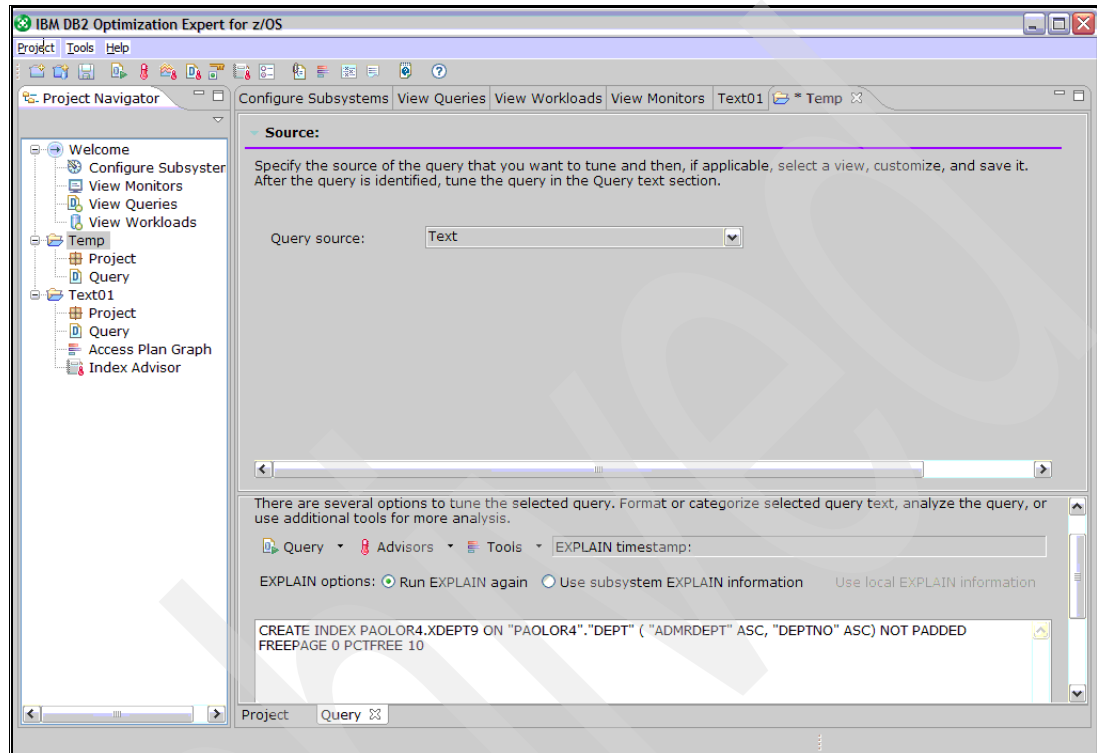


Figure 10-10 Using Optimization Expert to create the index

We have created the index, so now run Runstats. Again, we can take advantage of Optimization Expert to do this by switching back to the Text01 project. This project contains the original SQL statement that we investigated with the Index Advisor. We can run the Statistics Advisor to generate and run Runstats. You can find an example of running the Statistics Advisor against a single SQL statement in section 2.6, “Using the Statistics Advisor” on page 71. The Statistics Advisor is described in detail in Chapter 9, “Statistics Advisor” on page 223.

The recommendations of the Statistics Advisor are shown in Figure 10-11 on page 252. The high priority recommendation includes running Runstats on the new index.

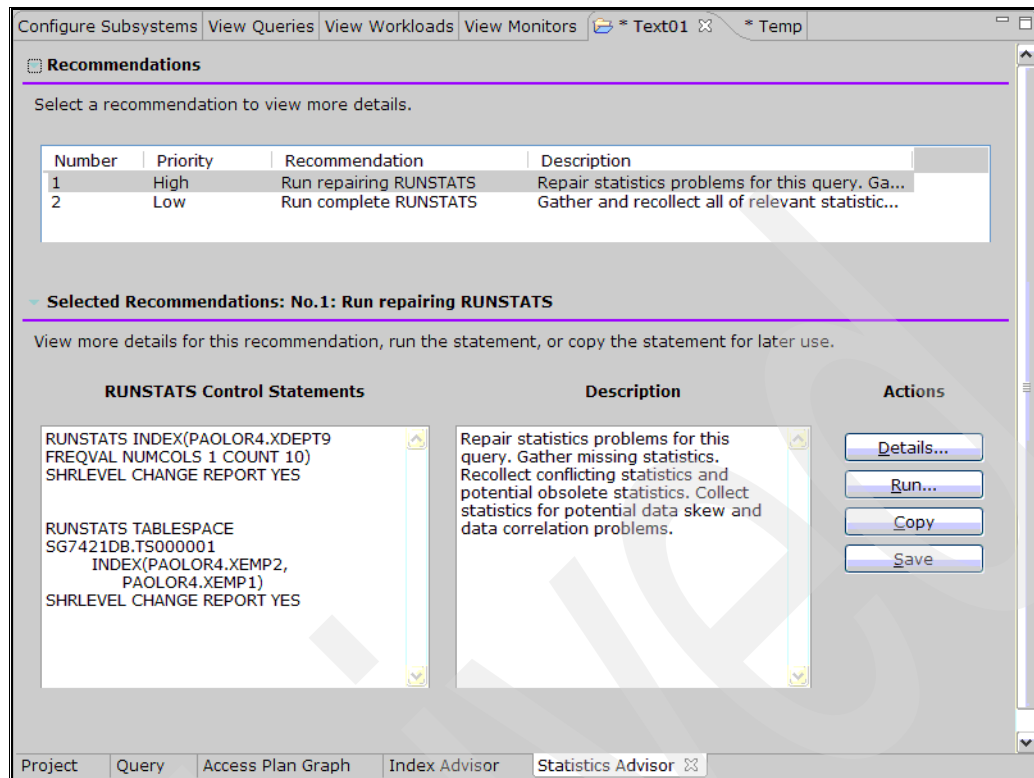


Figure 10-11 Statistics Advisor recommendations

We run the high priority recommendation. After that, we return to the Query tab and generate a new Access Plan Graph. The result is shown in Figure 10-12 on page 253. Compare this to the original graph in Figure 10-3 on page 245. You can see that our assumption was correct and that the access to the DEPT table has become an index-only access using the new index instead of a table space scan. Interestingly, the access to the EMP table has also changed from a table space scan to an index scan using XEMP2, an index on the WORKDEPT column of the EMP table. Our high priority Runstats included updating the statistics for the table space (TS000001), containing the EMP table and the two indexes based on the EMP table. It looks like these new statistics have had a benefit.

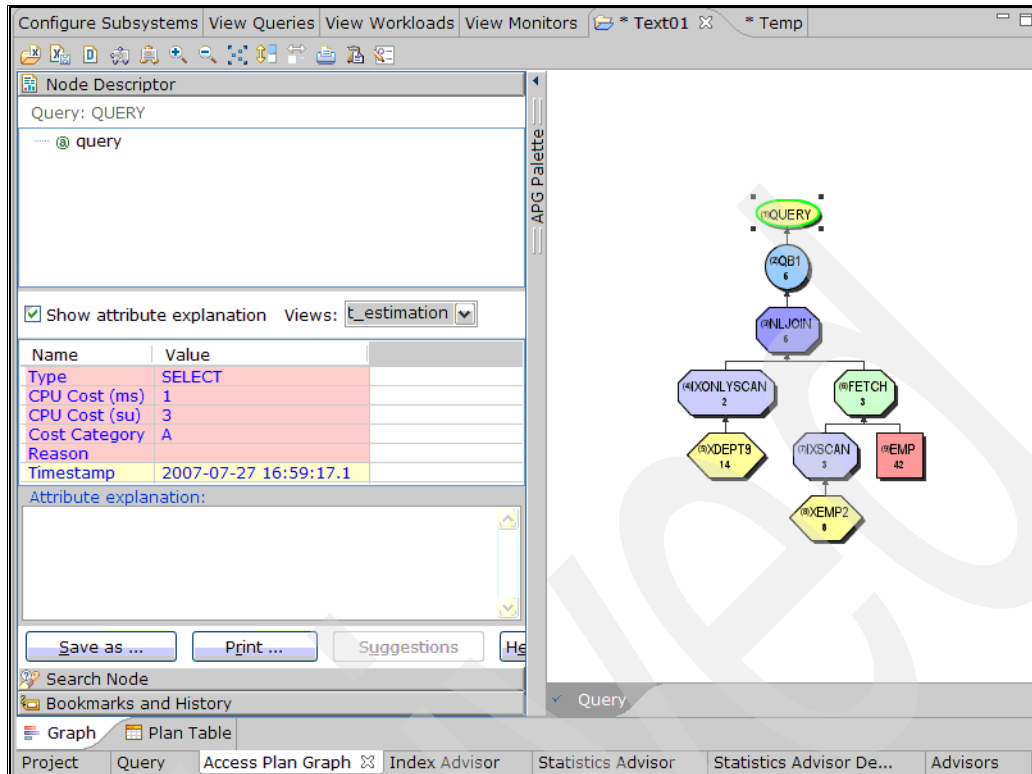


Figure 10-12 The new Access Plan Graph

## 10.2.4 Setting Query Index Advisor options for the current project

You can set the options that control operation of the Query Index Advisor before you run the Index Advisor. From the Project tab, click **Run Advisors**, shown in Figure 10-13 on page 254.

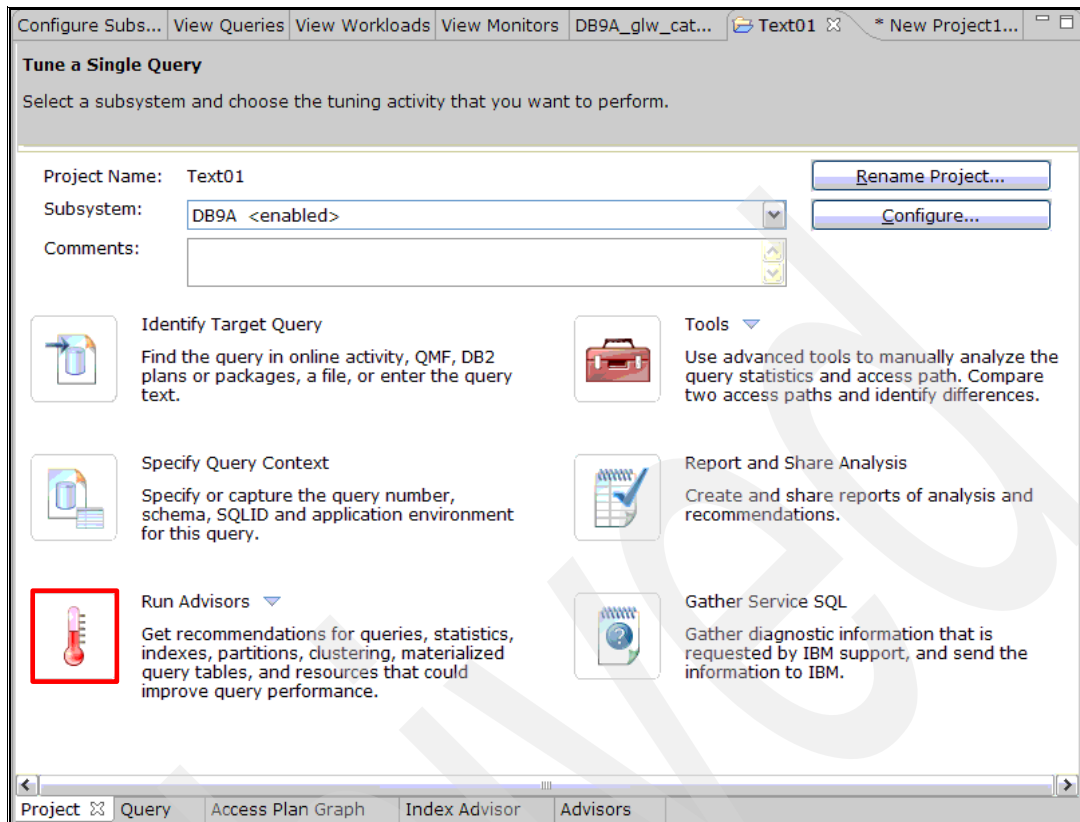


Figure 10-13 The Project tab

The Advisors tab opens as shown in Figure 10-14 on page 255.

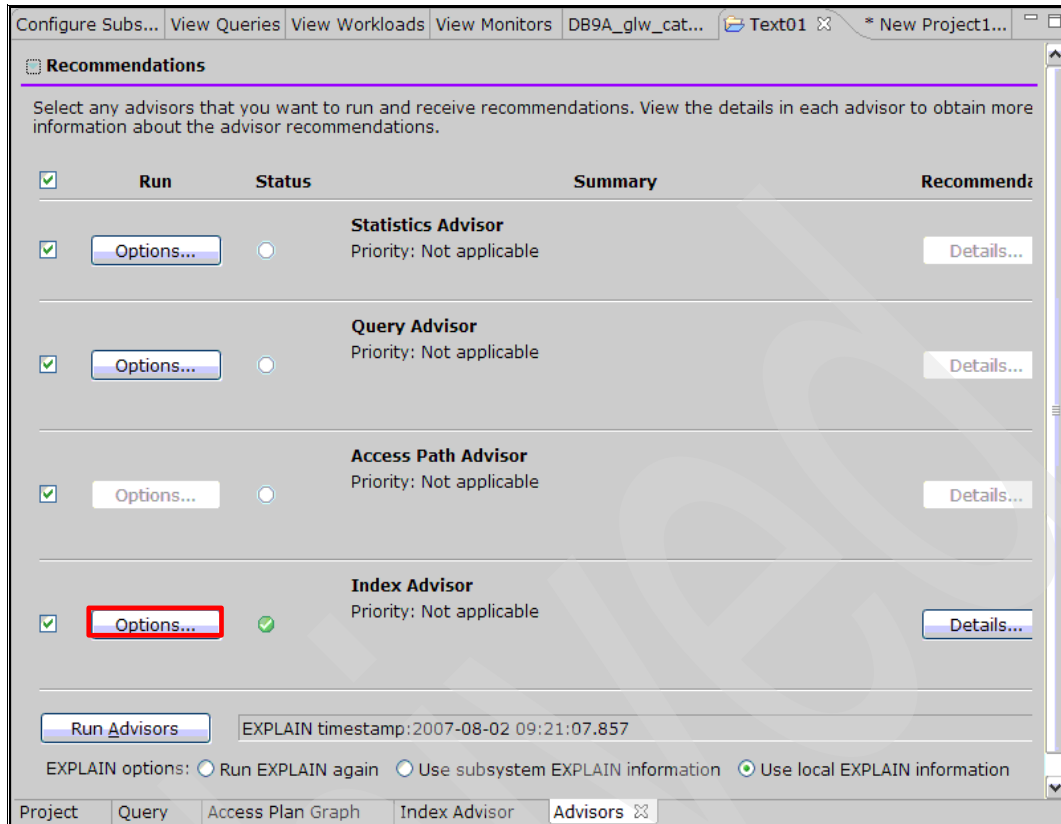


Figure 10-14 The Advisors tab

If you click **Options** next to the words Index Advisor, the Options panel opens, as shown in Figure 10-15.

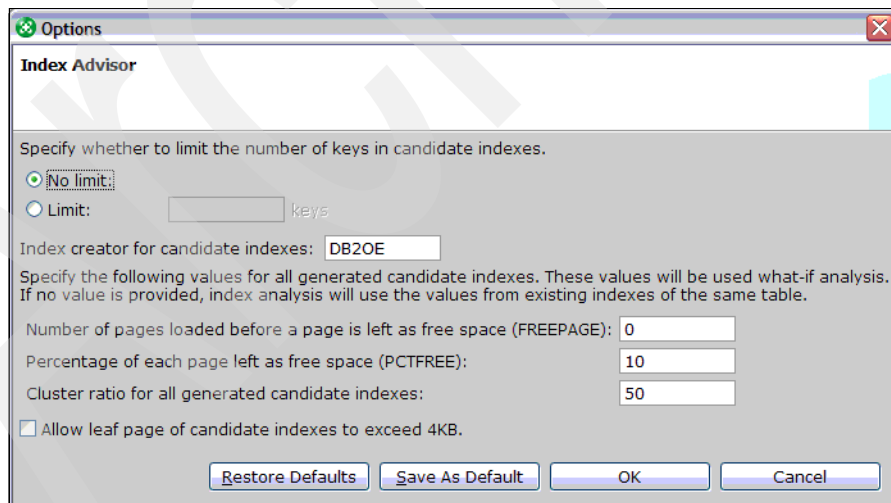


Figure 10-15 Query Index Advisor Options panel

You can specify a maximum for the number of columns that can be part of an index key and change the Creator ID that is used when index DDL is generated. You can also specify the FREEPAGE, PCTFREE, and CLUSTERRATIO values to be assumed for new indexes, and whether or not index leaf page size can exceed 4 KB.

## 10.2.5 Setting Query Index Advisor option defaults for new projects

You can also set the Query Index Advisor defaults that apply to all new query projects. From the menu at the top of the main window, choose the **Tools** pull-down menu and select the **Preferences** option, as shown in Figure 10-16.

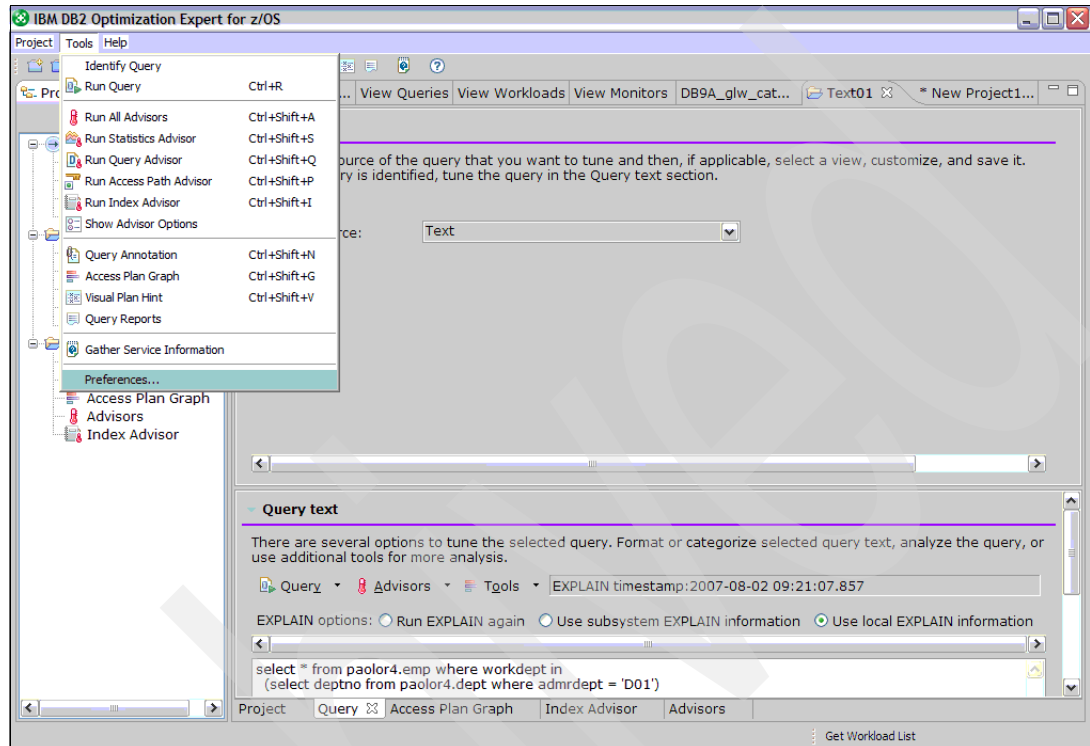


Figure 10-16 Tools menu Preferences option

The Preferences panel displays, as shown in Figure 10-17 on page 257. In the hierarchy of options on the left of the panel, click the plus sign to the left of the Tune Query option. This opens the sub-menu from which you select the Index Advisor. The options shown are the same as those you saw when setting the preferences for an individual project.



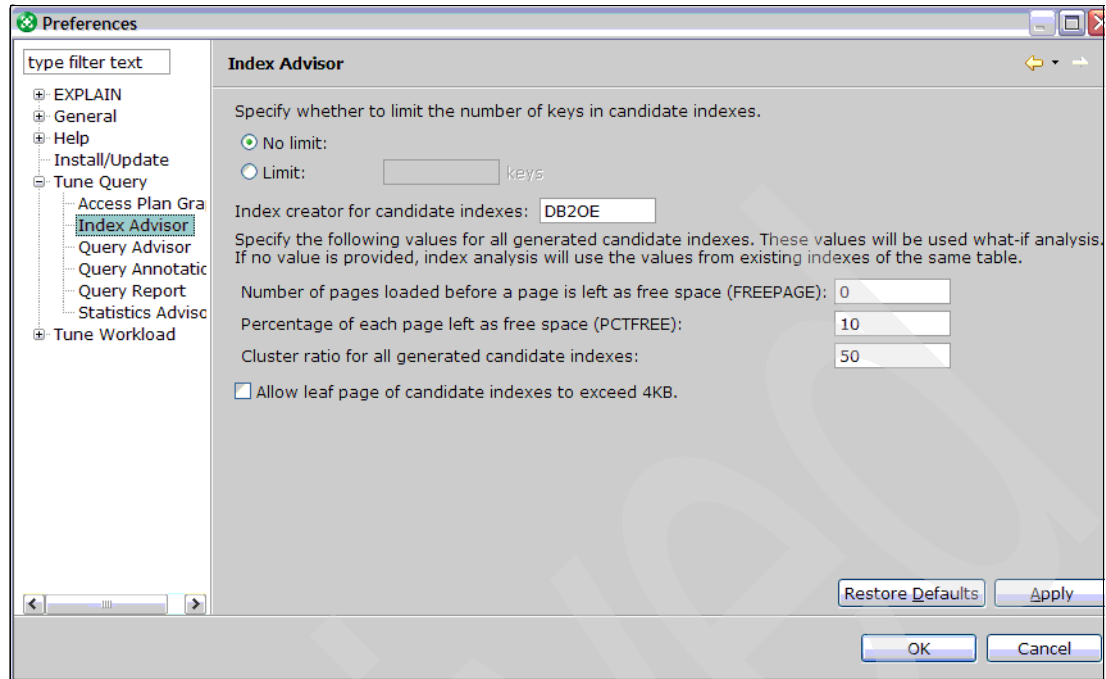


Figure 10-17 The Preferences panel

## 10.3 Workload Index Advisor

This section looks at how you can use the Index Advisor against a workload.

### 10.3.1 Before running Workload Index Advisor

We recommend that you run the Statistics Advisor against the workload before you run the Index Advisor. The Index Advisor gives the best results if up-to-date statistics have been gathered for all the objects referred to by the SQL statements that make up the workload. If any object has no statistics, the Index Advisor will insist that you run Runstats before running the Index Advisor. After applying the RUNSTATS recommendations from the Statistics Advisor, you need to EXPLAIN the workload again to reflect the changes.

**Note:** If your workload includes references to the table SYSIBM.SYSDUMMY1, you can avoid problems if you run a basic Runstats against the table space DSNDB06.SYSEBCDC that contains it before you run the Index Advisor. If the Index Advisor detects that there are no statistics for SYSIBM.SYSDUMMY1, it stops and tells you to run the Statistics Advisor before proceeding (in our case, it stopped after about 10 minutes of processing). The alternative, which is to locate all the statements in the workload that refer to SYSIBM.SYSDUMMY1 and to delete them, is not practical because Optimization Expert gives you no easy way to locate the relevant statements other than by using the Mark 1 eyeball.

### 10.3.2 Running Workload Index Advisor

First, select a workload to run the Index Advisor against. In Figure 10-18, we have selected a workload from the View Workloads window. The workload we have chosen is named `glw_catalog`.

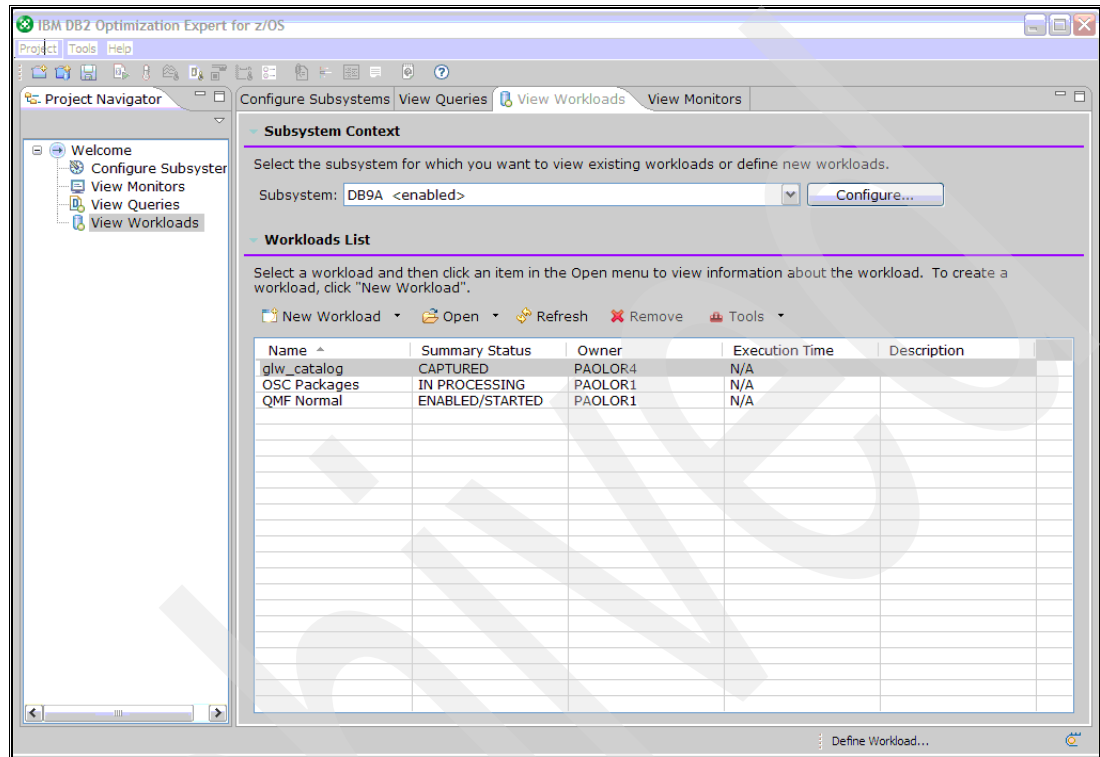


Figure 10-18 Selecting a workload

You are taken to a new project with the default name `DB9A_glw_catalog` (the subsystem name followed by the workload name), as shown in Figure 10-19 on page 259.

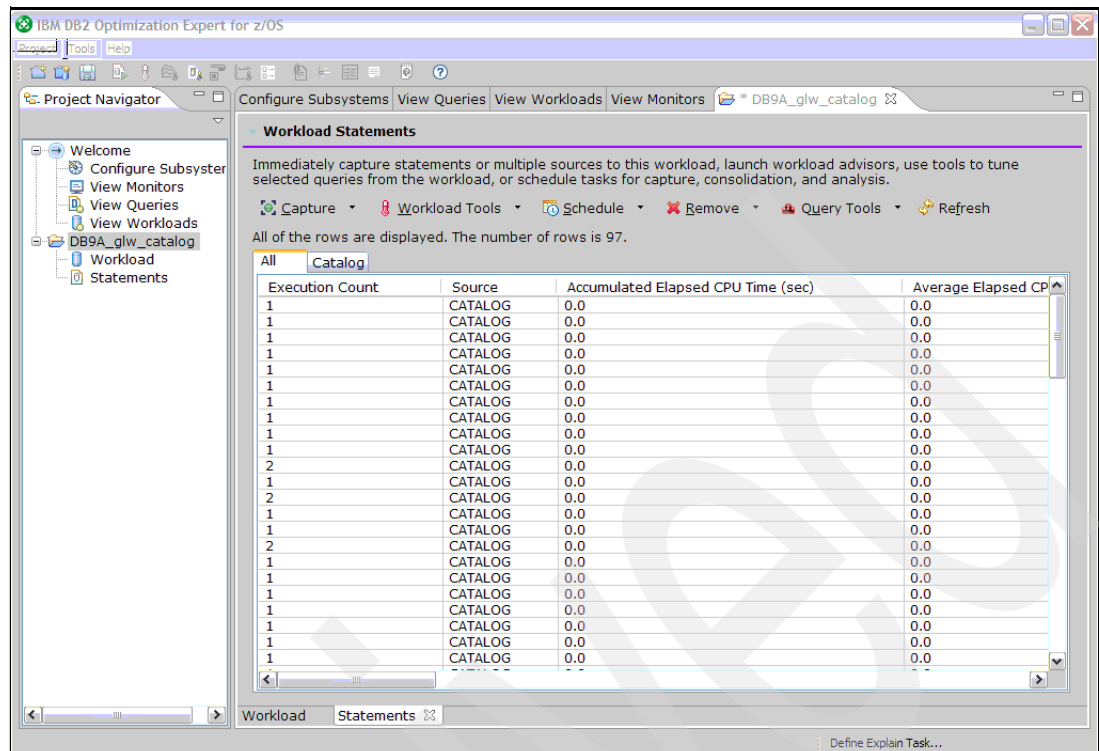


Figure 10-19 The new project

You can see the text of the SQL statements by scrolling the statement list to the right, as shown in Figure 10-20.

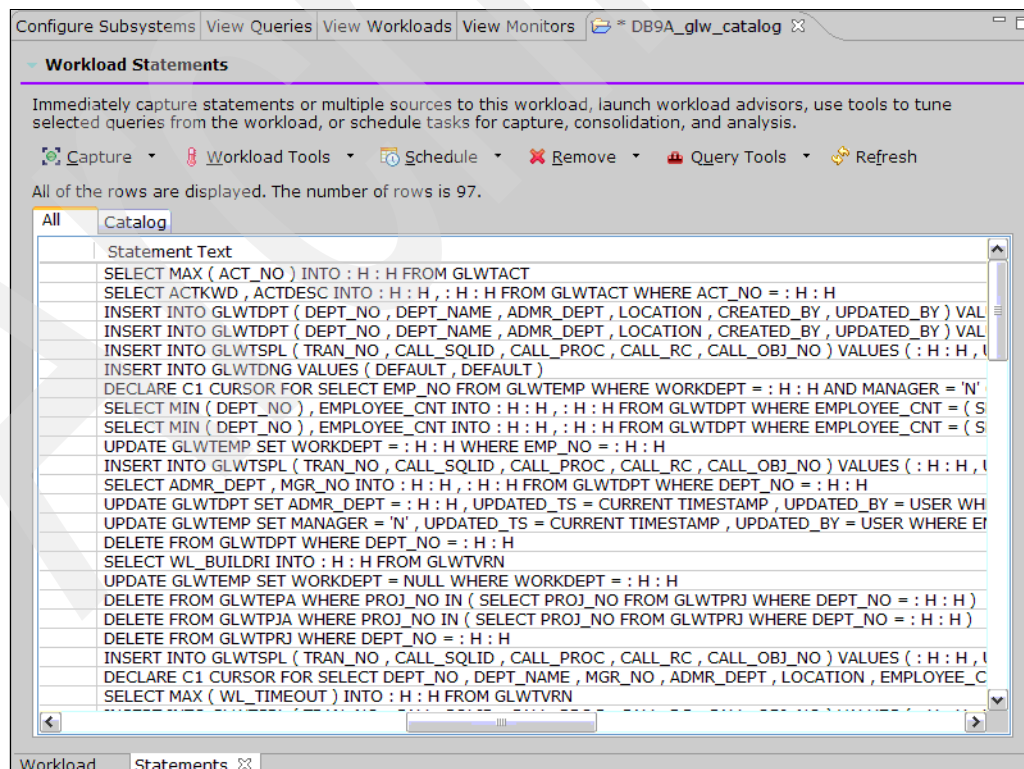


Figure 10-20 The text of the SQL statements

You can use the Workload Tools button above the list of SQL statements to open a pull-down menu (as shown in Figure 10-21) and select the option to Run Workload Index Advisor. For a large workload (our example contained almost 100 statements), the Index Advisor takes time to come up with recommendations.

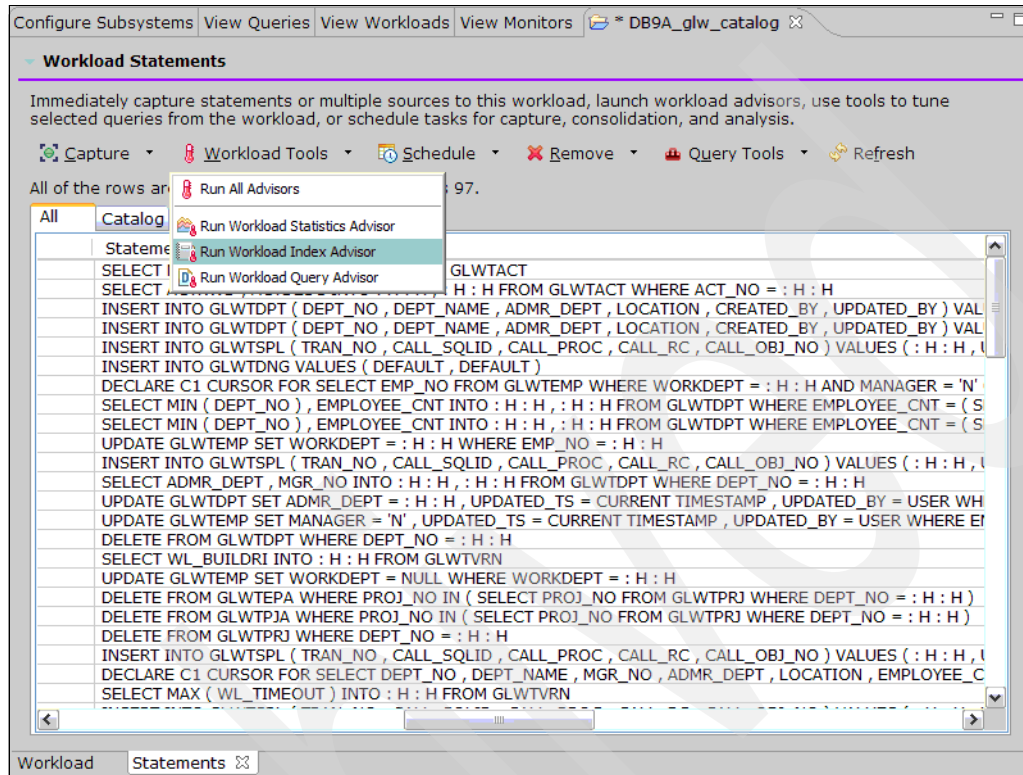


Figure 10-21 Run the Index Advisor

You might be requested to gather Explain information if there is not enough data for Workload Index Advisor to work on, as shown in Figure 10-22 on page 261.

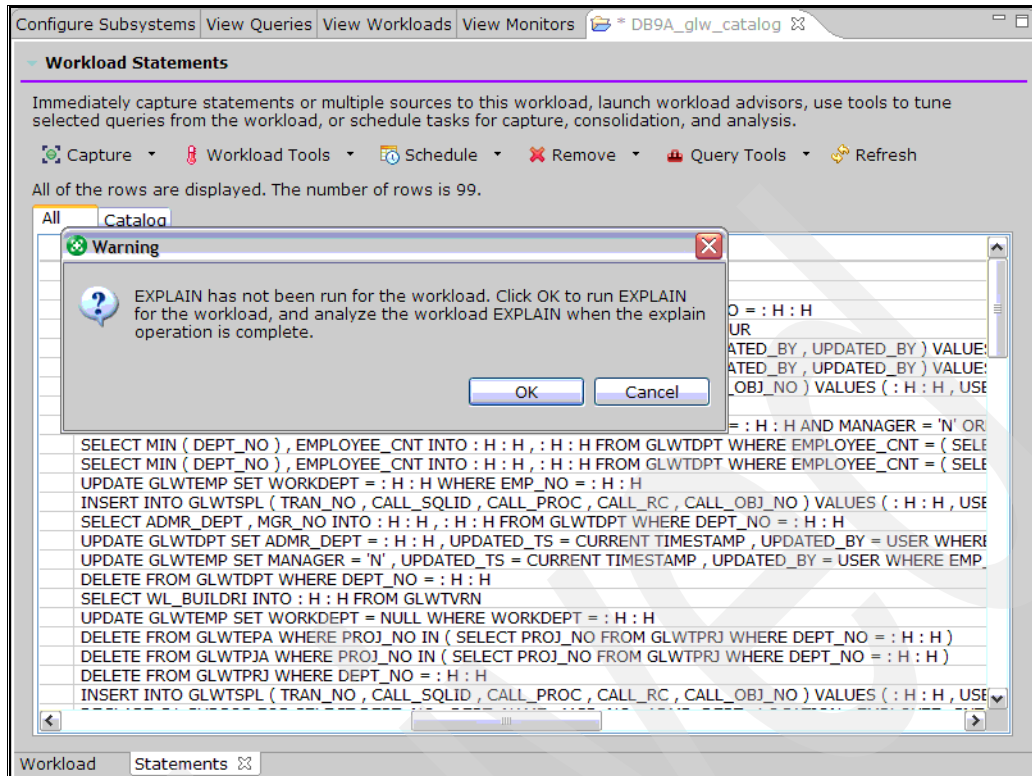


Figure 10-22 EXPLAIN information needed

If you click **OK**, a task runs in the background to gather the Explain information. This can take time, in our case about 15 minutes. Wait for the completion message (see Figure 10-23).

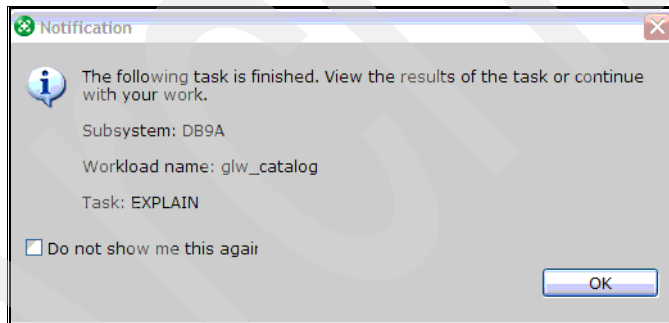


Figure 10-23 Notification that Explain task has completed

You can monitor the background Explain task by switching to the **Workload** tab and clicking **Schedule Tasks** (shown in Figure 10-24 on page 262). You might have to scroll down the page to see the button.



In our case, the task gathered enough Explain information for all except seven statements. This was indicated when we re-tried running the Workload Index Advisor (see Figure 10-26). We chose to proceed with the available information and, therefore, selected the option “Analyze the workload without running EXPLAIN”.

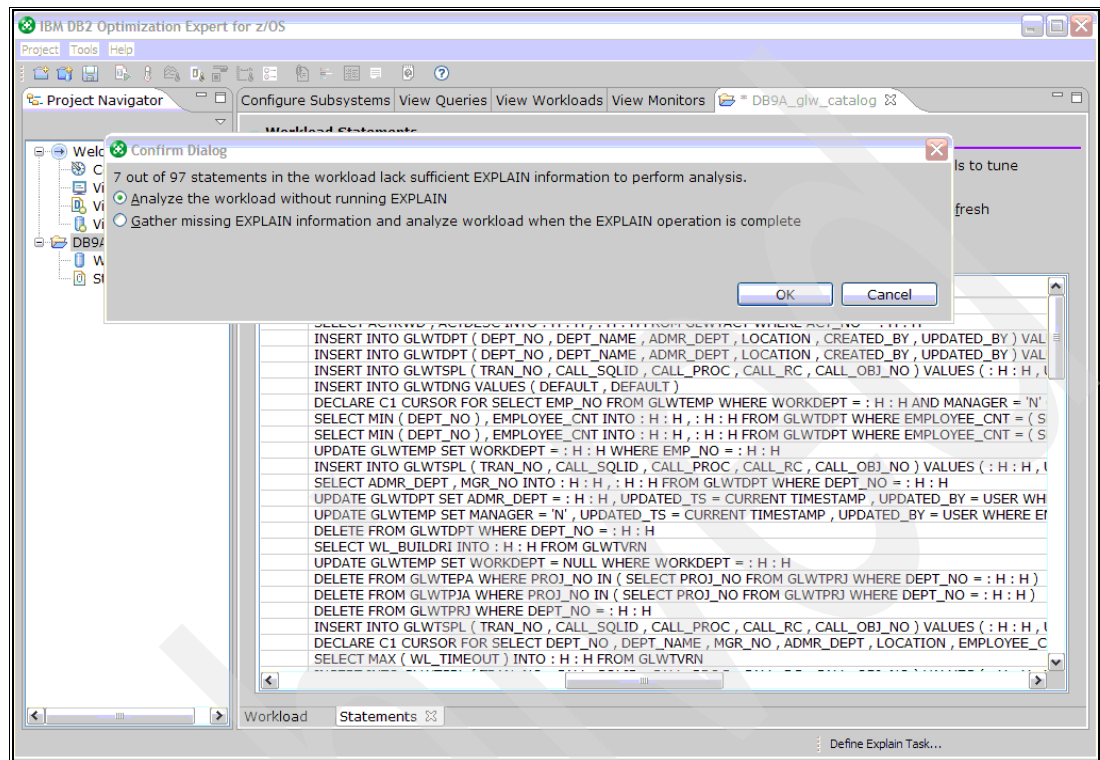


Figure 10-26 Explain information needed

When the Index Advisor starts running, you see a progress indicator (Figure 10-27 on page 264) that shows the elapsed time and the phase that the Index Advisor is currently in.



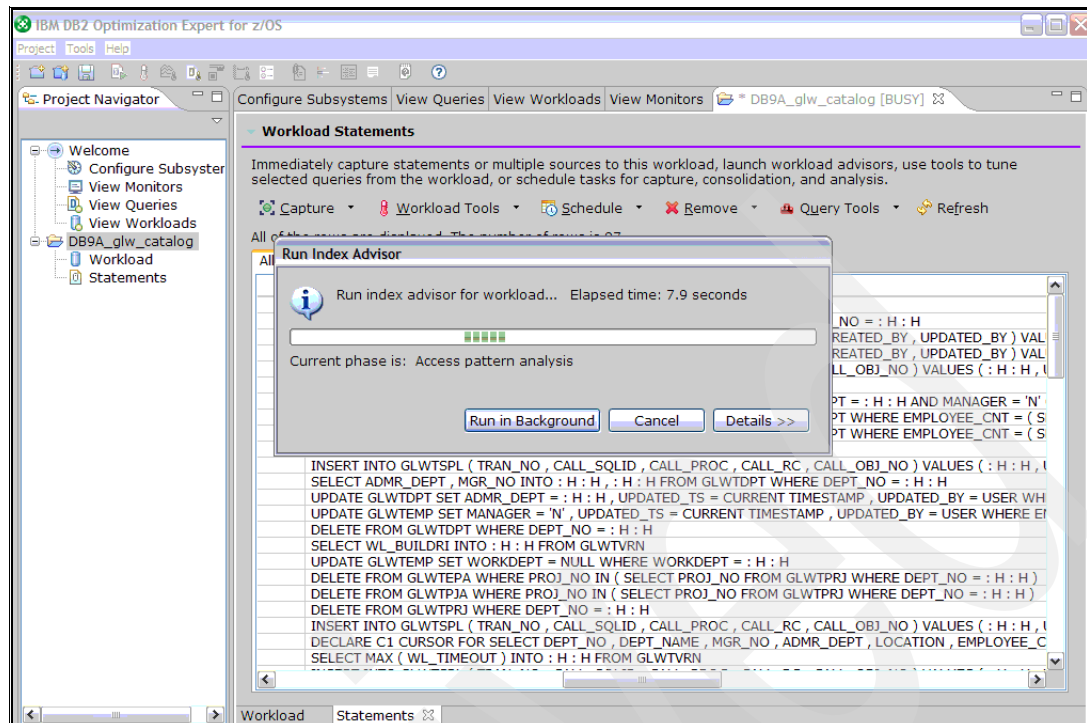


Figure 10-27 Progress indicator

The Index Advisor phases are:

1. *Access Pattern Analysis*: Extracts access pattern from Explain and Catalog information for each SQL statement in the workload.
2. *Referential Constraint Analysis*: Generates indexes based on referential integrity (foreign keys).
3. *Candidate Index Generation*: Generates candidate indexes, considering local predicates, join predicates, and sort avoidance. Favors fully matching index scan while considering local and join predicates.
4. *Candidate Index Expansion*: Expands candidate indexes from previous phase and generates candidate indexes by taking index screening or index only into consideration.
5. *Candidate Index Consolidation*: Consolidates candidate indexes based on the subset/superset and common column relationship.
6. *Candidate Index Recommendation*: Generates the final set of recommended indexes for the entire workload.

If you wish to work with other work in Optimization Expert while the Index Advisor is working on its recommendations, you can click **Run in Background**.

**Note:** When you run an Advisor in the background, the current project is locked. If you try to select any of the tabs within the project, nothing happens. You can, however, switch to another project and work on that.

The Workload Index Advisor might recommend indexes for the following reasons:

- ▶ Foreign keys that do not have indexes defined.
- ▶ Indexes that provide index filtering for SQL statements.
- ▶ Indexes that provide index filtering and screening for SQL statements.



- Indexes that provide index-only access for SQL statements.

If you do not understand the difference between index filtering and index screening, consider the example shown in Figure 10-9 on page 250 and the description that precedes it.

### 10.3.3 Workload Index Advisor recommendations

When the Index Advisor completes, you see the Workload Index Advisor Recommendations panel, as shown in Figure 10-28.

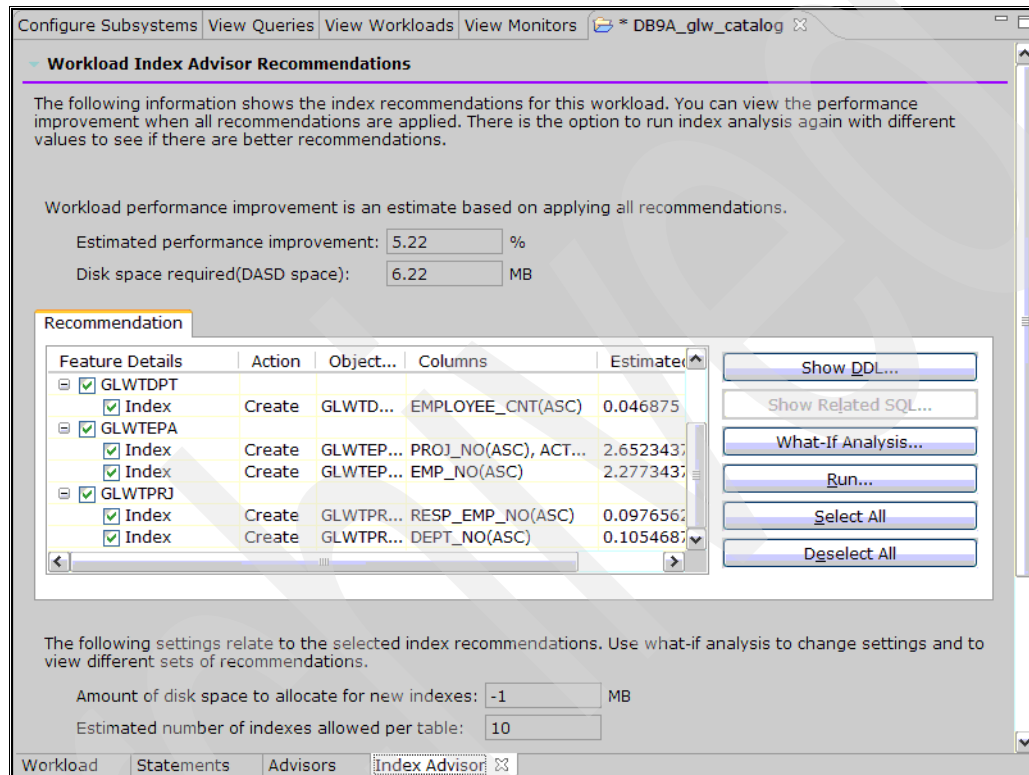


Figure 10-28 Index Advisor recommendations

Lets look at the list of index recommendations and see what the columns mean. The column Feature Details lists a table name (without the Creator ID) and, on the following lines, the indexes that are recommended for that table. The Action column shows the action required for each of the recommended indexes (Create). The next column is the Object Name, which shows the generated index name (again without Creator ID). If you want to see more of this or any other column, you can drag the right hand column separator further to the right. The column Columns shows which columns the recommended index is derived from. The last column is Estimated Disk Space. You can see this by dragging the scroll bar below the index list to the right, as shown in Figure 10-29 on page 266. You can see that, for example, the index on the EMPLOYEE\_CNT column of the table GLWTDPT requires 0.046875 MB of disk space.

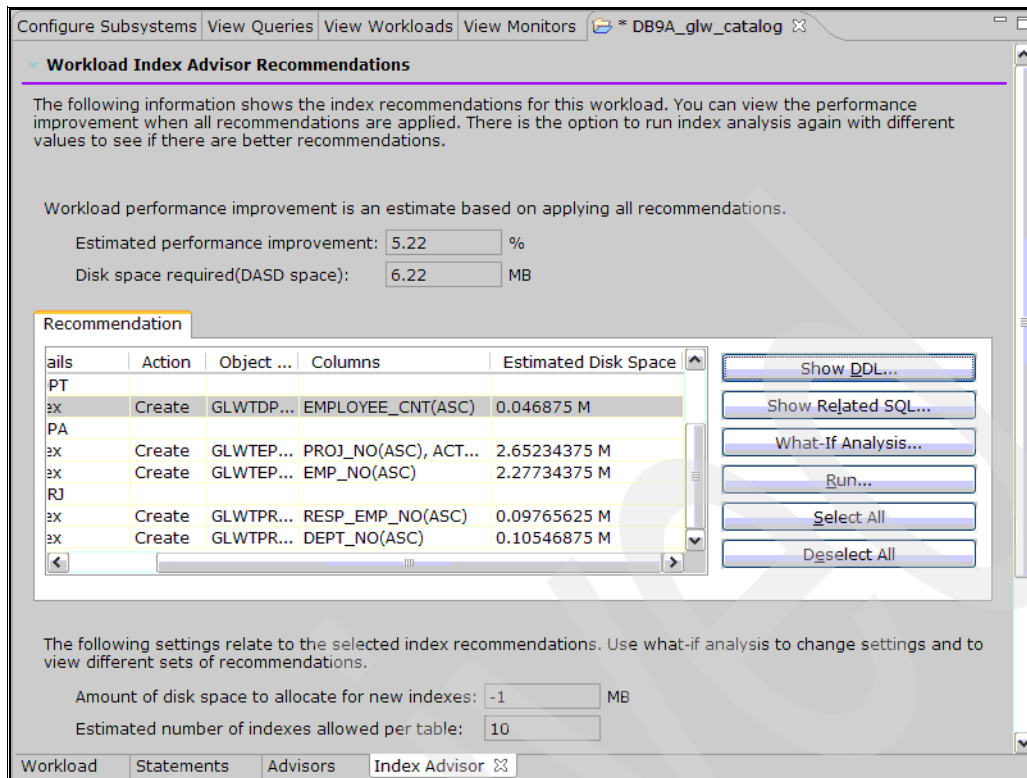


Figure 10-29 Scrolling the index list

Lets look at features you can do from this window. If you click **Show DDL**, the Index Advisor shows the DDL of the Create Index statements that generates all the recommended indexes (see Figure 10-30 on page 267).

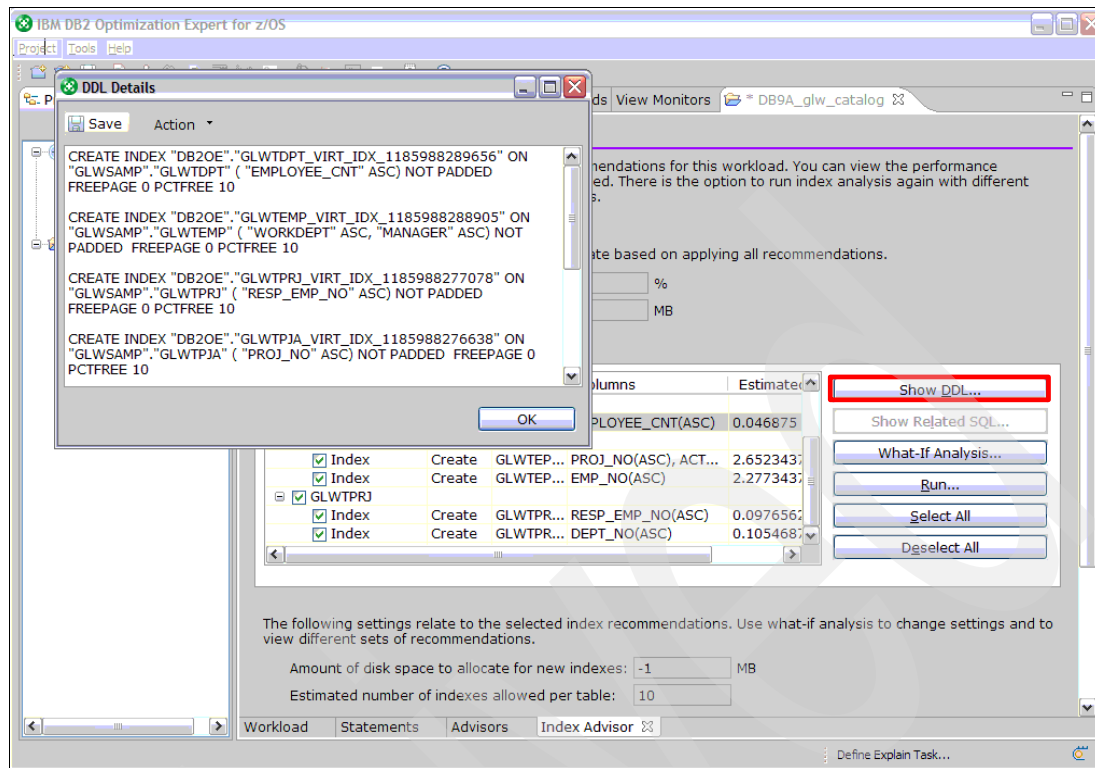


Figure 10-30 Show DDL

You can select a single index by clicking it. In Figure 10-31, we have selected the single index on the EMPLOYEE\_CNT column of the table GLWTDPT.

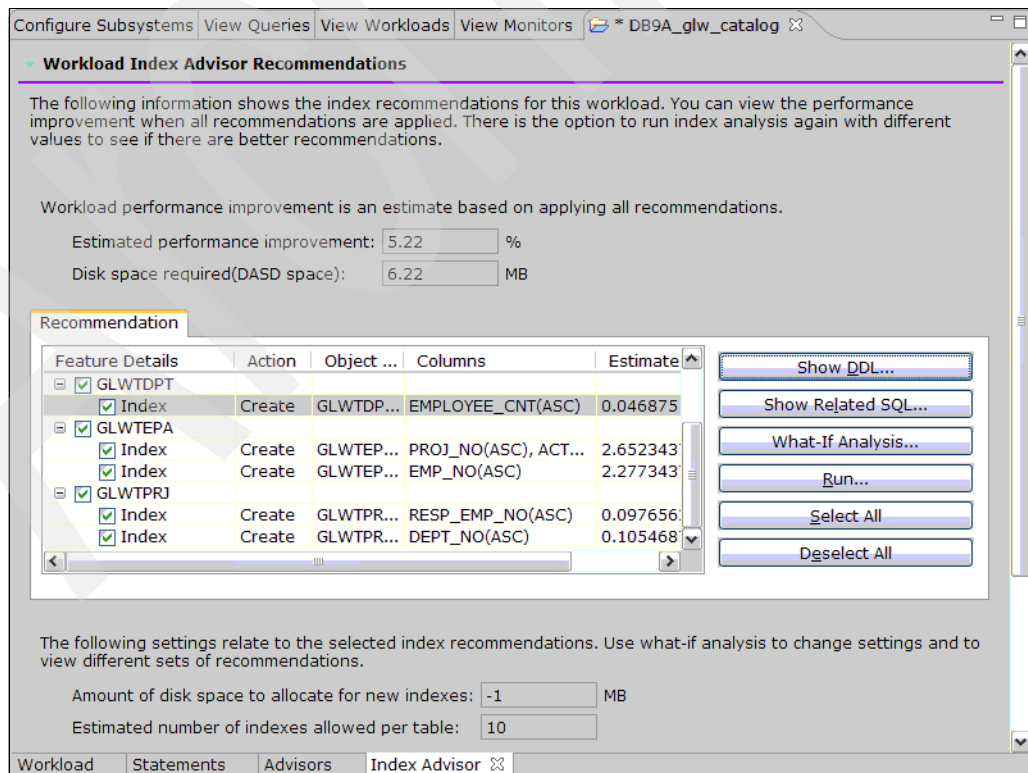


Figure 10-31 Selecting an index for investigation

When we have selected an index, the Show Related SQL button on the right becomes active. If you click this button, you can see the SQL statements from the workload that are expected to benefit from the creation of the selected index recommendation (shown in Figure 10-32).

**Note:** Not every index shows related SQL statements. For example, an index might be recommended to support a foreign key. These indexes do not show any related SQL.

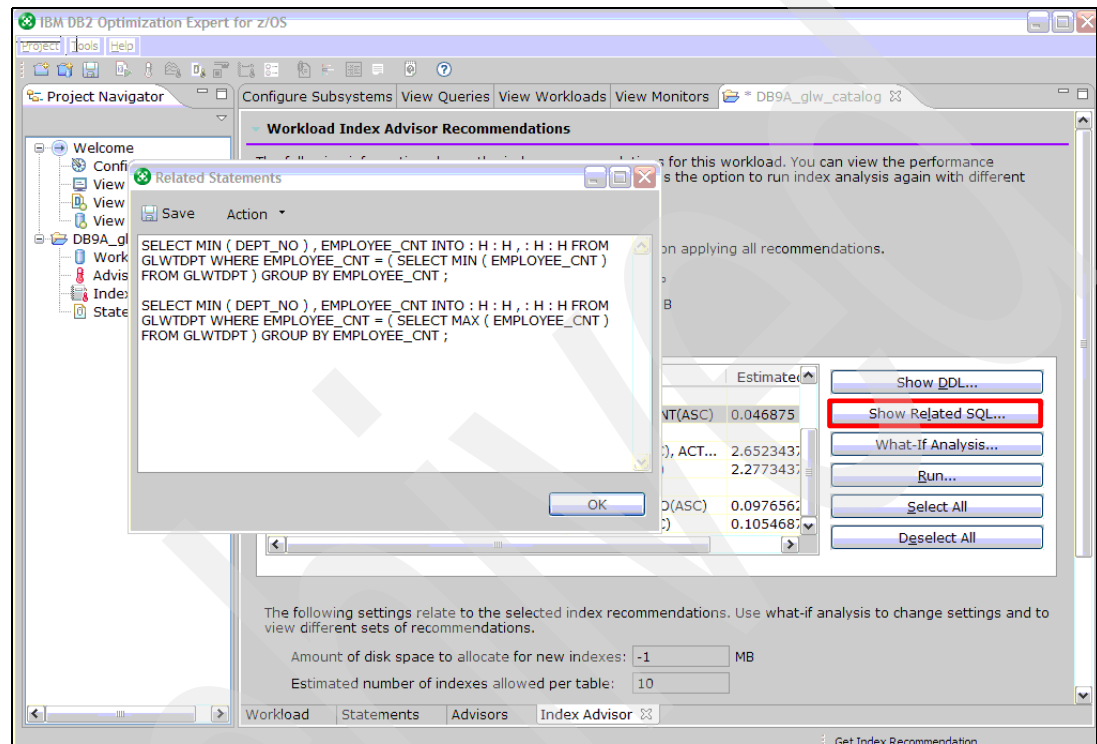


Figure 10-32 Related SQL panel

### 10.3.4 What-If Analysis

You can try applying various constraints to the Index Advisor and see how those constraints modify the recommendations. This function applies to the entire set of index recommendations, not to a single index.

If you click **What-If Analysis** to the right of the index list, a panel appears, as shown in Figure 10-33 on page 269.

Specify the settings for this analysis to output a new set of index recommendations. You can compare recommendations based on the different values specified.

Specify whether to limit the total amount of disk space to allocate for new indexes.

Amount of disk space to allocate for new indexes:

☒ No limit

☐ Limit:  MB

Specify whether to limit the maximum allowable number of indexes created for each table. You can also specify the number of indexes allowed for individual tables and tables created by specific creators.

Number of indexes allowed per table:

☐ No limit

☒ Limit:  [Apply to All](#)

The following items displays tables with customized number of indexes allowed.

Table Creator	Table Name	Number of indexes	

[Add...](#)  
[Remove](#)

[OK](#) [Cancel](#)

Figure 10-33 Specifying What-If options

You can ask the Index Advisor to revise its recommendations based on a restriction on the amount of DASD space allowed for new indexes or on a limit on the maximum number of indexes per table, or both. You can also specify the maximum number of indexes for specific tables, which override the global maximum.

You can see that the defaults are no DASD space restriction and a limit of 10 indexes per table. Those defaults were used for the original recommendations. Lets restrict the DASD space to 5 MB and restrict to a maximum of 5 indexes per table, as shown in Figure 10-34.

Specify the settings for this analysis to output a new set of index recommendations. You can compare recommendations based on the different values specified.

Specify whether to limit the total amount of disk space to allocate for new indexes.

Amount of disk space to allocate for new indexes:

☐ No limit

☒ Limit:  MB

Specify whether to limit the maximum allowable number of indexes created for each table. You can also specify the number of indexes allowed for individual tables and tables created by specific creators.

Number of indexes allowed per table:

☐ No limit

☒ Limit:  [Apply to All](#)

The following items displays tables with customized number of indexes allowed.

Table Creator	Table Name	Number of indexes	

[Add...](#)  
[Remove](#)

[OK](#) [Cancel](#)

Figure 10-34 Specifying new What-If options

When you click **OK**, the Index Advisor re-evaluates its recommendations and presents a new set of recommendations based on the constraints you specified, as shown in Figure 10-35.

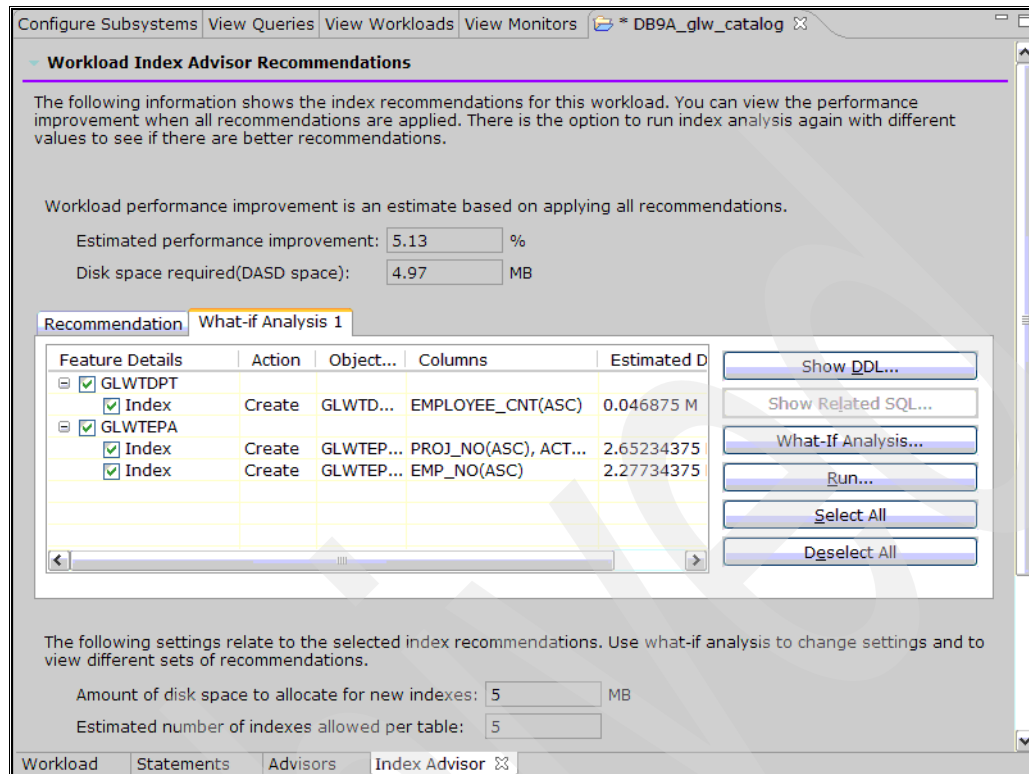


Figure 10-35 Results of What-If analysis

Note that a second tab has appeared at the top of the index list. It is named What-If Analysis 1. The original recommendations are still available on the tab named Recommendation.

If you look at the new recommendations, you see that the Index Advisor is now recommending only three new indexes. The original recommendation was for seven new indexes. The number of indexes has been reduced to meet our constraint that no more than 5 MB of DASD is used by the new indexes. Notice that, in the boxes above the index list, the DASD space requirement for the new indexes is estimated to be 4.97 MB. If you look back at the original recommendations (see Figure 10-28 on page 265), the estimated DASD space requirement was 6.22 MB. The other box shows the estimated performance improvement: 5.13% versus 5.22% for the original recommendations. We have lost performance gain by restricting the DASD space available for the new indexes.

The performance improvement is calculated by comparing the estimated total cost of all SQL statements when the new indexes are implemented with the estimated total cost without the new indexes. The total cost is the sum of the DB2 optimizer's estimated elapsed time for each SQL statement in the workload and is calculated based on both CPU and I/O time.

**Note:** The estimated elapsed time for each SQL statement is retrieved from the column `TOTAL_COST` in the DB2 V9 table `DSN_STATEMNT_TABLE`. This column is populated by the DB2 optimizer, during Explain, with the estimated elapsed time of the SQL statement based on CPU and I/O cost. The value that is placed in the floating point column represents milliseconds elapsed time. The DB2 optimizer uses this value when it compares the cost of different access paths before selecting the lowest cost access path. Do not take this value as an absolute value for performance estimation. It is only valid when using this number for comparisons between access paths at a single point in time. The optimizer estimate for an access path changes over time as new versions of DB2 are installed or as maintenance is applied. At present, this elapsed time calculation overestimates I/O time in most cases.

You can also see, below the index list, the constraints that were in force when the recommendations were generated. The constraints for What-If Analysis 1 were 5 MB DASD and no more than 5 indexes per table. You need to scroll down to see any individual table restrictions on the number of indexes.

We can repeat the What-If analysis a number of times using different constraints and select the best set of recommendations to implement.

**Note:** Although you can produce multiple sets of index recommendations based on different constraints, these multiple sets of recommendations only remain available for the duration of your Optimization Expert session. If you close down Optimization Expert and later re-start it, you see only the most recent What-If set of index recommendations for the project.

### 10.3.5 Acting on the recommendations

You can run the Create Index statements for either the original recommendation or one of the What-If scenarios directly from the tool if you are satisfied with the DDL as generated. Click **Run**. It is likely, however, that you want to modify the DDL to conform to your installation naming standards. To do this, first save the DDL as a text file. From the DDL details panel (see Figure 10-30 on page 267), you can click the **Save** menu, which allows you to specify a file name for the text file. You can then modify the DDL and run it using any method available to you for running DDL, such as SPUFI, QMF, and so on.

**Note:** Unlike the Query Index Advisor, you cannot currently change the default Creator ID that the Workload Index Advisor will use when generating the Create Index DDL.

You might also choose to implement only a subset of the indexes recommended by the Index Advisor. Again, you can do this by saving and editing the Create Index statements outside the Index Advisor. You can also use the tick boxes in the index list to select only the indexes you want to create before you click **Run**.

**Note:** You must have Create DB2 privileges to create the indexes.

### 10.3.6 Setting Workload Index Advisor options for the current project

Before you run the Workload Index Advisor, set the options specified for the What-If analysis and a number of others. From the Workload tab, click **Run Advisors** as shown in Figure 10-36.

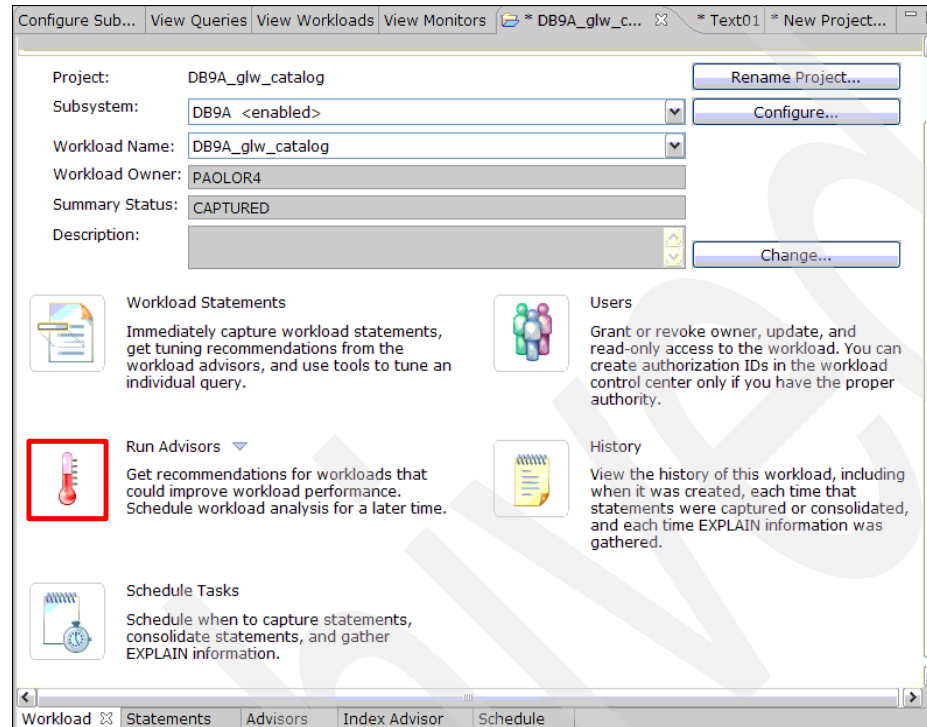


Figure 10-36 Open the Advisors tab



The Advisors tab looks like Figure 10-37. Click **Options** for the Index Advisor.



Figure 10-37 The Advisors tab

A panel opens, as shown in Figure 10-38.

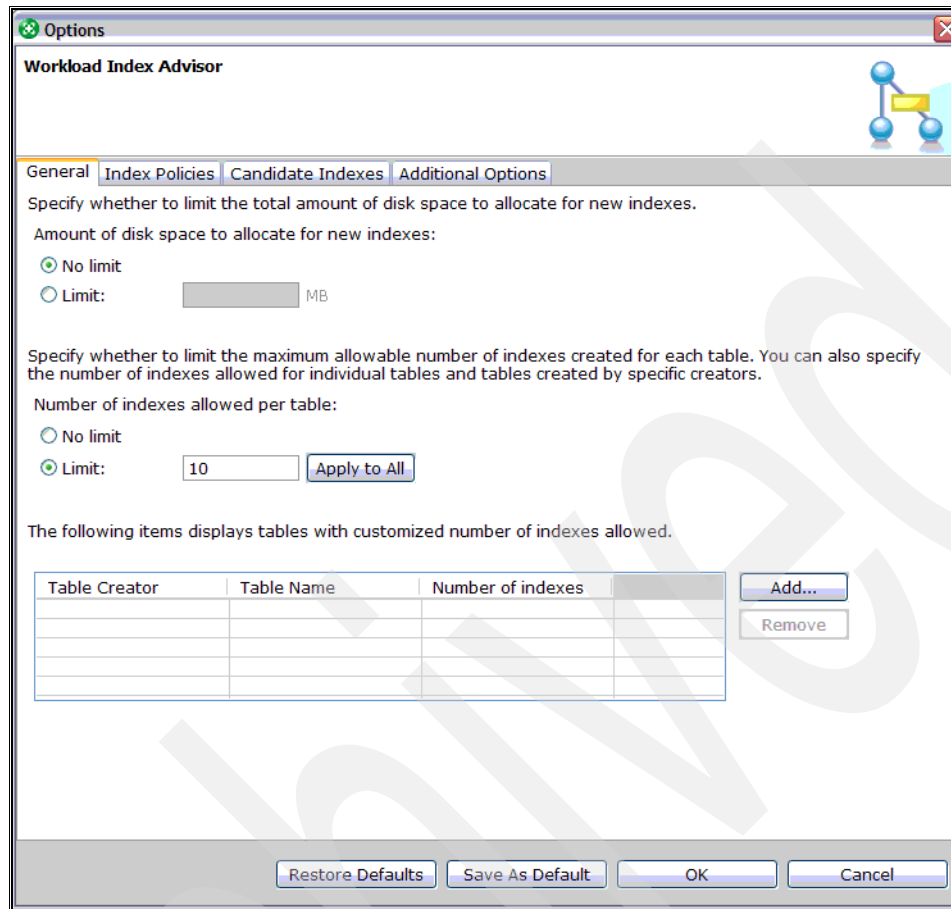


Figure 10-38 Workload Index Advisor options - Tab 1

On this panel, the first tab shows the options you have already seen when specifying the What-If criteria. However, there are other tabs along the top of the Preferences panel. The options on these other tabs are set before you run the Index Advisor. When set here, they affect subsequent Index Advisor runs for the current project. Click on the second tab named **Index Policies**. The result is shown in Figure 10-39 on page 275.

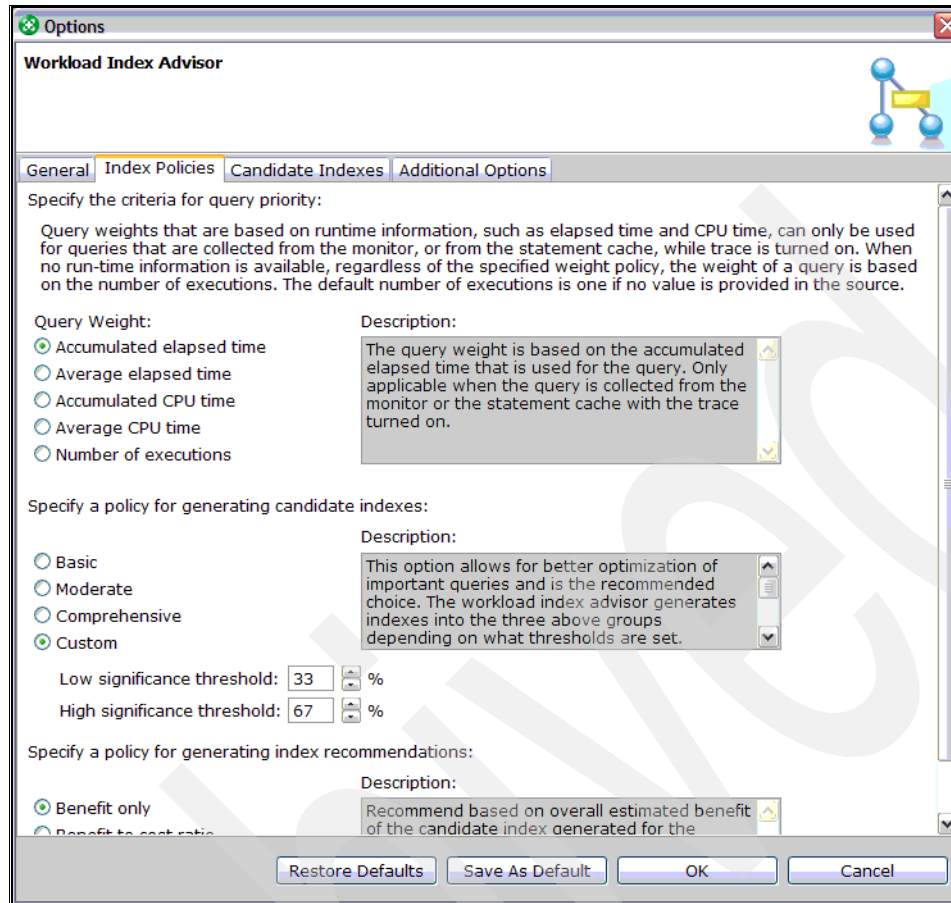


Figure 10-39 Workload Index Advisor options - Tab 2

The tab offers three criteria that you can set: the Query Weight, a policy for generating candidate indexes, and a policy for generating index recommendations.

You weigh the SQL statements in the workload by one of a number of criteria:

- ▶ **Accumulated elapsed time:** The relative weight of each statement is based on the accumulated elapsed time of all executions of the statement. It is only applicable when the statement is collected from the monitor or the statement cache with the trace turned on.
- ▶ **Average elapsed time:** The relative weight of each statement is based on the average elapsed time of a single execution of the statement. It is only applicable when the statement is collected from the monitor or the statement cache with the trace turned on.
- ▶ **Accumulated CPU time:** The relative weight of each statement is based on the accumulated CPU time of all executions of the statement. It is only applicable when the statement is collected from the monitor or the statement cache with the trace turned on.
- ▶ **Average CPU time:** The relative weight of each statement is based on the average CPU time of a single execution of the statement. It is only applicable when the statement is collected from the monitor or the statement cache with the trace turned on.
- ▶ **Number of executions:** The relative weight of each statement is based on the number of times the statement has been run. The default for the number of executions is one if the statement is not from the monitor nor from the statement cache with the trace turned on.

This weighting causes the Index Advisor to favor indexes that benefit high weighted SQL statements.

The tab also offers four policies for generating candidate indexes:

- ▶ *Basic*: This option generates indexes favoring matching index scans.
- ▶ *Moderate*: This option generates indexes favoring matching and screening index scans.
- ▶ *Comprehensive*: This option generates indexes favoring matching and screening index scans, and index only access.
- ▶ *Custom*: This option allows for better optimization of important statements and is the recommended choice. The Workload Index Advisor generates indexes into the three above groups, depending on what thresholds are set. Statements that are below the low significance threshold use the basic policy. Statements that are between the two thresholds use the moderate policy. Statements that are above the high significance threshold use the comprehensive policy.

The significance thresholds referred to in the last item are compared to a value for each SQL statement; that is, the weight of the single SQL statement as a percentage of the total weight of all SQL statements in the workload. You can set the threshold values as you wish.

The third criteria on the tab is the policy for generating index recommendations. There are only two options, which you can see by scrolling to the bottom of the panel (see Figure 10-40).

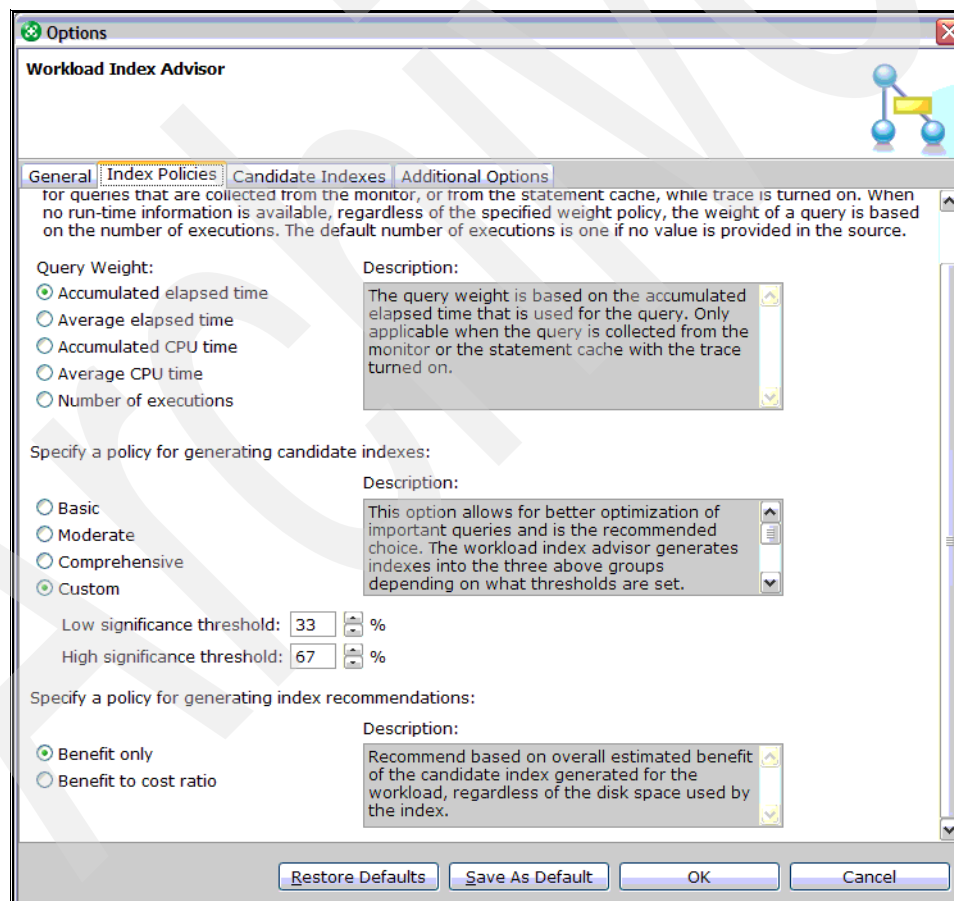


Figure 10-40 Workload Index Advisor options - Tab 2, scrolled down

The options are:

- ▶ **Benefit only:** Recommendations are based on overall estimated performance benefit of the candidate index generated for the workload, regardless of the disk space used by the index.
- ▶ **Benefit to cost ratio:** Recommendations are based on overall estimated performance benefit of the candidate index generated for the workload, compared to the disk space used by the index.

On the third tab, named Candidate Indexes (shown in Figure 10-41), you can limit the number of key columns in a candidate index, specify the FREEPAGE and PCTFREE values to be used for the indexes and specify whether indexes are created on foreign keys.

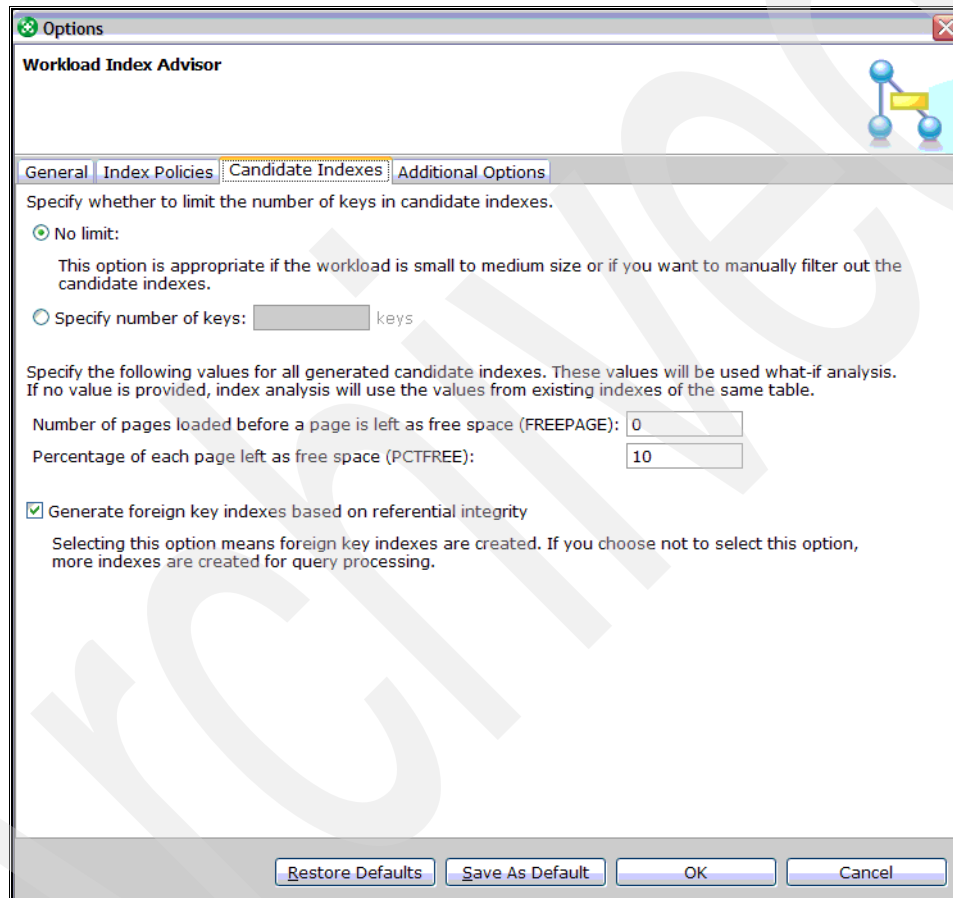


Figure 10-41 Workload Index Advisor options - Tab 3

The last tab is named Additional Options and is shown in Figure 10-42. The only available option allows index page sizes greater than 4 KB.

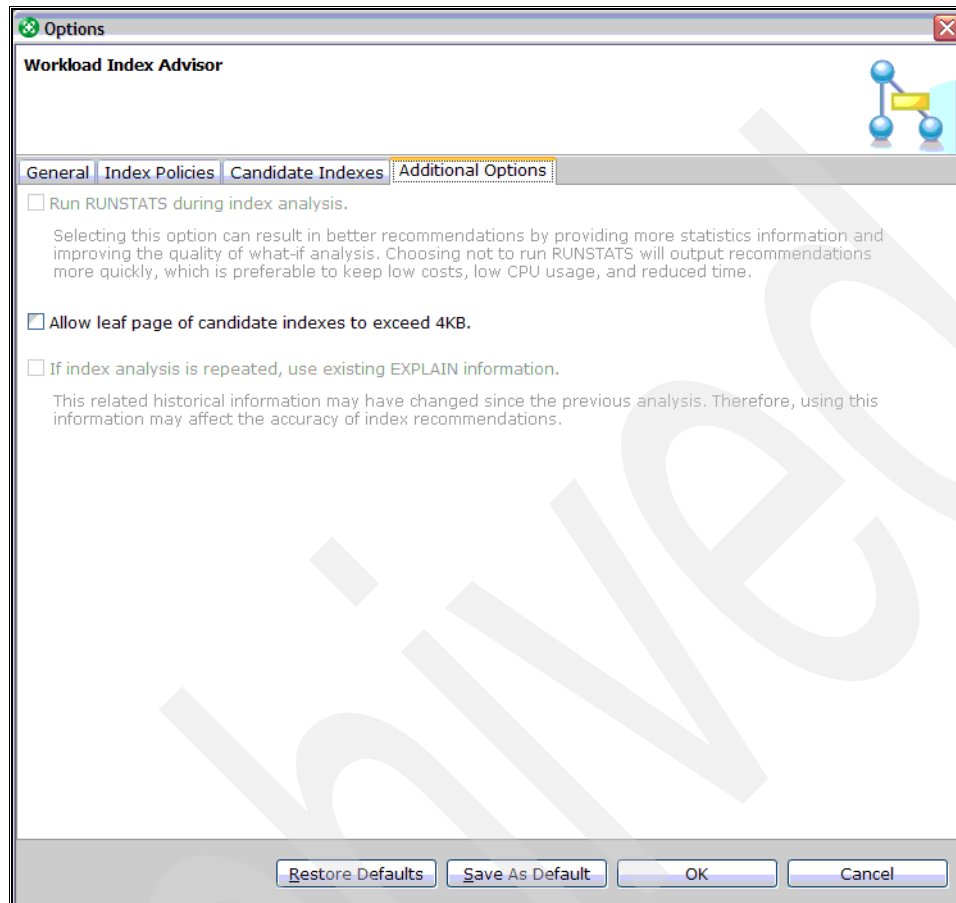


Figure 10-42 Workload Index Advisor options - Tab 4

### 10.3.7 Setting Workload Index Advisor option defaults for new projects

You can also set the Workload Index Advisor defaults that apply to all new workload projects. From the menu at the top of the main window, choose the **Tools** pull-down menu and select the **Preferences** option, as shown in Figure 10-43 on page 279.

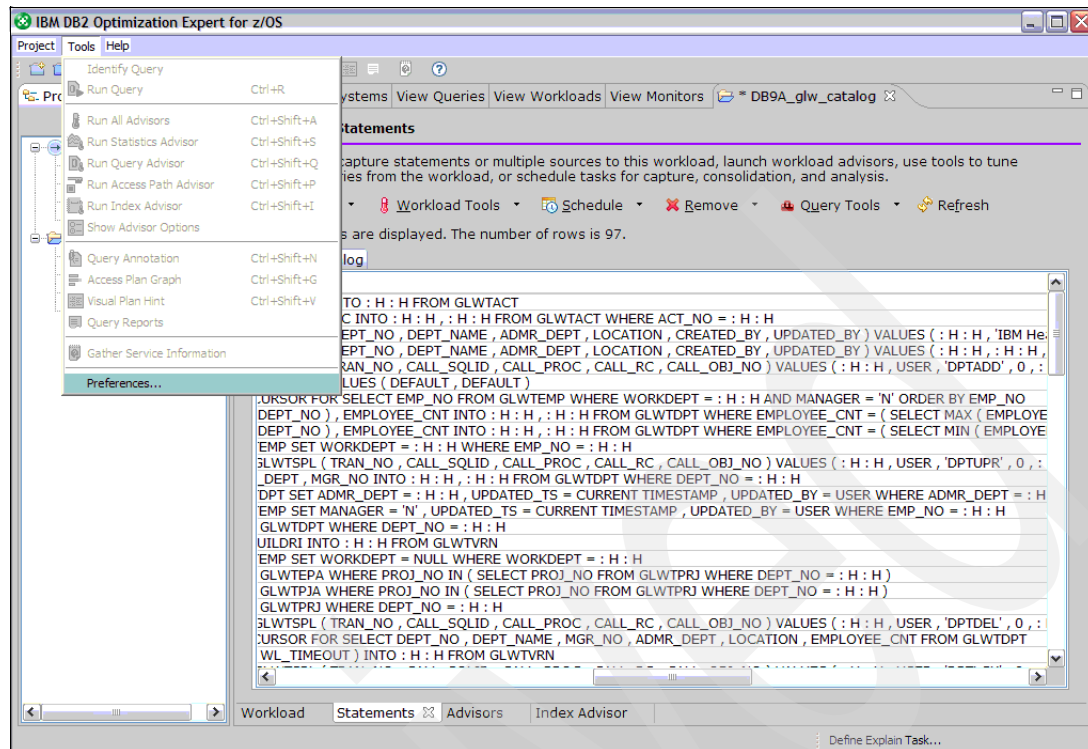


Figure 10-43 Tools Preferences menu

Open the Tune Workload folder by clicking the plus sign beside it and then select **Workload Index Advisor**, as shown in Figure 10-44.

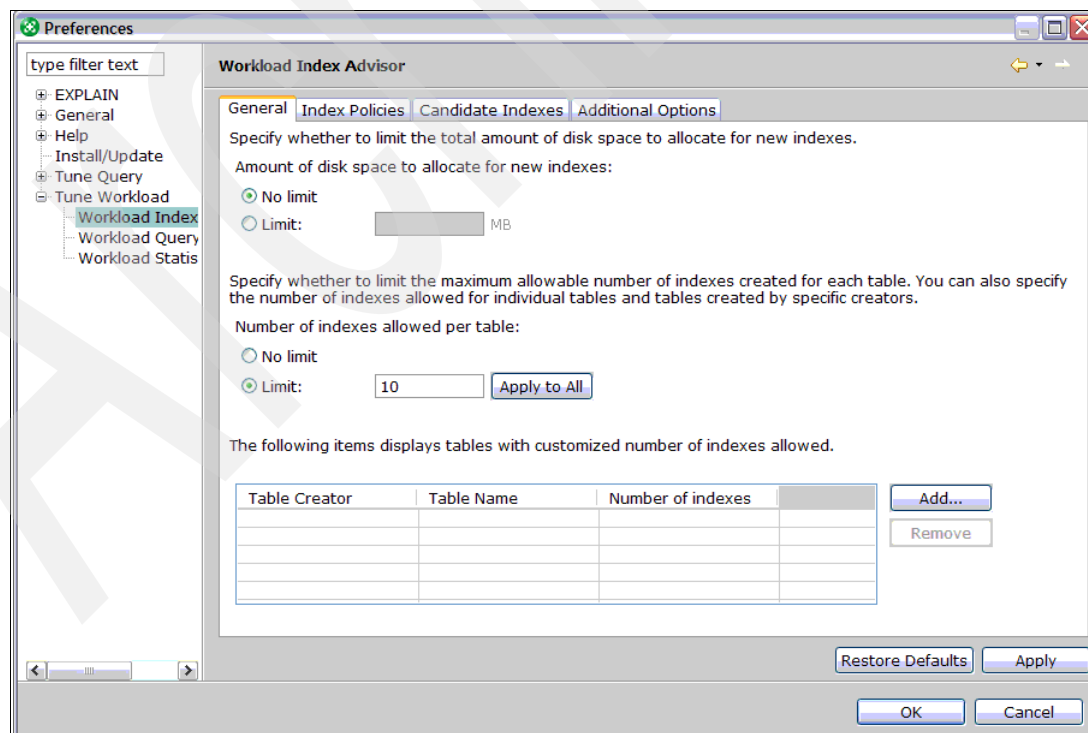


Figure 10-44 Workload Index Advisor preferences

The options shown here are the same as those you saw when setting the preferences for an individual project, except for the second tab, named Index Policies (see Figure 10-45).

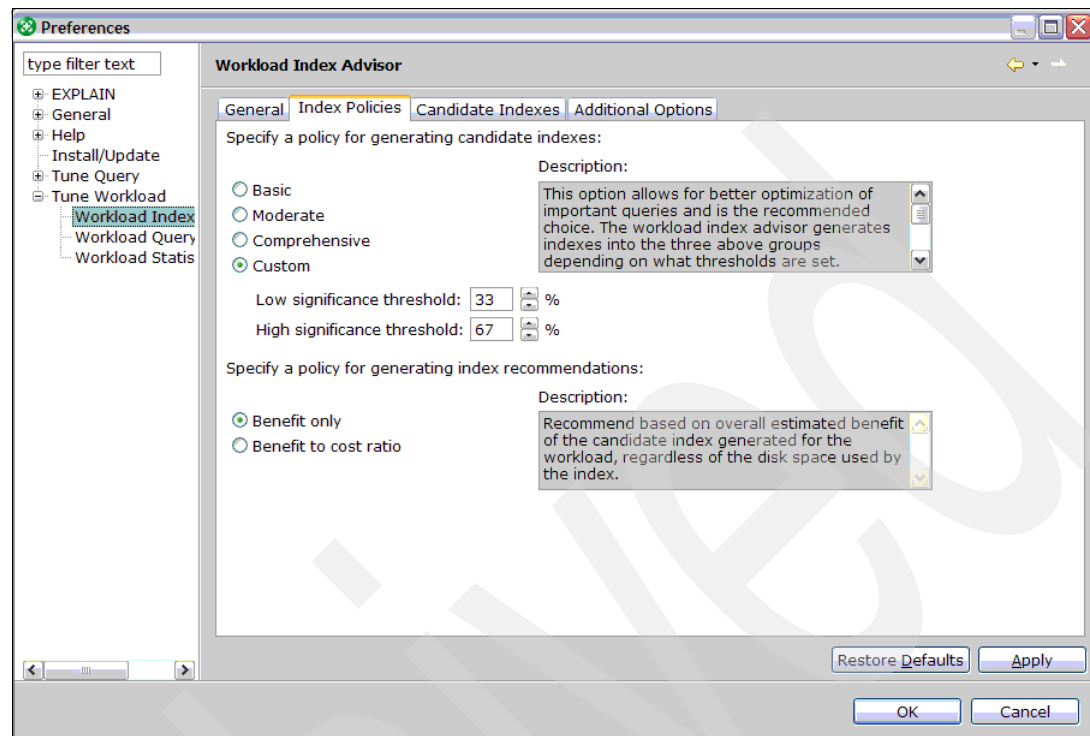


Figure 10-45 Index Policies tab

The option to set the query weighting preference, at the top of the tab when we set the preferences for the current project, is not shown here when we set the preferences for new projects. This is because the query weights apply to all the Advisors and not just the Index Advisor. Select the **Tune Workload** folder on the left to see this option, as shown in Figure 10-46.

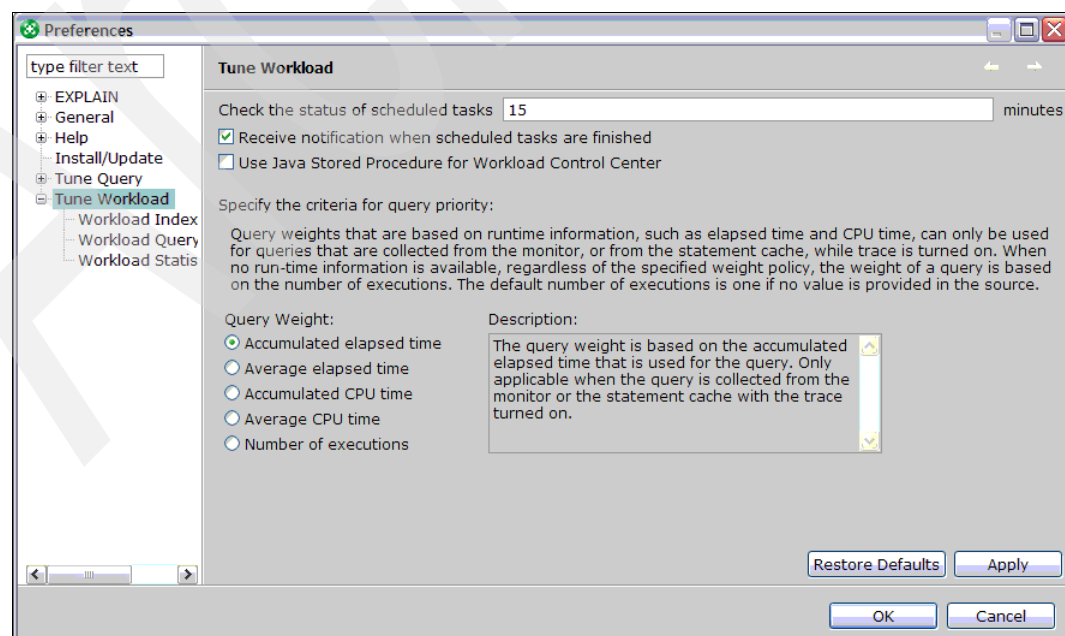


Figure 10-46 Tune Workload panel



On this panel, you can also set other preferences that apply to all Advisors. The top box sets the number of minutes between checks on the status of background tasks. The default means that it might be up to 15 minutes after a task ends before Optimization Expert notifies you. You can also choose whether or not Optimization Expert notifies you. You can choose whether or not the Workload Control Center is controlled directly by the tool or by a stored procedure.

Archived

Archived

## Access Path Advisor and Query Advisor

This chapter describes the functions of the Access Path Advisor and Query Advisor, which are components of Optimization Expert for z/OS.

This chapter contains the following topics:

- ▶ “Creating a workload from a file” on page 284
- ▶ “Access Path Advisor” on page 292
- ▶ “Query Advisor” on page 300
- ▶ “Workload Query Advisor” on page 307

## 11.1 Creating a workload from a file

To investigate the Access Path Advisor and Query Advisor, we use a workload derived from a file. In this section, we briefly look at the issues that arise when a workload is imported from a file. We use a text file containing SQL statements that are separated by semi-colons. It is important to remove any extraneous characters, such as “\*” or JCL, which prevent SQL statements from being processed correctly. Figure 11-1 shows the first part of the file we will use.

```
SELECT P_NAME, SUM(PS_AVAILQTY), COUNT(*)
  FROM PART, PARTSUPP
 WHERE P_PARTKEY  = PS_PARTKEY
    AND p_name like '%green%'
GROUP BY P_NAME
ORDER BY P_NAME;

SELECT o_orderpriority, sum(o_shippriority),count(*)
  FROM ORDER
 where O_Custkey between 00000 and 45000
GROUP BY o_orderpriority
ORDER BY o_orderpriority;

SELECT o_orderpriority, sum(o_shippriority),count(*)
  FROM ORDER
 where O_Custkey between 00000 and 60000
GROUP BY o_orderpriority
ORDER BY o_orderpriority;

SELECT N_NAME, COUNT(*), SUM(C_ACCTBAL)
  FROM CUSTOMER, NATION_NP
 WHERE C_NATIONKEY = N_NATIONKEY
    AND N_NATIONKEY < 10
GROUP BY N_NAME;

SELECT N_NAME, COUNT(*), SUM(C_ACCTBAL)
  FROM CUSTOMER, NATION
 WHERE C_NATIONKEY = N_NATIONKEY
    AND N_NATIONKEY < 10
GROUP BY N_NAME;
```

Figure 11-1 Sample file of SQL statements

Lets create a workload from the file of SQL statements. The file must be located on the workstation where we are running Optimization Expert. We create a new workload project (see Figure 11-2 on page 285).

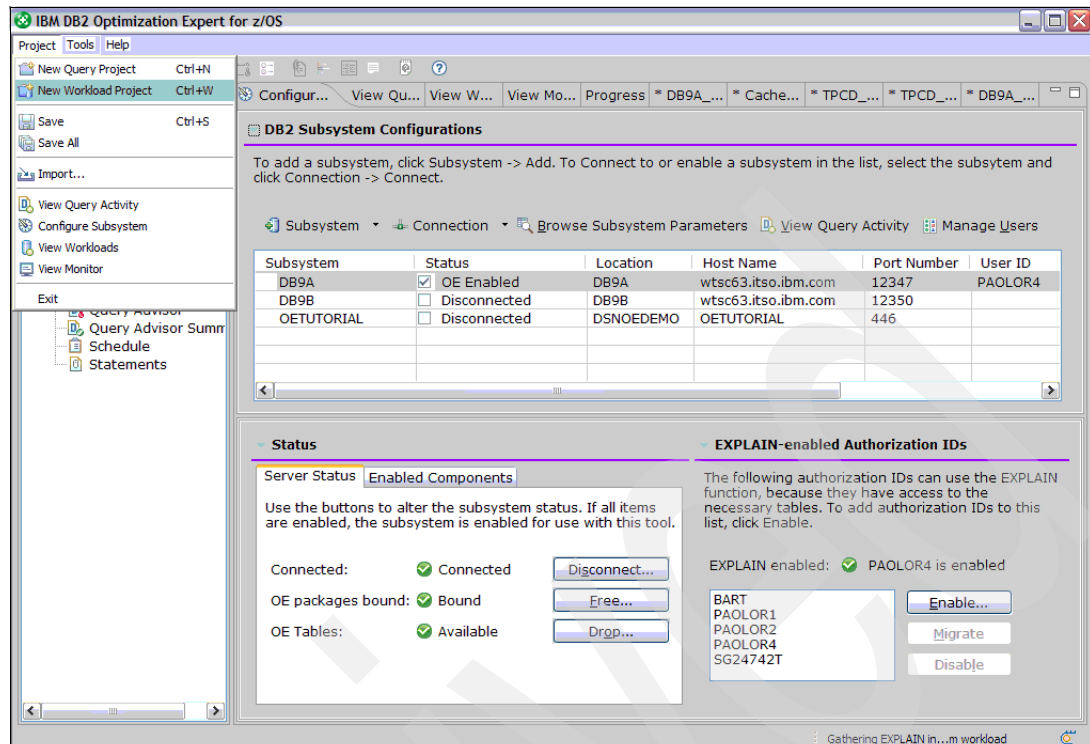


Figure 11-2 Create a new workload project

We name the project File\_Workload, as shown in Figure 11-3.

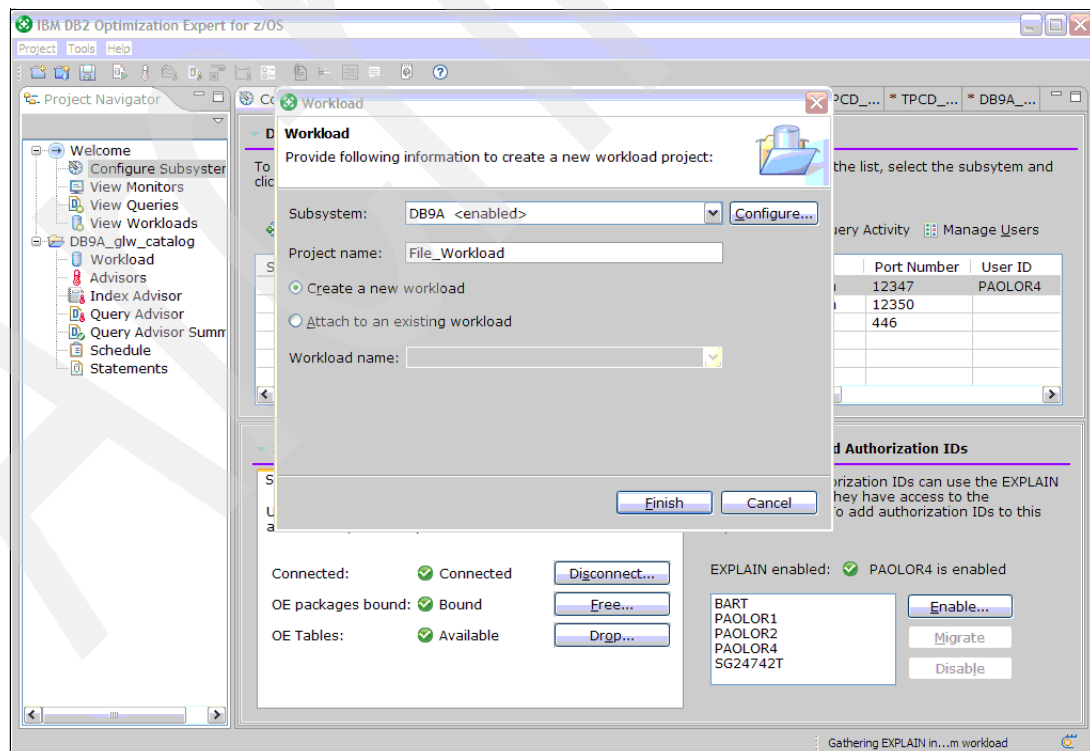


Figure 11-3 Name the project

You are taken to the first panel of the Workload Wizard, shown in Figure 11-4, where we decide to use the project name as the workload name (this is the default). Click **Next**.

The screenshot shows the 'Workload Wizard' dialog box. The title bar says 'Workload Wizard'. Below the title bar, the text 'Workload' is followed by 'Provide the following information to create a workload.' On the left, a 'Steps' pane lists four steps: 1. Workload (selected), 2. Source, 3. Filter, and 4. Capture. The main area contains three input fields: 'Workload name:' with the text 'File\_Workload', 'Owner:' with the text 'PAOLOR4', and 'Description:' with an empty text box. At the bottom, there are four buttons: '< Back', 'Next >', 'Finish', and 'Cancel'.

Figure 11-4 Set the workload name

On the next panel, assign a name to the source (File01), select the source type as **File** from the pull-down list, and add a comment, as shown in Figure 11-5 on page 287. Click **Next**.

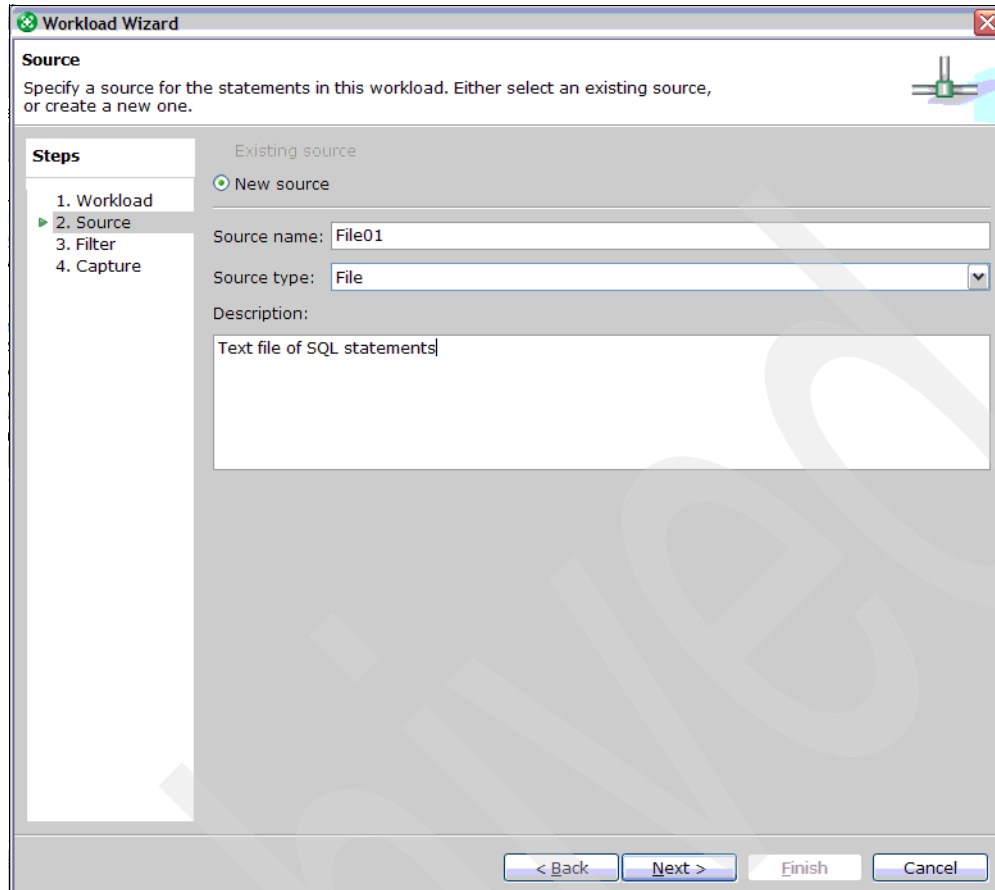


Figure 11-5 Define the workload source

On the next panel, specify the Qualifier if the input text file contains any SQL statements with unqualified table names. The Qualifier value is used as the table Creator ID for these unqualified table names. If this value is not set correctly, then Optimization Expert cannot explain these statements and they are ignored by the various Advisors and tools. You can also use **Browse** to locate the input file on the workstation.

**Note:** If you have SQL statements in your input file that use unqualified table names, this is the only point where you can specify the Creator ID. If you get it wrong here, you cannot correct the error later when you process the SQL statements. This is only an issue for workloads derived from a file. When a workload is extracted from the DB2 catalog or from the dynamic statement cache, the appropriate Creator ID is picked up from the plan, package information, or cache.

In our input file, all SQL statements use unqualified table names, so we have specified the Creator ID of the tables in the Qualifier field. We have also used **Browse** to select the file on our workstation (c:\Mike\tpcdsql1.txt). The statements are shown in the scrollable box below the file name, as shown in Figure 11-6 on page 288.

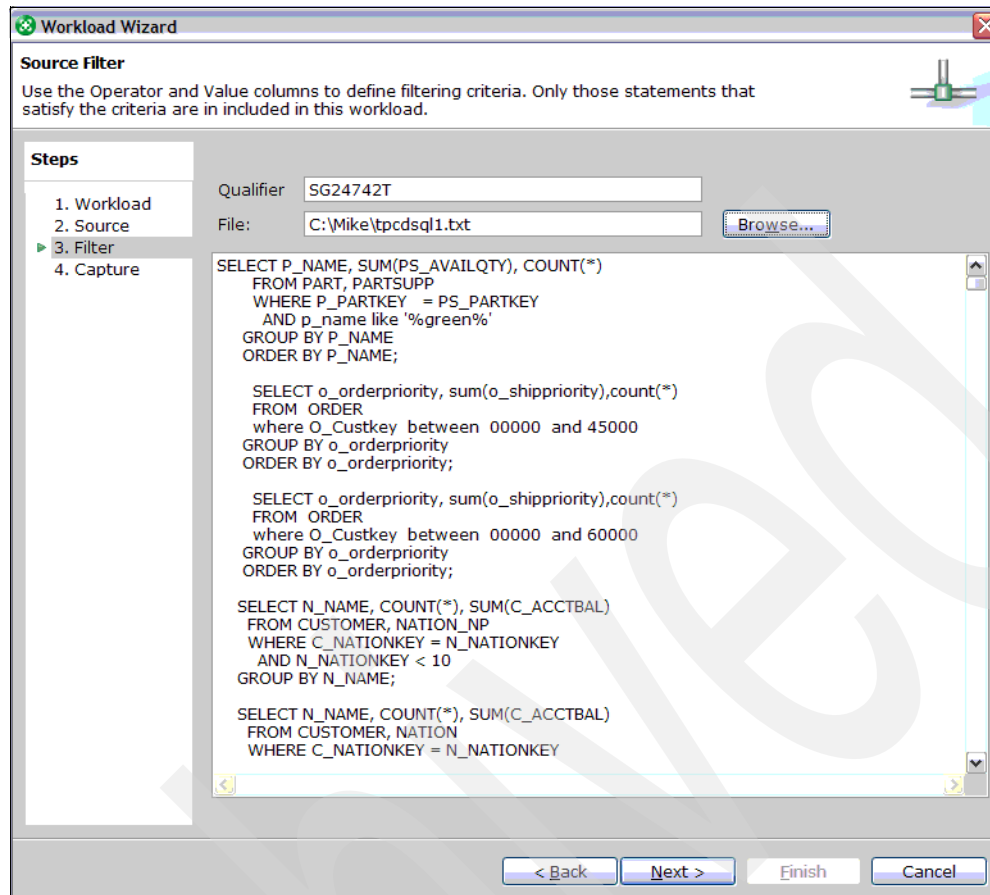


Figure 11-6 Specifying the input file and the table Creator ID

When you click **Next**, you are asked when to capture the workload. The only option for a file workload is “Immediately”, as shown in Figure 11-7 on page 289.



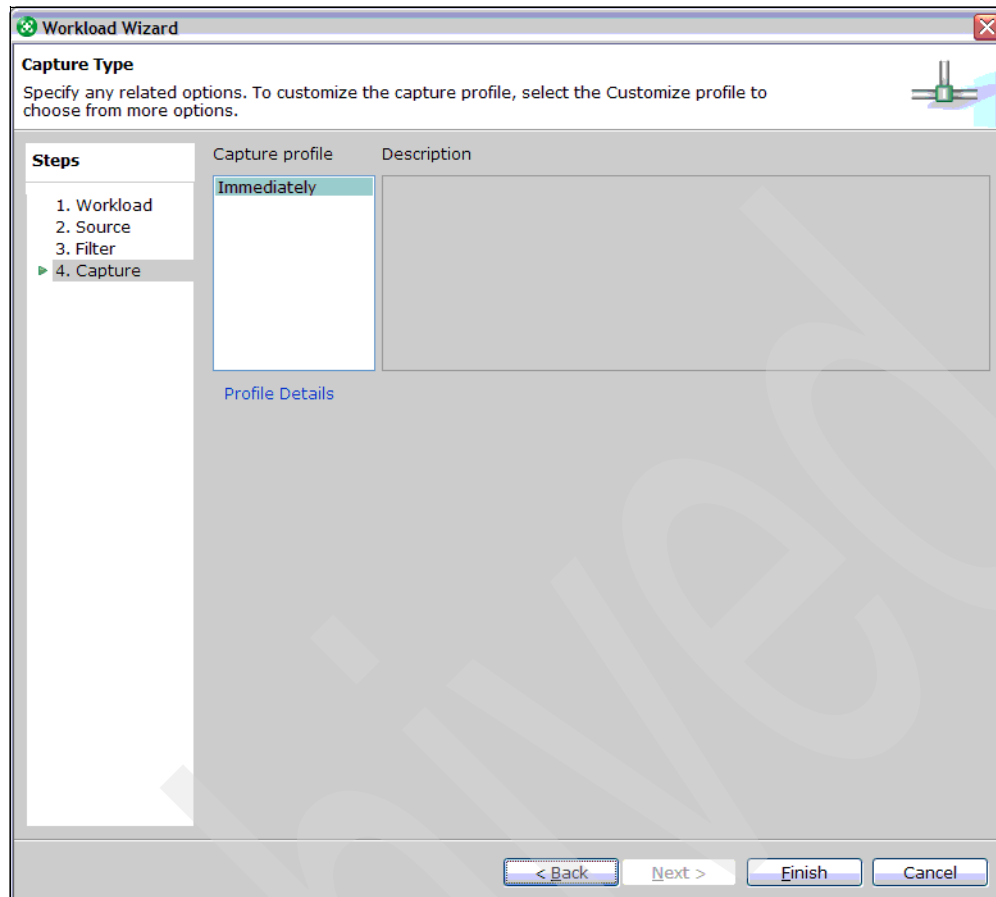


Figure 11-7 Capture workload immediately

When you click **Finish**, Optimization Expert creates the workload and imports the file contents. You see a progress indicator while this process is running as shown in Figure 11-8 on page 290.

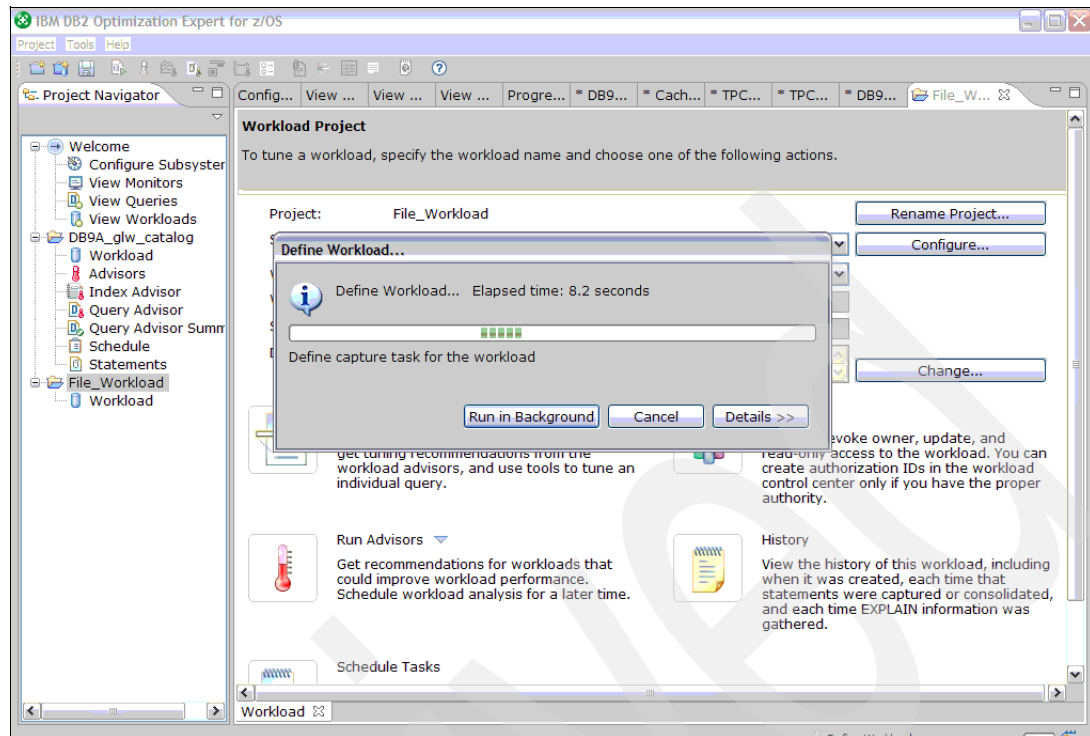


Figure 11-8 Setting up the workload

When the SQL statements have been captured (imported into the workload), you see the workload project panel showing the status of the workload as Captured (see Figure 11-9).

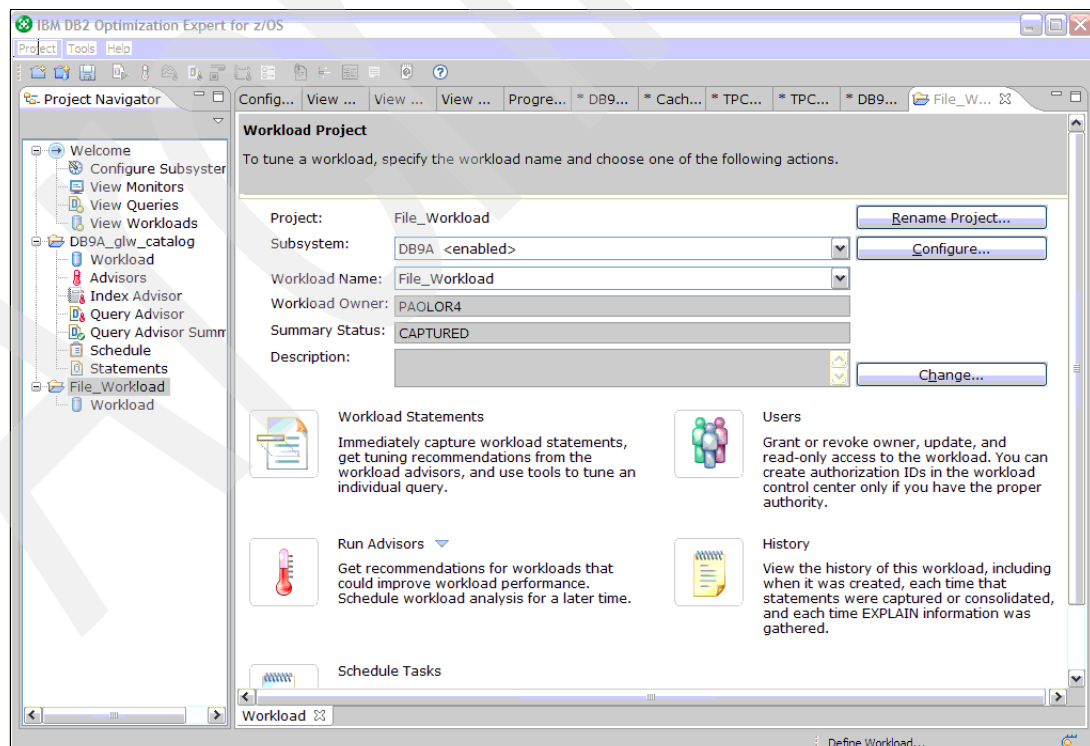


Figure 11-9 The workload is captured

If you click **Workload Statements**, you are taken to the list of statements. While the list is being built from the workload, you see a status indicator as shown in Figure 11-10.

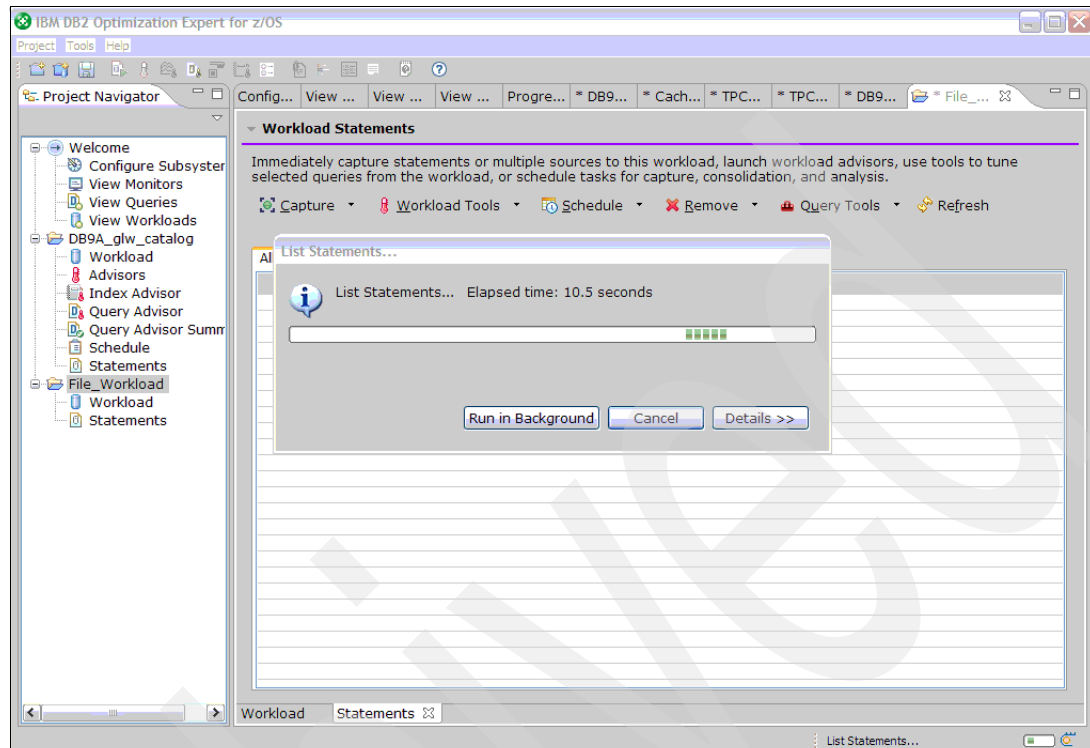


Figure 11-10 Building the statement list

When the statement list is complete, you are switched to the Statements tab as shown in Figure 11-11 on page 292.





The Access Path Advisor automatically creates a new Query Project for the single statement you selected. When it has explained the statement, it switches to this new project with the Access Path Advisor tab selected, as shown in Figure 11-14.

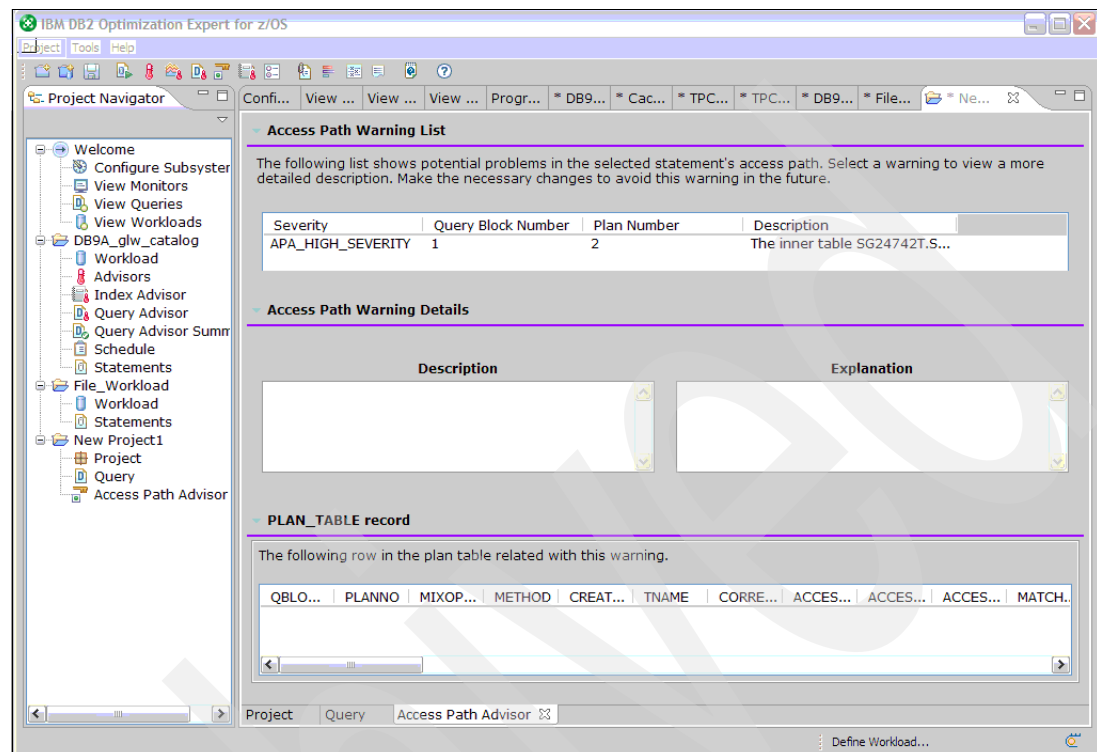


Figure 11-14 Access Path Advisor warnings

The new Query Project is named New Project1. You can rename this by selecting the **Project** tab and clicking **Rename Project**. We will leave the name as it is. If you select the **Query** tab, you can see the SQL statement text, as shown in Figure 11-15 on page 295.



Figure 11-15 The query text

## 11.2.2 Access Path Advisor recommendations

Lets switch back to the Access Path Advisor tab and look at the warning that was generated. There was one warning with a high severity level. If you select that warning in the top box, you can see more detail in the other boxes, as shown in Figure 11-16 on page 296.

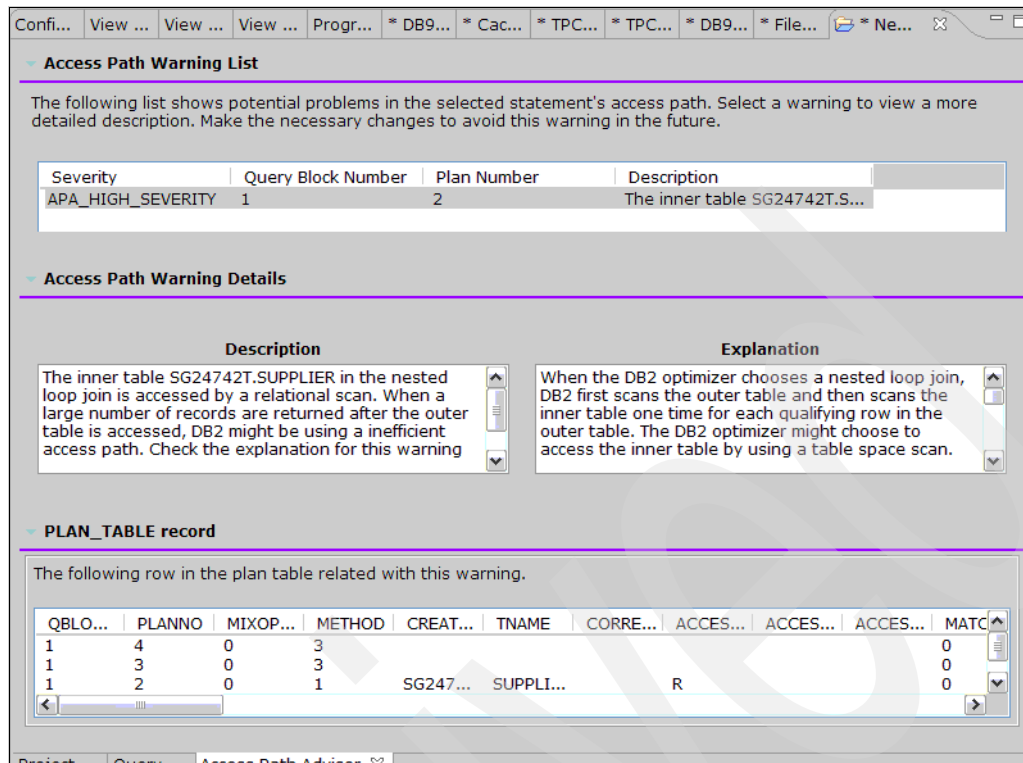


Figure 11-16 The Access Path Advisor warning detail

In the middle of the panel, the left hand box contains a description of the warning and the right hand box contains an explanation of the problem type. The box at the bottom of the panel shows the Plan Table rows that describe the access path for the statement.

Figure 11-7 on page 289 shows the SQL statement for reference.

```
SELECT C_MKTSEGMENT,SUM(C_ACCTBAL * 0.01),MAX(S_ACCTBAL) FROM CUSTOMER,
SUPPLIER
WHERE C_ACCTBAL = S_ACCTBAL AND C_ACCTBAL < 9000.00
AND C_MKTSEGMENT IN ('MACHINERY','FURNITURE','AUTOMOBILE')
GROUP BY C_MKTSEGMENT
ORDER BY 2
```

Figure 11-17 The SQL statement text

In our example, the full text in the Description box reads as shown in Figure 11-18.

The inner table SG24742T.SUPPLIER in the nested loop join is accessed by a relational scan. When a large number of records are returned after the outer table is accessed, DB2 might be using an inefficient access path. Check the explanation for this warning for more details about the possible cause and solution.

Figure 11-18 The Description box text



Figure 11-19 shows the the Explanation box.

When the DB2 optimizer chooses a nested loop join, DB2 first scans the outer table and then scans the inner table one time for each qualifying row in the outer table. The DB2 optimizer might choose to access the inner table by using a table space scan. However, a table space scan is generally not the best access path if the number of qualifying rows from the outer table is large, or if the inner table is large. This problem is usually caused by missing or out-of-date statistics in the catalog. Run the OSC Statistics Advisor to determine if any statistics need to be collected.

For more information about nested loop joins, see the "Monitoring and tuning DB2 performance" information in the Information Management Software for z/OS Solutions Information Center at <http://publib.boulder.ibm.com/infocenter/imzic>.

*Figure 11-19 The Explanation box text*

The Advisor has detected that the inner table of a nested loop join is accessed with a table space scan, which is a problem.

At present, the Access Path Advisor only checks for two things:

- ▶ For a Merge Scan join, the number of columns that DB2 will match during the join (shown in the PLAN\_TABLE column MERGE\_JOIN\_COLS) is less than the number coded in the statement join predicates.
- ▶ For a Nested Loop join, the access to the inner table uses a table space scan.

### 11.2.3 Acting on the recommendations

The warnings generated by the Access Path Advisor are based solely on the results of the Explain statement. While it generates warnings, it offers no specific recommendations as to how to resolve the problem. In most cases, the appropriate response is to run the Statistics Advisor and the Index Advisor and to act upon their recommendations. The Query Advisor might also help if the other two Advisors do not resolve the problem.

In our case, we choose (within the project created for this single SQL statement) to run all the Advisors, as shown in Figure 11-20 on page 298.

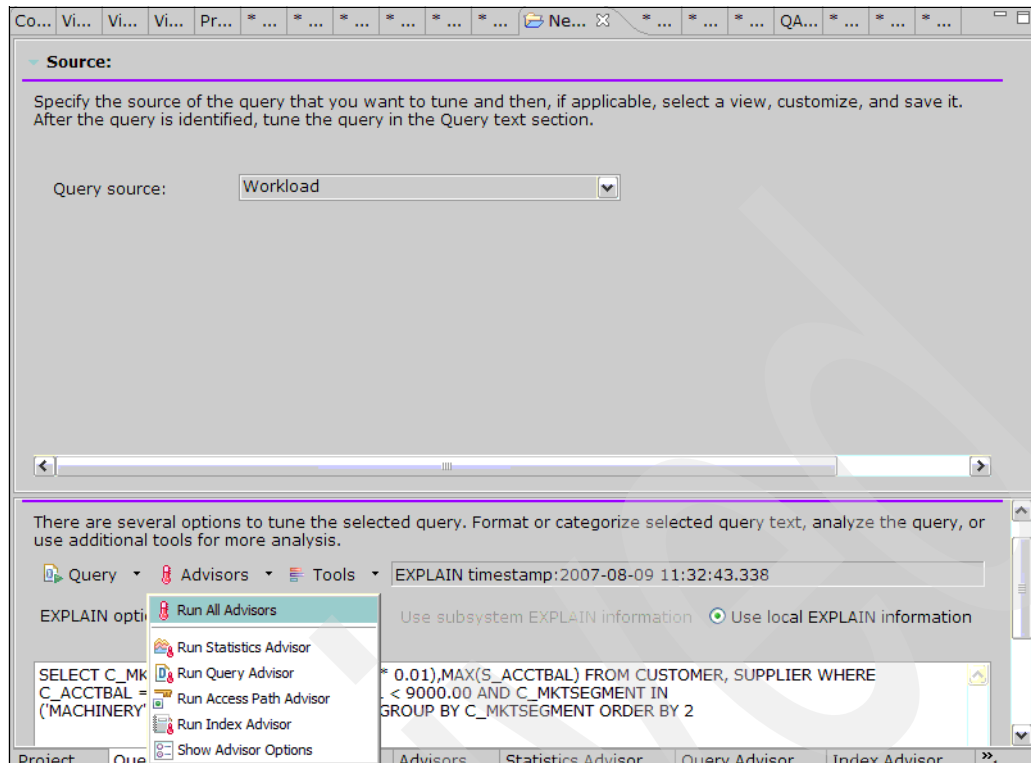


Figure 11-20 Run all the Query Advisors

On completion, you see the Advisors tab, as shown in Figure 11-21.

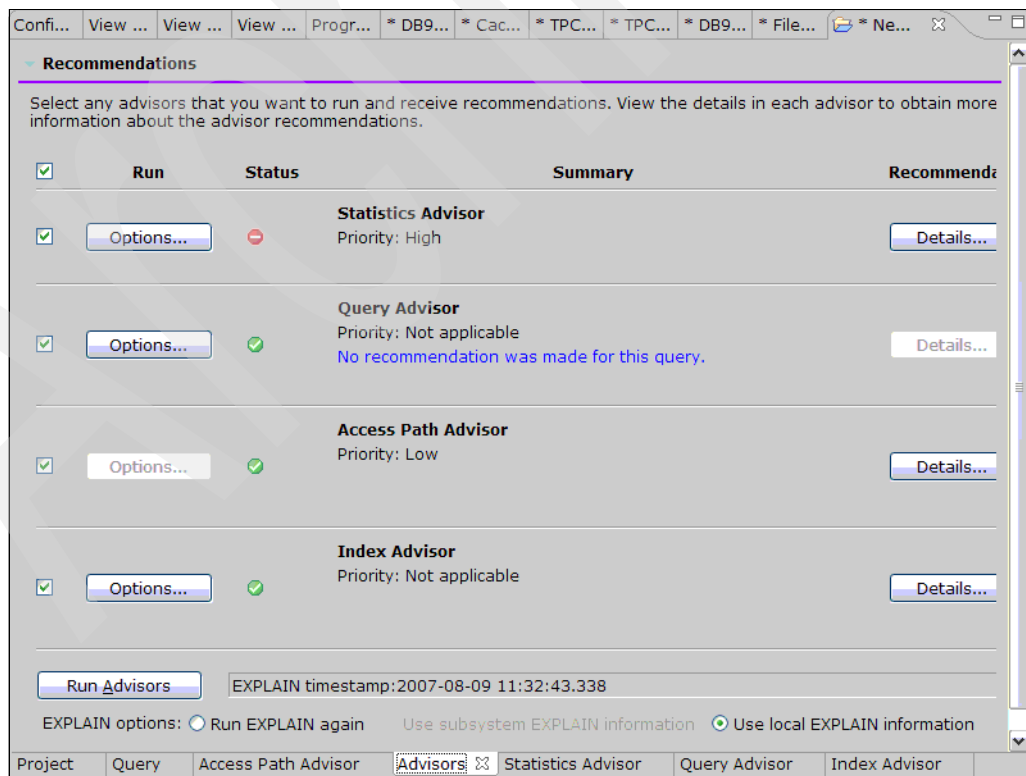


Figure 11-21 Result of running all Advisors

We can see that the Statistics Advisor has some high priority recommendations. The Query Advisor has made no recommendations. Check the Index Advisor. We already know what the Access Path Advisor has to say. Look at the Statistics Advisor recommendations by selecting the **Statistics Advisor** tab, as shown Figure 11-22.

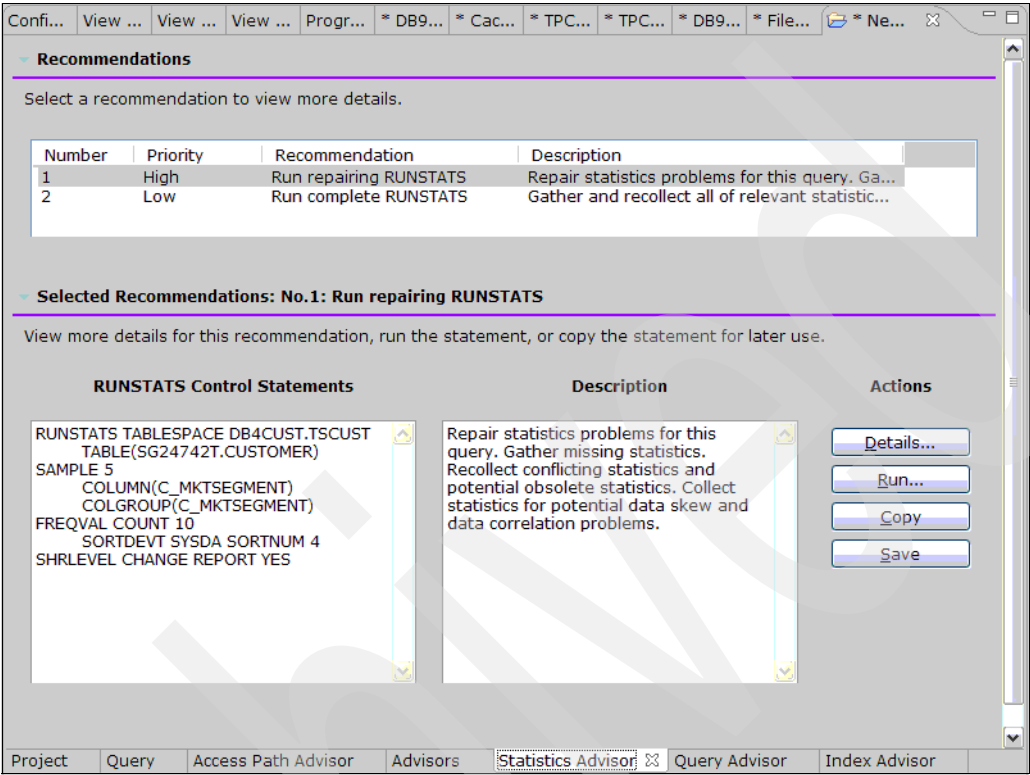


Figure 11-22 The Statistics Advisor recommendations

You can see the Index Advisor recommendations by selecting the **Index Advisor** tab as shown in Figure 11-23 on page 300.

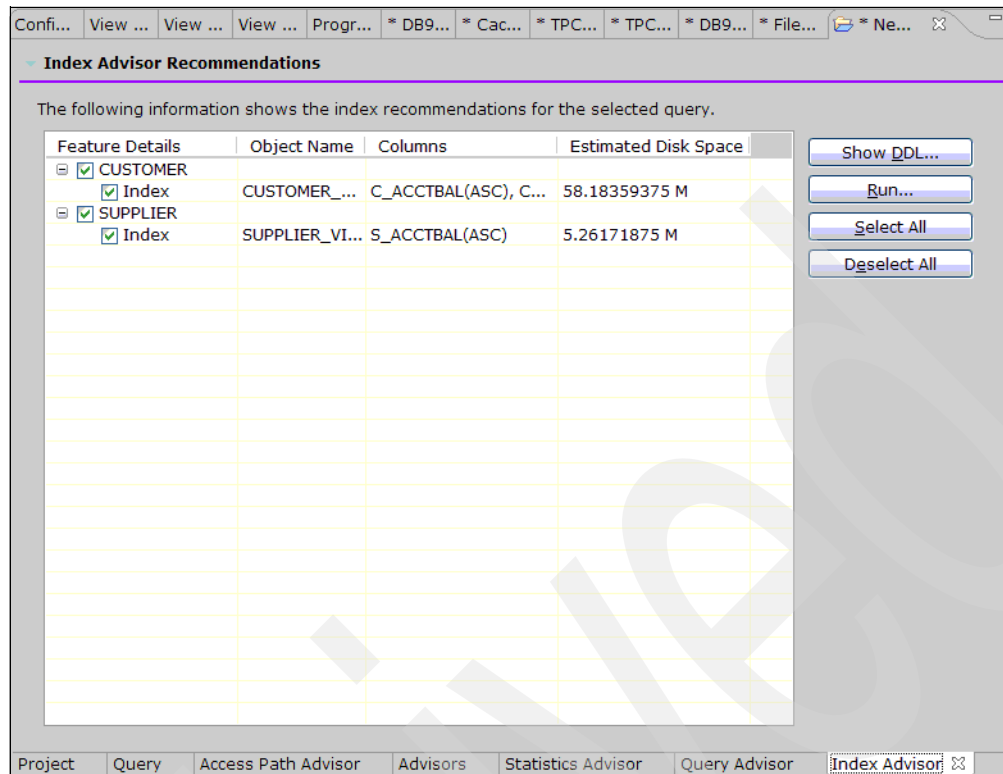


Figure 11-23 The Index Advisor recommendations

## 11.2.4 Setting Access Path Advisor options

The user cannot change the options for the Access Path Advisor.

## 11.3 Query Advisor

The Query Advisor, which is part of Optimization Expert for z/OS, operates in a single SQL statement or a workload comprising many statements and makes recommendations for improving performance by rewriting the SQL statement. This section looks at the Query Advisor operating on a single SQL statement. The next section looks at the Workload Query Advisor.

### 11.3.1 Running the Query Advisor

Lets set up a Query Project, named QA\_Single\_Query. We provide a SQL statement by typing in the text, as shown in Figure 11-24 on page 301. The text of the SQL statement is shown in Figure 11-25 on page 301.

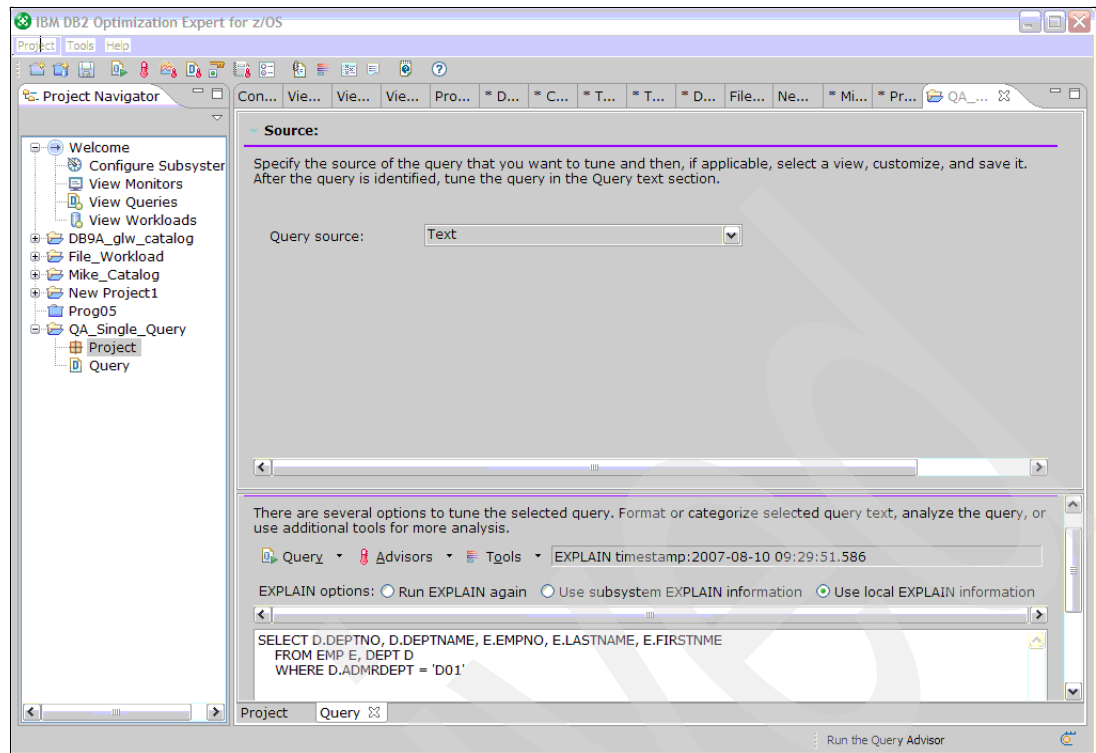


Figure 11-24 Typing the SQL statement to be analyzed

```
SELECT D.DEPTNO, D.DEPTNAME, E.EMPNO, E.LASTNAME, E.FIRSTNAME
FROM EMP E, DEPT D
WHERE D.ADMRDEPT = 'D01'
```

Figure 11-25 The SQL statement text

**Note:** The SQL statement uses unqualified table names. They will be qualified with the SQL ID that you specified when you connected to the DB2 subsystem.

To run the Query Advisor, select the **Run Query Advisor** option from the **Advisors** drop-down menu as shown in Figure 11-26 on page 302.

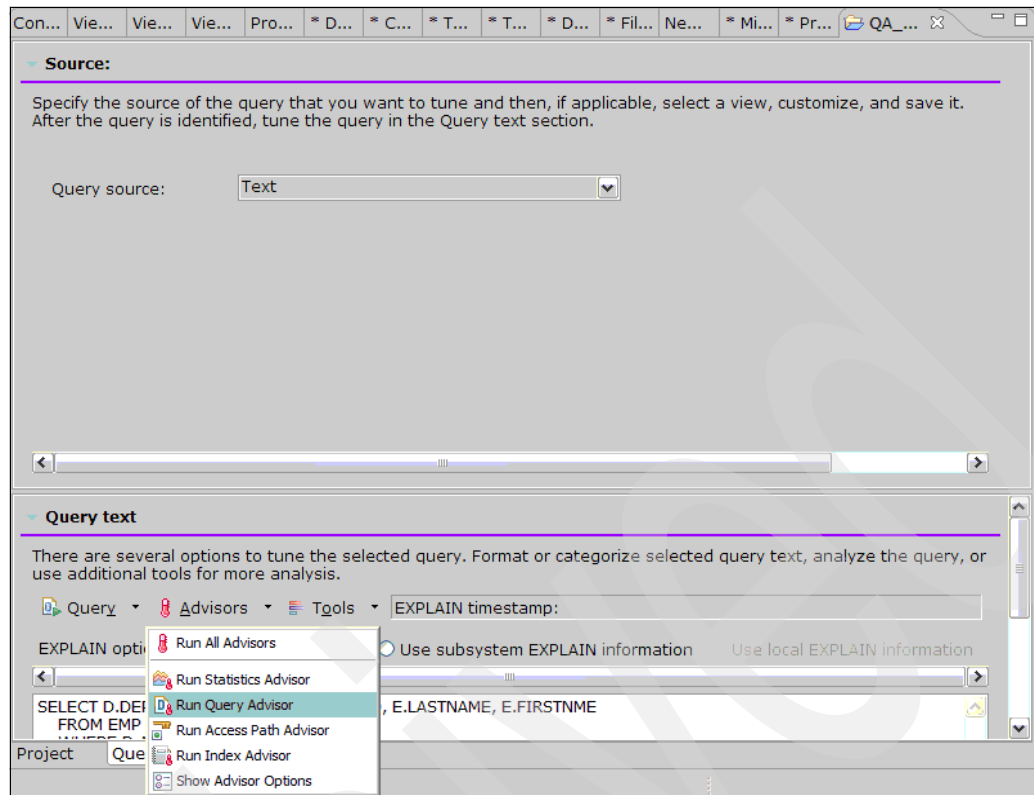


Figure 11-26 Run Query Advisor

### 11.3.2 Query Advisor recommendations

When the Advisor completes, a new Query Advisor tab is opened in the project as shown in Figure 11-27 on page 303. You can see the recommendation in the top box and the SQL text in the middle box.

If you select the recommendation in the top box, additional information becomes available in the bottom two boxes as shown in Figure 11-28 on page 303. We used the main scroll bar at the right of the panel to scroll down so that we can see more of the bottom boxes.

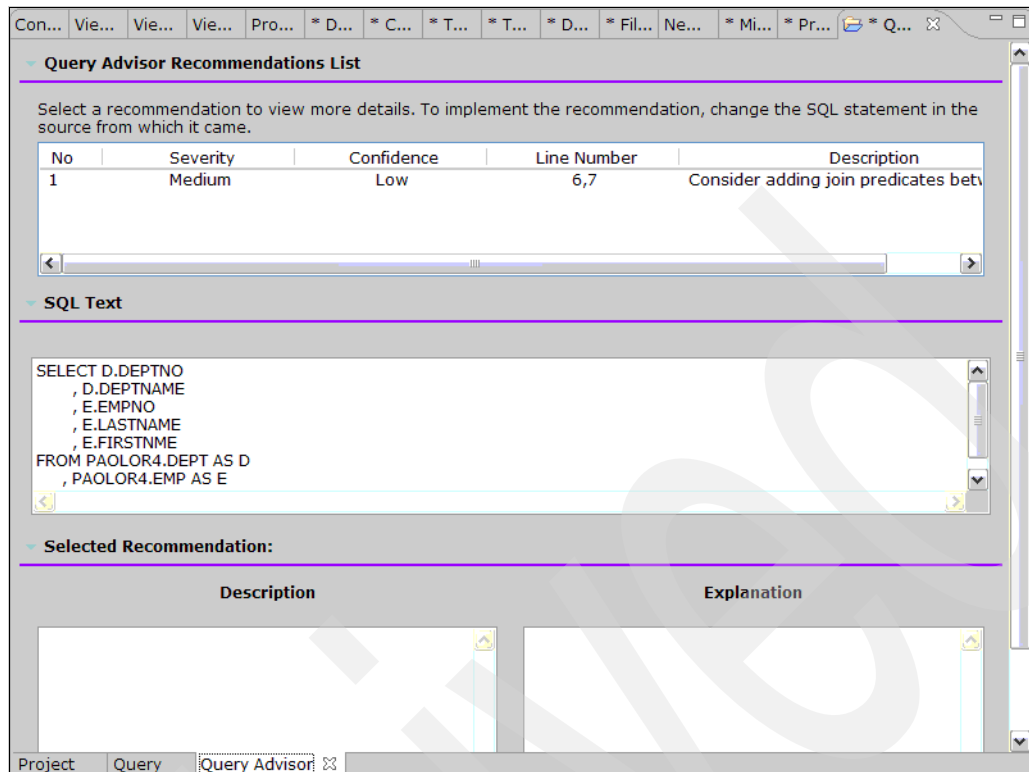


Figure 11-27 Query Advisor recommendations

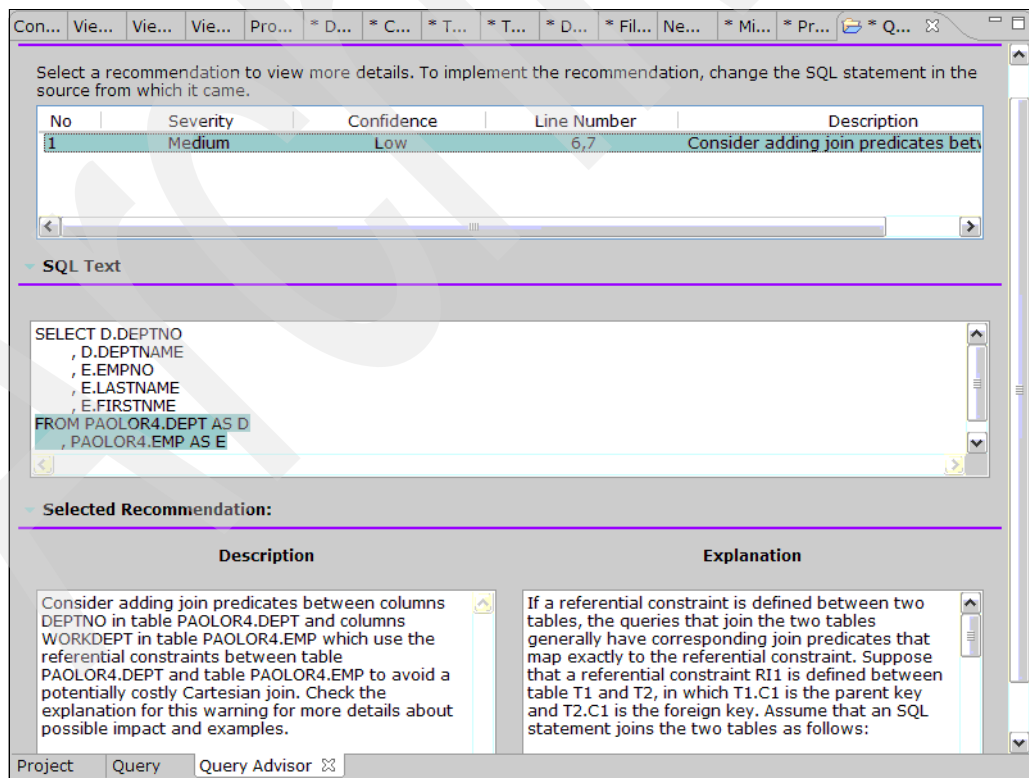


Figure 11-28 Seeing the detailed recommendation

The line or lines in the SQL text to which the recommendation applies is highlighted in the middle box. The full text of the Description (bottom left box) is shown in Figure 11-29.

Consider adding join predicates between columns DEPTNO in table PAOLOR4.DEPT and columns WORKDEPT in table PAOLOR4.EMP which use the referential constraints between table PAOLOR4.DEPT and table PAOLOR4.EMP to avoid a potentially costly Cartesian join. Check the explanation for this warning for more details about possible impact and examples.

*Figure 11-29 Recommendation description*

The full text of the Explanation is shown in the bottom right box (Figure 11-30). You can scroll using the small scroll bar in the box to read it all.

If a referential constraint is defined between two tables, the queries that join the two tables generally have corresponding join predicates that map exactly to the referential constraint. Suppose that a referential constraint RI1 is defined between table T1 and T2, in which T1.C1 is the parent key and T2.C1 is the foreign key. Assume that an SQL statement joins the two tables as follows:

```
SELECT T1.C2, T2.C2
FROM T1, T2
WHERE T1.C2 = :charHV3
AND T2.C2 = :charHV4
```

As written, the SQL statement does not include a join predicate that represents the referential constraint between the two tables. The result is likely to be a large number of meaningless rows. Therefore, consider rewriting the SQL statement as follows:

```
SELECT T1.C2, T2.C2
FROM T1, T2
WHERE T1.C2 = :charHV3
AND T2.C2 = :charHV4
AND T1.C1 = T2.C1
```

With the added join predicate, the rewritten SQL statement can filter out the meaningless rows and might provide more efficient access to the two tables. For such SQL statements, adding predicates that join the columns in the referential constraint can dramatically improve the performance, unless the "meaningless" rows actually need to be returned.

*Figure 11-30 Recommendation explanation*

The Query Advisor has identified that no join predicates were specified for the inner join of the EMP and DEPT tables. See Figure 11-25 on page 301 for the statement text.

**Note:** Currently, the Query Advisor only detects the missing join predicates when a foreign key is defined in DB2. In our example, a foreign key was defined on the EMP column WORKDEPT pointing to the DEPT table. When the same test was run without the foreign key defined, the Query Advisor did not detect the missing join predicates.



**Note:** Be *careful* if two tables have two RI constraints (foreign keys) defined between them. The Query Advisor might recommend join predicates that you cannot use. Consider the IBM sample tables EMP and DEPT. Employees work for departments identified by the WORKDEPT column in the EMP table. Departments are managed by employees identified by the MGRNO column in the DEPT table.

Assume that we ask the Query Advisor to analyze the following SQL statement:

```
SELECT * FROM DEPT D LEFT OUTER JOIN EMP E ON D.DEPTNO = E.WORKDEPT
```

If the MGRNO column is defined as a foreign key pointing to the EMP table, the Query Advisor will recommend adding the join predicate D.MGRNO = E.EMPNO. If you add this predicate, it changes the result in a way you probably do not want. Use your knowledge of the application requirement to validate the Query Advisor recommendations before implementing them.

Currently, the Query Advisor checks for:

- ▶ Missing join predicates, but only if a foreign key is defined (see note above).
- ▶ Stage 2 predicates that can improve performance if rewritten as Stage 1 or indexable.
- ▶ Stage 1 predicates that can improve performance if rewritten as indexable.
- ▶ Additional local predicates that are not automatically provided by DB2 can provide Predicate Transitive Closure.
- ▶ Predicates that are pushed down to a nested table expression or materialized view without changing the result, and not already done automatically by DB2.
- ▶ Additional predicates added to a complex WHERE clause, containing ORs and ANDs and parentheses. This might improve performance without changing the result.
- ▶ The use of SELECT \* to replace a specific column list.

### 11.3.3 Acting on the recommendations

Review the coding of the SQL statement to see if it needs to be rewritten, following the specific recommendations, to improve performance.

In our example, we add the join predicate as shown in Figure 11-31.

```
SELECT D.DEPTNO, D.DEPTNAME, E.EMPNO, E.LASTNAME, E.FIRSTNAME  
FROM EMP E, DEPT D  
WHERE D.ADMRDEPT = 'D01'  
AND D.DEPTNO = E.WORKDEPT
```

Figure 11-31 The revised SQL statement

Lets create a new Query Project, named QA\_Single\_Query\_2, for the revised SQL statement, as shown in Figure 11-32 on page 306.

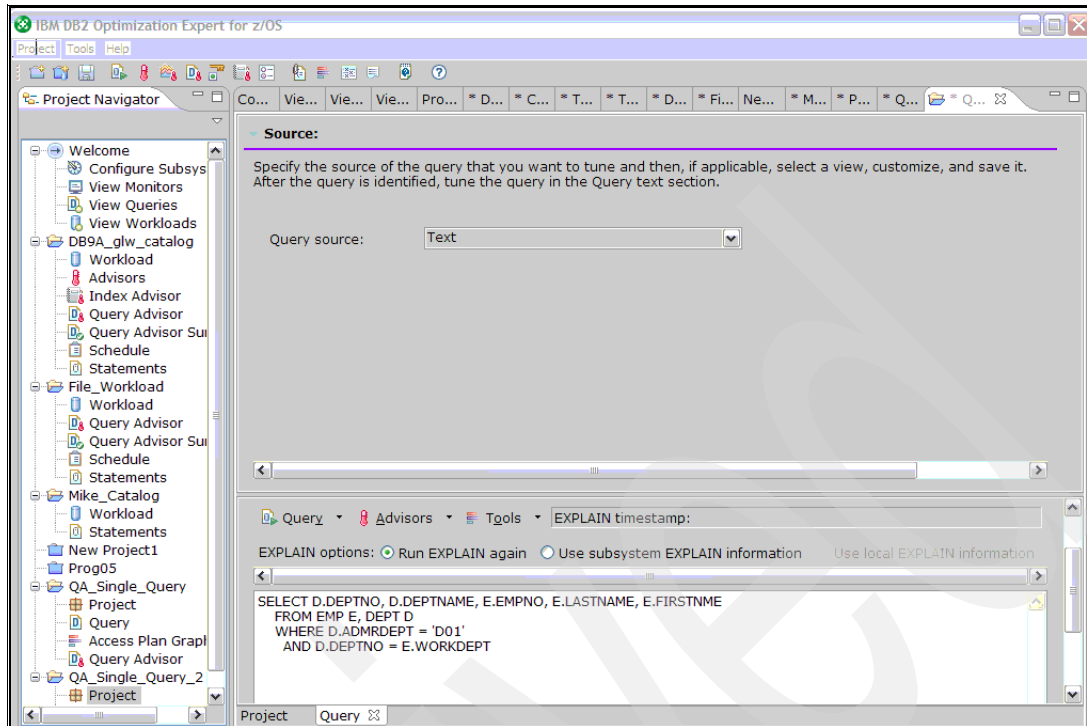


Figure 11-32 A new project for the revised query

Now, when you run the Query Advisor, no recommendation is provided, as shown in Figure 11-33.

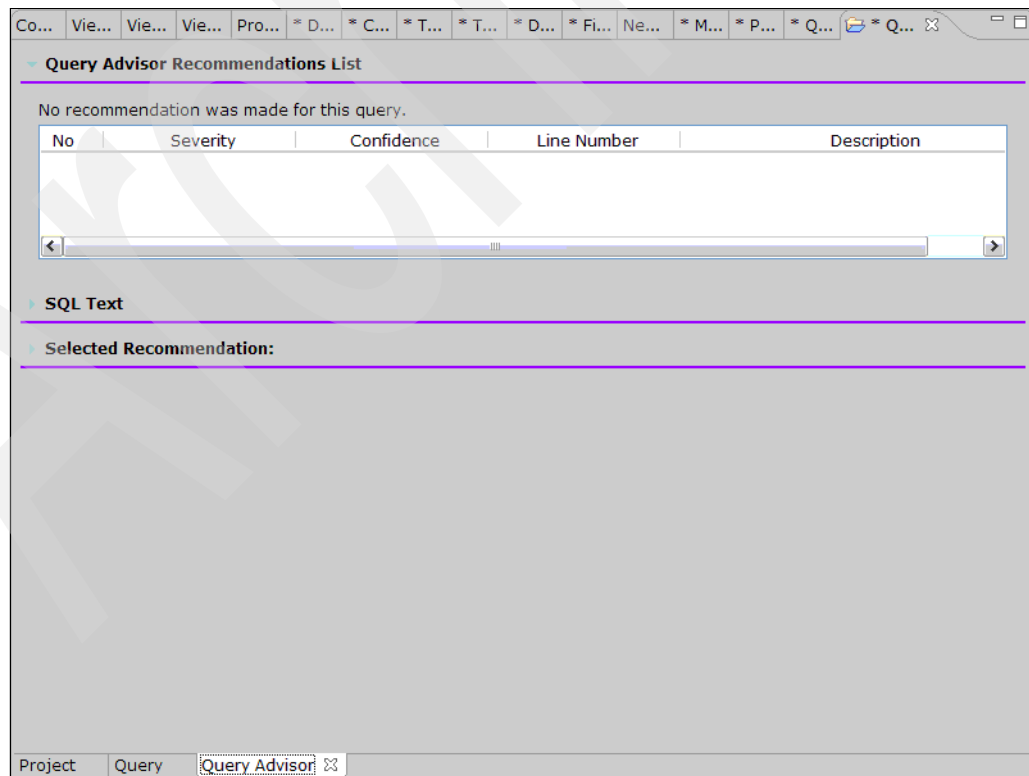


Figure 11-33 No recommendation for the revised statement

### 11.3.4 Setting Query Advisor options

You can access the Preferences panel for the Query Advisor from the main menu, Tools Preferences, to set the defaults for the new projects, as shown in Figure 11-34. You can set the same preferences for an existing project from the Advisors tab for the project.

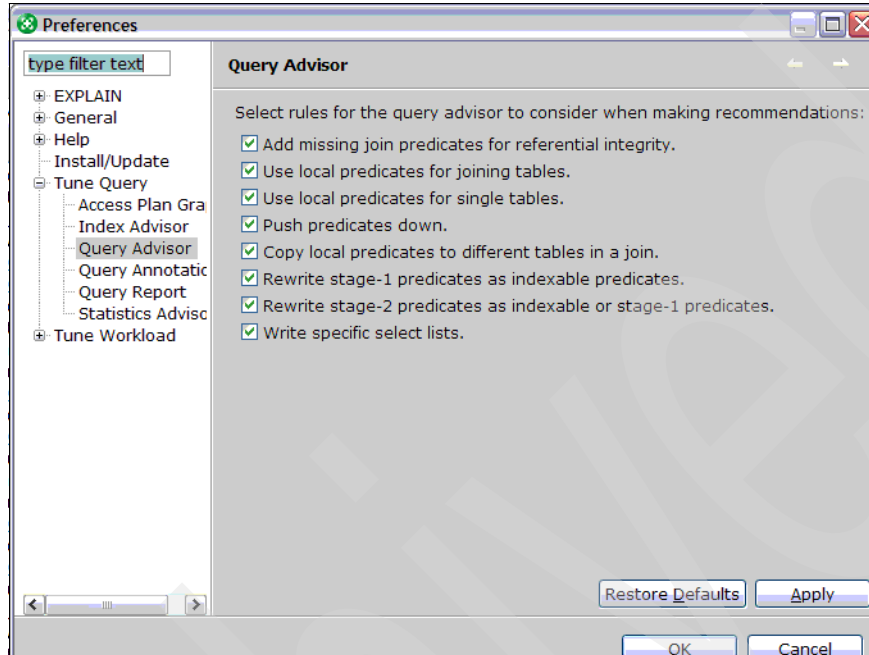


Figure 11-34 Query Advisor preferences panel

In the Preferences panel, you can select the problems that the Query Workload Advisor needs to check.

## 11.4 Workload Query Advisor

This section looks at the Workload Query Advisor. We use the workload that we created from a file in section 11.1, “Creating a workload from a file” on page 284.

### 11.4.1 Running the Workload Query Advisor

Switch to our File\_Workload project and select the **Statements** tab. Select the **Run Workload Query Advisor** option from the Workload Tools drop down menu, as shown in Figure 11-35 on page 308.

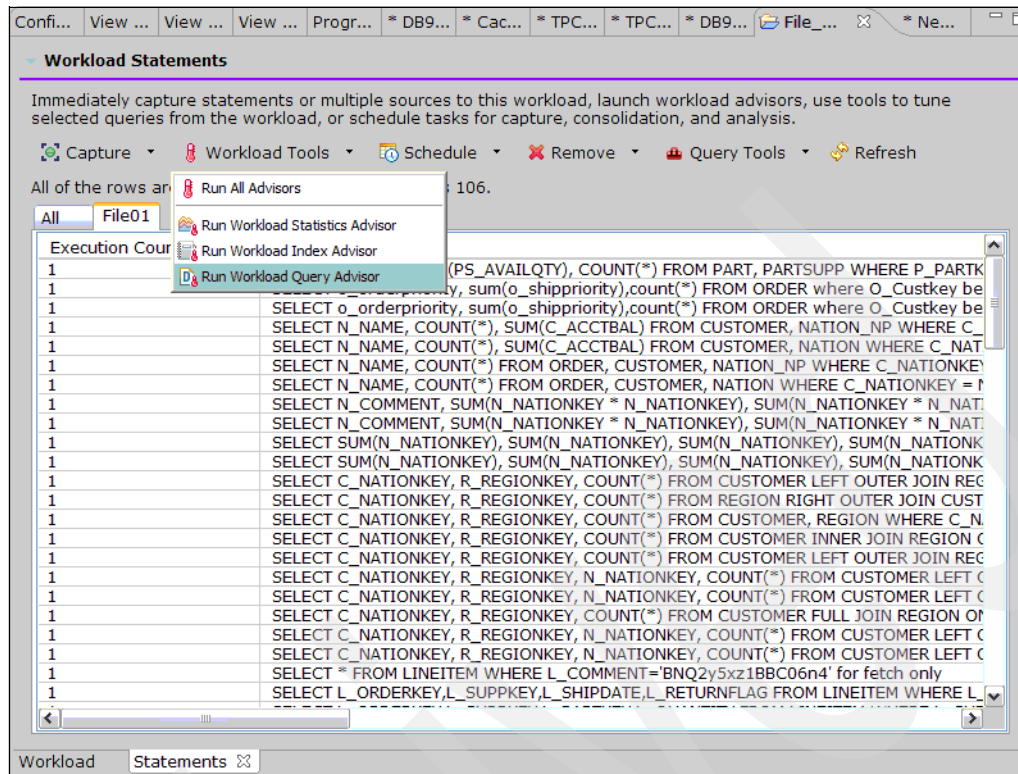


Figure 11-35 Run the Workload Query Advisor

Because you have not explained these statements yet, you are prompted to do so (see Figure 11-36).

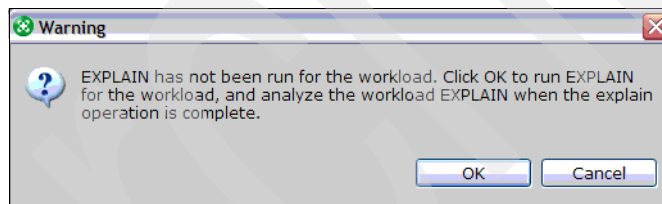


Figure 11-36 Explain is needed

Click **OK** and Optimization Expert prepares to run an Explain task in the background. You see the Schedule Explain panel, as shown in Figure 11-37 on page 309.

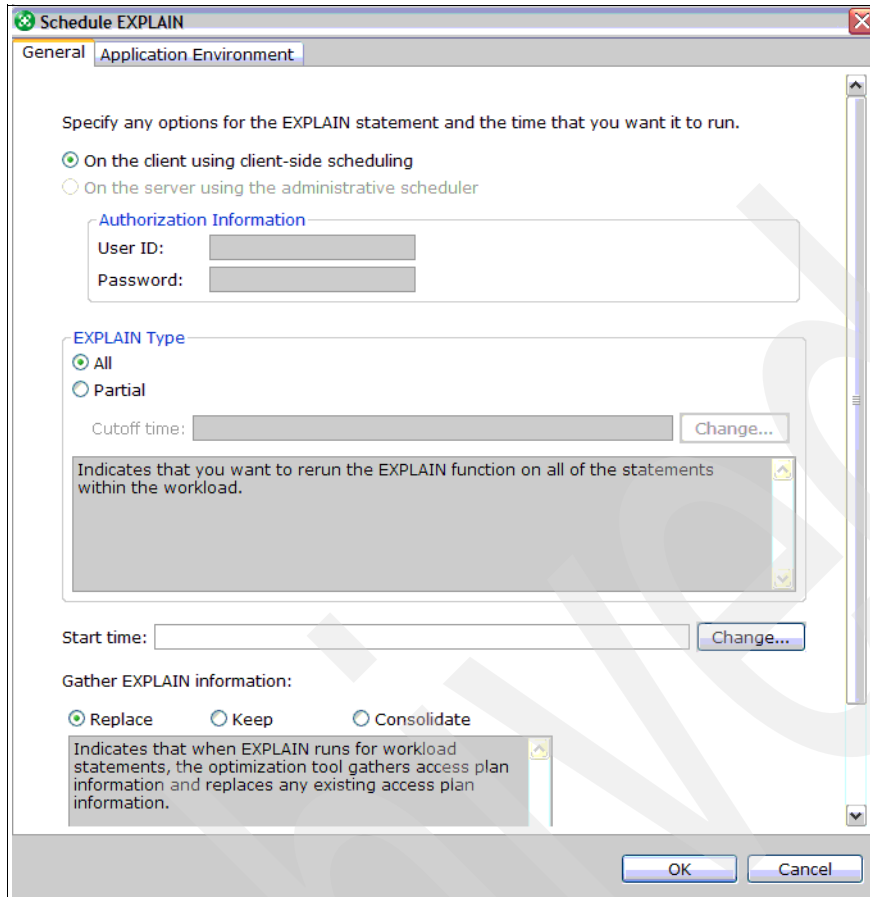


Figure 11-37 Schedule Explain panel

Note that there is a second tab at the top. Figure 11-38 on page 310 shows this second tab.

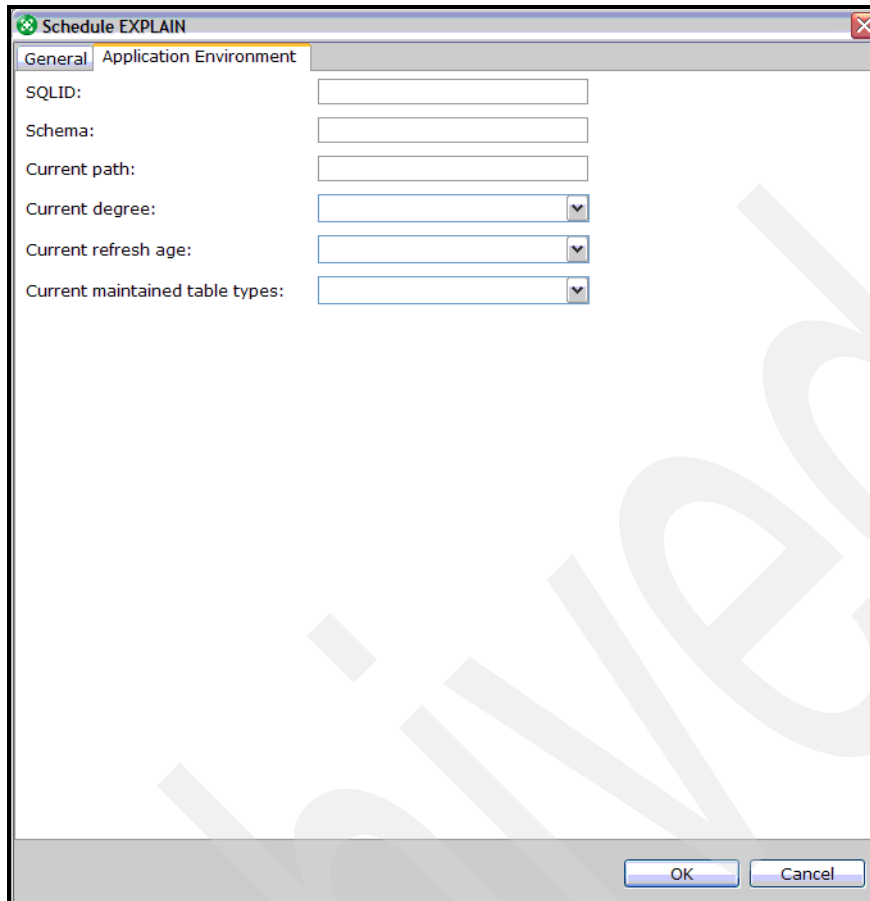


Figure 11-38 Schedule Explain second tab

You can see that the option of specifying SQLID or Schema on this tab. This looks like a way to handle the unqualified table name issue we discussed in section 11.1, “Creating a workload from a file” on page 284. Unfortunately, it does not work and so you need to specify the Qualifier when you capture the workload. If you take the defaults and click **OK**, the Explain task runs in the background. You can check the status of the Explain task by switching to the **Workload** tab and clicking **Schedule Tasks** (see Figure 11-39 on page 311).

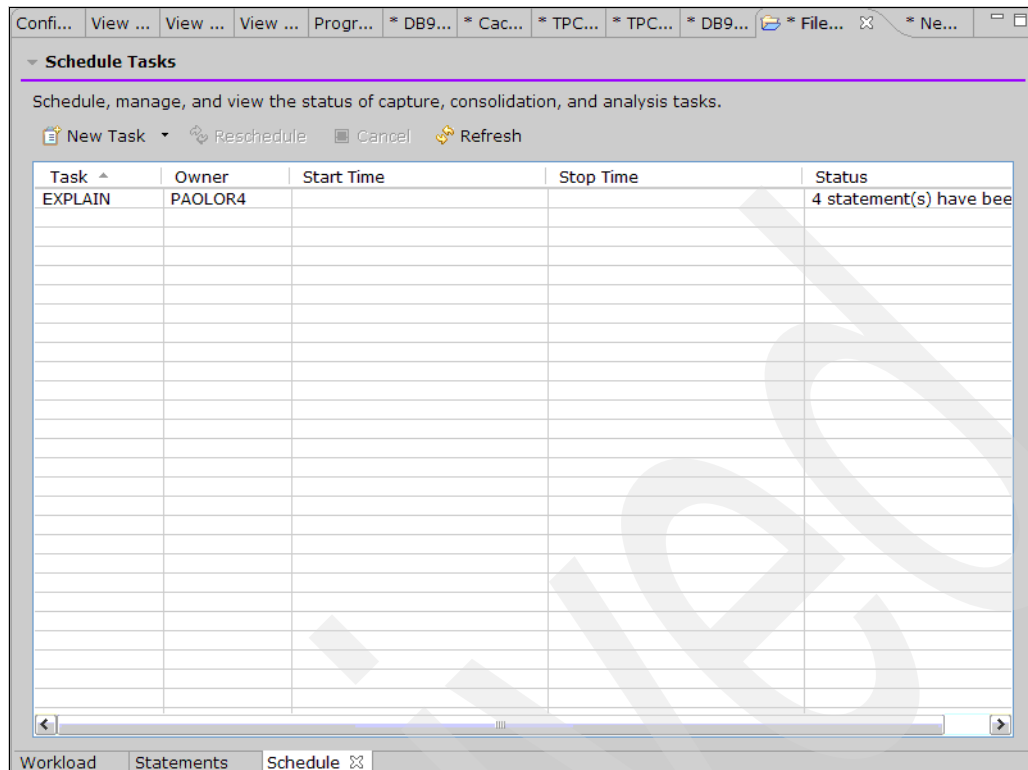


Figure 11-39 Checking the Explain task

When the Explain task completes, you can retry running the Workload Query Advisor. While it runs, a status indicator is displayed as shown in Figure 11-40.

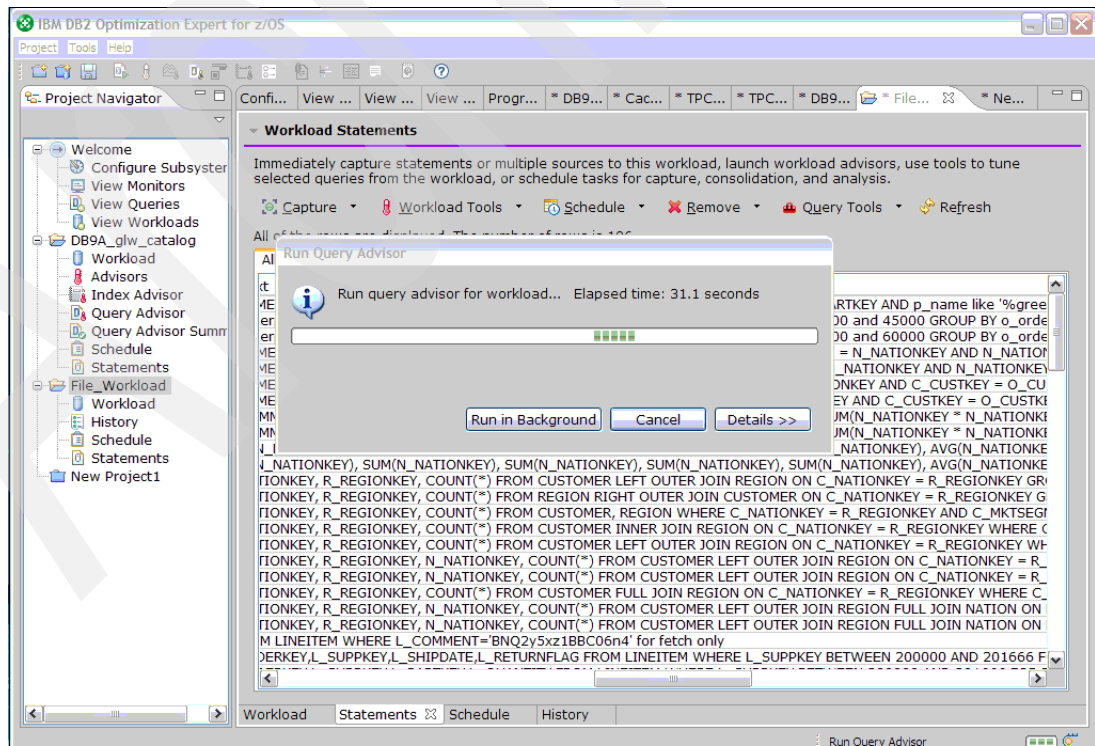


Figure 11-40 Workload Query Advisor is running

## 11.4.2 Workload Query Advisor recommendations

When the Workload Query Advisor ends, the results are displayed in a new Query Advisor Summary tab as shown in Figure 11-41.

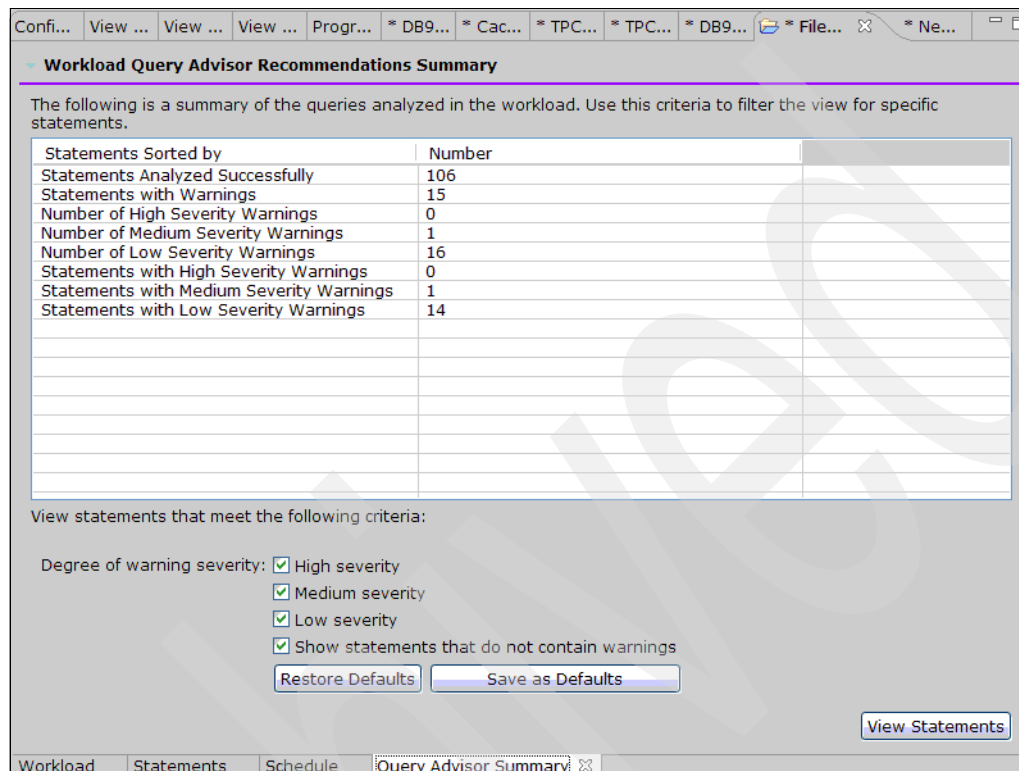


Figure 11-41 Workload Query Advisor recommendations summary

You can see that of the 106 statements in the workload, 15 have warnings. There is one medium severity warning and 16 low severity warnings. Note that a single SQL statement can have more than one warning. We are only interested in the statements with warnings, so unselect the Show statements that do not contain warnings box near the bottom of the window. Then click **View Statements** at the bottom right. You are taken to a new Query Advisor tab, as shown in Figure 11-42 on page 313.





Look at the first statement, which has the only medium severity warning. You can either select the statement and click **Details** (above the statement list), which only becomes active when a statement is selected, or you can double-click the statement. Optimization Expert creates another sub-tab at the top of the panel within the main Query Advisor tab, as shown in Figure 11-44.

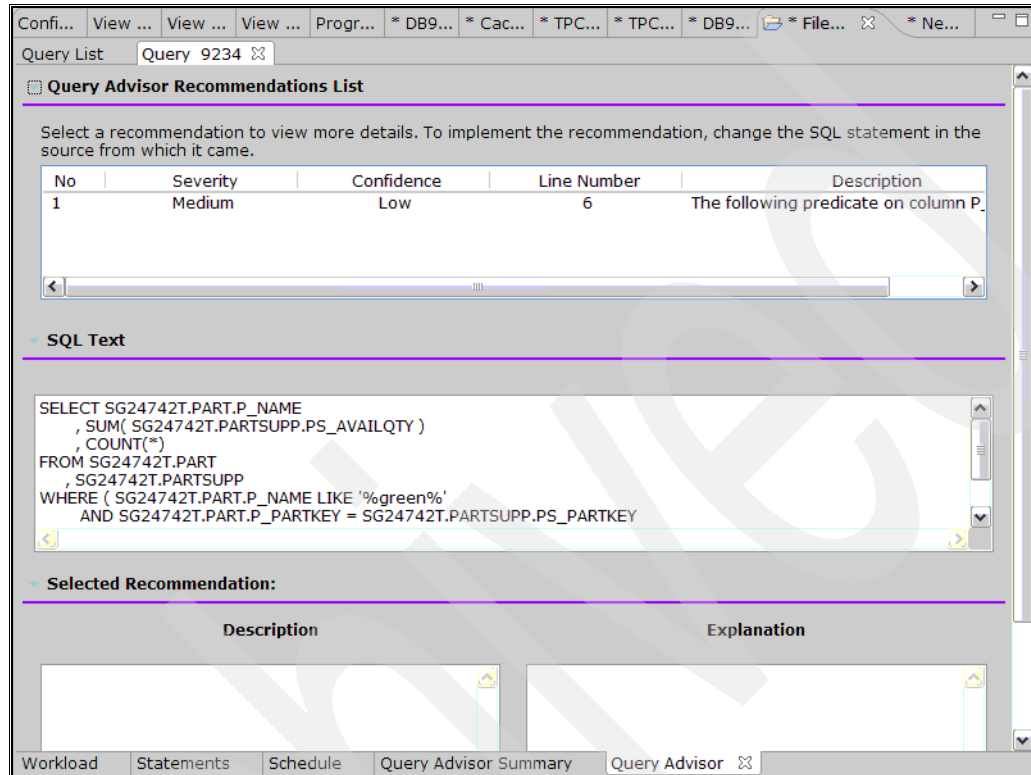


Figure 11-44 Individual statement recommendation

The sub-tab is named with a query number that Optimization Expert has assigned to the statement. You can see the recommendation in the top box and the SQL text in the middle box.

If you select the recommendation in the top box, additional information becomes available in the bottom two boxes, as shown in Figure 11-45 on page 315. We use the main scroll bar at the right of the panel to scroll down so that you can see more of the bottom boxes.

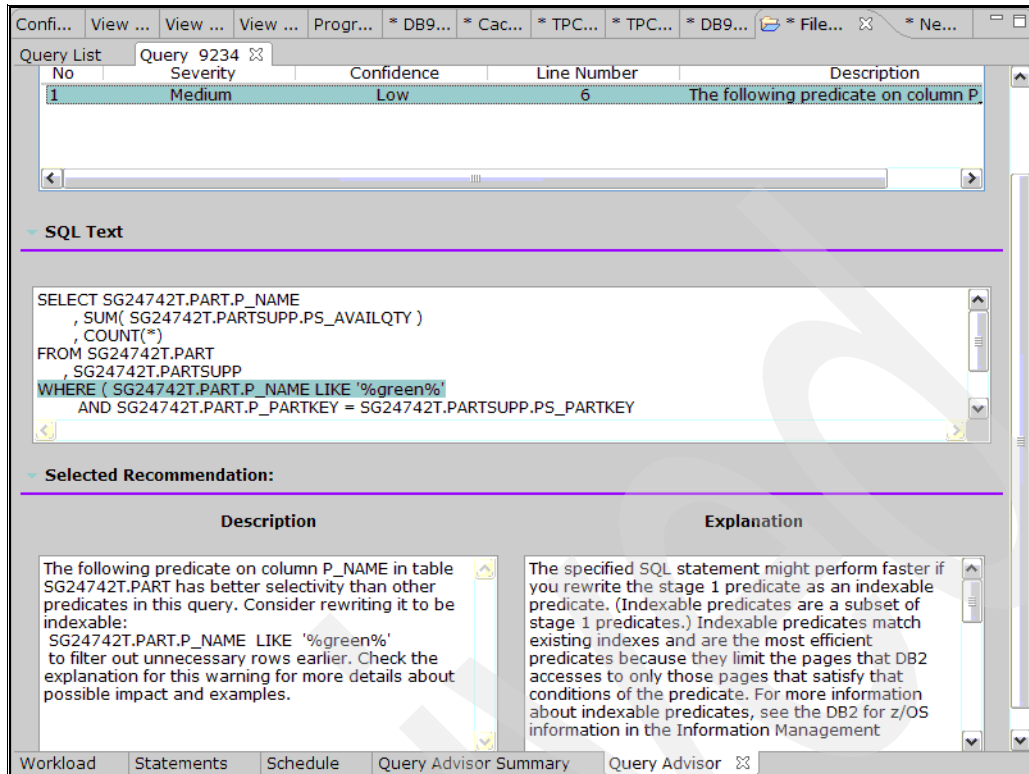


Figure 11-45 Recommendation details

The line or lines in the SQL text to which the recommendation applies is highlighted in the middle box. The full text of the Description (bottom left box) is shown in Figure 11-46.

The following predicate on column P\_NAME in table SG24742T.PART has better selectivity than other predicates in this query. Consider rewriting it to be indexable:  
 SG24742T.PART.P\_NAME LIKE '%green%'  
 to filter out unnecessary rows earlier. Check the explanation for this warning for more details about possible impact and examples.

Figure 11-46 Recommendation description

The full text of the Explanation in the bottom right box is shown in Figure 11-47 on page 316. You need to scroll using the small scroll bar in the box to read it all.

The specified SQL statement might perform faster if you rewrite the stage 1 predicate as an indexable predicate. (Indexable predicates are a subset of stage 1 predicates.) Indexable predicates match existing indexes and are the most efficient predicates because they limit the pages that DB2 accesses to only those pages that satisfy that conditions of the predicate. For more information about indexable predicates, see the DB2 for z/OS information in the Information Management Software for z/OS Solutions Information Center at <http://publib.boulder.ibm.com/infocenter/imzic>.

(Stage 1 predicates are sometimes called sargable predicates.)

The following examples show how to rewrite a stage 1 predicate as an indexable predicate:

**Example 1:**

Suppose that the column GENDER in table EMP represents the gender of each employee and has only two possible values: 'F' and 'M'. An index is created on column GENDER.

Stage 1 predicate: EMP.GENDER <> 'F'

Indexable predicate: EMP.GENDER = 'M'

**Example 2:**

Suppose that an index exists on the column DATE with type CHAR(8) in table PROJ.

Stage 1 predicate: PROJ.DATE > '1999-01-01'

(This predicate is stage 1 because the lengths of the operands do not match.)

Indexable predicate: PROJ.DATE > CHAR('1999-01-01', 8)

*Figure 11-47 Recommendation explanation*

What the recommendation is telling us is that the predicate SG24742T.PART.P\_NAME LIKE '%green%' is not an Indexable predicate, although it is a Stage 1 predicate. The problem is that the % at the beginning of the value matches any leading characters in the column P\_NAME and prevents the use of a Matching Index Scan. It is likely that the predicate, as written, is necessary to meet the business requirement. This needs to be confirmed.

If you click the top tab named **Query List**, you return to the list of statements with warnings. Select a statement with two low severity warnings. Another sub-tab is opened, as shown in Figure 11-48 on page 317.

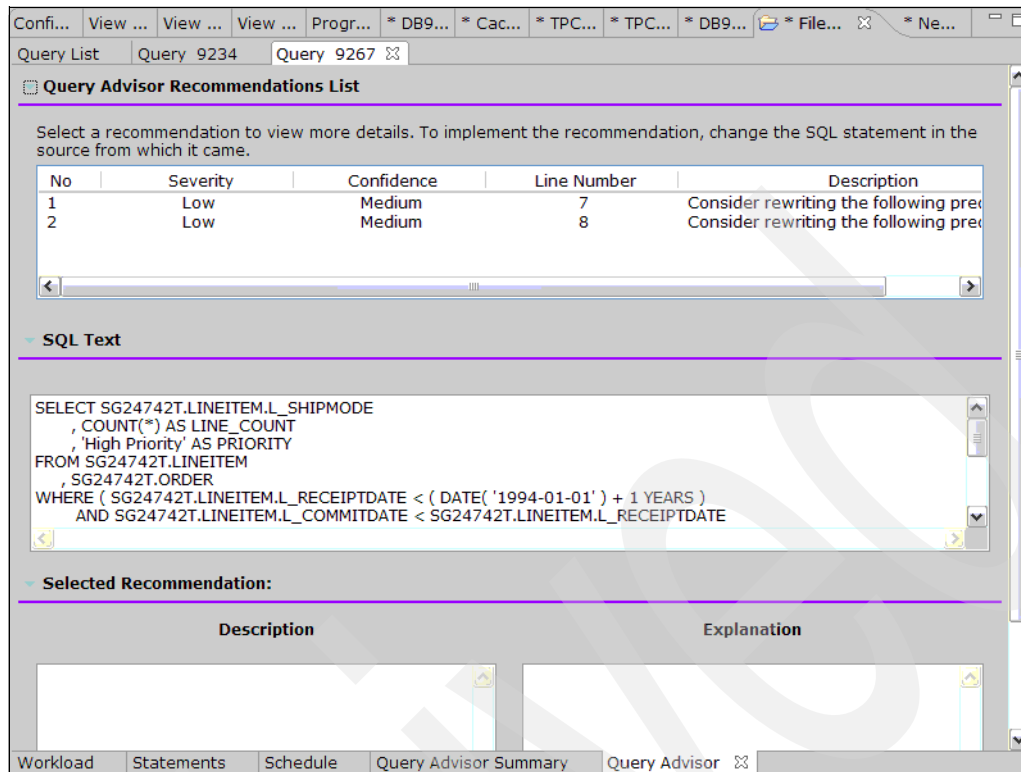


Figure 11-48 Recommendations for another SQL statement

Select the first recommendation, as shown in Figure 11-49.

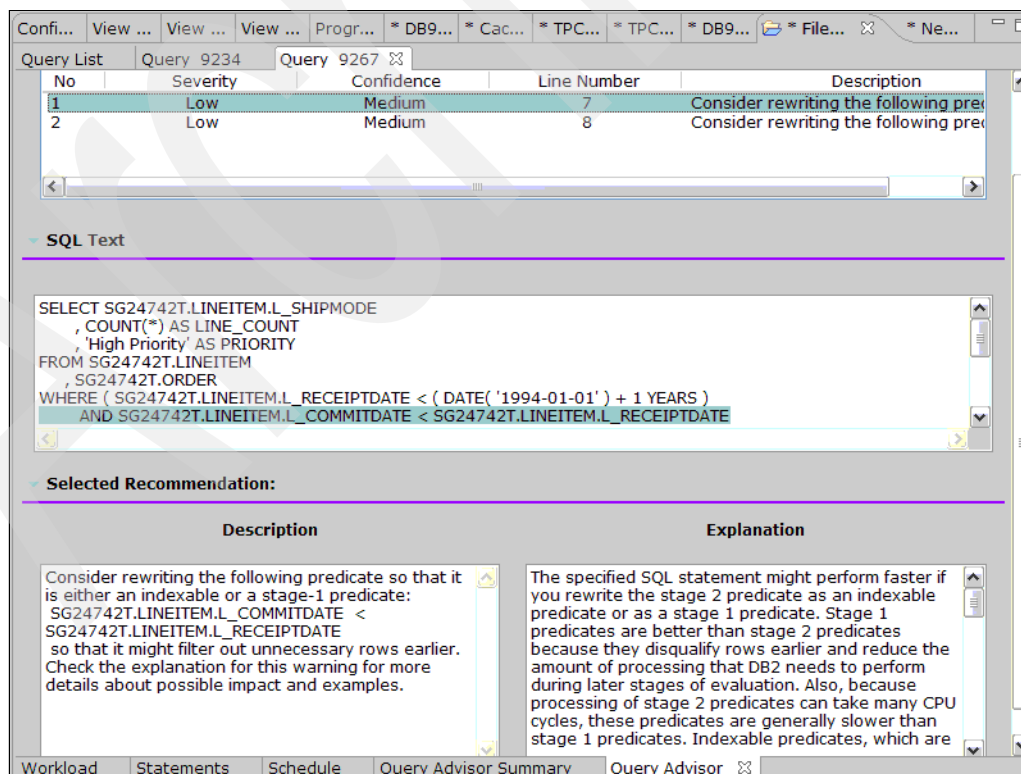


Figure 11-49 The first recommendation for the second statement

The text in the Description box is shown in Figure 11-50.

Consider rewriting the following predicate so that it is either an indexable or a stage-1 predicate:  
SG24742T.LINEITEM.L\_COMMITDATE < SG24742T.LINEITEM.L\_RECEIPTDATE  
so that it might filter out unnecessary rows earlier. Check the explanation for this warning for more details about possible impact and examples.

Figure 11-50 Description of first recommendation

The text of the Explanation box is shown in Figure 11-51.

The specified SQL statement might perform faster if you rewrite the stage 2 predicate as an indexable predicate or as a stage 1 predicate. Stage 1 predicates are better than stage 2 predicates because they disqualify rows earlier and reduce the amount of processing that DB2 needs to perform during later stages of evaluation. Also, because processing of stage 2 predicates can take many CPU cycles, these predicates are generally slower than stage 1 predicates. Indexable predicates, which are predicates that match existing indexes, are all stage 1 predicates.

(Stage 2 predicates are sometimes called residual predicates, and stage 1 predicates are sometimes called sargable predicates.)

For more information about what makes a predicate stage 1 or stage 2, search for information on predicate properties in the Information Management Software for z/OS Solutions Information Center at <http://publib.boulder.ibm.com/infocenter/imzic>. Generally the predicate type depends on the structure of the predicate (for example, a range predicate, an IN-list predicate, a subquery predicate, and so on).

The following examples show how to rewrite a stage 2 predicate as a stage 1 predicate.

Example 1:

Stage 2: 5 BETWEEN T1.C1 AND T2.C1

Stage 1: T1.C1 <= 5 AND T2.C1 >= 5

Example 2:

Stage 2: YEAR(T1.DATECOL) = 2005

Stage 1: T1.DATECOL BETWEEN '2005-01-01' AND '2005-12-31'

Example 3:

Stage 2: LOCATE('P', T1.LASTNAME) > 0

Stage 1: T1.LASTNAME LIKE '/cmvchome2/db2/vc/4/6/9/0/s.52'

Example 4:

Stage 2: T1.DATECOL + 10 YEARS < CURRENT DATE

Stage 1: T1.DATECOL < CURRENT DATE - 10 YEARS

Example 5:

Stage 2: T1.CHARCOL < :dateHV

Stage 1: T1.CHARCOL < CHAR(:dateHV)

Figure 11-51 Explanation of the first recommendation

The predicate SG24742T.LINEITEM.L\_COMMITDATE < SG24742T.LINEITEM.L\_RECEIPTDATE compares two columns of the same table and is, therefore, a Stage 2 predicate. The second warning is a similar problem, as shown in Figure 11-52 on page 319.

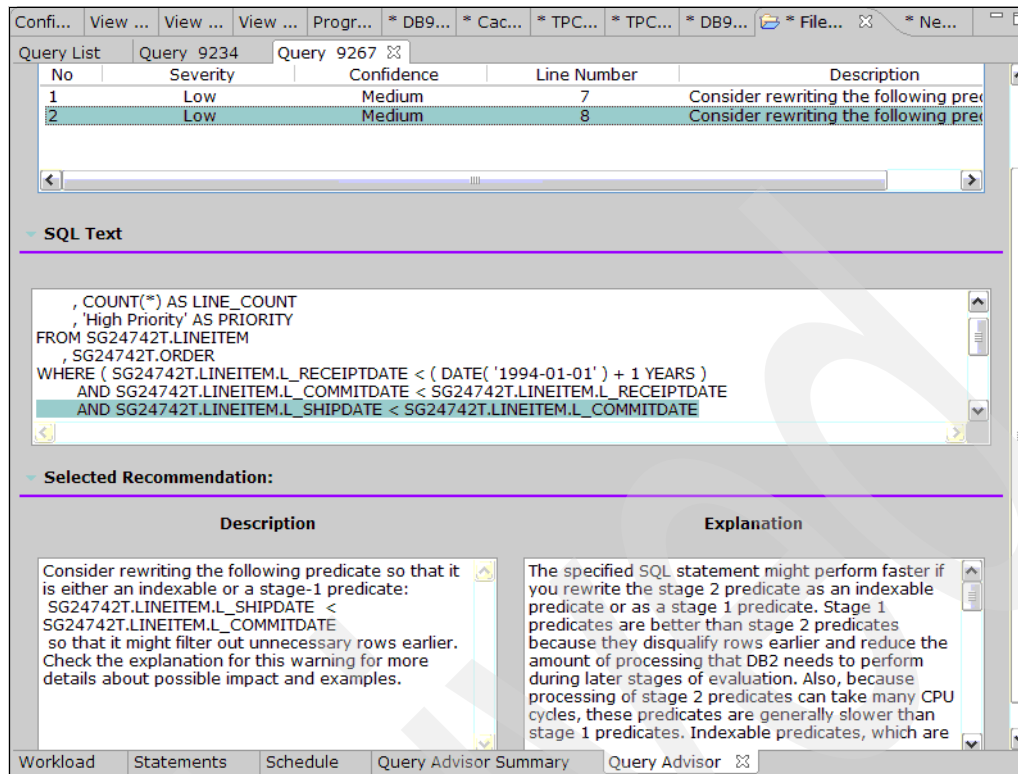


Figure 11-52 The second recommendation for the second statement

The Workload Query Advisor checks exactly the same things as the single-query version, such as:

- ▶ Missing join predicates, but only if a foreign key is defined (see note above).
- ▶ Stage 2 predicates that can improve performance if rewritten as Stage 1 or indexable.
- ▶ Stage 1 predicates that can improve performance if rewritten as indexable.
- ▶ Additional local predicates that are not automatically provided by DB2 can provide Predicate Transitive Closure.
- ▶ Predicates might be pushed down to a nested table expression or materialized view without changing the result. This is not already done automatically by DB2
- ▶ Additional predicates added to a complex WHERE clause containing ORs and ANDs and parentheses might improve performance without changing the result.
- ▶ Use of SELECT \* needs to be replaced with a specific column list.

### 11.4.3 Acting on the recommendations

Review the coding of the SQL statements to see if they need to be rewritten, following the specific recommendations, to improve performance.

### 11.4.4 Setting Workload Query Advisor options

You can access the Preferences panel for the Workload Query Advisor from the main menu, Tools Preferences, to set the defaults for new projects, as shown in Figure 11-53 on page 320. You can set the same preferences for an existing project from the **Advisors** tab for the project.

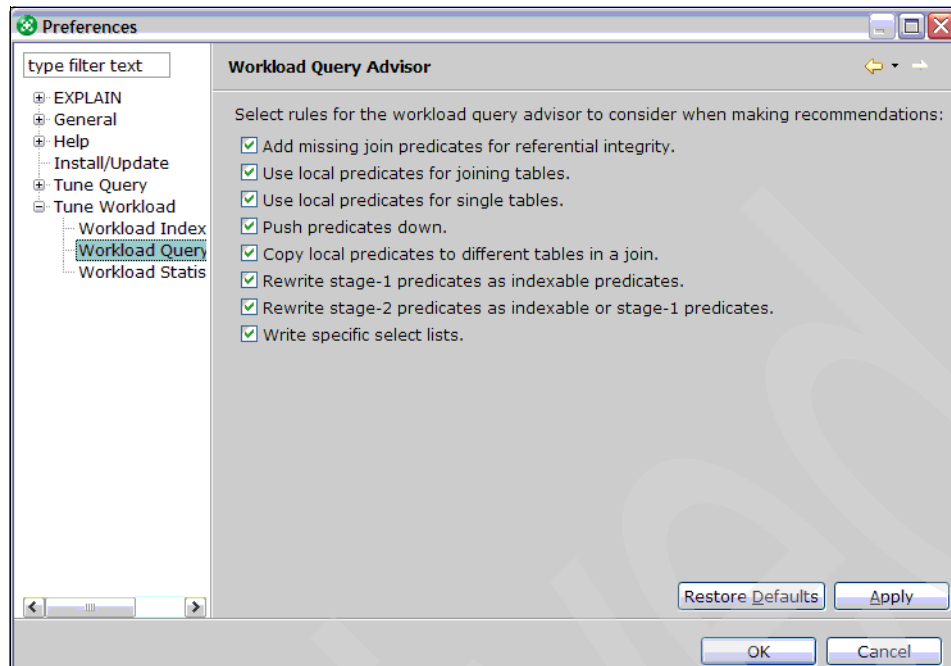


Figure 11-53 Workload Query Advisor preferences panel

In the Preferences panel, you can select the problems that the Query Workload needs to check.



# Tuning scenarios

This part assumes there are two typical scenarios for using the query optimization tools: the analysis of an individual problem query and the tuning of a general query workload.

The following chapters describe the steps for tuning these scenarios:

- ▶ Chapter 12, “Analysis of a problem query” on page 323
- ▶ Chapter 13, “Analysis of a problem workload” on page 343

We also examine using the tools during the life cycle of an application and common practices in tuning the steps and managing the tools in Chapter 14, “Common practices” on page 379.

Archived



## Analysis of a problem query

This chapter discusses how to analyze a problem query using the different components of both Optimization Service Center and Optimization Expert. We look at a scenario of a non-optimal access path caused by data correlation.

## 12.1 Not optimal access path because of data correlation

This scenario uses a simple query to illustrate the process of how to analyze an access path problem using Optimization Service Center or Optimization Expert.

### 12.1.1 The query

Lets assume that the query in Example 12-1 performs badly. There are many ways and tools to identify a problem SQL statement, which are discussed in this book. Lets assume that the task has been completed and that the next obvious question is: Why is this SQL statement performing poorly?

Chapter 6, “Profile monitoring” on page 131 provides information about this topic.

*Example 12-1 SQL statement*

```
SELECT *  
FROM CUSTOMERS  
WHERE ZIPCODE = ?  
AND CITY = ?  
AND STATE = ?
```

### 12.1.2 The current access path

First, explain the problem query and look at the current access path. Using Optimization Service Center or Optimization Expert, you can create a query project (**Project** → **New Query Project**). Next, click **Identify Target Query** and provide the SQL statement text.

If the problem SQL statement was identified as part of a Optimization Service Center or Optimization Expert workload, you can invoke the query tools directly from the workload statement. This automatically triggers the creation of a query project.

To look at the access path, use **Tools** → **Access Plan Graph**. By default, the Graph tab is selected and the graphical representation of the current access path is shown. Most database administrators are used to looking at the rows in the underlying PLAN\_TABLE. Therefore, this option has also been made available by clicking the **Plan Table** tab at the bottom, as shown in Figure 12-1.

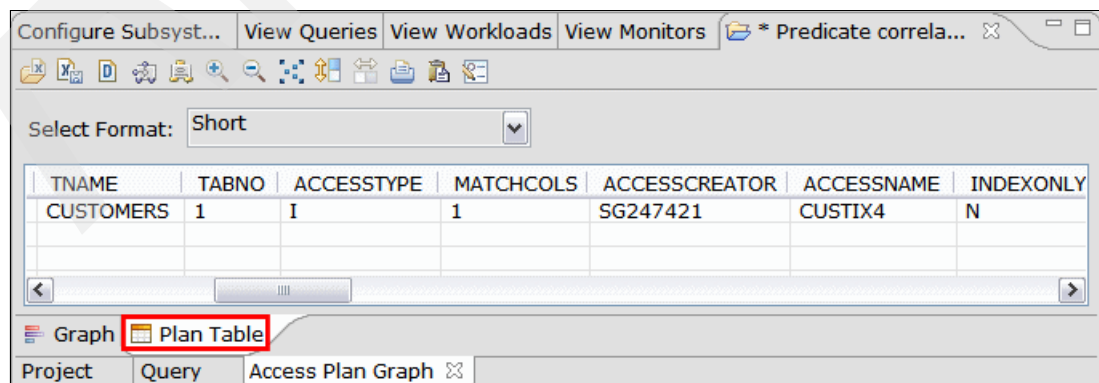


Figure 12-1 Access plan graph using the PLAN\_TABLE

The current access path shows that a one-column matching index access using CUSTIX4 is used. At first glance, this access path looks fine: no table space scan, no non-matching index scan. However, something is wrong. Otherwise, the user does not complain about the performance of this query.

### 12.1.3 Comparing optimizer estimate to real number

First, validate the number of rows the optimizer expects to be returned by the query, against the actual number when you execute the query.

#### Optimizer estimate when using parameter markers

A simple way to determine the estimated number of rows to be returned by this simple query is to look at the “QUALIFIED\_ROWS” value in the annotated query (**Tools → Query Annotation**), as shown in Example 12-2.

*Example 12-2 Annotated query*

---

SELECT SG247421.CUSTOMERS.CUSTNO	
, SG247421.CUSTOMERS.CITY	
, SG247421.CUSTOMERS.STATE	
, SG247421.CUSTOMERS.COUNTRY	
, SG247421.CUSTOMERS.ZIPCODE	
, SG247421.CUSTOMERS.STATUS	
, SG247421.CUSTOMERS.GENDER	
, SG247421.CUSTOMERS.RANDOMNO	
, SG247421.CUSTOMERS.BIRTHDATE	
FROM SG247421.CUSTOMERS	CARDF=192,960
	<b>QUALIFIED_ROWS=16.850983</b>
	NPAGESF=3,860
WHERE ( SG247421.CUSTOMERS.ZIPCODE = ?	COLCARDF=52
	MAX_FREQ=25.473%
	FF=0.019230768084526062
AND SG247421.CUSTOMERS.CITY = ?	COLCARDF=26
	MAX_FREQ=76.418%
	FF=0.038461536169052124
AND SG247421.CUSTOMERS.STATE = ?	COLCARDF=9
	MAX_FREQ=(missing)
	FF=0.1111111044883728

---

In this case, the optimizer expects to return 16.85 rows for this query.

For a more complex query, where multiple tables are involved, you might want to use the access path graph (**Tools → Access Plan Graph**). The top query block in the graph contains the number of rows the optimizer expects to return. Clicking that node opens up the node in the “Node Descriptor” area on the left of the access path graph. The node descriptor also shows the “Output Cardinality” of the query, as shown in Figure 12-2 on page 326.

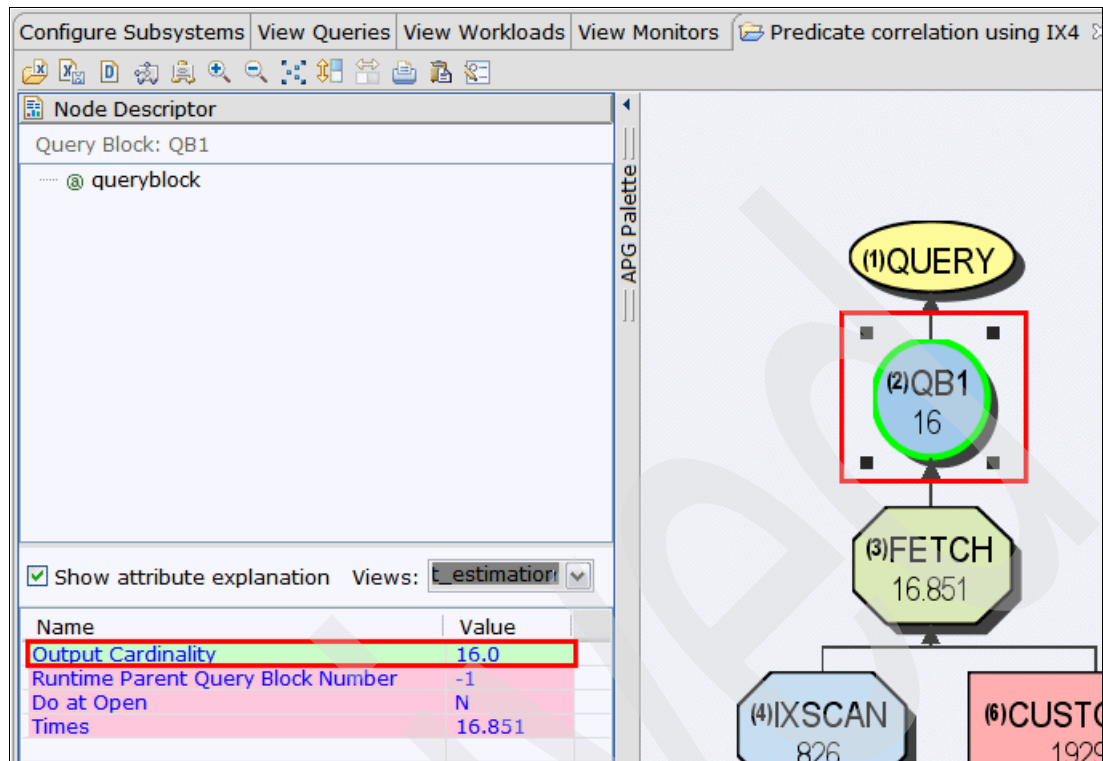


Figure 12-2 Query output cardinality

## Optimizer estimate when using literals

In this example, we use parameter markers. To verify the real number of rows that are returned, you need to know the actual values that are used at runtime. To obtain these values, ask the user for some typical values. Another way is to trace IFCID 247. IFCID 247 records input host variable data related to a user application program. IFCID247 is not present in any performance trace class and needs to be explicitly specified on the -START TRACE command, for example:

```
-STA TRA(P) CL(1,2,3) IFCID(247) PLANNAME(...) ...
```

In this case, the actual host variable values are shown in Example 12-3.

Example 12-3 SQL statement using literals

```
SELECT *
FROM CUSTOMERS
WHERE ZIPCODE = 60607
AND CITY = 'CHICAGO'
AND STATE = 'IL'
```

After explaining the query using these literal values, the optimizer's estimate (QUALIFIED\_ROWS) is 9.873992.

### Actual number of rows returned at execution time

Next, we compare the optimizer estimates to the number of rows returned by the query at execution time by actually running the query. The result is shown in Example 12-4.

Example 12-4 *COUNT(\*) of complete result*

```
***INPUT STATEMENT:
SELECT COUNT(*)
FROM SG247421.CUSTOMERS
WHERE ZIPCODE = 60607
AND CITY = 'CHICAGO'
AND STATE = 'IL'
```

```
+-----+
1_|      3072 |
+-----+
```

Comparing the different estimates against the actual numbers (Table 12-1), the use of parameter markers (“?”) or host variables instead of literals is not the cause of the optimizer’s underestimation of the number of returned rows. Both estimates are below the actual number of rows returned by the query.

Table 12-1 *Filtering of different query components*

	Optimizer estimate with parameter markers	Optimizer estimate with literals	Runtime values
Complete query	16.850983	9.873992	3072

### 12.1.4 Query breakdown

Underestimation of the number of rows to be returned by a query (or overestimation of the amount of filtering the predicates in the query provide) often leads to a poor access path choice. To zoom in on the problem, break the query down into smaller pieces.

#### Match the query to the available indexes

You can significantly improve query performance when the right indexes are defined and used. Which indexes are available for this query? Use the access plan graph and click the **CUSTOMERS** node. This opens the table in the node descriptor area. Click the indexes to see the index key columns (see Figure 12-3 on page 328).

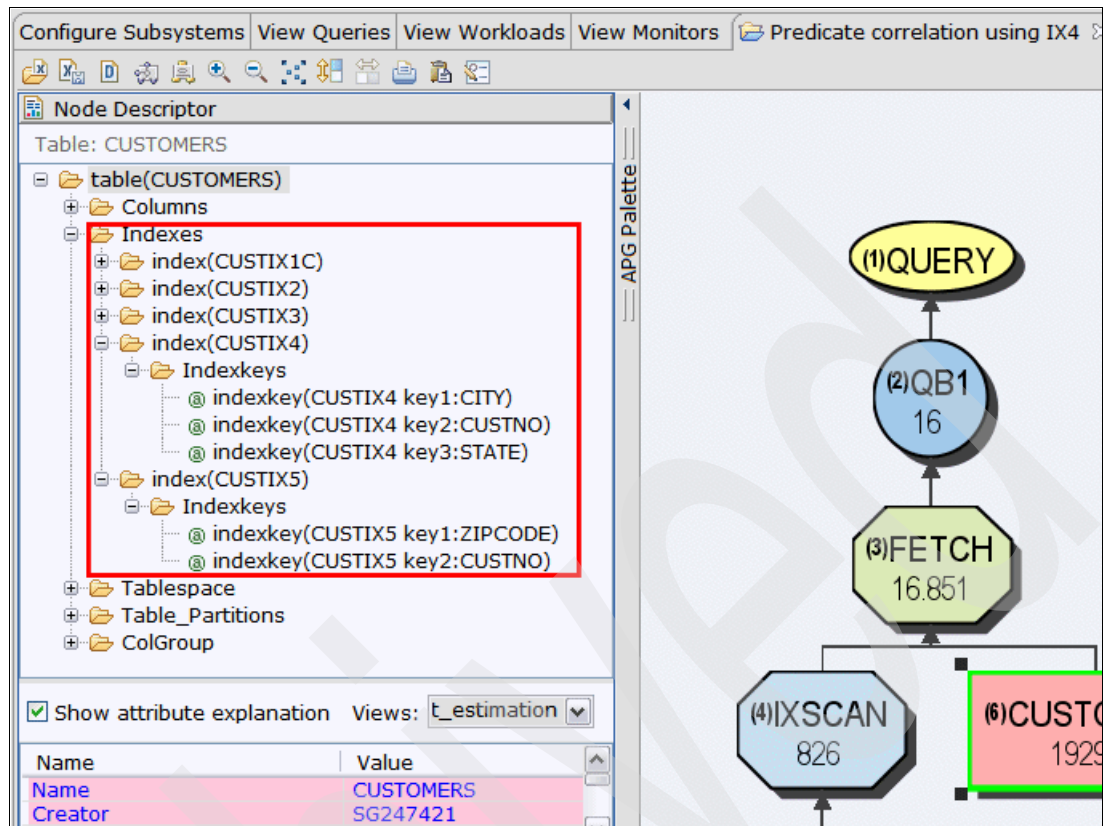


Figure 12-3 Available indexes in Node Descriptor

Another way is to use the **Tools** → **Query Reports** option, check the “Show index report” box, choose between text or HTML output, and click **Generate Report** (Figure 12-4).

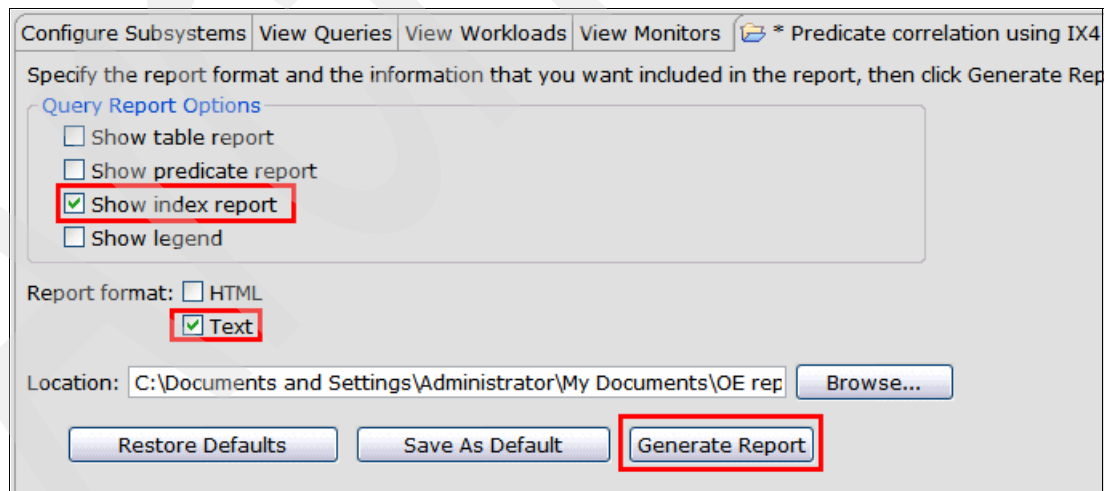


Figure 12-4 Generate index report

The result is shown in Example 12-5 on page 329. When matching the predicate columns to the indexed columns, only CUSTIX4 and CUSTIX5 are interesting because they allow for matching index access: on CITY when using CUSTIX4, or ZIPCODE when using CUSTIX5. When using CUSTIX4, DB2 can also perform index screening on the STATE column. The CITY column is also present in indexes CUSTIX2 and CUSTIX3, but not as a leading column. Since the leading columns of those indexes are not present in our query, DB2 can only do a



non-matching index scan using those indexes. This reduces the likelihood of them being selected by the optimizer.

#### Example 12-5 Available indexes report

Index Report				
== QB:1, PLAN: 1 ==				
=====				
TABLE	CORR_NAME			
SG247421.CUSTOMERS				
=====				
INDEX	INDEX_ONLY	ONE_FETCH	EQUAL_UNIQUE	GB_OB_DISTINCT
SG247421.CUSTIX1C	N	N	N	N
=====				
KEYCOL	ORDER	COLUMN_CARD	MULTI_COL_CARD	PRED
CUSTNO	ASCENDING	192960.0	192960.0	
=====				
INDEX	INDEX_ONLY	ONE_FETCH	EQUAL_UNIQUE	GB_OB_DISTINCT
SG247421.CUSTIX2	N	N	N	N
=====				
KEYCOL	ORDER	COLUMN_CARD	MULTI_COL_CARD	PRED
STATUS	ASCENDING	2.0	2.0	
CITY	ASCENDING	26.0	52.0	SG247421.CUSTOMERS.CITY=(EXPR) (FF:0.038461536169052124)
=====				
INDEX	INDEX_ONLY	ONE_FETCH	EQUAL_UNIQUE	GB_OB_DISTINCT
SG247421.CUSTIX3	N	N	N	N
=====				
KEYCOL	ORDER	COLUMN_CARD	MULTI_COL_CARD	PRED
STATUS	ASCENDING	2.0	2.0	
CITY	ASCENDING	26.0	52.0	SG247421.CUSTOMERS.CITY=(EXPR) (FF:0.038461536169052124)
CUSTNO	ASCENDING	192960.0	192960.0	
=====				
INDEX	INDEX_ONLY	ONE_FETCH	EQUAL_UNIQUE	GB_OB_DISTINCT
<u>SG247421.CUSTIX4</u>	N	N	N	N
=====				
KEYCOL	ORDER	COLUMN_CARD	MULTI_COL_CARD	PRED
CITY	ASCENDING	26.0	26.0	SG247421.CUSTOMERS.CITY=(EXPR) (FF:0.038461536169052124)
CUSTNO	ASCENDING	192960.0	-1.0	
STATE	ASCENDING	9.0	192960.0	SG247421.CUSTOMERS.STATE=(EXPR) (FF:0.1111111044883728)
=====				
INDEX	INDEX_ONLY	ONE_FETCH	EQUAL_UNIQUE	GB_OB_DISTINCT
<u>SG247421.CUSTIX5</u>	N	N	N	N
=====				
KEYCOL	ORDER	COLUMN_CARD	MULTI_COL_CARD	PRED
ZIPCODE	ASCENDING	52.0	52.0	SG247421.CUSTOMERS.ZIPCODE=(EXPR) (FF:0.019230768084526062)
CUSTNO	ASCENDING	192960.0	192960.0	
=====				

### Counting the number of qualified rows per index

Now that you know which columns of the query are indexed, you can run queries to determine how many rows are returned after index filtering (index matching and index screening) by the different indexes. Figure 12-5 on page 330 shows how the query predicates map to the indexed columns of CUSTIX4 and CUSTIX5.

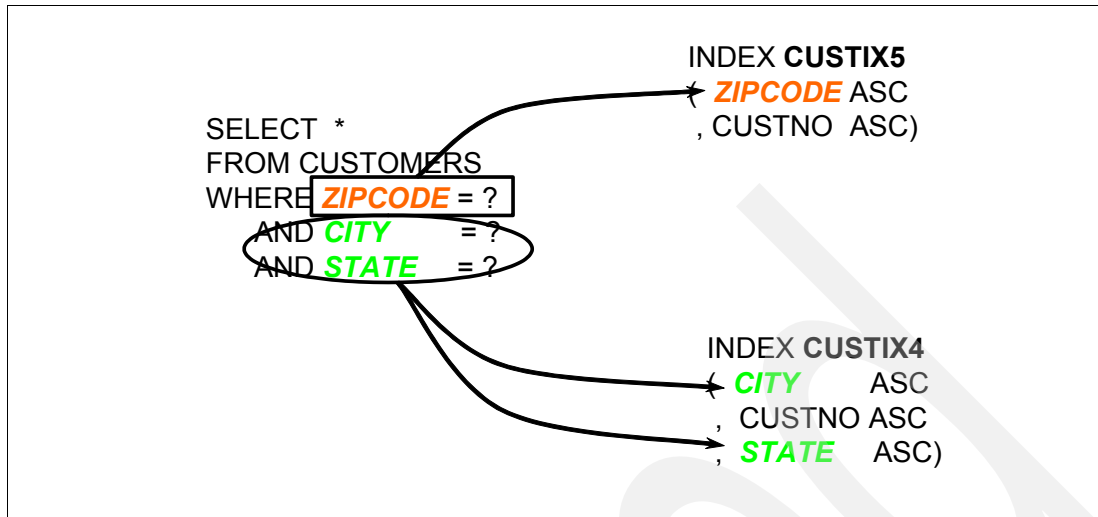


Figure 12-5 Predicate to index mapping

With this mapping in mind, run the following queries (Example 12-6) to determine how much filtering each of the indexes provides.

Example 12-6 Determine amount of index filtering

```
-- INDEX FILTERING FOR CUSTIX4
***INPUT STATEMENT:
  SELECT COUNT(*)
    FROM SG247421.CUSTOMERS
   WHERE CITY = 'CHICAGO'
     AND STATE = 'IL' ;
```

```
+-----+
1_|      21504 |
+-----+
```

```
INDEX FILTERING FOR CUSTIX5
***INPUT STATEMENT:
  SELECT COUNT(*)
    FROM SG247421.CUSTOMERS
   WHERE ZIPCODE = 60607;
```

```
+-----+
1_|      3072 |
+-----+
```

Note that after applying the matching and screening predicates to CUSTIX4 (the index that is currently picked by the optimizer), 21504 rows are returned. After applying the ZIPCODE = 60607 predicate to the CUSTIX5 index, only 3072 are returned. This is also summarized in Table 12-2.

Table 12-2 Filtering of different query components

	Optimizer estimate with parameter markers	Optimizer estimate with literals	Runtime values
Complete query	16.850983	9.873992	3072

	Optimizer estimate with parameter markers	Optimizer estimate with literals	Runtime values
CUSTIX4 filtering			21504
CUSTIX5 filtering			3072

Clearly, CUSTIX5 provides more filtering than CUSTIX4, but the optimizer preferred to use CUSTIX4.

Note that the number of rows returned after applying the ZIPCODE = 60607 predicate to CUSTIX5 is the same than when you run the complete query. In this case, CITY = 'CHICAGO' and STATE = 'IL' provide *no additional filtering*. Based on this information, CUSTIX5 is clearly the better index choice.

When you click the IXSCAN node in the access plan graph for the current access path, Optimization Service Center or Optimization Expert opens the node in the node descriptor window, as shown in Figure 12-6. The figure shows the result of index filtering for the problem query using parameter markers.

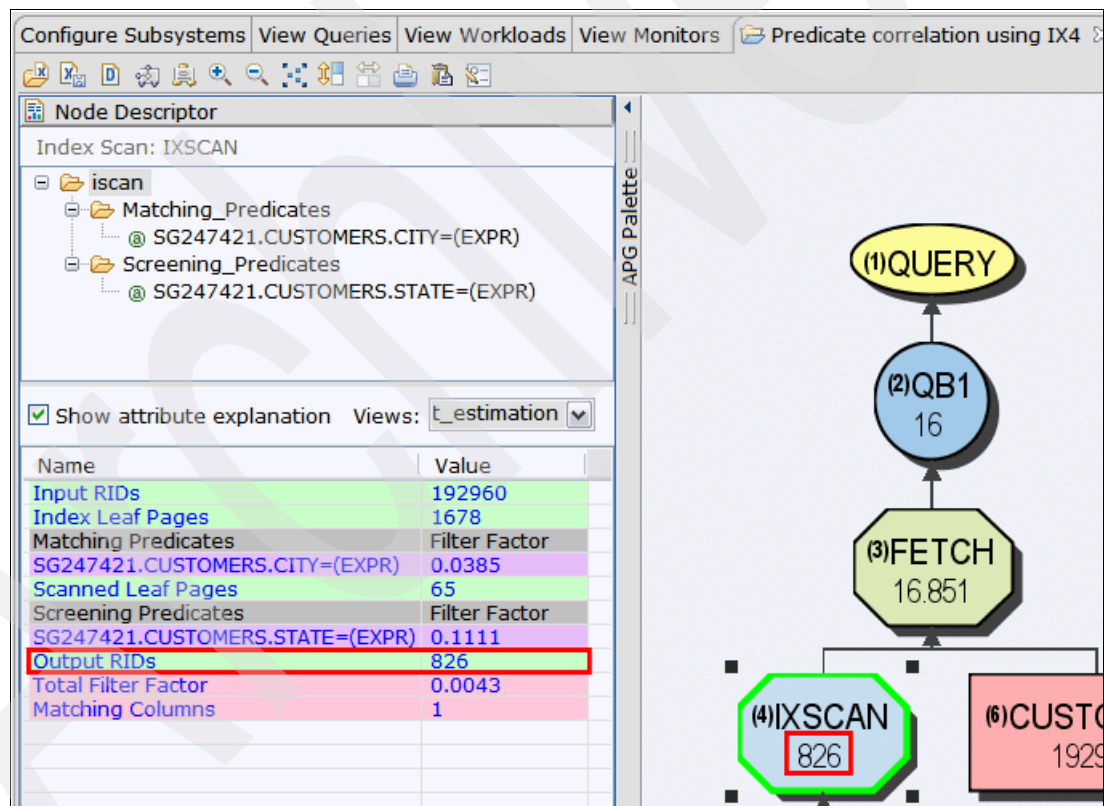


Figure 12-6 Index filtering for CUSTIX4 with parameter markers

The number of rows expected by the optimizer after index filtering for CUSTIX4 is 826. When doing this for the query using literals, the optimizer expects to return 2391 rows after index filtering using CUSTIX4. The result is shown in Table 12-3 on page 332.

Table 12-3 Filtering of different query components

	Optimizer estimate with parameter markers	Optimizer estimate with literals	Runtime values
Complete query	16.850983	9.873992	3072
CUSTIX4 filtering	826	2391	21504
CUSTIX5 filtering			3072

### Counting the number of qualified rows per predicate

Now that you know how many rows qualify for both of the indexes of interest, let us look at the filtering that each individual predicate provides. We do this by obtaining the count for each of the individual predicates as shown in Example 12-7.

Example 12-7 Filtering per predicate

```
-- FILTERING CITY =
***INPUT STATEMENT:
  SELECT COUNT(*)
    FROM SG247421.CUSTOMERS
    WHERE CITY = 'CHICAGO';
```

```
+-----+
1_|      21504 |
+-----+
```

```
-- FILTERING STATE =
**INPUT STATEMENT:
  SELECT COUNT(*)
    FROM SG247421.CUSTOMERS
    WHERE STATE = 'IL' ;
```

```
+-----+
1_|      21792 |
+-----+
```

```
-- FILTERING ZIPCODE (same as CUSTIX5 filtering)
***INPUT STATEMENT:
  SELECT COUNT(*)
    FROM SG247421.CUSTOMERS
    WHERE ZIPCODE = 60607
```

```
+-----+
1_|      3072 |
+-----+
```

The results are added to Table 12-4.

Table 12-4 Filtering of different query components

	Optimizer estimate with parameter markers	Optimizer estimate with literals	Runtime values
Complete query	16.850983	9.873992	3072

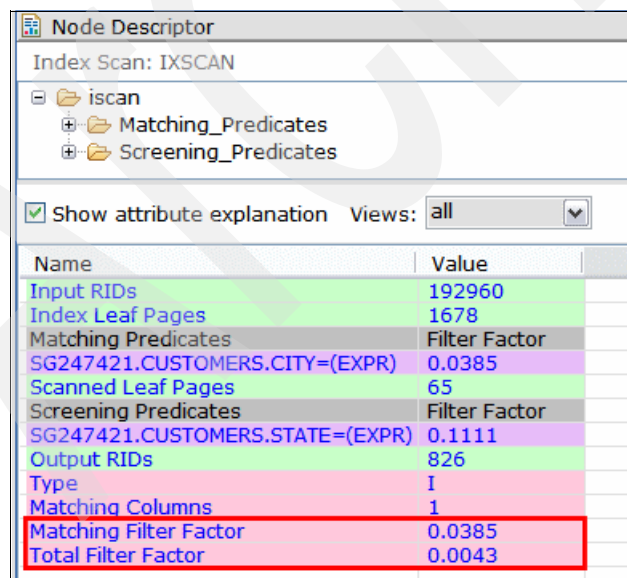
	Optimizer estimate with parameter markers	Optimizer estimate with literals	Runtime values
CUSTIX4 filtering	826	2391	21504
CUSTIX5 filtering			3072
<i>CITY = filtering</i>			21504
<i>STATE = filtering</i>			21792
<i>ZIPCODE = filtering</i>			3072

Note that the CITY= predicate returns the same number of rows as applying both the CITY = and STATE= predicate to CUSTIX4. This means that the STATE= screening predicate provides no additional filtering. As mentioned before, the total number of rows returned after applying the ZIPCODE = predicate is the same as the total number of rows returned by the complete query, indicating that CITY and STATE do not provide any further filtering.

Now compare this with the optimizer's view of reality.

Looking at Figure 12-7, the filter factor of the matching index predicate (CITY=), is not the same as the total filter factor - matching and screening (CITY= and STATE=). However, when we obtained the actual counts above, the screening predicate does not provide additional filtering.

**Note:** The filter factor (FF) is a number between 0 and 1 that provides the estimated filter percentage. It is calculated based on the available catalog statistics. For example, a FF of 0.25 means 25% of the rows are estimated to qualify. For more information about FF, see "Filter factors and catalog statistics" in *DB2 9 for z/OS Performance Monitoring and Tuning Guide*, SC18-9851-00.



Name	Value
Input RIDs	192960
Index Leaf Pages	1678
Matching Predicates	Filter Factor
SG247421.CUSTOMERS.CITY=(EXPR)	0.0385
Scanned Leaf Pages	65
Screening Predicates	Filter Factor
SG247421.CUSTOMERS.STATE=(EXPR)	0.1111
Output RIDs	826
Type	I
Matching Columns	1
Matching Filter Factor	0.0385
Total Filter Factor	0.0043

Figure 12-7 Matching versus total filter factor

In general, and unless there are statistics to indicate otherwise, the optimizer considers each predicate to be independent of each other. Therefore, the total filter factor (FF) for predicates

that are AND-ed together is the FF of the first predicate multiplied by the FF of the second predicate. In our example:

- ▶ FF of CITY= ? is 1/ COLCARD, or 1/26, or 0.0385
- ▶ FF of STATE= ? is also 1/COLCARD, or 1/9, or 0.1111
- ▶ The total FF is 1/26 \* 1/9 = 0.0043

Out of the total number of rows in the table (192 260), 0.43% or 826 (output RIDs) rows are expected to qualify.

It looks like we discovered the root cause of our access path problem. The optimizer expects the predicates to be independent of each other, whereas the actual counts show that this is not the case. Certain columns are not independent, they are *correlated*.

## 12.1.5 Detecting column correlation

The information above strongly suggests that CITY and STATE are not independent columns. To determine whether or not these two columns are correlated, run simple queries, as shown in Example 12-8.

*Example 12-8 Queries to determine correlation*

```
-- DETERMINE CORRELATION
--
```

```
***INPUT STATEMENT:
```

```
SELECT COUNT(DISTINCT CITY) AS #_CITIES
, COUNT(DISTINCT STATE) AS #_STATES
FROM SG247421.CUSTOMERS ;
```

	#_CITIES	#_STATES
1_	26	9

```
SUCCESSFUL RETRIEVAL OF 1 ROW(S)
```

```
PAGE 1
```

```
***INPUT STATEMENT:
```

```
SELECT COUNT(*) AS #_CITIES_STATES
FROM ( SELECT DISTINCT CITY, STATE
FROM SG247421.CUSTOMERS ) AS A;
```

	#_CITIES_STATES
1_	31

If CITY and STATE were independent, there are 26 \* 6 = 234 different combinations present in the data. However, only 31 exist in reality.

**Tip:** When the product of the individual column count (234) is greater than the group count (31), the columns are correlated.

The current catalog statistics clearly fail to show that a correlation exists between the CITY and STATE column. Otherwise, the correlation is taken into account during the optimizer's calculations. This is not the case because the total filter factor calculated by the optimizer matches up with our calculation for independent predicates.

Another way to verify whether correlation information is present is to click the table node in the access path graph and look at the colgroup information in the table's node descriptor, as shown below in Figure 12-8. COLGROUP information describes the cardinality of a combination of columns (other than FULLKEYCARD). This information is obtained from the SYSIBM.SYSCOLDIST table. FULLKEYCARDF statistics can also provide correlation information. It is used when all columns in the index are present in the query. However, this information is not shown in the COLGROUP folder.

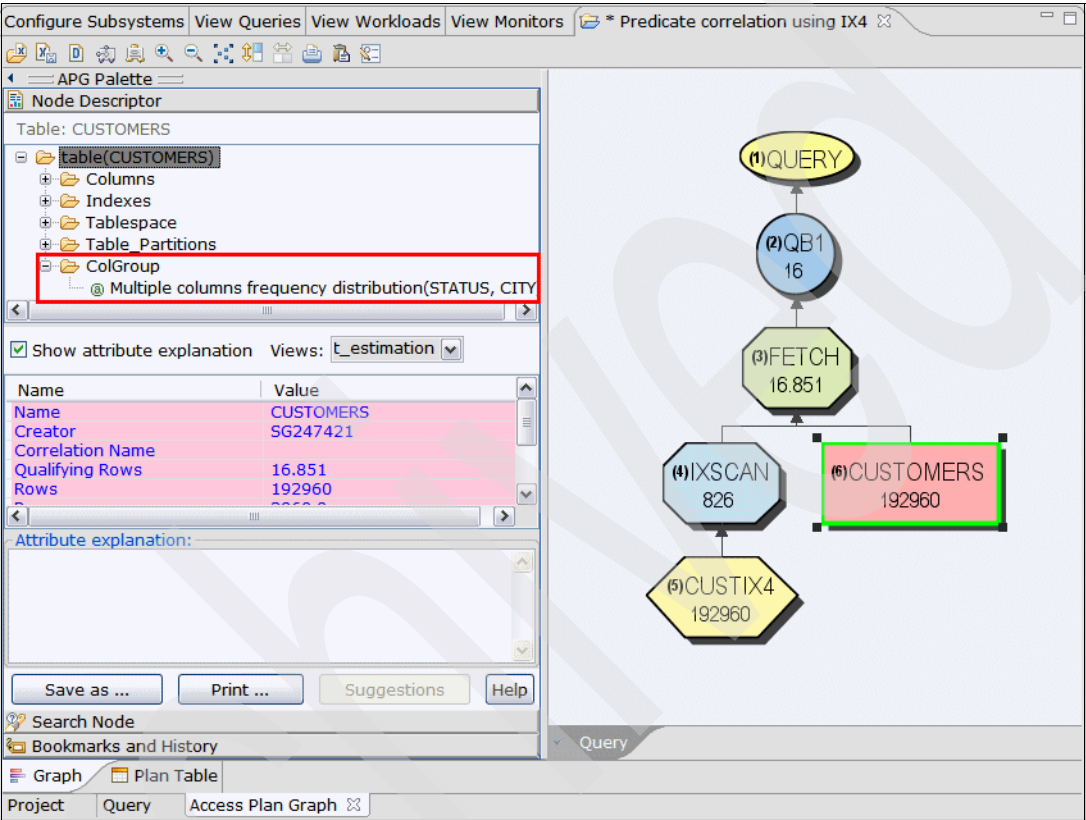


Figure 12-8 Colgroup statistics in node descriptor

There is no colgroup information for the (CITY;STATE) column combination.

Another way to verify whether correlation statistics exist is by using **Tools → Query Reports** and selecting **Show table report**. The result is shown in Example 12-9.

Example 12-9 Column group statistics in the table report

Table Report								
== 1 ==								
TABLE_SPACE	NACTIVEF	PARTS	SEGSIZE	PG_SIZE				
TPTTEST.TPTSTTS1	5040.0	1	4	4				
TABLE	CARDF	NPAGESF	TABNO	QUALROWS				
SG247421.CUSTOMERS	192960	3860	1	16.850983				
INDEX	CLU	UR	NLEAF	NLEVEL	CR	KEYCOLNAME	COLCARDF	MCARDF
SG247421.CUSTIX1C	Y	U	528	3	1.0	CUSTNO	192960	192960
SG247421.CUSTIX2	N	D	251	3	0.998	STATUS	2	2
						CITY	26	52

SG247421.CUSTIX3	N	D	1709	3	0.998	STATUS	2	2
						CITY	26	52
						CUSTNO	192960	192960
SG247421.CUSTIX4	N	D	1678	3	0.999	CITY	26	26
						CUSTNO	192960	-1
						STATE	9	192960
SG247421.CUSTIX5	N	D	815	3	0.999	ZIPCODE	52	52
						CUSTNO	192960	192960
-----								
COLUMN_GROUPMCARDF								
(STATUS,CITY) 52.0								
-----								

The report shows that no COLUMN\_GROUP MCARDF information is available for (CITY,STATE).

**Note:** Note that the CUSTNO column in CUSTIX4 has an MCARD statistic of -1. This means there is no multi-column cardinality statistic (MCARD) value for the combined (CITY, CUSTNO) columns. This is because only basic RUNSTATS information was gathered: RUNSTATS TABLESPACE... TABLE(ALL) INDEX(ALL). The KEYCARD keyword was not specified. If it was specified, there is an actual value instead of -1. However, this additional information, if available, does not help the optimizer to determine a better access path for our problem query.

## 12.1.6 Collecting correlation statistics

As shown in Example 12-9 on page 335, the catalog has the FULLKEYCARDF value of the column combination (CITY,CUSTNO,STATE), namely 192 960, but you need the cardinality of the combination (CITY,STATE) without CUSTNO.

Since DB2 V8, the collection of correlation statistics (MCARD) is easy, as you can now collect this information for any combination of columns (in an index or not, leading or non-leading, or consecutive or non-consecutive columns in an index).

To achieve this, use the COLGROUP option on the RUNSTATS utility statement. For our problem query, you need (CITY,STATE) information. Therefore, execute the following RUNSTATS statement (Example 12-10).

*Example 12-10 RUNSTATS COLGROUP(CITY,STATE)*

```
RUNSTATS TABLESPACE TPTST.TPTSTTS1
TABLE (SG247421.CUSTOMERS)
COLGROUP (CITY,STATE)
REPORT YES
```



After executing the RUNSTATS statement, run explain again, and look at the table report (shown in Example 12-11) to verify that the (CITY,STATE) COLGROUP information was added.

*Example 12-11 Table report after COLGROUP RUNSTATS*

Table Report

== 1 ==

TABLE_SPACE	NACTIVEF	PARTS	SEGSIZE	PG_SIZE				
TPTEST.TPTSTTS1	5040.0	1	4	4				
TABLE	CARDF	NPAGESF	TABNO	QUALROWS				
SG247421.CUSTOMERS	192960	3860	1	120.651276				
INDEX	CLU	UR	NLEAF	NLEVEL	CR	KEYCOLNAME	COLCARDF	MCARDF
SG247421.CUSTIX1C	Y	U	528	3	1.0	CUSTNO	192960	192960
SG247421.CUSTIX2	N	D	251	3	0.998	STATUS	2	2
SG247421.CUSTIX3	N	D	1709	3	0.998	CITY	26	52
						STATUS	2	2
						CITY	26	52
SG247421.CUSTIX4	N	D	1678	3	0.999	CUSTNO	192960	192960
						CITY	26	26
						CUSTNO	192960	-1
SG247421.CUSTIX5	N	D	815	3	0.999	STATE	9	192960
						ZIPCODE	52	52
						CUSTNO	192960	192960
COLUMN_GROUPMCARDF								
(CITY,STATE) 31.0								
(STATUS,CITY) 52.0								

## 12.1.7 Enhanced access path after correlation statistics

Note that when looking at the access path graph (Figure 12-9 on page 338) after re-explaining the statement with the correlation statistics in place, the access path has changed. The optimizer now uses CUSTIX5 instead of CUSTIX4.

After providing the correlation statistics, the optimizer realizes that STATE does not provide any additional filtering, and that the filtering of the CITY predicate alone is less than the filtering provided by ZIPCODE. Therefore, using CUSTIX5 provides more filtering. This is confirmed by the earlier count(\*) we ran using the predicates that were used in the different indexes.

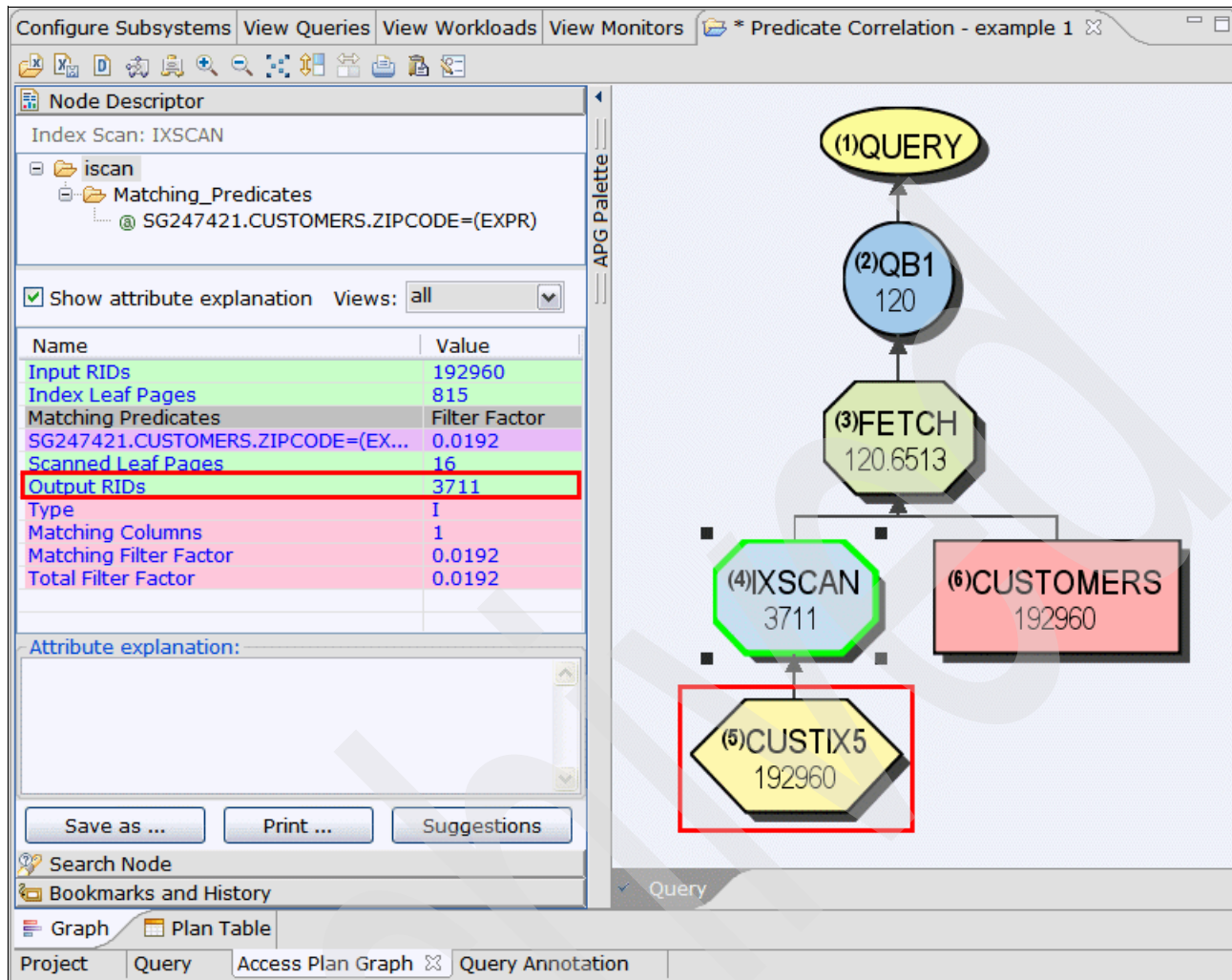


Figure 12-9 Improved access path

The optimizer estimates that 3711 rows are returned after the CUSTIX5 is done with filtering using the ZIPCODE column. Table 12-5 is also updated with this information.

Table 12-5 Filtering of different query components

	Optimizer estimate with parameter markers	Optimizer estimate with literals	Runtime values
Complete query	16.850983	9.873992	3072
CUSTIX4 filtering	826	2391	21504
CUSTIX5 filtering	3711	718(*)	3072
CITY = filtering			21504
STATE = filtering			21792
ZIPCODE = filtering			3072

	Optimizer estimate with parameter markers	Optimizer estimate with literals	Runtime values
(*) The reason for the underestimation we see here is that the top 10 most values have been gathered (FREQVAL). The value 60607 is outside the top 10, but has a similar number of occurrences than the top 10 values. The filter factor is calculated based on: $(1 - \text{SUM}(\text{top\_10\_frequencies})) / (\text{COLCARD} - 10)$			

## 12.1.8 Data access cost

After accessing the CUSTIX5 index, the CUSTOMER table is accessed to apply the remaining predicates. Figure 12-10 shows the data filtering.

The “Input Cardinality” is the number of rows that are left after the index filtering is done. The “Output Cardinality” is the number of rows that are left after the remaining Stage 1 and Stage 2 predicates have been applied to the data rows.

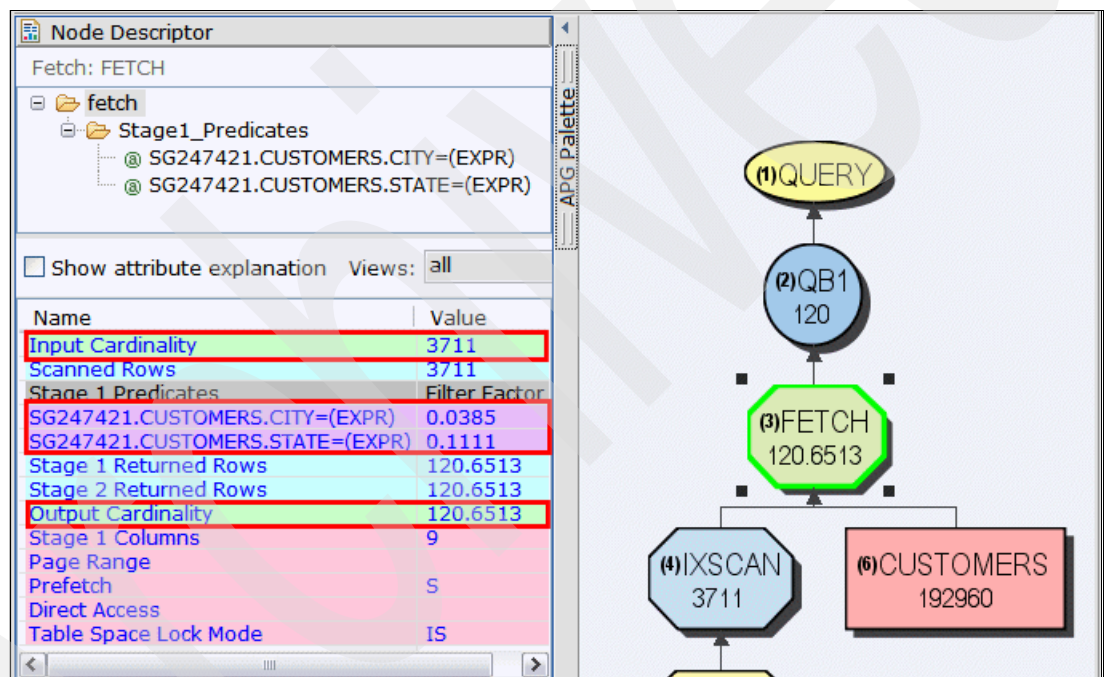


Figure 12-10 Accessing the CUSTOMER table

- Note that the optimizer has used the multi-column cardinality statistics to calculate the filtering provided by “CITY= ? AND STATE = ?”:
  - If the optimizer had used the individual filter factors of both predicates, the estimated output cardinality is:  
 $3711 * 0.0385 * 0.1111 \pm 16 \text{ rows}$
  - However, the optimizer estimates 120 rows to be returned, or:  
 $3711 * 1/31 \pm 120 \text{ rows, where 31 is the MCARD value for (CITY,STATE)}$
- Note also that the input and output cardinalities are different. This means that the optimizer expects that the CITY= and STATE= predicates to provide additional filtering, reducing the result set from 3711 to 120 rows. However, we know that after applying the ZIPCODE= predicate, CITY= and STATE= do not provide any additional filtering because

the number of rows returned by the complete query is the same as the number of rows returned by only applying the ZIPCODE= filtering (IXCUST5). See Table 12-5 on page 338 for details.

To make the optimizer aware of this, gather additional correlation statistics, namely on (ZIPCODE,CITY,STATE). The RUNSTATS statement is shown in Example 12-12.

*Example 12-12 RUNSTATS to gather correlations statistics on (ZIPCODE,CITY, STATE)*

---

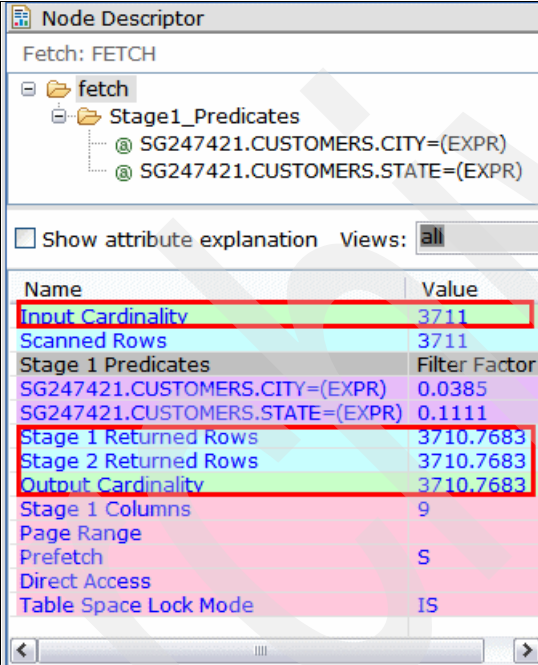
```

RUNSTATS TABLESPACE TPTEST.TPTSTTS1
TABLE (SG247421.CUSTOMERS)
COLGROUP (CITY,STATE,ZIPCODE)
REPORT YES

```

---

After executing the RUNSTATS statement and explaining the statement again, we have another look at the table access part of the access path (Figure 12-11). The access path has not changed, and we are still using CUSTIX5 as expected.



Name	Value
Input Cardinality	3711
Scanned Rows	3711
Stage 1 Predicates	Filter Factor
SG247421.CUSTOMERS.CITY=(EXPR)	0.0385
SG247421.CUSTOMERS.STATE=(EXPR)	0.1111
Stage 1 Returned Rows	3710.7683
Stage 2 Returned Rows	3710.7683
Output Cardinality	3710.7683
Stage 1 Columns	9
Page Range	
Prefetch	S
Direct Access	
Table Space Lock Mode	IS

Figure 12-11 Data access after additional MCARD statistics

The number of rows after index filtering (input cardinality) is still 3711. However, the numbers of qualifying rows after the remaining predicates (CITY= and STATE =) are applied are now higher (3710.7683). Actually, it is the same as the number of rows that we started with (input cardinality). By providing the additional MCARD statistic, the optimizer realizes that CITY= AND STATE= do not provide additional filtering.

## 12.1.9 Conclusion

The optimizer normally considers columns independent unless catalog statistics demonstrate otherwise. Note that with DB2 9, the optimizer no longer assumes complete column independence, even when no correlations statistics are present. Analyzing many different query types over time has shown that when the number of predicates increase, the chances that columns are correlated also increase. The V9 optimizer is taking this into account. Therefore, multiplying the individual predicate filter factors might not give the same filter factor

than the access path graph or query reports show. What is shown in the graph and reports is the value used by the optimizer.

When the optimizer incorrectly assumes that there is no correlation, this can lead to a completely wrong estimate of the total filtering provided by a combination of predicates. As shown in the example, this leads to a bad access path and non-optimal performance.

To collect correlation information, use the following RUNSTATS utility keywords:

- ▶ The KEYCARD option to gather correlation statistics for all multi-column indexes:
  - When using KEYCARD, RUNSTATS gathers multi-column frequency statistics for all leading columns in the index. For example, if IXA is a four-column index on (COL1,COL2,COL3,COL4), specifying KEYCARD will gather MCARD statistics for:
    - (COL1,COL2)
    - (COL1,COL2,COL3)

The column cardinality of COL1 already gathered as FIRSTKEYCARD, and the column cardinality for (COL1,COL2,COL3,COL4) is also part of the default index statistics as FULLKEYCARD.

- ▶ The COLGROUP option to collect correlation statistics:
  - For matching and screening cases for non-consecutive index columns, this is shown in our problem query for CITY,STATE. CUSTIX4 consists of CITY,CUSTNO,STATE so KEYCARD does not gather MCARD statistics on CITY,STATE.
  - For non-indexed columns to allow the optimizer to better determine the total predicate filtering, this is demonstrated by gathering MCARD statistics on CITY,STATE,zipcode.

While indexes can also provide the optimizer with correlation information, the primary goal of an index is to provide filtering.

**Tip:** Note that correlation does *not* require the use of literal values in your SQL statement. The DB2 optimizer can exploit correlation statistics for any kind of SQL statement.

Archived

## Analysis of a problem workload

This chapter provides a sample scenario that demonstrates how you can use Optimization Service Center or Optimization Expert to analyze and tune a workload containing a number of DB2 queries. It uses a set of queries from the IBM WebSphere® Commerce product to analyze.

This chapter also demonstrates how to use Workload Statistics Advisor and Workload Index Advisor, which is only available with Optimization Expert. Both tools can help you:

- ▶ Analyze a set of queries.
- ▶ Identify the problem queries.
- ▶ Make suggestions on how to improve performance by gathering additional statistics or providing a more optimal set of indexes.

## 13.1 The selected workload

For the scenario, we randomly selected 7 queries used by the WebSphere Commerce product. For ease of use, we only selected a rather small subset of all queries used by WebSphere Commerce, but you can use the exact same procedure for a much larger workload.

## 13.2 Connect to Optimization Service Center or Optimization Expert

As noted in the introduction, this scenario uses both Workload Statistics Advisor and Workload Index Advisor. The Index Advisor is only available when you use the Optimization Expert product and it is not available in Optimization Service Center.

If only Optimization Service Center is available, it is still a valuable tool to make sure the optimizer has the best possible statistics information available when deciding on which access path to use. Optimization Service Center is also valuable when analyzing the access path of a particular query.

The first thing to do is to start the product and connect to the DB2 system on which you want to analyze queries.

## 13.3 Defining a monitor profile

In this sample scenario, we use the Optimization Service Center or Optimization Expert workload monitoring capabilities to capture the query runtime information. We can also use the information available in the dynamic statement, cache, but because profile monitoring is new in Optimization Service Center or Optimization Expert and DB2 9 for z/OS, we want to illustrate this feature.

The steps are:

1. To capture the workload, first define a monitor profile, as shown in Figure 13-1 on page 345.



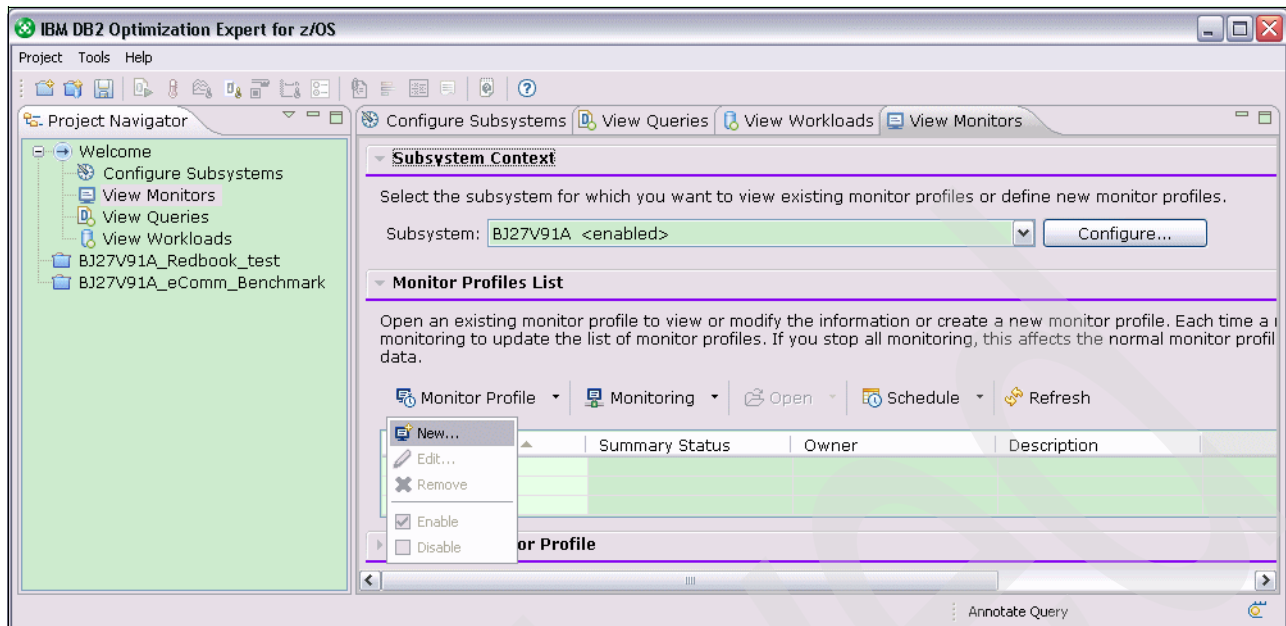


Figure 13-1 Define a monitor profile

2. Select the **View Monitors** tab and select **Monitor Profile** → **New**. This takes you to the window shown in Figure 13-2.

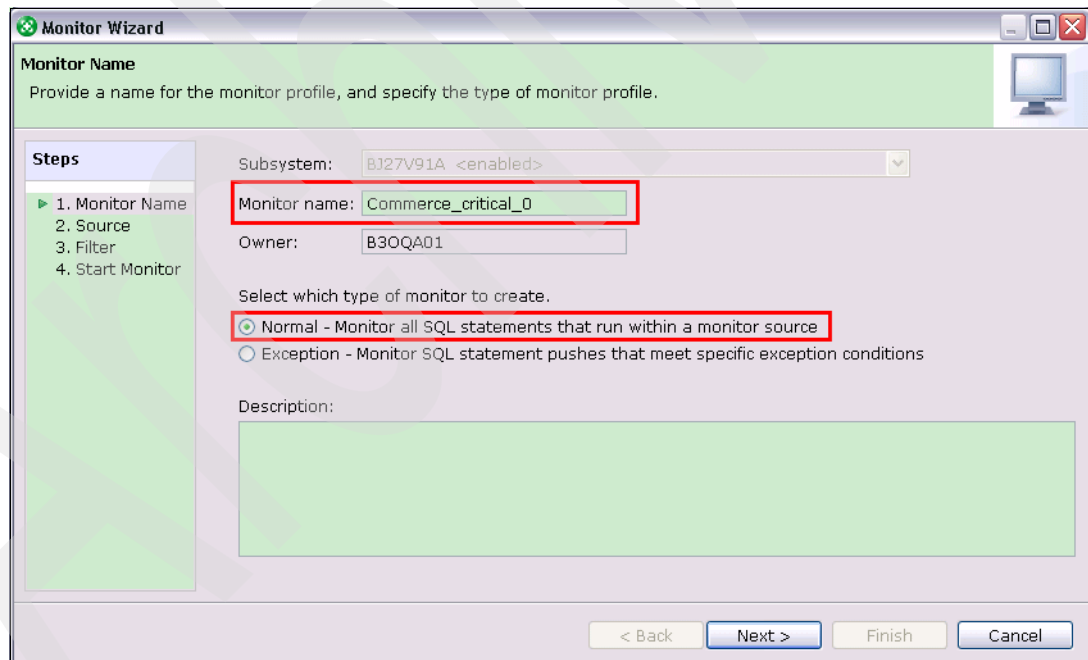


Figure 13-2 Define a 'normal' monitor

You must specify a monitor name. We use "Commerce\_critical\_0", and select that we want to use *normal monitoring*.

Use normal monitoring when you want to capture all statements that qualify the workload criteria, for example, when analyzing a new application that is ready for a performance review. If you are doing periodical monitoring of an existing application to verify whether its

performance is still valid, you can use exception monitoring. This way, only statements that pass the exception thresholds are written out.

3. Clicking **Next** takes you to the window to select the source criteria. This is the selection criteria for the SQL statements that you want to monitor (Figure 13-3).

**Monitor Wizard**

**Monitor Source**  
Define sources for the monitor profile. You can specify more than one statement source in a single monitor profile. Specify Authorization ID and IP address for each dynamic statement source, and specify

**Steps**

1. Monitor Name
2. Source
3. Filter
4. Start Monitor

☒ **Dynamic statements**

Authorization ID:

IP address:

☐ **Embedded statements**

Plan name:

Collection ID:

Package name:

**Source List**

Authorizati...	IP Address	Plan Name	Collection ID	Package N...
TGCOMM01	9.186.66.71			

< Back   Next >   Finish   Cancel

Figure 13-3 Define monitor source

As our SQL statements come in via the network using DRDA®, we select the statements based on authorization ID and IP address. After supplying them, click **Add** to add the selection to the source list.

4. Click **Next** to continue. This takes you to the Settings window shown in Figure 13-4 on page 347.

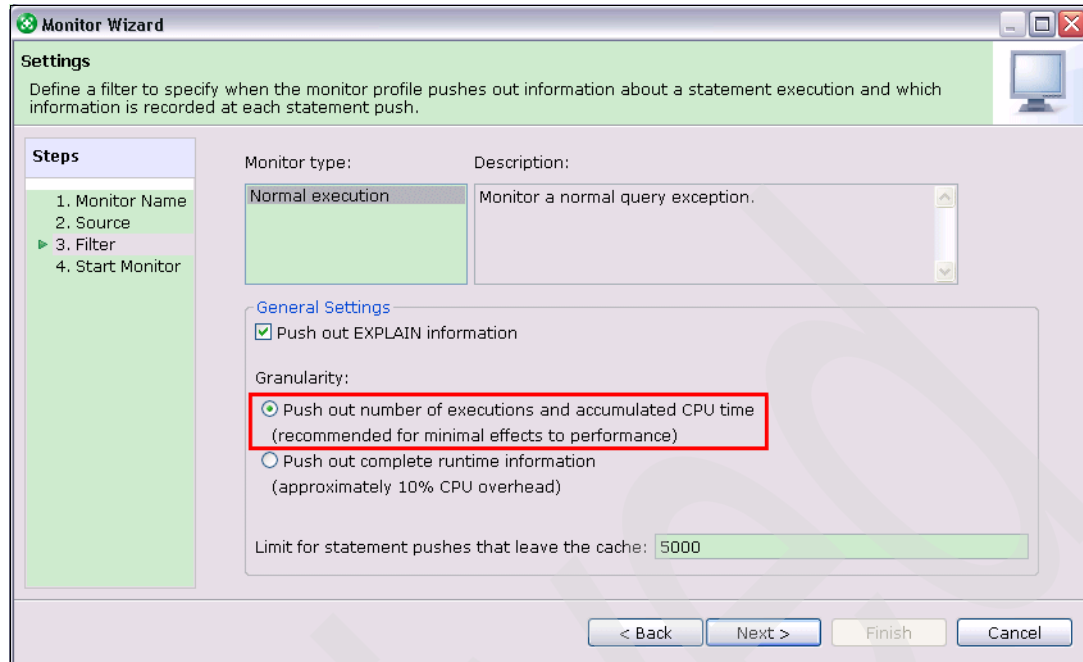


Figure 13-4 Specify monitor granularity

This window allows you to specify which information you want the monitor profile to collect. To get an initial idea about how a workload performs, it is sufficient to collect the number of executions and the elapsed time and CPU time of the queries that qualify the selection criteria.

You can also request to gather the complete set of runtime information, but there is a performance overhead to gather this additional information. As mentioned above, to get a high level view of how a workload is performing, it is normally sufficient to gather the “basic” performance data.

5. Click **Next** to go to the last window (Figure 13-5 on page 348) of the monitor profile specification.

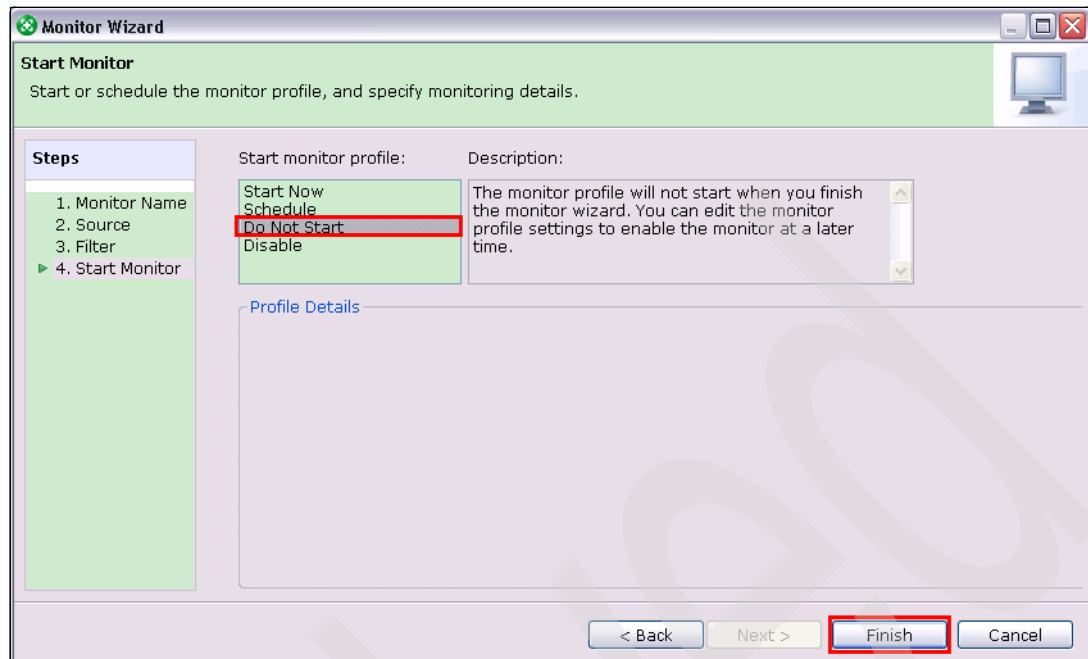


Figure 13-5 Monitor start option

You have the option to start the monitor immediately after the definition is complete, or you can schedule when you want the monitor profile to become active. We want to manually control the start of the profile, and therefore we select the **Do Not Start** option.

6. When done, click **Finish**. The monitor definition is now complete and the monitor profile is defined in the monitor table. Because we specified “Do Not Start”, the monitor profile is not yet started. The result of our definition is shown in Figure 13-6 on page 349. Note that you can also create a monitor profile that is initially disabled. To activate the monitoring profile, you need to enable it first, before starting the monitoring profile.

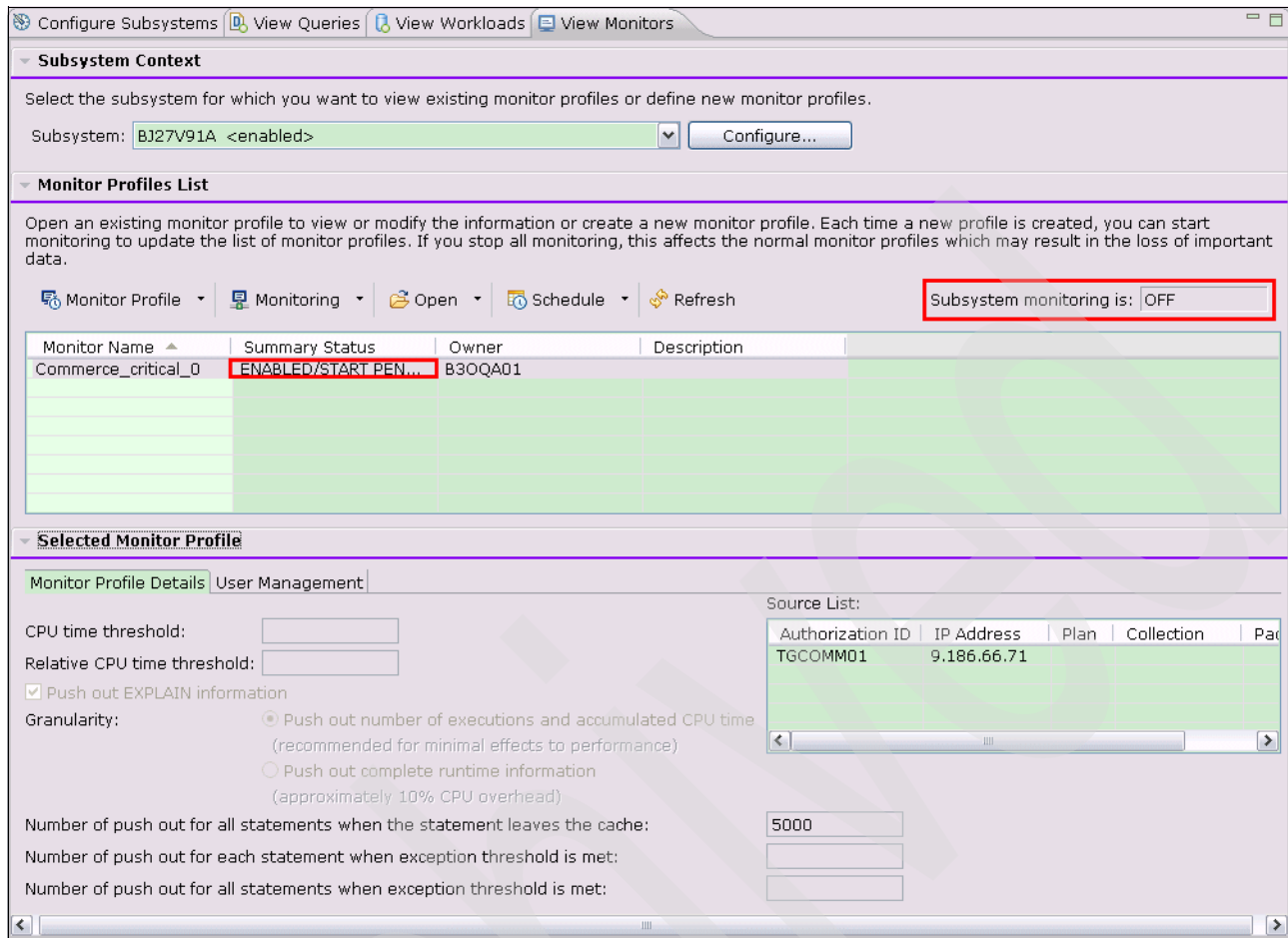


Figure 13-6 Monitor definition complete

The monitor definition is now complete and ready to use.

## 13.4 Activating monitoring and measuring the initial run

We are now ready to do our initial performance measurements.

### 13.4.1 Starting the monitor profile

When the application is ready to run, start the monitor profile (Figure 13-7).

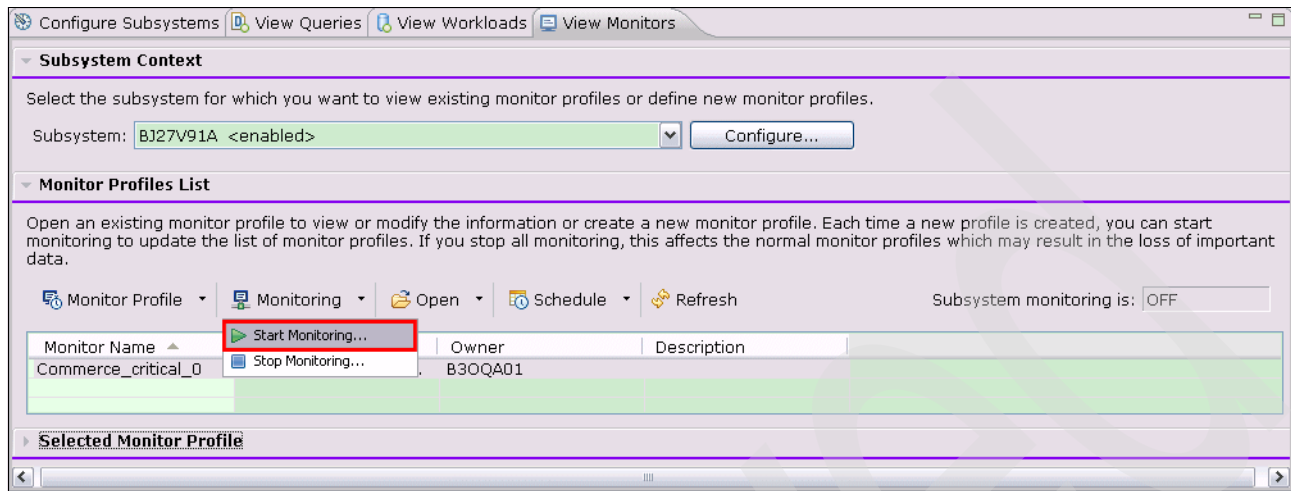


Figure 13-7 Start monitor

At startup time, the Monitor Options window appears as shown in Figure 13-8.

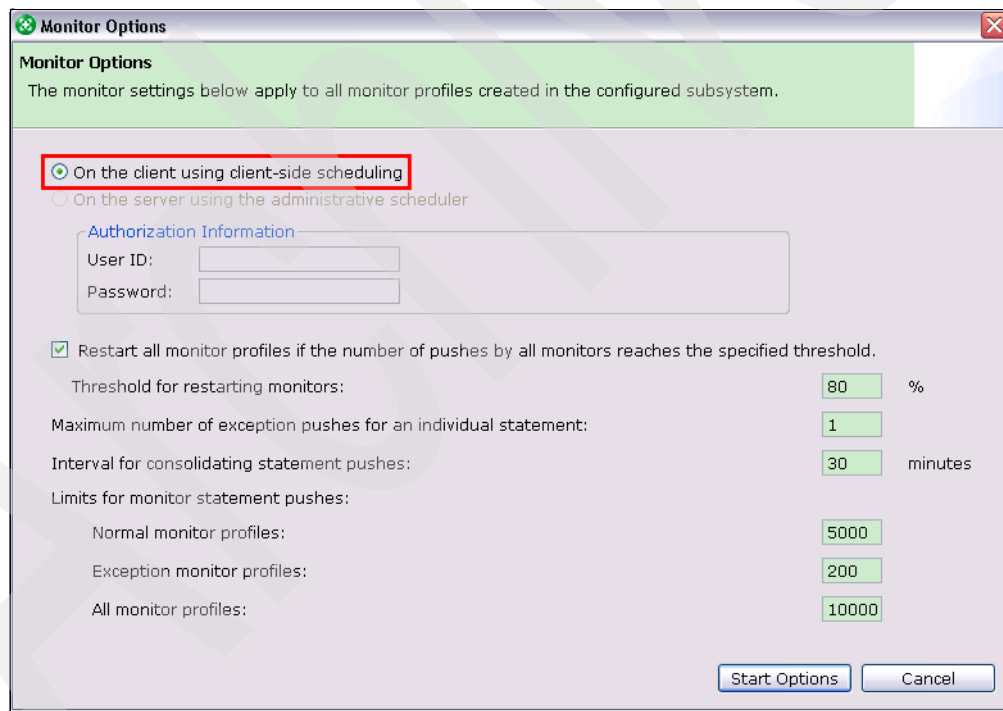


Figure 13-8 Start monitor from the client

This window allows you to specify whether the Optimization Service Center or Optimization Expert client starts the profile monitoring, or whether the start is to be triggered by the Administrative Server. We choose to start the monitoring directly from our workstation.

Accept the defaults for the other options as suggested by Optimization Service Center or Optimization Expert. These options are more geared towards monitoring for a longer period of time and are in place to limit the overhead and amount of output produced by the monitor

profile. For more details about these options, see Chapter 6, “Profile monitoring” on page 131.

Clicking **Start Options** starts the monitor for our profile. This button indicates to start the monitor using the options specified in this window. This returns you to the Monitor Profile List window as shown in Figure 13-9.

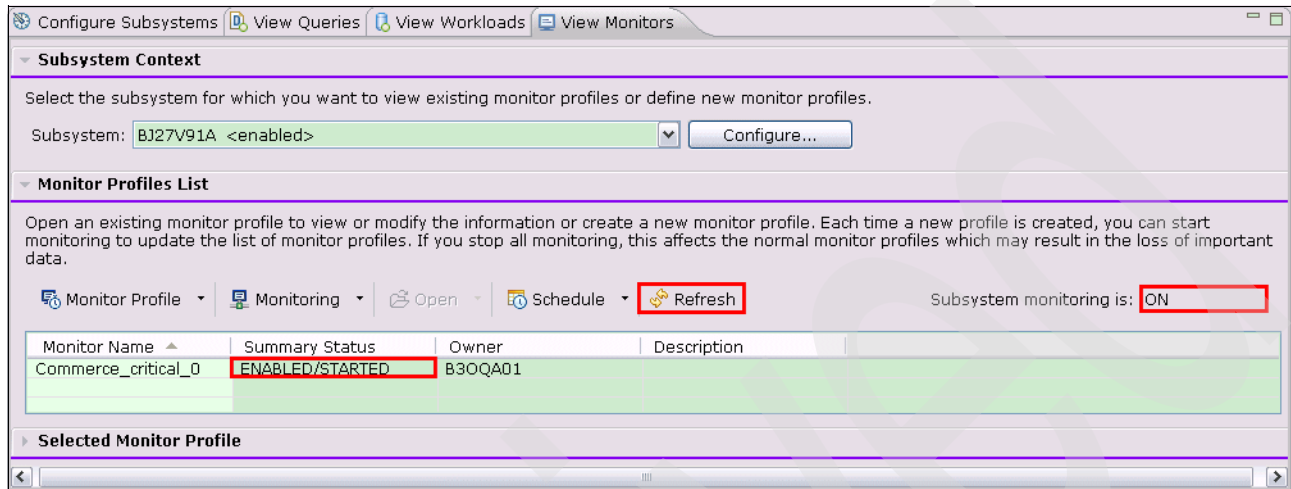


Figure 13-9 Verifying the monitor is started

The summary status for the profile changes to “ENABLED/STARTED”. If it does not, click **Refresh** a few times. The Subsystem Monitoring status also changes to “ON”. If it does not, even after clicking Refresh a few times, you might want to issue the -DIS PROFILE command to verify the status of the monitor.

### 13.4.2 Executing the workload that you want to capture

Now that the monitoring profile has been successfully activated, it is time to run the workload.

### 13.4.3 Obtaining monitoring profile data

After a successful run, look at the data gathered for the monitored profile. There are two ways to externalize the data captured by the monitoring profile:

- ▶ Take a monitor snapshot.
- ▶ Stop the complete profile monitoring.

#### Monitoring snapshot

The first way to externalize the monitoring information gathered by the profile monitor is to snapshot the monitor profile by using **Schedule** → **Snapshot...**, as shown in Figure 13-10 on page 352.

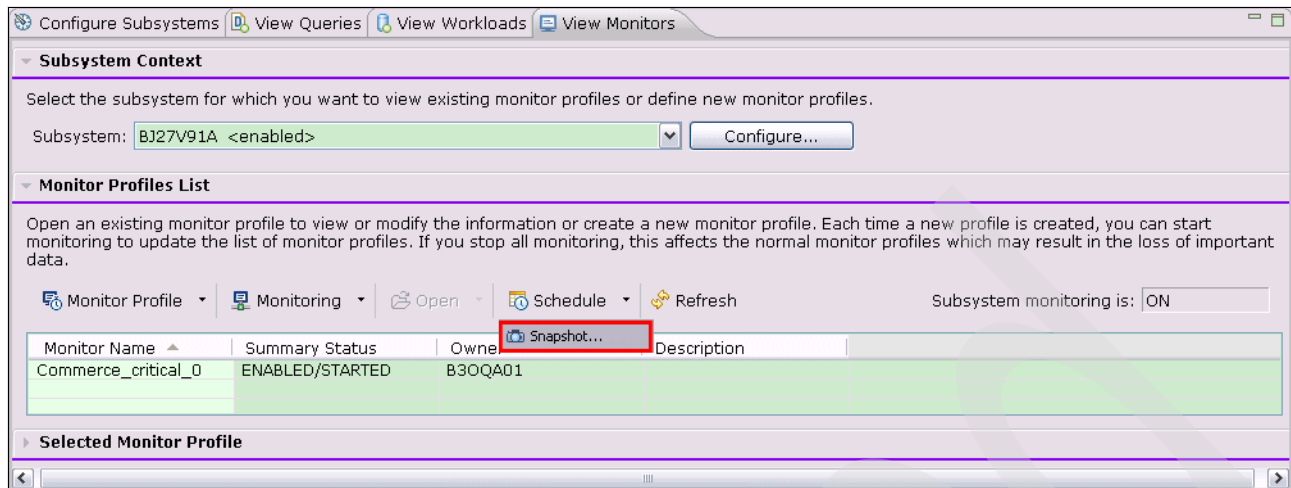


Figure 13-10 Monitoring snapshot

This takes you to the Schedule Snapshot window (Figure 13-11).

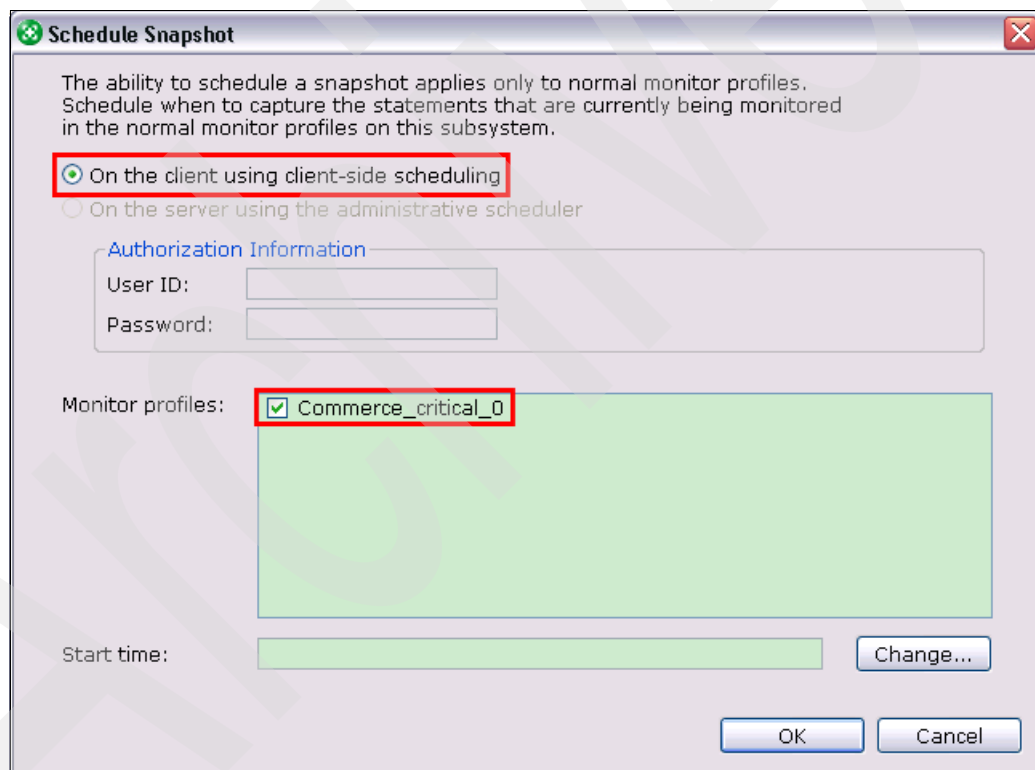


Figure 13-11 Profile snapshot selection

You can either take the snapshot directly using the client workstation, or schedule it through the Administrative Scheduler. You also need to select which monitor profile you want to externalize information for. When done, click **OK** to initiate the snapshot.

### Stopping the monitor completely

Because there is only a single monitoring profile running, you can also stop the profile monitor completely by selecting **Monitoring** → **Stop monitoring**. Optimization Service Center or



Optimization Expert asks you to confirm that you want to stop monitoring because certain data might be lost. The confirmation window is shown in Figure 13-12.

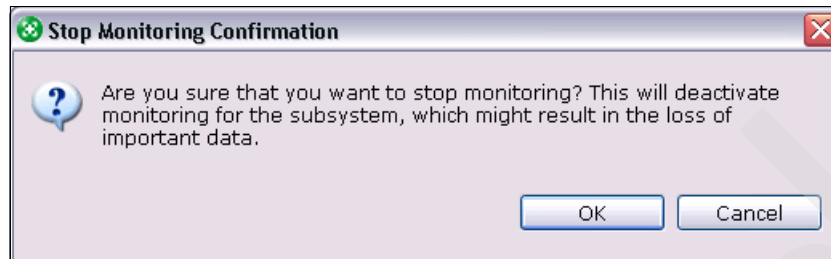


Figure 13-12 Stop monitor warning

Click **OK** to continue. After the monitor is stopped, the subsystem monitoring indicator shows "OFF".

## 13.5 Analyzing the results

You can look at the data that was captured by the monitoring profile by clicking it and clicking **Open** to look at the captured statements (Figure 13-13).

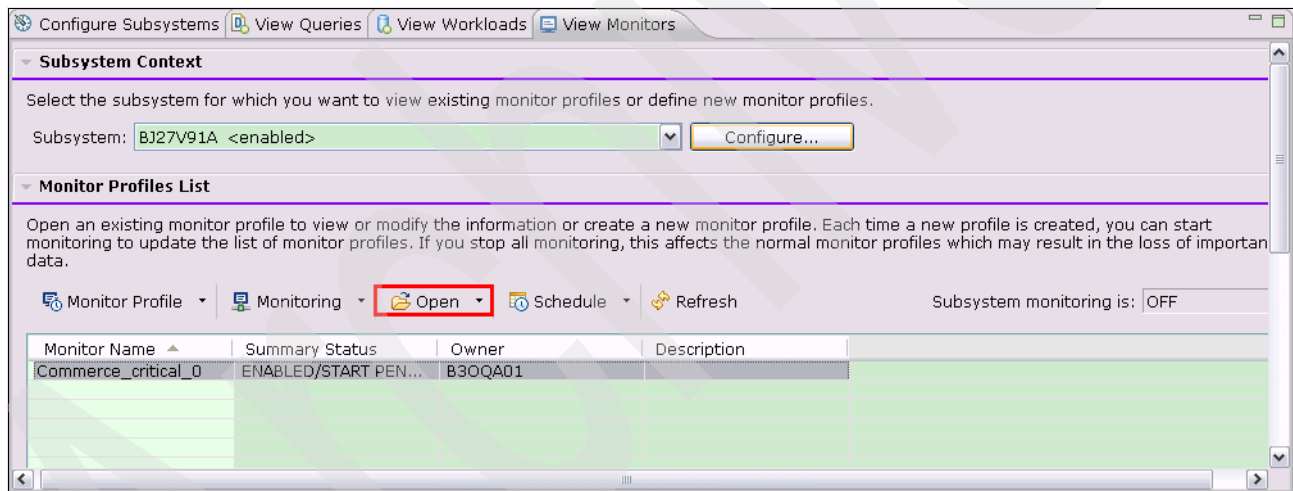


Figure 13-13 Open monitor results

Opening a workload adds a new project tab to the top for this workload project. The name of the project is the subsystem name concatenated with the profile name, DBJ27V91A\_Commerce\_critical\_0 in our case.

After the project is automatically created for the workload, the statement tab opens. The statements that were captured by the monitoring profile are displayed. The result of the workload run is shown in Figure 13-14 on page 354.

**Workload Statements**

Immediately capture statements or multiple sources to this workload, launch workload advisors, use tools to tune selected queries from the workload, or schedule tasks for capture, consolidation, and analysis.

Capture Workload Tools Schedule Remove Query Tools Refresh

All of the rows are displayed. The number of rows is 7.

All Monitor

Execution Count	Source	Accumulated Elapsed...	Average Elapsed...	Accumulated CPU...	Average CPU ...	Statement Text
12	MONI...	170.36246	14.196872	167.11783	13.926486	SELECT DISTINCT T1.PARTNUM
3	MONI...	61.14922	20.383074	60.074955	20.024984	SELECT B1.MEMBER_ID , B1.L
2	MONI...	10.850196	5.425098	10.13576	5.06788	SELECT DISTINCT T1.CATENTR
14	MONI...	3.0855324	0.22039518	2.3212483	0.16580345	SELECT T1.ENDTIME , T1.POLI
11	MONI...	0.037618086	0.003419826	0.024586668	0.0022351516	SELECT T1.CATGROUP_ID , T1
5	MONI...	0.020077297	0.0040154597	0.010990039	0.002198008	SELECT T1.NAME , T2.CATALO
8	MONI...	0.011632756	0.0014540945	0.0032596118	4.0745147E-4	SELECT T1.NAME , T1.AUXDES

Workload Statements

Figure 13-14 Initial results

As with all windows in Optimization Service Center or Optimization Expert, you can resize each individual column and sort the data on each of the columns.

In our case, we are interested in which statement executed the most CPU time, either because it was executed many times, or because an individual execution is CPU intensive, or both.

From the data above, it is clear that:

- ▶ The first statement not only uses a lot of CPU when executed once (13.9 CPU seconds), but it is also executed quite frequently by this workload (12 times).
- ▶ Statement 2 is the individual statement that uses the most CPU when executed (20 CPU seconds).
- ▶ The third statement is also intensive in terms of CPU usage.
- ▶ The other statements run subsecond.
- ▶ There are definitely tuning opportunities in this workload.

## 13.6 Running Workload Statistics Advisor

Before sitting down to analyze the individual queries in great detail to determine whether there is room for performance improvement, always verify whether the statements used by the workload have enough DB2 statistics information in the DB2 catalog tables for the optimizer to determine a good access path.

Workload Statistics Advisor provides an easy way to verify. The Statistics Advisor not only verifies whether Workload Statistics Advisor sufficient statistics are available, it also recommends which statistics are missing (partial RUNSTATS), and which statistics are gathered on a regular basis for this workload (complete RUNSTATS).

You can easily invoke Workload Statistics Advisor from a workload by selecting **Workload Tools** → **Run Workload Statistics Advisor** as shown in Figure 13-15 on page 355.

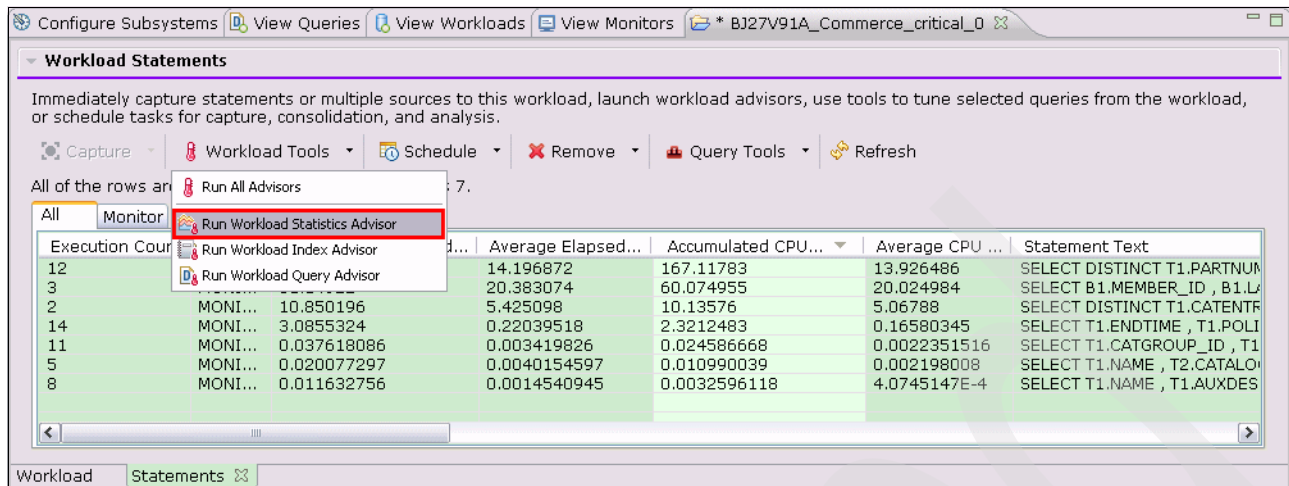


Figure 13-15 Start Workload Statistics Advisor

To operate, Workload Statistics Advisor needs explain and catalog information related to the statements in the workload. Because not all this information is available, Workload Statistics Advisor invokes the explain function to gather the necessary information to perform its analysis. Workload Statistics Advisor informs you about this with the following pop-up window (Figure 13-16).

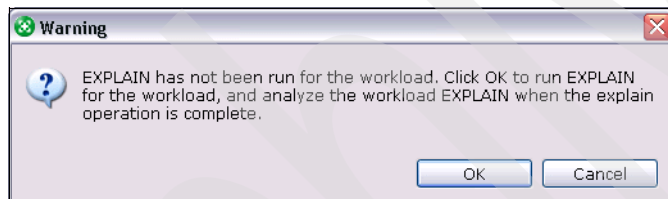


Figure 13-16 Explain not run warning

Click **OK** to continue. This brings up the Schedule EXPLAIN task window, shown in Figure 13-17 on page 356.

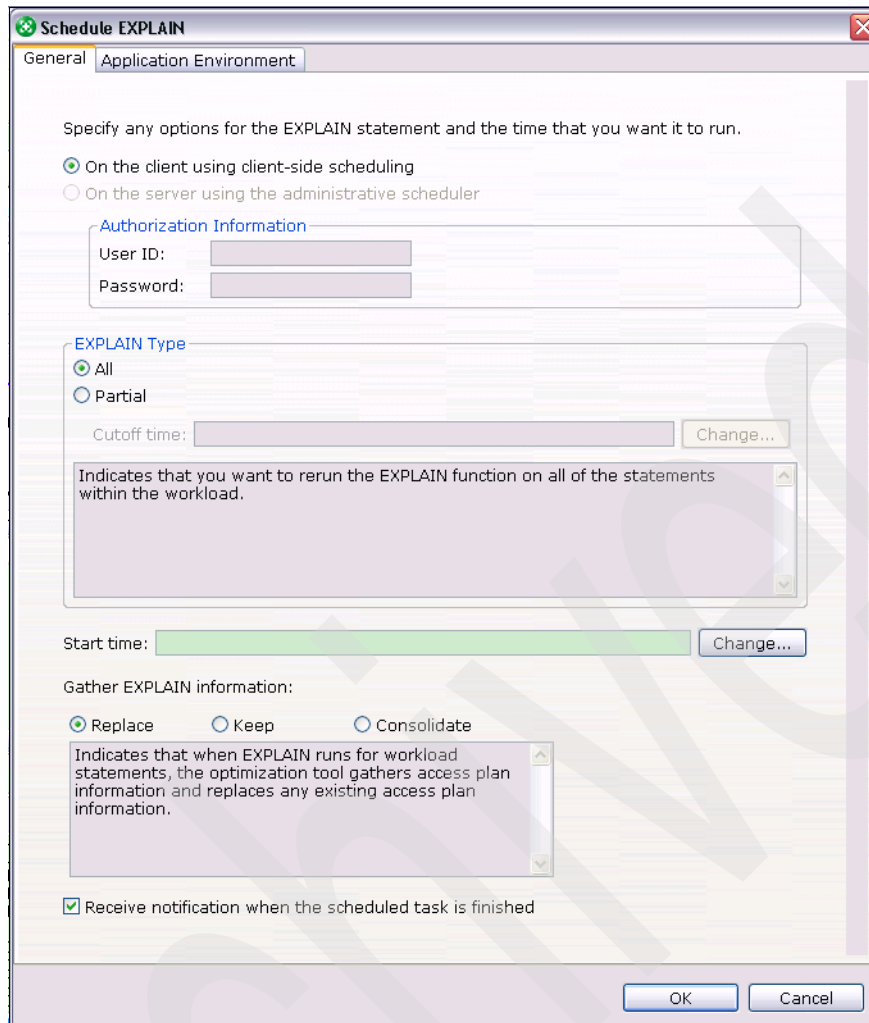


Figure 13-17 Schedule EXPLAIN task

Accept the defaults and click **OK**. Optimization Service Center or Optimization Expert informs you when the task has finished via the following pop-up window (Figure 13-18).

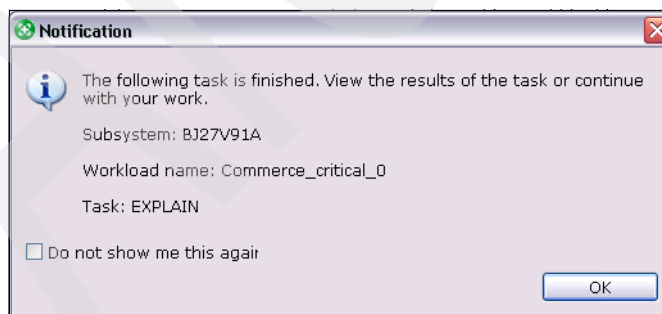


Figure 13-18 Explain task ended

Click **OK** to continue.

Now that the EXPLAIN task has ended, you can invoke Workload Statistics Advisor again by selecting **Workload Tools** → **Run Workload Statistics Advisor** as shown in Figure 13-15 on page 355. Workload Statistics Advisor has completed its analysis and an Advisors tab is

added to the project, representing the status of *all* available Advisors for this project. The Statistics Advisor tab is also added with the recommendations produced by Workload Statistics Advisor. The latter tab is shown in Figure 13-19.

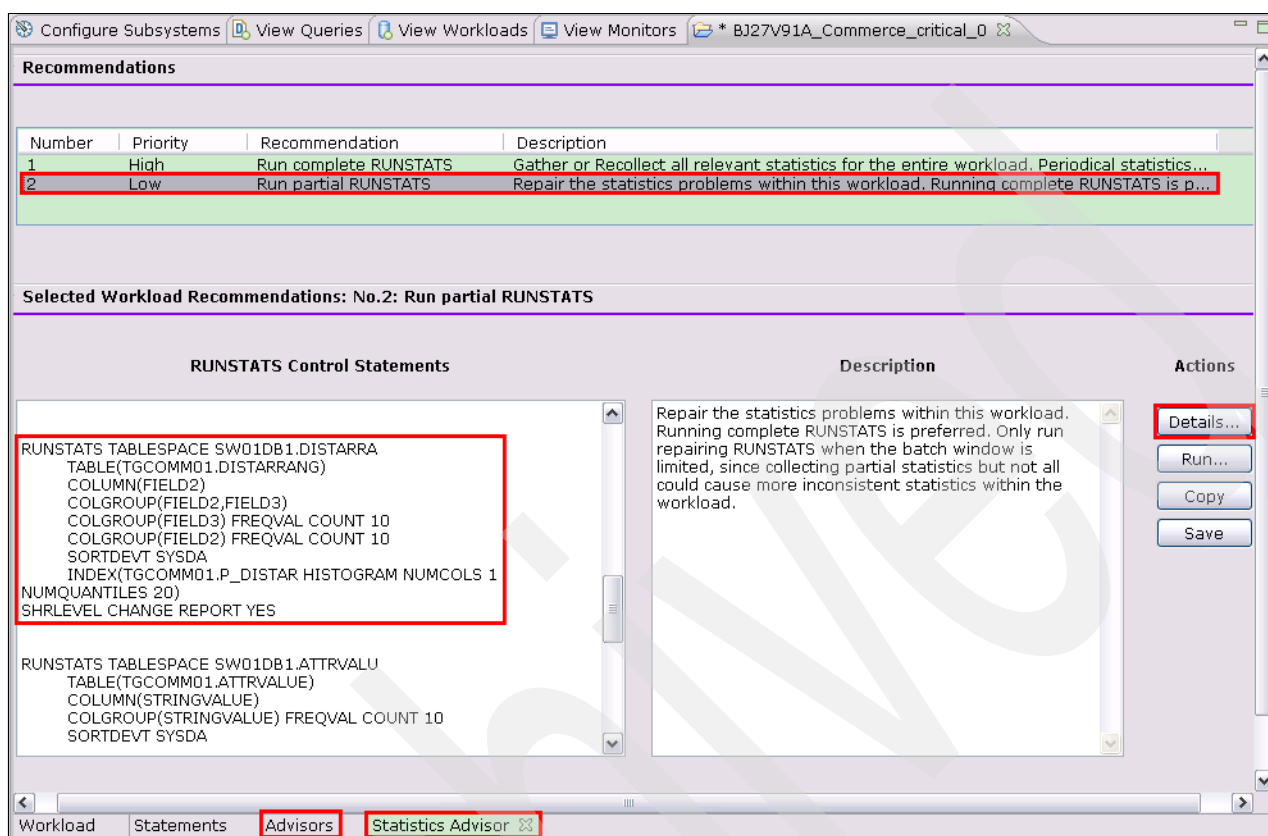


Figure 13-19 Workload Statistics Advisor results

The recommendations returned by Workload Statistics Advisor for this scenario are twofold:

- ▶ A set of complete RUNSTATS statements to gather a complete set of statistics relevant for this workload: This set of RUNSTATS statements is good to save and run on a regular basis, to keep the statistics information in the DB2 catalog up-to-date.
- ▶ A set of RUNSTATS statements to supplement the statistics that are currently available in the DB2 catalog: For a big workload, it is not easy to justify to run the complete set of statistics again when you are just trying to improve the performance of a few individual queries. Most often, it is sufficient to gather additional statistics to supplement the statistics that are already available. For this purpose, Workload Statistics Advisor provides you with a targeted set of additional RUNSTATS statements to only collect the statistics that are currently “missing” from the catalog, but cab benefit one or more queries from the workload.

**Note:** Note that running only partial statistics might introduce inconsistencies in your catalog statistics, as the “missing statistics” have been gathered just now. However, the original statistics might have been gathered weeks ago and the data might have changed. Therefore, we suggest that you run Workload Statistics Advisor again after implementing the initial set of recommended statistics. If there are inconsistencies, Workload Statistics Advisor alerts you to those after running a second time. You can decide if it is appropriate to gather a complete set of RUNSTATS statements.

To demonstrate that it is not required to gather all statistics from scratch, we use the “partial RUNSTATS” in this scenario. Select the **Run partial RUNSTATS** row from the recommendations table (Figure 13-19 on page 357).

The bottom left part of the window now shows the RUNSTATS statements that will gather the “missing” statistics. You can save the statements to a file using the **Save** button, copy them to the clipboard (**Copy**), look at the detailed information as to why these statistics are recommended (**Details**), or execute the statements directly from Optimization Service Center or Optimization Expert (**Run**).

We want to know why these RUNSTATS statements are being suggested, so click **Details**.

This opens the Statistics Advisor Details tab. At the top, you have the table, index, column and column group details, and at the bottom, the Conflicts detail section.

Lets look closer at the statistics recommendation for one of the SQL statements. We use the worst performing SQL statement (average CPU time) as an illustration. Select the query from the workload statement list, right-click and select **Query Annotation** (Figure 13-20).

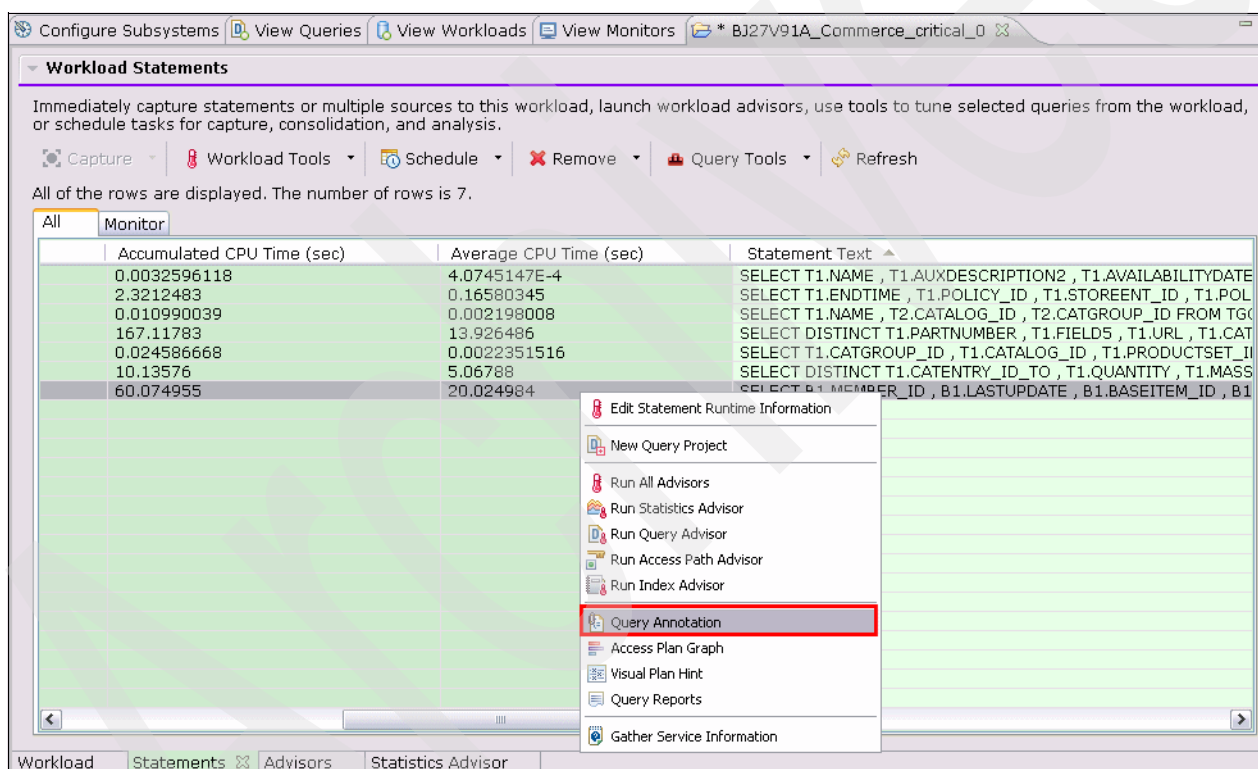


Figure 13-20 Invoke query annotation from monitored statement list

The annotated query is shown in Figure 13-21.

Original Transformed

**Query Annotation**

Indicate which annotation to display and customize your view. Selecting a row will highlight all of the relevant rows for this table. Selecting a join predicate will highlight all of the join predicate rows in both of the joined tables. Click Reset Text to return to the original text view.

Annotation to display: All

Expand All Collapse All Customize Save Print Reset Text

Formatted Query	Annotation
SELECT B1.MEMBER_ID	
, B1.LASTUPDATE	
, B1.BASEITEM_ID	
, B1.MARKFORDELETE	
, B1.PARTNUMBER	
, B1.QUANTITYMULTIPLE	
, B1.QUANTITYMEASURE	
, B1.ITEMTYPE_ID	
FROM TGCOMM01.BASEITEM AS B1	CARDF=90,017 QUALIFIED_
WHERE B1.BASEITEM_ID IN ( SELECT T1.BASEITEM_ID	COLCARDF=90,017 MAX_FR
FROM TGCOMM01.CATGROUP AS C1	CARDF=50,075 QUALIFIED_
, TGCOMM01.CATENTRY AS T1	CARDF=360,000 QUALIFIED_
TGCOMM01.DISTARRANG AS D1	CARDF=90,016 QUALIFIED_
WHERE ( C1.CATGROUP_ID BETWEEN 10004990 AND 10005945	COLCARDF=50,075 MAX_FR
AND D1.DISTARRANG_ID BETWEEN 1000000008975 AND 1000000009200	COLCARDF=90,016 MAX_FR
AND D1.FIELD2 = 'Albany'	COLCARDF=11 MAX_FRE
AND D1.FIELD3 = 'NewYork(NY)'	COLCARDF=8 MAX_FRE
AND T1.CATENTRY_ID BETWEEN 1000000023905 AND 1000000024709	COLCARDF=360,000 MAX_FF
AND D1.FIELD2 = C1.FIELD1	COLCARDF=11/544 MAX_FR
AND D1.FIELD3 = C1.FIELD2	COLCARDF=8/7 MAX_FRE
AND D1.FIELD2 = T1.FIELD4	COLCARDF=11/16 MAX_FRI
AND D1.FIELD3 = T1.FIELD5	COLCARDF=8/13 MAX_FRE
)	
ORDER BY 5 ASC	
FOR READ ONLY	

Project Query Query Annotation

Figure 13-21 Query #5 annotated version

Note that the DISTARRANG table is used in the subselect, having both local and join predicates. Clicking the drop down lists next to the FIELD2 and FIELD3 predicates of the DISTARRANG table also reveal that frequency statistics were not collected for these columns - MAX\_FREQ (missing).

The access plan graph of the query block, in which the DISTARRANG table is accessed, is shown in Figure 13-22 on page 360.

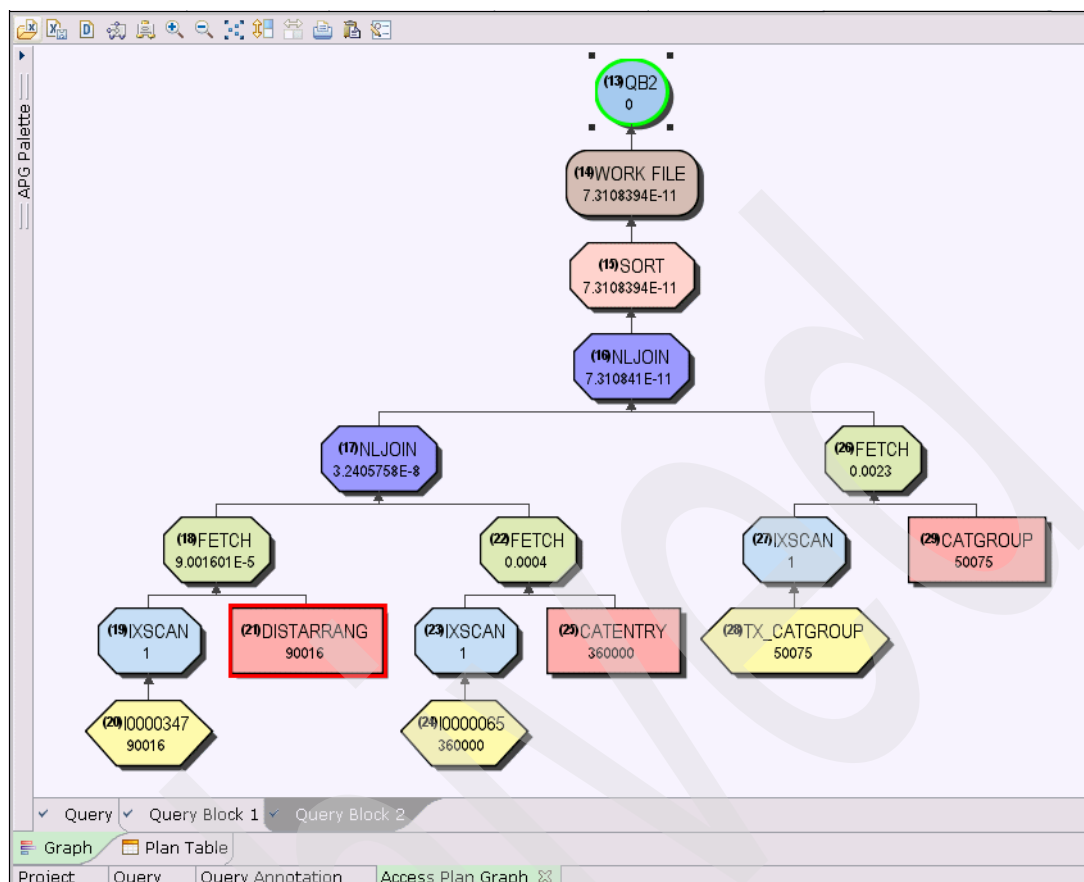


Figure 13-22 Access plan graph for query block accessing DISTARRANG

Note that the join order for query block 2 is DISTARRANG → CATENTRY → CATGROUP. The number of qualifying rows after applying local predicates is shown in Table 13-1. This information is found by looking at the number of QUALIFIED\_ROWS in the drop down lists, next to each of the tables in the annotated query of Figure 13-21 on page 359.

Table 13-1 Number of qualified rows for the tables of query block 2

Table name	QUALIFIED_ROWS
DISTARRANG	9.001601 E-5
CATENTRY	0.0004
CATGROUP	0.0023

The value for DISTARRANG is extremely low, making it a good candidate to become the first table of the join. However, if the optimizer based its decision on incomplete statistics, this might not be the ideal join order. Lets look at it in more detail.

Example 13-1 on page 361 shows an excerpt from the table, index, column, and column group details from the Workload Statistics Advisor recommendations for the DISTARRANG table.



-----

**TABLE TGCOMM01.DISTARRANG**

Cardinality: 90016.0  
Collection Time: 2007-09-24 08:20:24.425057  
Statistics Status: OK

INDEXES:

**TGCOMM01.I0000095 (BASEITEM\_ID,MERCHANTSTORE\_ID,WHOLESALESTORE\_ID,STARTDATE,ENDDATE)**

First Key Cardinality: 90016.0  
Full Key Cardinality: 90016.0  
Data Repetition Factor: 1839.0  
Collection Time: 2007-09-24 08:20:24.425057  
Statistics Status: OK

**TGCOMM01.I0000347 (DISTARRANG\_ID,MERCHANTSTORE\_ID)**

First Key Cardinality: 90016.0  
Full Key Cardinality: 90016.0  
Data Repetition Factor: 1839.0  
Collection Time: 2007-09-24 08:20:24.425057  
Statistics Status: OK

**TGCOMM01.I0000560 (MERCHANTSTORE\_ID)**

First Key Cardinality: 2.0  
Full Key Cardinality: 2.0  
Data Repetition Factor: 1838.0  
Collection Time: 2007-09-24 08:20:24.425057  
Statistics Status: OK

**TGCOMM01.I0000561 (WHOLESALESTORE\_ID)**

First Key Cardinality: 2.0  
Full Key Cardinality: 2.0  
Data Repetition Factor: 1838.0  
Collection Time: 2007-09-24 08:20:24.425057  
Statistics Status: OK

**TGCOMM01.P\_DISTAR (DISTARRANG\_ID)**

First Key Cardinality: 90016.0  
Full Key Cardinality: 90016.0  
Data Repetition Factor: 1839.0  
Collection Time: 2007-09-24 08:20:24.425057  
Statistics Status: OK

Interesting Columns:

**DISTARRANG\_ID**

Cardinality: 90016.0  
Uniform Statistics Collection Time: 2007-09-24 08:20:24.425057  
Uniform Statistics Status: OK  
Frequency Statistics Collection Time: 2007-09-24 08:20:24.425057  
Frequency Statistics Status: OK  
**Histogram Statistics Collection Time: null**  
**Histogram Statistics Status: missing**  
Possibly Point Skewed: No  
**Possibly Range Skewed: Yes**

**Symptom:** LIKE or BETWEEN operator. Columns in LIKE or BETWEEN operators are more likely to be skewed.

**FIELD3**

Cardinality: 8.0  
Uniform Statistics Collection Time: 2007-09-24 08:20:24.425057  
Uniform Statistics Status: OK  
**Frequency Statistics Collection Time:** null  
**Frequency Statistics Status:** missing  
Histogram Statistics Collection Time: null  
Histogram Statistics Status: missing  
**Possibly Point Skewed:** Yes  
**Symptom:** Low column cardinality relative to table cardinality. Columns with low cardinality relative to table cardinality are more likely to be skewed.  
Possibly Range Skewed: No

**FIELD2**

Cardinality: 11.0  
Uniform Statistics Collection Time: 2007-09-24 08:20:24.425057  
Uniform Statistics Status: OK  
**Frequency Statistics Collection Time:** null  
**Frequency Statistics Status:** missing  
Histogram Statistics Collection Time: null  
Histogram Statistics Status: missing  
**Possibly Point Skewed:** Yes  
**Symptom:** Low column cardinality relative to table cardinality. Columns with low cardinality relative to table cardinality are more likely to be skewed.  
Possibly Range Skewed: No

**Interesting Column Groups:**

**(FIELD2, FIELD3)**

Cardinality: -1.0  
Uniform Statistics Collection Time: null  
Uniform Statistics Status: missing  
Frequency Statistics Collection Time: null  
Frequency Statistics Status: missing  
Histogram Statistics Collection Time: null  
Histogram Statistics Status: missing

---

**Note that:**

- ▶ The DISTARRANG\_ID column appears in a BETWEEN predicate, so additional histogram statistics can improve the filter factor estimation of this predicate.
- ▶ The FIELD3 column has a low column cardinality; only eight distinct values are present. A low cardinality column is often subject to skewed data, and therefore, frequency statistics for the FIELD3 column can help the optimizer's filter factor estimate for predicates using FIELD3.
- ▶ The same applies to the FIELD2 column as only 11 distinct values exist for that column.
- ▶ Workload Statistics Advisor also recommends to gather column group statistics for the combined columns FIELD2 and FIELD3. Both columns are used to join with the CATGROUP(C1) and CATENTRY (T1) table. Because the cardinality of the individual

columns (FIELD2 and FIELD3) is low and is likely to be skewed, it is also likely that both columns are correlated. To determine the optimal join sequence, knowing the cardinality of the combined columns is useful.

This explains why Workload Statistics Advisor suggests the following RUNSTATS statement for the DISTARRANG table (Example 13-2).

*Example 13-2 RUNSTATS statement for DISTARRA*

---

```
RUNSTATS TABLESPACE SW01DB1.DISTARRA
          TABLE(TGCOMM01.DISTARRANG)
          COLUMN(FIELD2)
          COLGROUP(FIELD2,FIELD3)
          COLGROUP(FIELD3) FREQVAL COUNT 10
          COLGROUP(FIELD2) FREQVAL COUNT 10
          SORTDEVT SYSDA
          INDEX(TGCOMM01.P_DISTAR HISTOGRAM NUMCOLS 1 NUMQUANTILES 20)
SHRLEVEL CHANGE REPORT YES
```

---

The other recommendations have a similar origin and are not explained in detail here.

## 13.7 Implementing the Workload Statistics Advisor recommendations

As indicated above, you can invoke the RUNSTATS utility from within Optimization Service Center or Optimization Expert. Many installations might not allow the execution of a DB2 utility from a workstation, or want to execute the RUNSTATS during the night time. In that case, you save the recommendations in a file, import the file into a JCL file, and submit the RUNSTATS statement as a batch job, either directly or through a job scheduler.

After the RUNSTATS statements that were suggested by Workload Statistics Advisor have been run, we recommend to invoke Workload Statistics Advisor a second time to see whether, based on these additional statistics, Workload Statistics Advisor recommends to collect extra statistics. This second Workload Statistics Advisor run also allows you to verify whether the new statistics gathered in the first run did not create inconsistencies in the DB2 catalog statistics.

## 13.8 Re-measuring the workload after Workload Statistics Advisor recommendations

After rerunning Workload Statistics Advisor until no further statistics are recommended, we rerun our workload measurement to see whether the additional Workload Statistics Advisor statistics improved the performance of our workload.

**Tip:** Our workload contains dynamic SQL statement. In case you use static SQL, you must rebind the plans and packages involved to allow them to select a better access path after implementing the Workload Statistics Advisor recommended statistics.

To rerun the measurement, you need to delete and redefine the monitor profile or create a new profile. Otherwise, the data of the new run is added to the existing data and the result is

a set of numbers that includes the elapsed and CPU time of both runs, which is not what we are looking for. Our second run only includes data from the second run, not from both runs combined.

In our case, we created a new monitor called Commerce\_critical\_1, and deleted the original one.

**Note:** You *cannot* have two monitoring profiles that use the exact same monitoring criteria, not even when one is enabled and the other one is disabled.

The results of the second run, after implementing the Workload Statistics Advisor recommendations, is shown in Figure 13-23.

Execution Count	Source	Accumulated Elapsed...	Average Elapsed ...	Accumulated CPU ...	Average CPU...	Statement Text
2	MONI...	10.2071295	5.1035647	10.041289	5.0206447	SELECT DISTINCT T1.CATENT
3	MONI...	4.8935933	1.6311978	4.789024	1.5963413	SELECT B1.MEMBER_ID , B1.L
12	MONI...	2.006385	0.16719876	1.9519875	0.16266562	SELECT DISTINCT T1.PARTNU
14	MONI...	1.3561871	0.096870504	1.338278	0.09559129	SELECT T1.ENDTIME , T1.POL
11	MONI...	0.02383647	0.0021669518	0.023281867	0.0021165332	SELECT T1.CATGROUP_ID , T
5	MONI...	0.009361555	0.001872311	0.0091694305	0.0018338861	SELECT T1.NAME , T2.CATALC
8	MONI...	0.0025934374	3.2417968E-4	0.0025685586	3.2106982E-4	SELECT T1.NAME , T1.AUXDES

Figure 13-23 Run after implementation of Workload Statistics Advisor RUNSTATS

When you compare the performance of the first and second run, you can see that the performance has dramatically improved.

For example, the second query in the list (the one with three executions; this is the one that is using the DISTARRANG table), performance dropped from 20 CPU seconds to 1.6 CPU seconds. The same applies to other statements.

When we look at the access path (query block 2) of the query again after implementing the Workload Statistics Advisor recommendations (Figure 13-24 on page 365), you notice that the join order has changed from DISTARRANG → CATENTRY → CATGROUP to CATGROUP → DISTARRANG → CATENTRY.

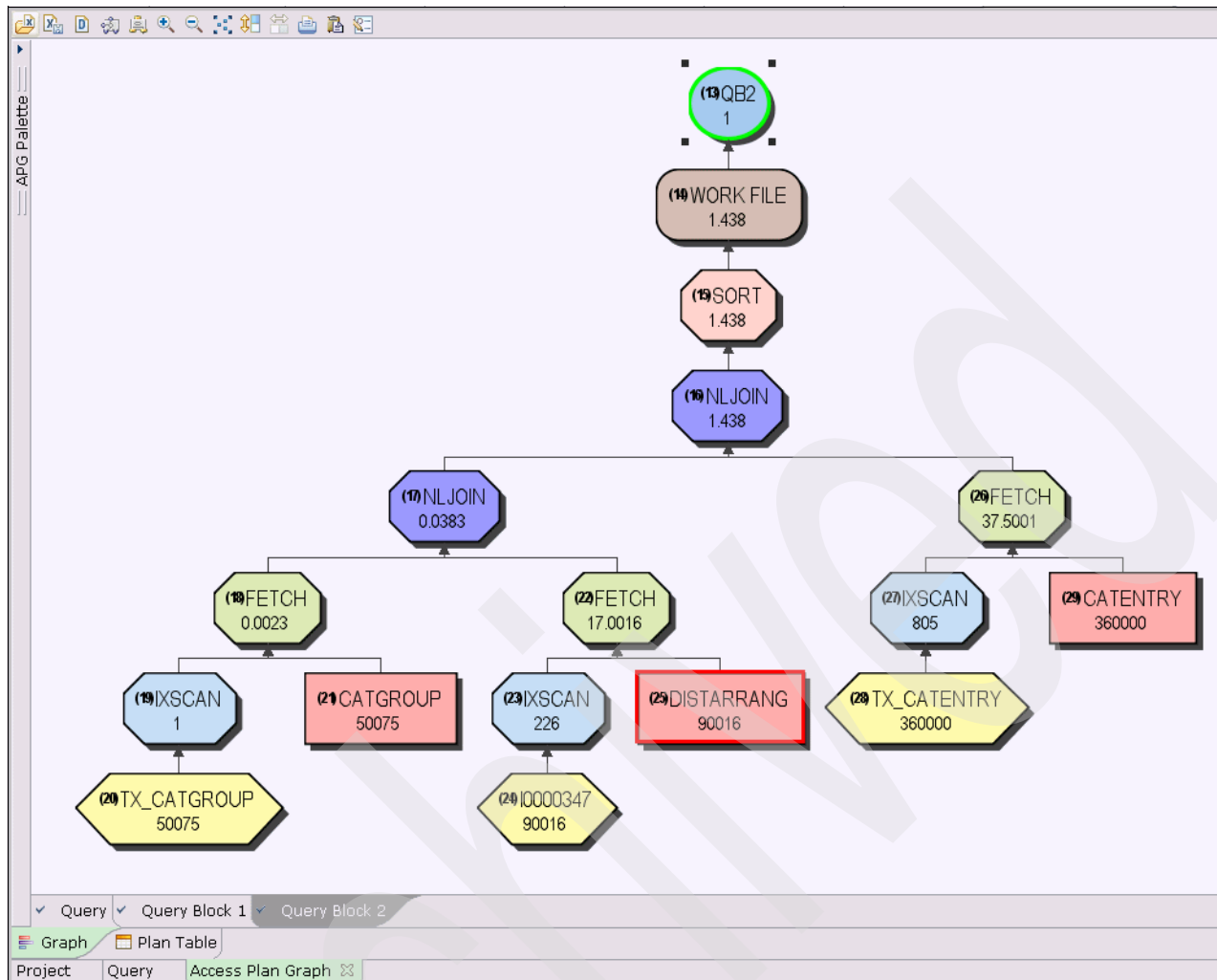


Figure 13-24 Access plan graph of Q5 after Workload Statistics Advisor

The number of qualifying rows after applying local predicates is shown in Table 13-2.

Table 13-2 Number of qualified rows for the tables of query block 2 after Workload Statistics Advisor

Table name	QUALIFIED_ROWS
CATGROUP	0.0023
DISTARRANG	17.006
CATENTRY	1.0003

The change in the join order was clearly triggered by the additional statistics, giving DB2 a better idea about the number of qualifying rows for these tables.

### 13.8.1 Using Workload Index Advisor

Because we want to push our workloads as hard as we can, we also want to have a look at whether additional indexes can help improve the performance of certain, and potentially all, queries in this workload.

To achieve this, you can analyze each individual query for opportunities to improve query performance by adding additional indexes. However, because this is a publication on Optimization Service Center and Optimization Expert, we use the Index Advisor for this purpose. Note that Index Advisor, both for an individual query and at the workload level, is only available when using the Optimization Expert product, not in Optimization Service Center.

Because we are looking to improve the performance of the entire workload, we use the Workload Index Advisor for this purpose.

To invoke the Index Advisor, open the workload that was created when the data of the second run was captured by the profile monitor. Opening the Commerce\_critical\_1 workload creates a project called DBJ27V91A\_Commerce\_critical\_1, and the Statements tab is automatically opened. Then click **Workload Tools** → **Run Workload Index Advisor**, as shown in Figure 13-25.

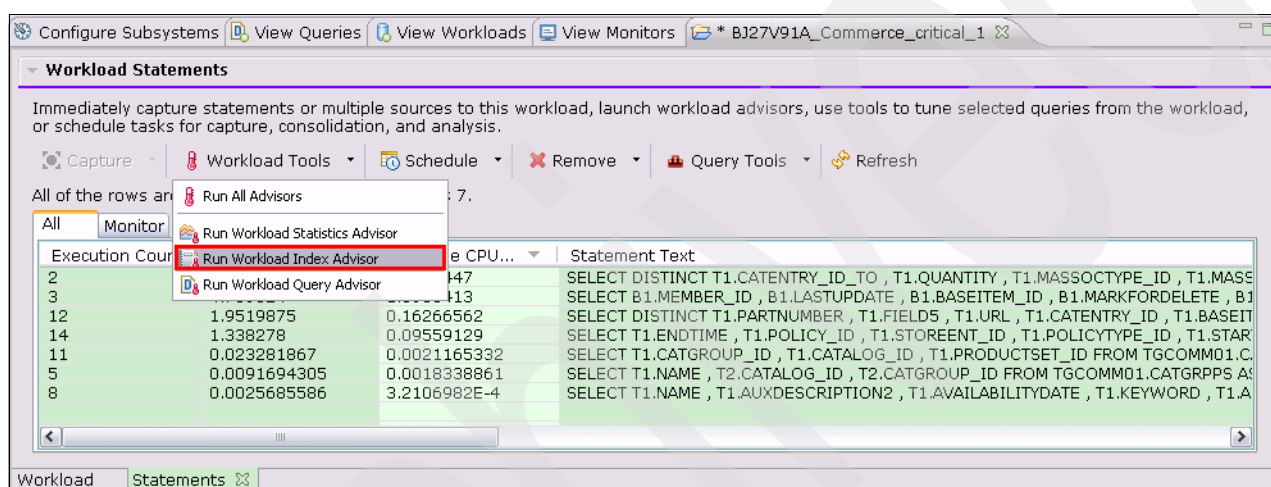


Figure 13-25 Start Workload Index Advisor

When the Index Advisor starts, you receive a similar warning when invoking Workload Statistics Advisor, indicating that “EXPLAIN has not been run for this workload” (Figure 13-16 on page 355). See section 13.6, “Running Workload Statistics Advisor” on page 354 for details.

Click **OK** twice to invoke the EXPLAIN function for the newly captured workload. Then invoke the Index Advisor again as shown in Figure 13-25. The results of the invocation is shown in Figure 13-26 on page 367.

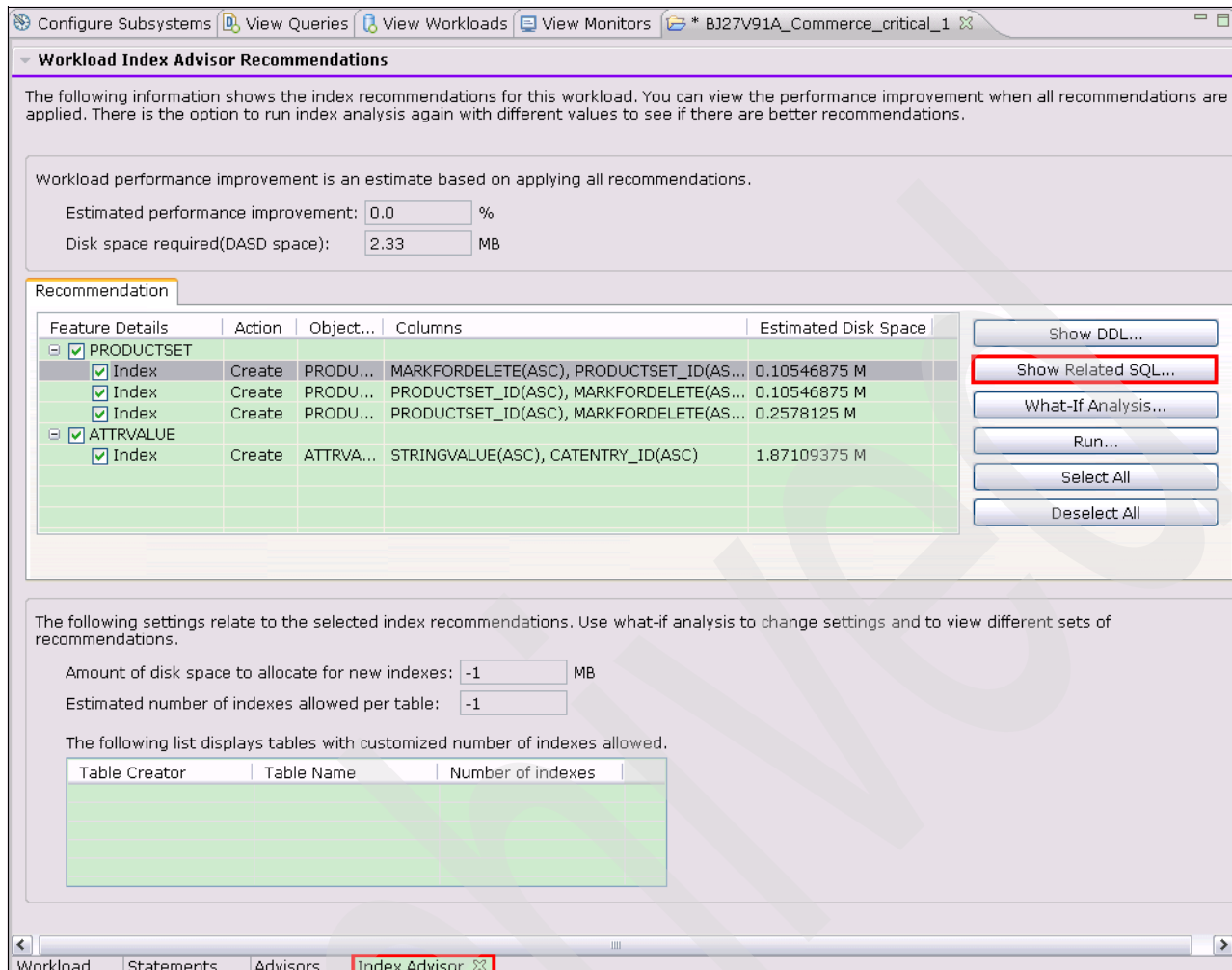


Figure 13-26 Workload Index Advisor results

The Index Advisor suggests to create additional indexes on the PRODUCTSET and ATTRVALUE table. In case you wonder which SQL statement is likely to benefit from one of the proposed indexes, select the index you are interested in, and click **Show Related SQL** (see Figure 13-26). The related SQL statement is shown in a new window (see Figure 13-27 on page 368).

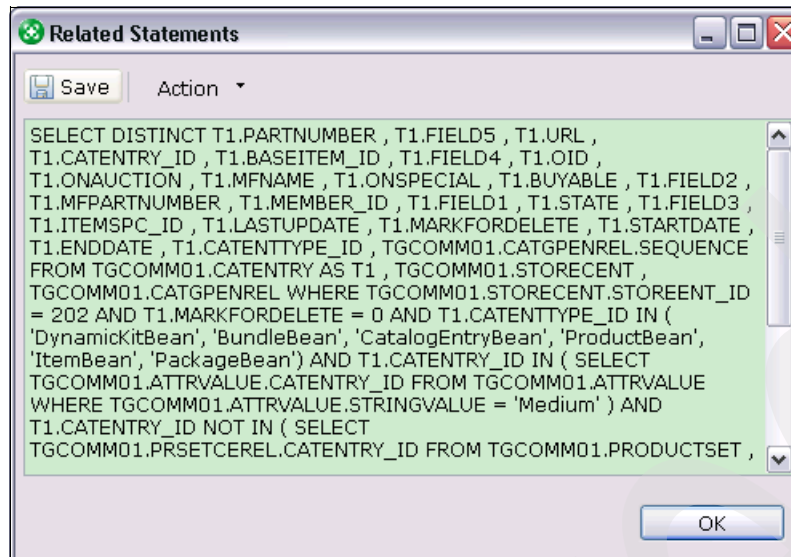


Figure 13-27 SQL statement expected to benefit from the suggested index

Now that we know the statement, look at the statement's current access path to see where the suggested index might help to improve performance.

To do so, click the **Statement** tab for the workload project, select the statement from the list, and invoke Access Plan Graph for the statement, either by right-mouse clicking **Access Plan Graph** or **Query Tools** → **Access Plan Graph**. The result is shown in Figure 13-28 on page 369.



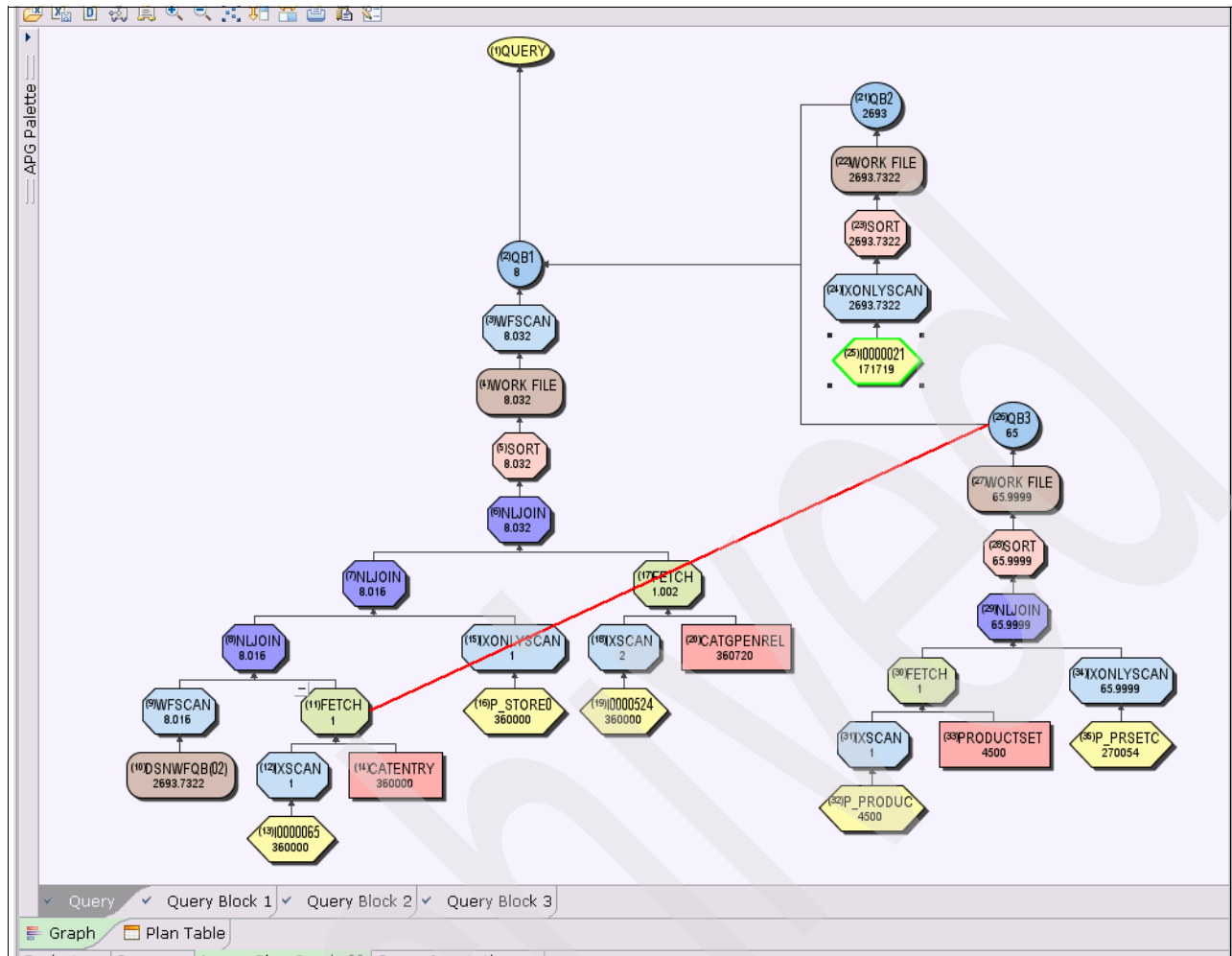


Figure 13-28 Access plan graph of Q7

Note that the I0000021 index is used via a non-matching index access according to the PLAN\_TABLE output shown in Figure 13-29.

QUERYNO	QBLO...	PLANNO	METH...	TNAME	TABNO	ACCESS...	MATCHC...	ACCESSN...	INDEXONLY	SORTN_UNIQ	SORTN_IC
16905	1	1	0	DSNWFQB(02)	7	R	0		N	N	N
16905	1	2	1	CATENTRY	1	I	2	I0000065	N	N	N
16905	1	3	1	STORECENT	2	I	2	P_STORE0	Y	N	N
16905	1	4	1	CATGPENREL	3	I	1	I0000524	N	N	N
16905	1	5	3		0		0		N	N	N
16905	2	1	0	ATTRVALUE	4	I	0	I0000021	Y	N	N
16905	2	2	3		0		0		N	N	N
16905	3	1	0	PRODUCTSET	5	I	1	P_PRODUC	N	N	N
16905	3	2	1	PRSETCEREL	6	I	1	P_PRSETC	Y	N	N
16905	3	3	3		0		0		N	N	N

Figure 13-29 PLAN\_TABLE output of Q7

When looking at the indexes proposed by the Index Advisor (see Figure 13-26 on page 367), you notice that an index on ATTRVALUE is suggested, specifically on the STRINGVALUE and CATENTRY\_ID columns.

Looking at the annotated query in Figure 13-30, you notice that the suggested index is geared towards making the subquery use more matching columns.

**Query Annotation**

Indicate which annotation to display and customize your view. Selecting a row will highlight all of the relevant rows for this table. Selecting a join predicate will highlight all of the join predicate rows in both of the joined tables. Click Reset Text to return to the original text view.

Annotation to display: All

Expand All Collapse All Customize Save Print Reset Text

Formatted Query	Annotation
FROM TGCOMM01.CATENTRY AS T1	CARDF=360,000 QUALIFIED.
, TGCOMM01.STORECENT	CARDF=360,000 QUALIFIED.
, TGCOMM01.CATGPENREL	CARDF=360,720 QUALIFIED.
WHERE ( TGCOMM01.STORECENT.STORECENT_ID = 202	COLCARDF=1
AND T1.MARKFORDELETE = 0	COLCARDF=1
AND T1.CATENTTYPE_ID IN ( 'DynamicKitBean', 'BundleBean', 'CatalogEntryBean', 'ProductBean', 'Item	COLCARDF=2
AND T1.CATENTRY_ID NOT IN ( SELECT TGCOMM01.PRSETCEREL.CATENTRY_ID	COLCARDF=360,000
FROM TGCOMM01.PRODUCTSET	CARDF=4,500 QUALIFIED.
, TGCOMM01.PRSETCEREL	CARDF=270,054 QUALIFIED.
WHERE ( TGCOMM01.PRODUCTSET.PRODUCTSET_ID = 10000002	COLCARDF=4,500
AND TGCOMM01.PRODUCTSET.MARKFORDELETE = 0	COLCARDF=1
AND TGCOMM01.PRSETCEREL.PRODUCTSET_ID = 10000002	COLCARDF=4,500
AND TGCOMM01.PRODUCTSET.PRODUCTSET_ID = TGCOMM01.PRSETCEREL	COLCARDF=4,500/4,500
)	
AND TGCOMM01.CATGPENREL.CATENTRY_ID = TGCOMM01.DSNWFQB(02).CATENTRY_ID	COLCARDF=360,000/(not ap
AND TGCOMM01.CATGPENREL.CATENTRY_ID = TGCOMM01.STORECENT.CATENTRY_ID	COLCARDF=360,000/360,00
AND TGCOMM01.STORECENT.CATENTRY_ID = TGCOMM01.DSNWFQB(02).CATENTRY_ID	COLCARDF=360,000/(not ap
AND T1.CATENTRY_ID = TGCOMM01.CATGPENREL.CATENTRY_ID	COLCARDF=360,000/360,00
AND T1.CATENTRY_ID = TGCOMM01.DSNWFQB(02).CATENTRY_ID	COLCARDF=360,000/(not ap
DB2 creates a virtual table, SYSADM.DSNWFQB(02), to process the following non-correlated subquery	
SELECT TGCOMM01.ATTRVALUE.CATENTRY_ID	
FROM TGCOMM01.ATTRVALUE	CARDF=171,719 QUALIFIED.
WHERE TGCOMM01.ATTRVALUE.STRINGVALUE = 'Medium'	COLCARDF=25
End of the subquery for the SYSADM.DSNWFQB(02) virtual table.	
AND TGCOMM01.STORECENT.CATENTRY_ID = T1.CATENTRY_ID	COLCARDF=360,000/360,00
)	
ORDER BY 24 ASC	
FOR READ ONLY	

Project Query Access Plan Graph Query Annotation

Figure 13-30 Annotated version of the Q7 query

Which indexes are generated for a workload not only depends on the queries themselves, but also on the setting of the different Index Advisor options. To look at, or change them, you can either use the **Advisors** tab at the bottom of the workload project and clicking **Option** next to Index Advisor, or use **Tools** → **Preferences** → **expand the Tune Workload entry** → **Workload Index Advisor** (see Figure 13-31 on page 371).

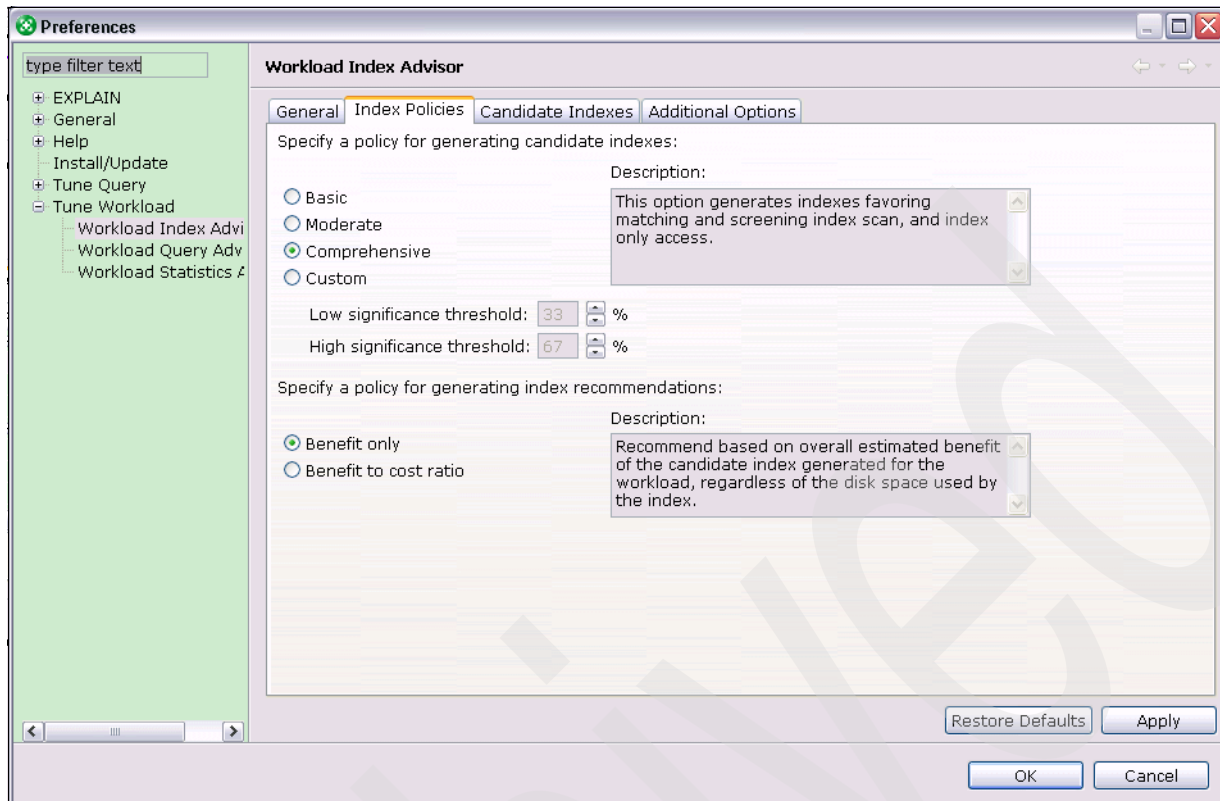


Figure 13-31 Workload Index Advisor Index options - Index policies

In this scenario, we use the Comprehensive policy, allowing the Index Advisor to look for indexes that allow index matching, index screening, and index only access.

In case the suggested indexes look incorrect, or you find that too many indexes are suggested, using these options allow you to set boundaries or indicate which types of indexes you prefer.

After returning to the list of suggested indexes by the Index Advisor (Figure 13-26 on page 367), click **Show DDL** to look at the DDL for the suggested indexes. The result is shown in Figure 13-32 on page 372.

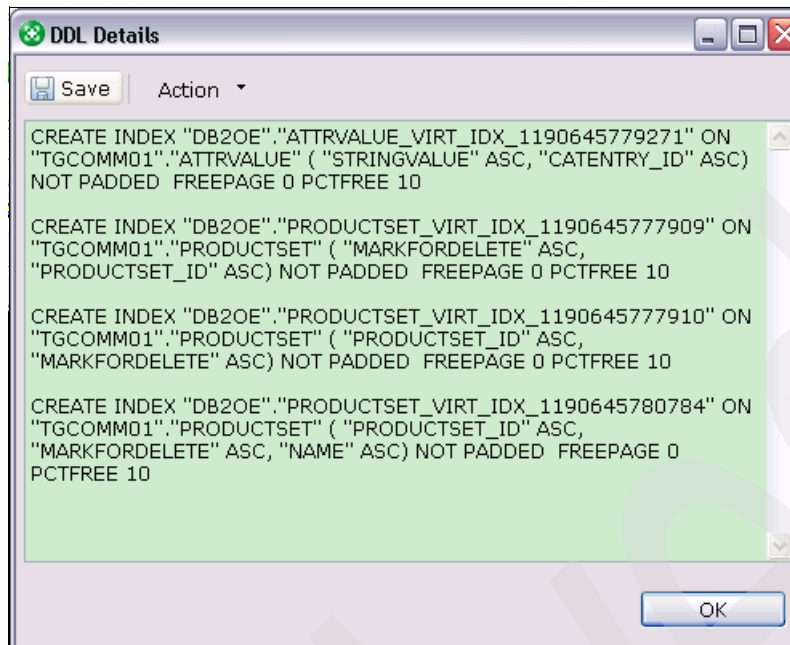


Figure 13-32 CREATE INDEX DDL from Workload Index Advisor

## 13.9 Implementing the indexes suggested by Workload Index Advisor

When you are satisfied with the indexes generated by the Index Advisor, implement the recommendations.

### 13.9.1 Creating the indexes

The CREATE INDEX DDL generated by the Index Advisor does not allow you to create the index with the DEFER YES option. It does not allow you to specify the buffer pool in which the indexes are created, to provide space allocations, or to change the index name. Therefore, we use **Save** to save the DDL in a file and import it into an MVS data set to edit, and then submit it.

**Tip:** We suggest that you create the indexes using the DEFER YES option and invoke the REBUILD INDEX utility to populate the indexes. REBUILD INDEX is more efficient for large indexes.

Currently, the Index Advisor provides DDL, where all recommended indexes are allocated to the default buffer pool (BP0). We strongly suggest that you specify the BUFFERPOOL option to allocate a specific and dedicated buffer pool for the indexes.

### 13.9.2 Gathering statistics for the newly created indexes

Now that the indexes are created, gather relevant statistics for the indexes so that the optimizer can effectively exploit the indexes.

When you build the indexes with REBUILD INDEX as suggested, you can use INLINE statistics. In that case, statistics are gathered during the utility run, so you save an extra scan of the index compared to running RUNSTATS later. However, you cannot gather histogram statistics by the INLINE statistics process. A separate RUNSTATS INDEX is required in case histogram statistics are important.

An alternative is to invoke Workload Statistics Advisor again for the workload after creating the indexes that were recommended by the Index Advisor. Workload Statistics Advisor recognizes that these newly created indexes are missing catalog statistics and suggests the RUNSTATS statement to collect them. You can use the same process as we used in section 13.6, “Running Workload Statistics Advisor” on page 354 to collect and implement these statistics.

## 13.10 Re-measuring the workload after Workload Index Advisor recommendations

Now that we have implemented all the suggested Workload Statistics Advisor statistics and Workload Index Advisor indexes, we are ready to re-measure our workload.

**Tip:** Our workload contains dynamic SQL statements. In case you use static SQL, you must rebind the plans and packages involved. This allows them to select a better access path after implementing the Index Advisor recommended indexes and gathering RUNSTATS information for them.

As before, we create a new monitor profile for our measurements, called Commerce\_critical\_2 (and delete the old one). The results of the third run, after implementing both the Workload Statistics Advisor and Workload Index Advisor recommendations, is shown in Figure 13-33.

Execution Count	Source	Accumulated Elapsed...	Average Elapsed ...	Accumulated CPU...	Average CPU...	Statement Text
2	MONITOR	10.168441	5.0842204	9.990055	4.9950275	SELECT DISTINCT T1.CATE
3	MONITOR	4.9170833	1.6390277	4.8146396	1.6048799	SELECT B1.MEMBER_ID , B
14	MONITOR	1.3624089	0.09731492	1.3431723	0.09594088	SELECT T1.ENDTIME , T1.P
12	MONITOR	0.24389805	0.020324837	0.237565	0.019797083	SELECT DISTINCT T1.PART
5	MONITOR	0.012779904	0.0025559808	0.012575832	0.0025151665	SELECT T1.NAME , T2.CAT
11	MONITOR	0.027881233	0.0025346575	0.02755241	0.0025047644	SELECT T1.CATGROUP_ID
8	MONITOR	0.0024316248	3.039531E-4	0.002402449	3.0030613E-4	SELECT T1.NAME , T1.AUX

Figure 13-33 Workload Index Advisor results

For the statement in our workload that is executed 12 times, which is the statement that can benefit from additional indexes, the average CPU time per execution dropped from 0.162 to 0.0197 CPU seconds, an eight times CPU reduction.

The new access path, after implementing the new indexes recommended by the Index Advisor, is shown in Figure 13-34 on page 374. Note that two virtual indexes (newly created

indexes) are used by this query, which are the major contributors to the CPU time reduction of the query.

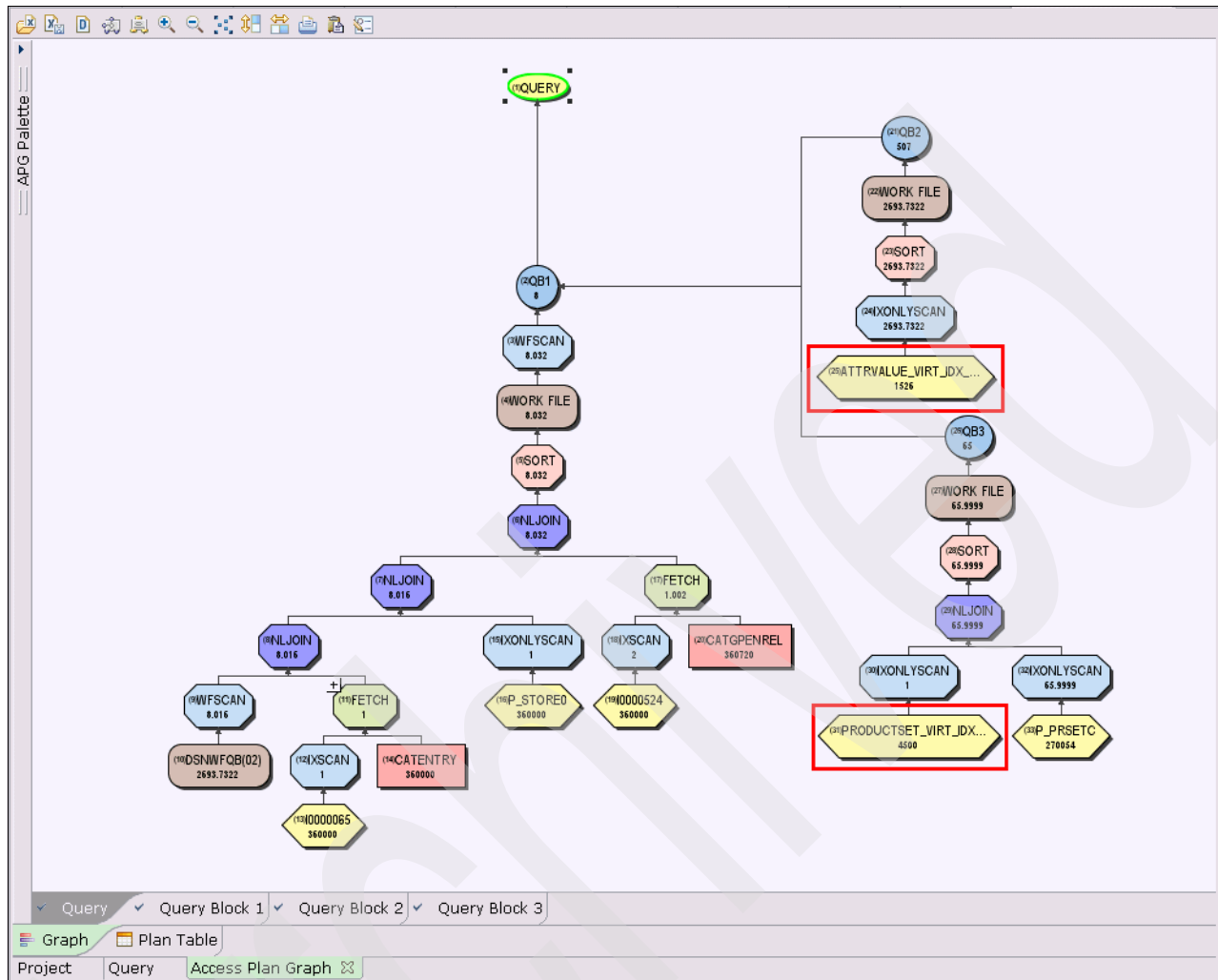


Figure 13-34 Access plan graph of Q7 after Workload Index Advisor run

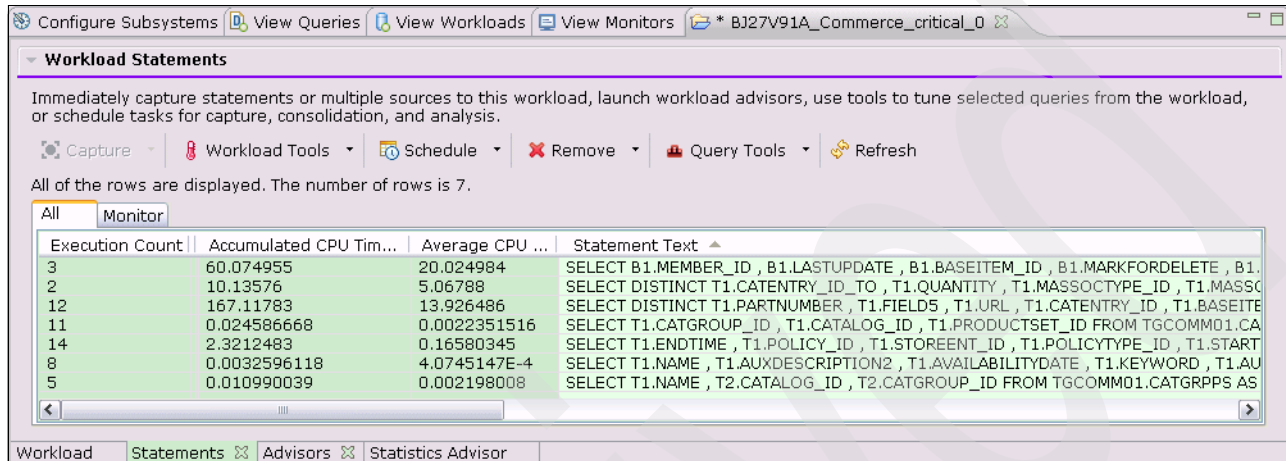
Looking at just the CPU time of a *single* execution of this statement and before implementing additional indexes, you might decide not to investigate further to improve its performance. 0.162 seconds is acceptable, considering the complexity of the statement. However, you also need to consider the number of executions and the total CPU time consumed by this statement; in other words, its weight in the entire profile.

**Tip:** You need to not only focus on the average execution time of an individual statement, but also take the number of executions into account. A relatively inexpensive SQL statement that is executed a million times can still consume a considerable amount of CPU.

## 13.11 Comparing all three workload runs

To summarize the evolution of the performance of our workload, we provide a screen shot after each run. The statements are sorted on the SQL statement text. This makes it easier to compare the different runs because an individual statement is on the same row in each run.

Figure 13-35 shows the initial run.

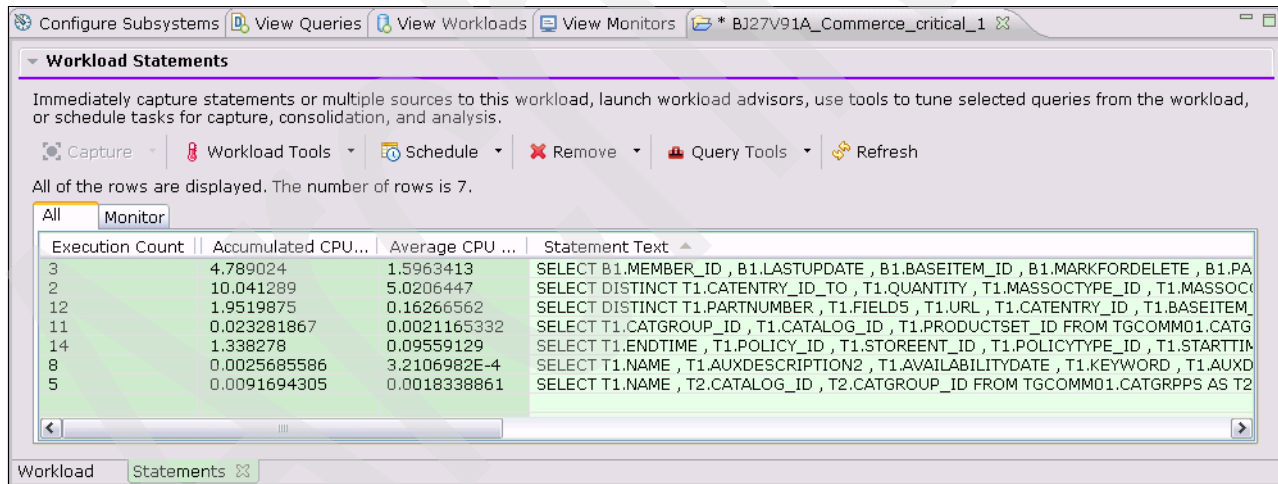


The screenshot shows the 'Workload Statements' window in a database management tool. The window has a menu bar with 'Configure Subsystems', 'View Queries', 'View Workloads', 'View Monitors', and a tab for 'BJ27V91A\_Commerce\_critical\_0'. Below the menu bar, there's a section for 'Workload Statements' with a description: 'Immediately capture statements or multiple sources to this workload, launch workload advisors, use tools to tune selected queries from the workload, or schedule tasks for capture, consolidation, and analysis.' Below this, there are buttons for 'Capture', 'Workload Tools', 'Schedule', 'Remove', 'Query Tools', and 'Refresh'. A status bar indicates 'All of the rows are displayed. The number of rows is 7.' The main table has columns: 'Execution Count', 'Accumulated CPU Tim...', 'Average CPU ...', and 'Statement Text'. The table contains 7 rows of data, with the first row having an execution count of 3 and an accumulated CPU time of 60.074955. The statements are sorted by their SQL text.

Execution Count	Accumulated CPU Tim...	Average CPU ...	Statement Text
3	60.074955	20.024984	SELECT B1.MEMBER_ID , B1.LASTUPDATE , B1.BASEITEM_ID , B1.MARKFORDELETE , B1.
2	10.13576	5.06788	SELECT DISTINCT T1.CATENTRY_ID_TO , T1.QUANTITY , T1.MASSOCTYPE_ID , T1.MASSC
12	167.11783	13.926486	SELECT DISTINCT T1.PARTNUMBER , T1.FIELD5 , T1.URL , T1.CATENTRY_ID , T1.BASEITE
11	0.024586668	0.0022351516	SELECT T1.CATGROUP_ID , T1.CATALOG_ID , T1.PRODUCTSET_ID FROM TGCOMM01.CA
14	2.3212483	0.16580345	SELECT T1.ENDTIME , T1.POLICY_ID , T1.STOREENT_ID , T1.POLICYTYPE_ID , T1.START
8	0.0032596118	4.0745147E-4	SELECT T1.NAME , T1.AUXDESCRIPTION2 , T1.AVAILABILITYDATE , T1.KEYWORD , T1.AU
5	0.010990039	0.002198008	SELECT T1.NAME , T2.CATALOG_ID , T2.CATGROUP_ID FROM TGCOMM01.CATGRPPS AS

Figure 13-35 Initial run with statements sorted by SQL statement text

Figure 13-36 shows the SQL statements after running Workload Statistics Advisor.



The screenshot shows the 'Workload Statements' window in a database management tool, similar to the previous one but with a different tab 'BJ27V91A\_Commerce\_critical\_1'. The table contains 7 rows of data, with the first row having an execution count of 3 and an accumulated CPU time of 4.789024. The statements are sorted by their SQL text.

Execution Count	Accumulated CPU Tim...	Average CPU ...	Statement Text
3	4.789024	1.5963413	SELECT B1.MEMBER_ID , B1.LASTUPDATE , B1.BASEITEM_ID , B1.MARKFORDELETE , B1.PA
2	10.041289	5.0206447	SELECT DISTINCT T1.CATENTRY_ID_TO , T1.QUANTITY , T1.MASSOCTYPE_ID , T1.MASSOC
12	1.9519875	0.16266562	SELECT DISTINCT T1.PARTNUMBER , T1.FIELD5 , T1.URL , T1.CATENTRY_ID , T1.BASEITEM
11	0.023281867	0.0021165332	SELECT T1.CATGROUP_ID , T1.CATALOG_ID , T1.PRODUCTSET_ID FROM TGCOMM01.CATG
14	1.338278	0.09559129	SELECT T1.ENDTIME , T1.POLICY_ID , T1.STOREENT_ID , T1.POLICYTYPE_ID , T1.STARTTIM
8	0.0025685586	3.2106982E-4	SELECT T1.NAME , T1.AUXDESCRIPTION2 , T1.AVAILABILITYDATE , T1.KEYWORD , T1.AUXD
5	0.0091694305	0.0018338861	SELECT T1.NAME , T2.CATALOG_ID , T2.CATGROUP_ID FROM TGCOMM01.CATGRPPS AS T2

Figure 13-36 Run after Workload Statistics Advisor with statements sorted by SQL statement text

Figure 13-37 shows the SQL statements after running Workload Index Advisor.

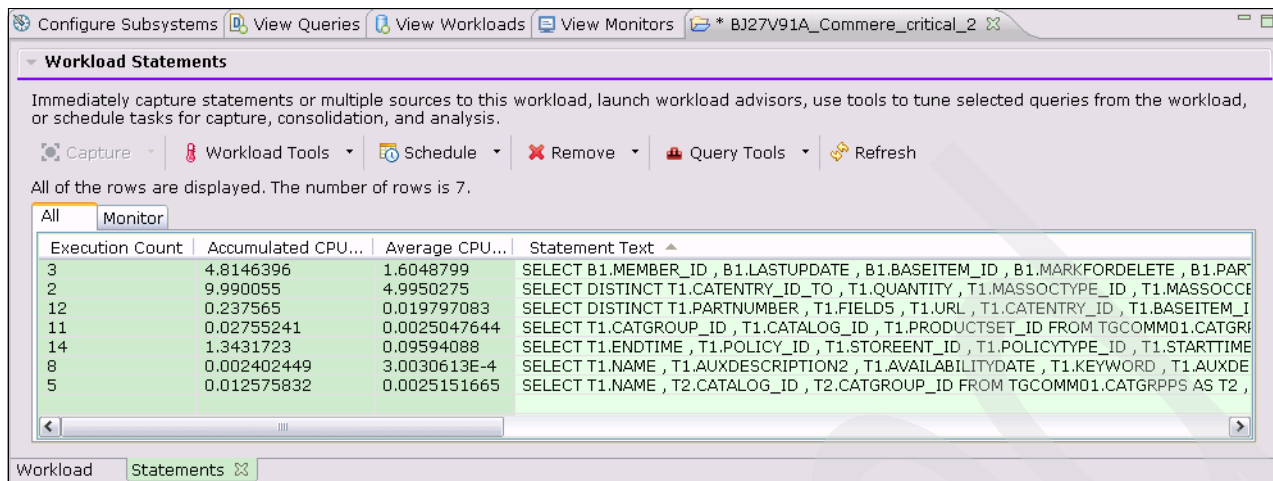


Figure 13-37 Workload Index Advisor results with statements sorted by SQL statement text

The chart in Figure 13-38 shows the initial total CPU for the entire workload, the total CPU after the Workload Statistics Advisor contribution, and the CPU after both Workload Statistics Advisor and Workload Index Advisor contributions to reduction. It clearly indicates that the CPU percentage reductions are obtainable with Optimization Service Center or Optimization Expert advisory functions and where the most benefits are gained for this specific CPU-bound workload.

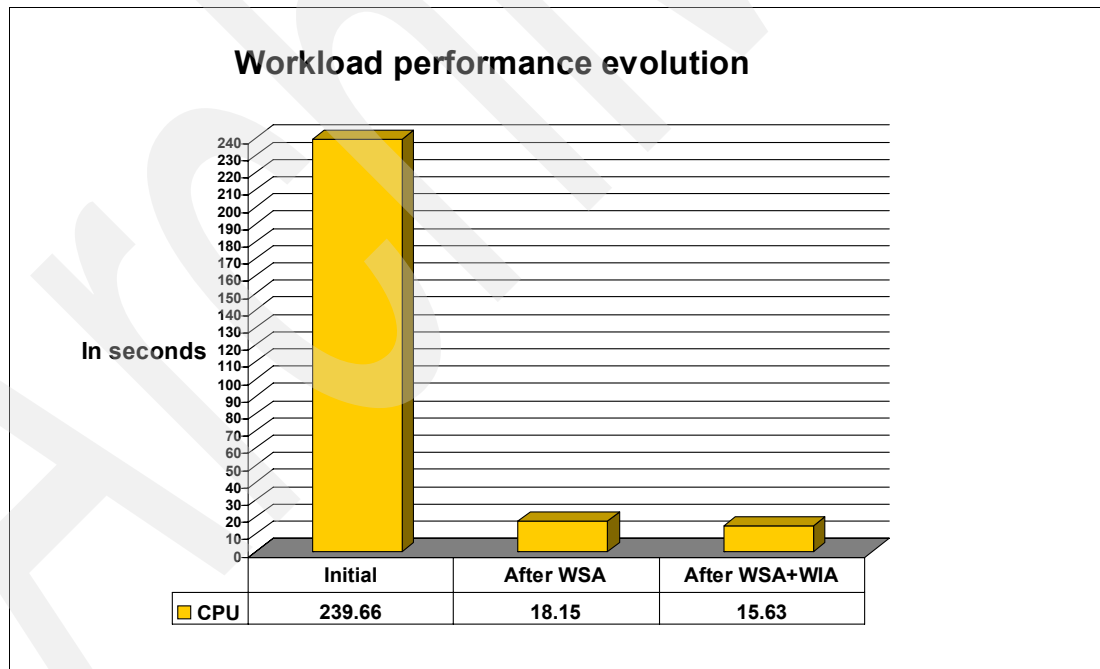


Figure 13-38 Workload performance evolution

## 13.12 Conclusion

For this relatively small workload, we have demonstrated the powerful features of Optimization Service Center and Optimization Expert. These features not only allow you to



quickly analyze the access path of a query that is not performing adequately, but they also allow you to improve the performance of the workload by using both the Workload Statistics Advisor and Workload Index Advisor.

As we have seen, these tools are easy to use and allow you to analyze and improve the performance of a query workload in no time. This allows more time for other tasks, such as investigating more complex access paths in details and queries which performance cannot easily enhance by adding an index or adding statistics information.

Archived

## Common practices

This chapter describes practical usage scenarios, tips and techniques, and common practices using Optimization Service Center or Optimization Expert in various phases of an application's life cycle. The first part of this chapter deals with identifying problem queries in each phase. When the problem query is identified, the second part provides step-by-step guidance for tuning this query, with considerations for both individual execution and workload environment.

This chapter contains the following topics:

- ▶ “Using Optimization Service Center or Optimization Expert in application's life cycle” on page 380:
  - “Usage during application design and development” on page 380
  - “Usage during testing or quality assurance” on page 381
  - “Usage during production phase” on page 381
- ▶ “Common practice for tuning steps” on page 384:
  - “Tuning a single query” on page 384
  - “Tuning a query workload” on page 388

## 14.1 Using Optimization Service Center or Optimization Expert in application's life cycle

Optimization Service Center or Optimization Expert provides flexibility for use in various phases in the application's life cycle. The relevant phases are design and development, testing or quality assurance, and production. Each phase has specific tools and advisors that fit well with the characteristics of the phase.

The main objectives of this section are two fold, which are problem prevention and identification:

- ▶ Prevent occurrence of problem queries by providing assistance to write them well in the first place.
- ▶ Identify problem queries in various phases of the application's life cycle.

### 14.1.1 Usage during application design and development

At the application design and development phase, the goal is to specify a well-formulated, correct, and efficient query. They are:

- ▶ **Correctness:**

Is the query written correctly to return the expected results? You can actually use Optimization Service Center or Optimization Expert to run the query and look at the result. For example, after rewriting some of the predicates, you want to make sure the query still returns the same results.

- ▶ **Efficiency:**

Does it use efficient expressions? Does it include the necessary predicates: screening, matching, join, and other predicates to enable the query to run efficiently? Can you rewrite the query into a more simple form?

Basically, all single Query Advisors and tools are applicable. However, the most appropriate features to use are:

- ▶ *Query Advisor and Access Path Advisor*: Run the Query Advisor to see if the query can be re-written into a more efficient form. Access Path Advisor checks if the generated access path shows any sign of potential poor performance. Apply Query Advisor's recommendation to add predicates or to rewrite the query, if any.

See Chapter 11, "Access Path Advisor and Query Advisor" on page 283 for the description of the Access Path Advisor and Query Advisor.

- ▶ *Query Annotation tool*: Run the query through the Query Annotation tool. Analyze the annotated query. Compare the original and the transformed query to see if you can make them more efficient. Observe the transformed query and look for rewrites done by the transformation step, for example, additional predicates.

See 8.3, "Query Annotation tool" on page 180.

- ▶ *Access Plan Graph tool*: Explain the query and run it through Access Plan Graph tool. Analyze the access plan to see if it makes sense. Observe if the query uses the most efficient access path available. Use this tool to measure the effectiveness of the Query Advisor's recommendations before and after applying them. See section 8.2, "Access Plan Graph" on page 171.

- ▶ *Index Advisor*: You can run the query through Index Advisor too, to see if additional indexes can help improve the query's performance. See Chapter 10, "Index Advisor" on page 243. However, you might consider postponing this step into a later stage or apply the Workload Index Advisor instead, also at a later stage.

Because the focus is having a correct and efficient query, it might be too early to run Workload Advisors and tools, especially if not all components are completely written yet. However, toward the end of the development process, when all components are completed and the workloads are defined, we can run the workload through the Workload Statistics Advisor to generate the RUNSTATS command. This includes a follow on to run Workload Index Advisor to get recommendations on additional indexes required by the workload. You do or repeat these two steps during the quality assurance phase. Therefore, they are described in details in the next section.

Lastly, it is not useful to run Statistics Advisor at this phase because the database, most likely, is not populated with substantial and meaningful data.

### 14.1.2 Usage during testing or quality assurance

In this phase, it is assumed that unit testing and levels of functional testing have been done with success. Each individual query has been verified to generate correct results and performs reasonably well. It is now the right time to apply the following features:

- ▶ *RUNSTATS utility*: Assuming that the database has been populated with data distribution resembling the actual production data, we need to run RUNSTATS for the first time to collect statistics.
- ▶ *Statistics Advisor and Workload Statistics Advisor*: It is easier to run the Statistics Advisor to generate the appropriate RUNSTATS commands, instead of generating it manually as described in the previous point. Because all the application has been written, we can define a workload to capture statements in this application and run this workload through Workload Statistics Advisor. This approach is probably faster when compared to the alternative of running each individual query through the Statistics Advisor.
- ▶ *Workload Index Advisor*: Run the above workload through the Workload Index Advisor to check whether this workload can benefit from creating additional indexes. If you ran the Index Advisor for each query in the design and development phase, you might notice that indexes are similar, as illustrated in section 14.2.2, "Tuning a query workload" on page 388. In this case, you have the opportunity to decide which index to choose.
- ▶ *Verify your index design against the Workload Index Advisor recommendations*: This is an optional step. The purpose is to verify your index design compared to the Workload Index Advisor's recommendations. First, record the existing indexes you have for the workload. Second, delete all indexes, run this workload through Workload Index Advisor, and then compare your index list with the recommendations. Depending on the outcome of your analysis, you might either apply the recommendations or restore the deleted indexes.
- ▶ *Rebind static queries*: This step is necessary to take advantage of the new statistics and additional indexes.

### 14.1.3 Usage during production phase

Assuming that we have done our due diligence in the development and test phases, we have a well-tested application with correct queries performing at an acceptable level. However, this is not always the case. The production system can go wrong in different places:

- ▶ Data distributions and characteristics in the production system are totally different from the test system. Consequently, the statistics are inaccurate causing the workload to not perform well.
- ▶ The production system evolves as more rows are added, deleted, and updated. New tables are created and altered. At some point, the statistics become stale, causing performance degradation.
- ▶ More users are added into the system, generating more queries.
- ▶ More applications are introduced into the system. This might require the DBA to define different mixture of workloads.
- ▶ Unexpected exceptions occur, which might cause outages.
- ▶ More data is fed into the application because of a merger or acquisition.
- ▶ A business area has enabled a function that was dormant, thus pumping increased volume of data into the application.

The first and essential recommendation to overcome the above potential problems is to update the statistics periodically, especially after the database has undergone substantial changes. The next recommendation is to use the Profile Monitor to monitor query executions in the production runs. You can use the “complete RUNSTATS” from the Workload Statistics Advisor for this purpose, or have Workload Statistics Advisor periodically regenerate “complete RUNSTATS”.

The query monitoring is done in two modes: *reactive* and *proactive* modes, as described in the following section.

### Reactive mode of query monitoring

Monitor query executions for exceptional conditions, and react to resolve the problem as it occurs. For example, monitor CPUTIME exception, which is when the query execution CPU time exceeds the specified absolute threshold. Another exception to monitor is the SPIKE exception; that is, when the query execution times goes higher or lower than a specified percentage threshold over an average execution time.

To monitor for exceptions, you need to define monitor profile for the application, and specify the filter or scope for exception monitoring. Specify a reasonable threshold to capture misbehaving queries. For OLTP applications with sub-seconds queries, a couple seconds threshold is a reasonable value to begin with. The steps for setting up and running monitor profiles for detecting exceptions are described in Chapter 6, “Profile monitoring” on page 131 and section 13.1, “The selected workload” on page 344 through section 13.5, “Analyzing the results” on page 353.

When the problem queries are identified, proceed to the steps described in section 14.2, “Common practice for tuning steps” on page 384 to tune them.

### Proactive mode of query monitoring

Monitoring is done in a proactive way on a database without any outstanding problems as a periodic health check or tuning for growth. In addition to running RUNSTATS periodically to update statistics, we recommend this health check to determine if the application has developed performance degradations because of updates and other changes. As the database evolved, the performance of queries might have regressed into sub-optimal level undetected, which consequently increases the possibility of application outages.

The recommended practice is to run Normal Monitor to gather the baseline performance of selected applications or query portfolios. Repeat this run periodically, comparing the result with the previous run to see if there are significant performance regressions. In this run, it is

important to gather enough information in the report. For example, the access path, which is essential for problem analysis later on.

In case of migration, a baseline performance data is extremely useful to facilitate problem determination when performance regressions occur after a database migration to a new release of DB2 for z/OS.

When the regressed queries are identified, the one with significant performance regressions is subjected to analysis and tuning procedures outlined in section 14.2, “Common practice for tuning steps” on page 384.

## Performance consideration

The Profile Monitor incurs overheads on query executions. However, compared to other monitor tools or products, the Optimization Service Center or Optimization Expert Profile Monitor incurs less overhead for the following reasons:

- ▶ Monitoring is limited to queries within the scope specified when defining the monitor profile, as opposed to monitoring all queries in the system. In other words, the scope provides a filtering mechanism; only queries that fall within the scope are monitored. Recall from section 6.1.1, “Monitor profiles” on page 132 that a scope is defined to cover plan name, collection, and package; or authorization ID and IP address.
- ▶ The Profile Monitor provides options to tailor the desirable levels of data collection and reporting required by users. In our experience, the overhead is a function of the amount of data collection and reporting requested by users, so minimal request for data collection yields less overhead.

The performance graph for the Profile Monitor, as compared to IFCID 0318, is shown in Figure 14-1 on page 384. There are several important points to highlight in this graph:

- ▶ Monitoring with base reporting incurs negligible overhead, which is less than 1% of CPU time. With this data, we can conclude that the performance overhead of monitoring with base reporting is insignificant, such that it is affordable to turn on the monitor for exception for a long period of time. This is done for diagnosing and tuning purposes.
- ▶ The overhead for minimum reporting is a bit higher than base reporting, but it is still under 1%. This value is still insignificant, thus affordable.
- ▶ The overhead for medium and full or complete reporting are relatively higher, at < 5% and around 9% CPU time, respectively. Therefore, use them sparingly.
- ▶ The overhead for monitoring CPUTIME exception is about the same as the SPIKE exception.
- ▶ The overhead of complete reporting for monitor exceptions is higher than IFCID 0318 because it collects more statistics than the one reported by IFCID 0318.
- ▶ The overhead of minimum and medium reporting for normal monitoring is not shown in the graph, but they are expected to be within 0.58 - 0.95% and 4.89 - 5.04% range, respectively. The overhead of complete reporting for normal monitoring is also expected to be within the 4.89 - 5.04% range.

Note that there are works in progress to improve this performance, both in optimizing the code and in the use of the zIP processor.

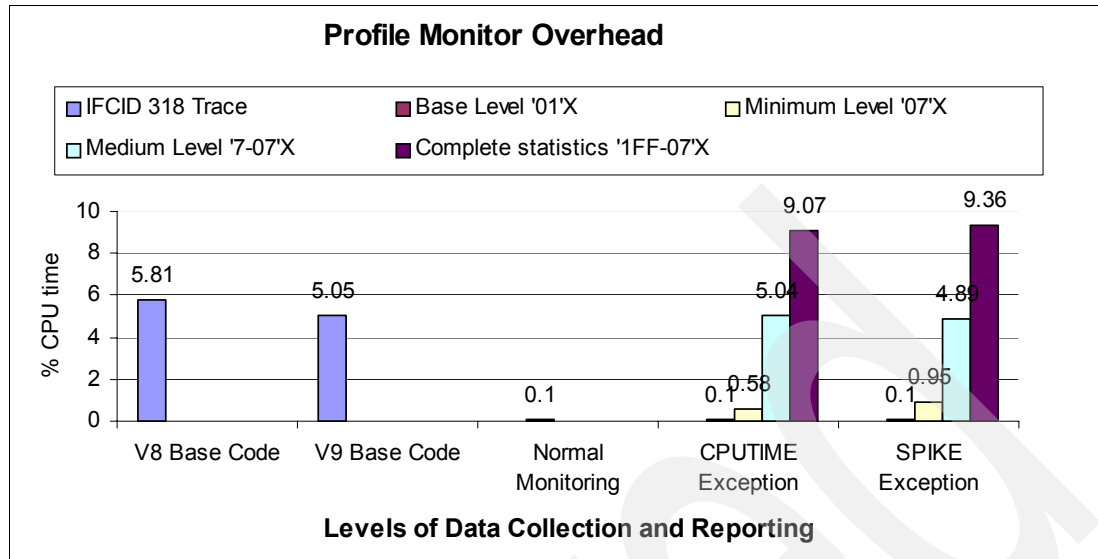


Figure 14-1 Profile monitor overhead

**Note:** Currently, only base and complete reporting is enabled by Optimization Service Center or Optimization Expert. Minimum and medium levels are only available at the server side, but they are not enabled in the Optimization Service Center or Optimization Expert client yet.

## 14.2 Common practice for tuning steps

This section describes general recommendations and common practices for analyzing and tuning problem queries when they are identified. Besides providing guidelines to tune queries, these steps might help you to gain a deeper understanding on how to write an efficient query, how to create an optimal index design, and how the query optimizer works in general. The description is divided into two sections: recommendations for tuning a single query and for tuning a query workload.

### 14.2.1 Tuning a single query

The procedure to resolve a problem query vary from case to case, depending on the type of problem, the information, and resources available at hand. Assuming that you have the query text and access to the server, you can use Optimization Service Center or Optimization Expert directly to tune the query by following the common steps described below. In a production system, where access or problem debugging is prohibitive, you might need to gather service information using Service SQL function to collect relevant information to be reproduced in another accessible system. See section 8.5, “Gather service information” on page 198 for the description of Service SQL features.

It is a common practice to follow the steps below for analyzing and tuning a problem query. Because these steps have been described in detail in other parts of this book, they are not described in detail, but references are provided. You might need to follow all of these steps. If you have good insight into the problem, consider taking a shortcut to use the specific tool or



Advisor to best tune the query. In general, the problem might be resolved at any of the following steps before completing all of them.

The steps are:

1. Explain the query:

This step is pre-required by other Advisors and tools, so do it first.

Follow the directions outlined in section 8.1.4, “Explaining an SQL statement” on page 168 to perform this task. Depending on the situation, you have several choices for inputting the query text, entering the text directly using cut and paste, from dynamic statement cache, and so forth.

2. Run the Query Advisor: Follow the direction outlined in section 11.3, “Query Advisor” on page 300 to perform this task.

The purpose is to see if there is any recommendation to re-write the query into a more efficient form. For example, missing a primary key to foreign key join predicate, or missing predicates in general. Apply the recommendations. Display the access plan graph by using the Access Plan Graph tools before and after applying the recommendations to observe any performance improvements.

3. Run the Query Annotation tool: Follow directions described in section 8.3, “Query Annotation tool” on page 180 to run the Query Annotation. Analyze the results. Observe if there any missing statistics, for example, frequency statistics, column group cardinality, column correlations, and so on. Confirm the results with the recommendations from the Statistics Advisors below. Figure 13-21 on page 359 and Figure 13-30 on page 370 show examples of an annotated query.

4. Run the Statistics Advisor: This detects any missing, obsolete, or conflicting statistics. Compare the recommendations with the results from the Query Annotation above for confirmation. Apply the recommendations prior to running the Index Advisor below. Production sites usually have established policies to only allow running RUNSTATS at a specific time slots; adhere to this policy, if any.

Follow directions in Chapter 9, “Statistics Advisor” on page 223 to perform this task.

Figure 14-2 on page 386 shows an annotated query in which the maximum frequency statistics of some columns are missing. Check to see if such conditions are detected by the Statistics Advisor and the appropriate RUNSTATS recommendations are given.

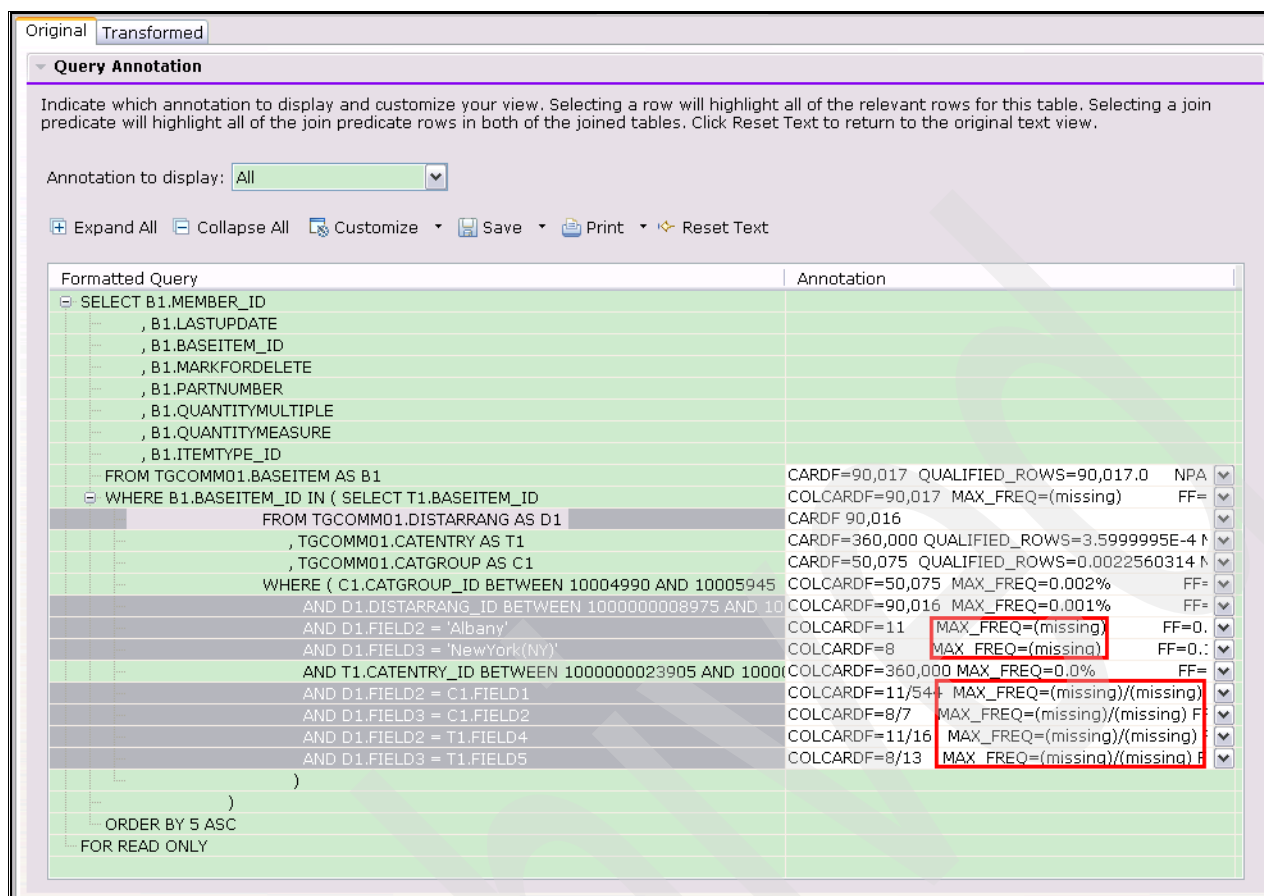


Figure 14-2 Missing maximum column frequency statistics

- Run the Index Advisor: The Index Advisor pre-requires the Statistic Advisor to be run, because it is not able to generate good index recommendations without good statistics. Follow directions in Chapter 10, "Index Advisor" on page 243 to perform this task. Evaluate the recommendations and consider implementing them. As we all know, missing important indexes might cause poor performance. Follow the local site procedure for implementing the recommendations for creating new indexes.

Figure 14-3 shows that the Index Advisor recommends two indexes with suggested columns composition and ordering.

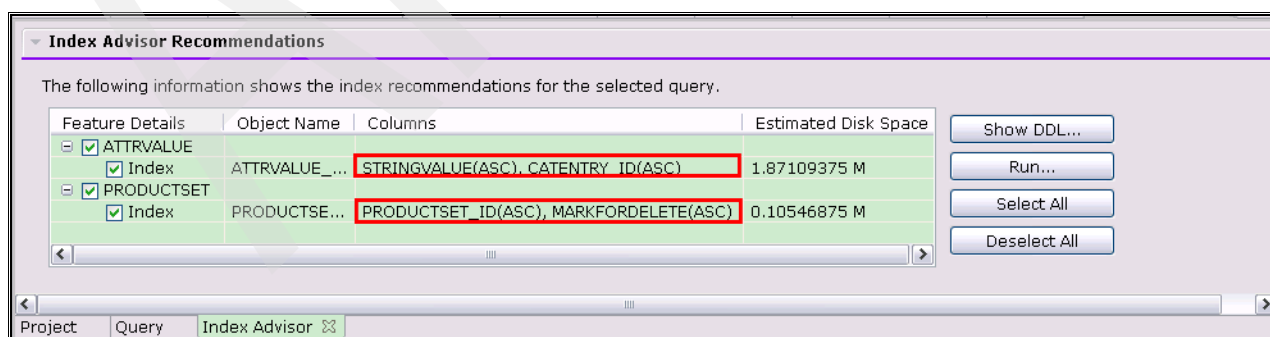


Figure 14-3 Index Advisor recommendations

6. Use the Access Path Advisor and Access Path Graph tool: At this point, the statistics and indexes are at the optimal level, according to the Statistics Advisor and Index Advisor recommendations. We can check the access path for confirmation, although we do not expect to find any problems. Use the Access Plan Graph tool to analyze the generated access path in more detail. Observe if there is a noticeable improvement depicted in the Access Plan Graph.

Refer to directions in section 11.2, “Access Path Advisor” on page 292 to perform this task.

Figure 14-4 shows a recommendation from the Access Path Advisor for a query with a join having two join predicates.

The screenshot displays the 'Access Path Warning List' window. It contains a table with the following data:

Severity	Query Block Number	Plan Number	Description
APA_DISASTER_SEVERITY	1	3	1 columns are joined in a s...

Below the table is the 'Access Path Warning Details' section, which is divided into two panes:

- Description:** 1 columns are joined in a sort merge join, but 2 join predicates are used. This discrepancy in number means that some join predicates are applied after the join. If a large number of records are returned after the join, this query is a potential problem. Check the explanation for this warning for more details about the possible cause and solution.
- Explanation:** DB2 used a sort merge join to process the specified query. For such queries, ensure that all join predicates are in the form TABLE1.COL1 = TABLE2.COL2 and that the two columns have the same data type. If you are using DB2 for z/OS, Version 7 or earlier, also ensure that these two columns have the same length and nullability. Otherwise, DB2 does not evaluate these predicates before the join, and the query processing might take a long time.

At the bottom, the 'PLAN\_TABLE record' section shows a table with columns: QBLO..., PLANNO, MIXOP..., METHOD, CREAT..., TNAME, CORRE..., ACCES..., ACCES..., ACCES..., MATCH..., MERGE..., INDEX..., and PREFE... The table is currently empty.

Figure 14-4 Access Path Advisor recommendation

7. Use the Visual Plan Hints tool: When you reach this point, the access path is optimal and your problem might have been solved already. In rare cases, the query performance only improves slightly or not at all. If so, the query might already have an optimum access path and, if necessary, the optimization focus shifted to consider other parameters. In case of a performance regression, it might also mean that the optimizer, for some reason, cannot generate an efficient access path within the time limit using the available information in the new environment setting. There are several alternatives to try to resolve this problem, such as increasing the optimizer time via DSNZPARM, re-write the query to reduce its complexity, or use Visual Plan Hints tool to provide hints to help the optimizer compose a better access path.

The Visual Plan Hints is a good tool to use for an emergency fix when a better access path is available. However, it is not selected by the optimizer.

Follow directions in section 8.6, “Visual Plan Hint” on page 204 to use the Visual Plan Hints tool.

8. Use Service SQL: If none of the above steps help in resolving the problem query, then the last resort is to consult resident experts or IBM service. Use the Service SQL tool to gather the required information to send to IBM Service. Refer to section 8.5, “Gather service information” on page 198 to perform Service SQL. Service SQL helps you reproduce the SQL situation on another system.

You can also save the access plan graph as an XML file. This saves the necessary information to be passed along for consultation, to allow someone else to look at the access path. See “Access Plan Graph buttons” on page 174.

## 14.2.2 Tuning a query workload

When the individual query is tuned, you might want to see how well the query performs in a workload environment along with other queries in the application. The Workload Advisors and tools can help you in this effort. The motivation to perform this task is to gain further performance improvement for all queries in the workload and for each individual query. There is no strict rule in the order of applying these Advisors and tools. If you have a workload defined containing some number of problem queries to tune, you might consider using the Workload Advisors and tools directly to save time, instead of tuning each individual query first one by one. It is a choice.

See also Chapter 13, “Analysis of a problem workload” on page 343 for details and examples on how to analyze a query workload.

The steps are:

1. Run the Workload Query Advisor: At this stage, we assume that each individual query is tuned or performs reasonably well. However, we do not know if the queries perform well in a workload and if there is more room left for further improvements.

A query workload is defined by capturing a set of queries from supported sources, such as text files, the statement cache, monitor profiles, QMF, and so forth.

Run the Workload Query Advisor against this new workload. We do not expect anything unusual if each query has been run through the Query Advisor before.

The Workload Query Advisor generates warnings for SQL statements that potentially cause poor performance within the workload. The report provides a summary of how many statements are analyzed, how many with warnings, breakdown on the severity of warnings (high, medium, and low), and the frequency count of statements within each category. When there are high severity warnings, it is advisable to tune these queries individually before proceeding further with the workload analysis.

See section 11.4, “Workload Query Advisor” on page 307 on how to define and run the Workload Query Advisor.

Figure 14-5 shows a summary of Workload Query Advisor result. Click **View Statements** to see the messages for each statement.

**Workload Query Advisor Recommendations Summary**

The following is a summary of the queries analyzed in the workload. Use this criteria to filter the view for specific statements.

Statements Sorted by	Number
Statements Analyzed Successfully	7
Statements with Warnings	3
Number of High Severity Warnings	0
Number of Medium Severity Warnings	2
Number of Low Severity Warnings	4
Statements with High Severity Warnings	0
Statements with Medium Severity Warnings	1
Statements with Low Severity Warnings	3

View statements that meet the following criteria:

Degree of warning severity: ☒ High severity ☒ Medium severity ☒ Low severity ☒ Show statements that do not contain warnings

[Restore Defaults](#) [Save as Defaults](#) [View Statements](#)

Workload | Statements | **Advisors** | Statistics Advisor | Statistics Advisor Details | Query Advisor Summary

Figure 14-5 Workload Query Advisor - summary result

Figure 14-6 lists the statements and warnings associated with them. Double-click the statement you want to analyze. In this example, we click the last statement in the list to see the Workload Query Advisor's recommendations.

**Query List**

**Workload Query Advisor Recommendations**

This table shows the filtered view of queries based on the confidence level of the recommendation and the degree of warning severity as specified in the filter.

[Details](#)

Status	Warnings	High Severi...	Medium Se...	Low Severi...	SQL Text
No Warning	0	0	0	0	SELECT T1.CATGROUP_ID , T1.CATALOG_ID , T1.PRODUCTSET_I...
No Warning	0	0	0	0	SELECT T1.NAME , T2.CATALOG_ID , T2.CATGROUP_ID FROM TGC...
No Warning	0	0	0	0	SELECT T1.NAME , T1.AUXDESCRIPTION2 , T1.AVAILABILITYDATE ...
No Warning	0	0	0	0	SELECT B1.MEMBER_ID , B1.LASTUPDATE , B1.BASEITEM_ID , B1...
Warnings w...	1	0	0	1	SELECT T1.ENDTIME , T1.POLICY_ID , T1.STOREENT_ID , T1.POLI...
Warnings w...	1	0	0	1	SELECT DISTINCT T1.PARTNUMBER , T1.FIELD5 , T1.URL , T1.CAT...
Warnings w...	4	0	2	2	SELECT DISTINCT T1.CATENTRY ID TO , T1.QUANTITY , T1.MASS...

Workload | Statements | **Advisors** | Statistics Advisor | Statistics Advisor Details | Query Advisor Summary | **Query Advisor**

Figure 14-6 Workload Query Advisor - query list

Figure 14-7 shows the recommendations for this statement. Click one of the recommendation to see the details, in this case, the last one. The recommendation basically suggest to re-write the inequality predicate into a more efficient form when possible.

Query List Query 16907

**Query Advisor Recommendations List**

Select a recommendation to view more details. To implement the recommendation, change the SQL statement in the source from which it came.

No	Severity	Confidence	Line Number	Description
1	Medium	Medium	23,24,25,26,27,28,...	The following predicate on column CATENTR...
2	Medium	Low	16,17	Consider adding join predicates between c...
3	Low	Low	16	Consider replacing the asterisk (*) or the lo...
4	Low	Low	21	Consider rewriting the following predicate t...

**SQL Text**

```

, T1.OID
, T1.STORE_ID
, T1.DATE1
FROM TGCOMM01.MASSOCCECE AS T1
, TGCOMM01.CATENTRY
, TGCOMM01.STORECENT
WHERE ( TGCOMM01.CATENTRY.MARKFORDELETE = 0
AND TGCOMM01.STORECENT.STORECENT_ID = 202
AND T1.MASSOCTYPE_ID != 'X-SELL'

```

**Selected Recommendation:**

Description	Explanation
Consider rewriting the following predicate to be indexable on column MASSOCTYPE_ID in table TGCOMM01.MASSOCCECE: T1.MASSOCTYPE_ID <> 'X-SELL' so that it might filter out unnecessary rows earlier. Check the explanation for this warning for more details about possible impact and examples.	The specified SQL statement might perform faster if you rewrite the stage 1 predicate as an indexable predicate. (Indexable predicates are a subset of stage 1 predicates.) Indexable predicates match existing indexes and are the most efficient predicates because they limit the pages that DB2 accesses to only those pages that satisfy that conditions of the predicate. For more information about indexable predicates, see the DB2 for z/OS information in the Information Management Software for z/OS Solutions Information Center at <a href="http://publib.boulder.ibm.com/infocenter/imzic">http://publib.boulder.ibm.com/infocenter/imzic</a> .  (Stage 1 predicates are sometimes called sargable predicates.)  The following examples show how to rewrite a stage 1 predicate as an indexable predicate:

Workload Statements Advisors Statistics Advisor Statistics Advisor Details Query Advisor Summary Query Advisor

Figure 14-7 Workload Query Advisor - query recommendation

2. Run the Workload Statistics Advisor: The Workload Statistics Advisor provides some RUNSTATS recommendations for the whole workload. Consider applying these recommendations to update the statistics.

See section 9.4.4, “Workload Statistics Advisor sample” on page 240 and section 13.6, “Running Workload Statistics Advisor” on page 354.

Figure 14-8 on page 391 shows an example of the Workload Statistics Advisor's recommendations. One of the recommendations is to collect statistics on column groups MARKFORDELETE and PRODUCTSET\_ID. To understand this recommendation, click **Details...** under **Actions** in the right hand side of the panel.

**Recommendations**

Number	Priority	Recommendation	Description
1	High	Run complete RUNSTATS	Gather or Recollect all relevant statistics for...
2	Low	Run partial RUNSTATS	Repair the statistics problems within this w...

**Selected Workload Recommendations: No.2: Run partial RUNSTATS**

**RUNSTATS Control Statements**

```

RUNSTATS TABLESPACE SW01DB1.PRODUCTS
TABLE(TGCOMM01.PRODUCTSET)
COLUMN(NAME,PRODUCTSET_ID,MARKFORDELETE)
COLGROUP(MARKFORDELETE) FREQVAL COUNT 1
COLGROUP(MARKFORDELETE,PRODUCTSET_ID) FREQVAL COUNT
10
SORTDEVT SYSDA
INDEX(TGCOMM01.I0000191,
TGCOMM01.I0000700,
TGCOMM01.P_PRODUC FREQVAL NUMCOLS 1 COUNT 10)
SHRLEVEL CHANGE REPORT YES

RUNSTATS TABLESPACE SW01DB1.BASEITEM
TABLE(TGCOMM01.BASEITEM)
COLUMN(BASEITEM_ID)
INDEX(TGCOMM01.P_BASEIT,
TGCOMM01.I0000040)

```

**Description**

Repair the statistics problems within this workload. Running complete RUNSTATS is preferred. Only run repairing RUNSTATS when the batch window is limited, since collecting partial statistics but not all could cause more inconsistent statistics within the workload.

**Actions**

Details...

Run...

Copy

Save

Workload | Statements | Advisors | **Statistics Advisor** |

Figure 14-8 Workload Statistics Advisor - recommendations

The explanation is shown in Figure 14-9 on page 392. The reason for the recommendation to gather this particular column group statistics is because the Workload Statistics Advisor detects an interesting column group relationship between the MARKFORDELETE and PRODUCTSET\_ID columns.

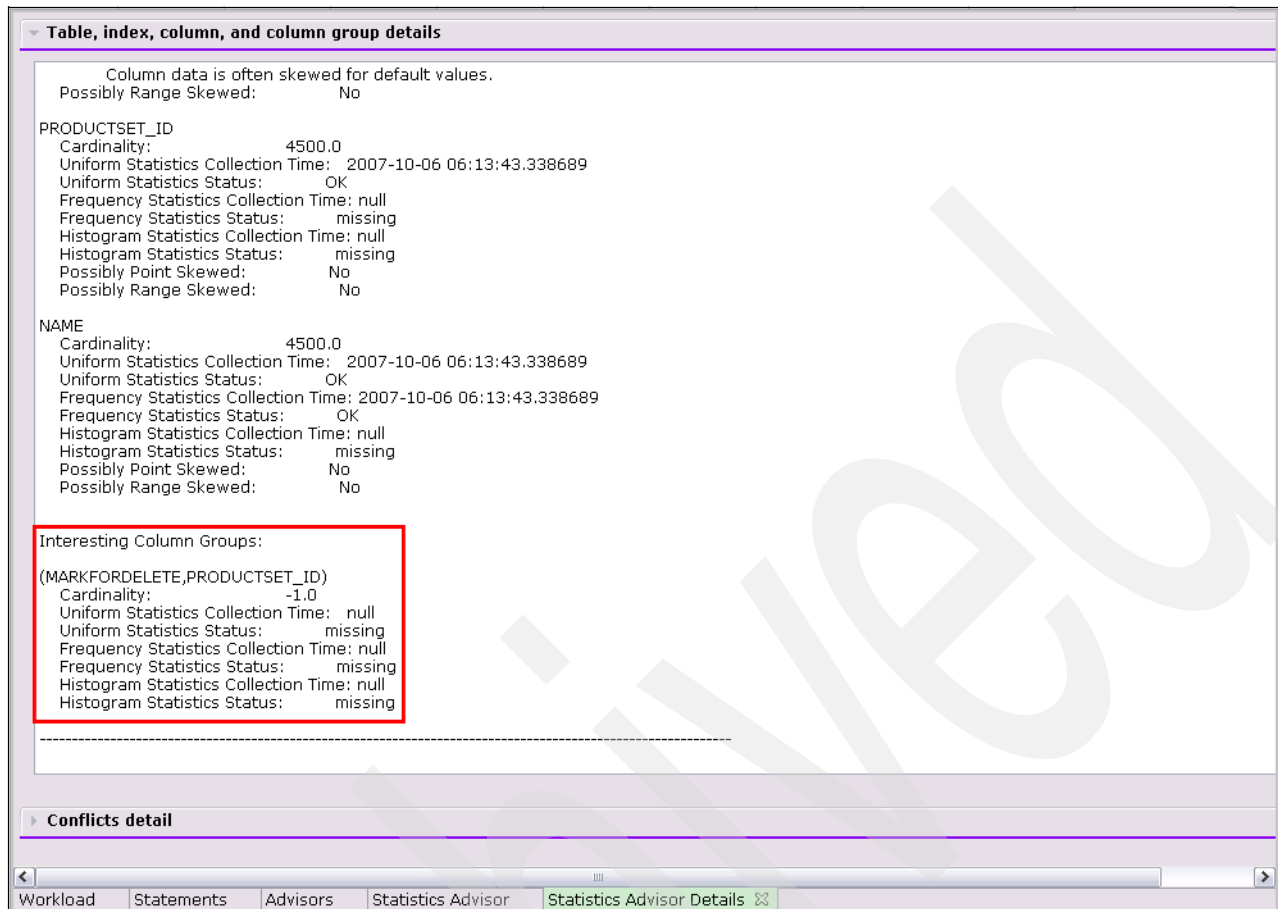


Figure 14-9 Workload Statistics Advisor - details

See also Figure 13-19 on page 357 for an additional example.

3. Run the Workload Index Advisor: See section 10.3, “Workload Index Advisor” on page 257 and section 13.8.1, “Using Workload Index Advisor” on page 365.

The Workload Index Advisor might recommend additional indexes. Before applying these recommendations, check for similar indexes; that is, an existing index with similar composition as the recommended indexes. In this case, you need to choose wisely which index you want to create, if not all.

Figure 14-10 on page 393 illustrates this situation in which the Workload Index Advisor recommends three similar indexes for the PRODUCTSET table.



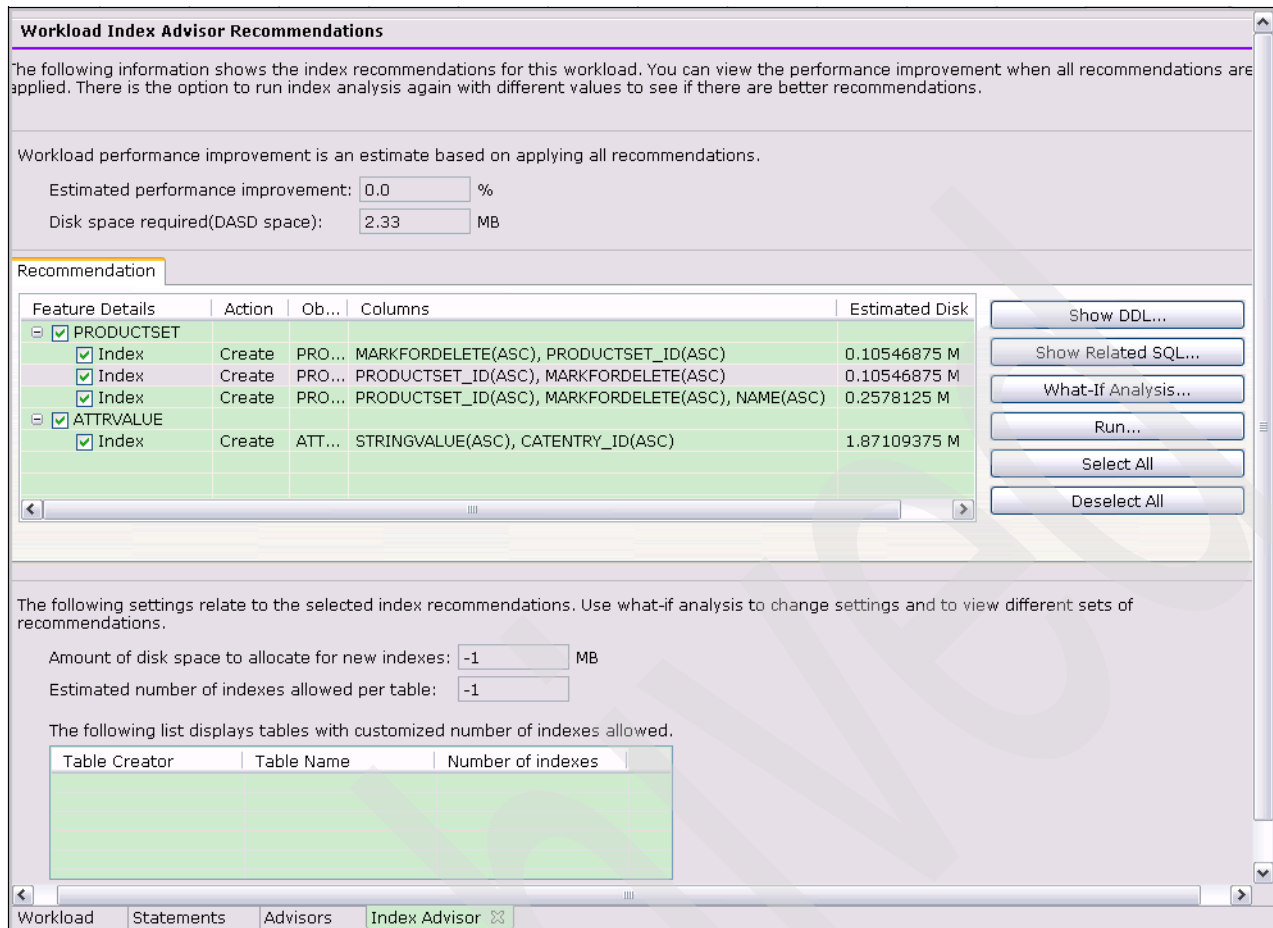


Figure 14-10 Workload Index Advisor - recommendations

See also Figure 13-26 on page 367 for an additional example of the Workload Index Advisor results.

Archived

## Appendixes

This part includes the following appendixes:

- ▶ Appendix A, “Tips for optimizing very large static workloads” on page 397
- ▶ Appendix B, “Administrative Scheduler” on page 405
- ▶ Appendix C, “Relevant DB2 objects” on page 409

Archived

## Tips for optimizing very large static workloads

This appendix details two tips that help optimize overall performance for a query population contained in very large static workloads. These procedures help in situations where the scale of the workloads (thousands or tens of thousands of SQL statements) prevents convenient processing by the Optimization Expert.

Appendix A.1, “Report all column expressions (index candidates)” on page 398 demonstrates a batch procedure that discovers all column expressions in SQL contained in a set of packages. Those most frequently used are reasonable candidates for INDEX ON (column-expression) or rewrites (where possible.)

Appendix A.2, “Automated RUNSTATS column lists” on page 401 shows how a list of column statistics is formulated, paring down unnecessary COLUMN statistics and adding first-cut frequency statistics. They provide a list of reasonable first-cut recommendations. After they are run, run the Statistics Advisor on significant subsets to refine the recommendations further. See Chapter 9, “Statistics Advisor” on page 223.

Both examples exploit the extended EXPLAIN information externalized by DB2 and utilized by the Statistics Advisor, as documented in Appendix E, *DB2 Version 9.1 for z/OS Performance Monitoring and Tuning Guide*, SC18-9851.

## A.1 Report all column expressions (index candidates)

Every predicate that must be evaluated in Stage 2 represents a possible performance problem (depending on the number of times the predicate is evaluated).

The demonstration example looks for all occurrences of column expressions (COLEXP):

```
FUNCTION(colname) = literal|marker  
for example DATE(TIMESTAMP) = CURRENT DATE
```

Many of these are of a type that is rewritten to a Stage 1 alternative. DB2 V9 also introduces a more general capability to create indexes directly over these column expressions.

### A.1.1 Mass-explain procedure

This procedure operates on a set of packages bound into DB2. That set can include:

- ▶ All static packages in a subsystem.
- ▶ All packages in a given COLLID.
- ▶ Packages bound recently.
- ▶ Packages dependent on a particular table or list of tables.
- ▶ Any list to be generated as a `SELECT FROM SYSIBM.SYSPACKAGE`.

For each package, the procedure formats a `BIND COPY` into a test COLLID with `EXPLAIN`, then an immediate `FREE` of that package. The `BIND` is done solely to externalize the `EXPLAIN` information and the `FREE` cleans out the (unneeded) package from the DB2 catalog.

Each package `EXPLAIN` populates `DSN_PREDICAT_TABLE`, which, at the end of the process, contains a categorization of all the predicates in the workload.

#### Setup prerequisites

For this exercise, you must have available an SQLID that:

- ▶ Is one of your secondary AUTHIDs, so that you can specify it as the `OWNER` parameter in a `BIND` command.
- ▶ Has been granted authority over all the tables referenced in the packages, as `BIND` requires.
- ▶ Has `BINDADD` authority.
- ▶ Has `CREATE IN COLLECTION` for the TEST COLLIDs to be employed (your choice).

`EXPLAIN` requires that you define the tables listed in Table A-1.

Table A-1 *EXPLAIN tables needed*

Table	Where to find definition
<sqlid>.PLAN_TABLE	DB2 Version 9.1 for z/OS SQL Reference, SC18-9854
<sqlid>.DSN_PREDICAT_TABLE	DB2 Version 9.1 for z/OS Performance Monitoring and Tuning Guide, SC18-9851

## Generating the predicate information

The procedure in Example A-1 gathers information for all packages in a collection:

*Example: A-1 Generating predicate information for a set of packages*

```
/*-----*
/*  GENERATE "BIND COPY" <= SELECT FROM SYSIBM.SYSPACKAGE
/*-----*
//UNLOAD  EXEC  PGM=IKJEFT01,DYNAMNBR=20
//SYSTSPRT DD  SYSOUT=*
//SYSTSIN  DD  *
DSN SYSTEM(DSN)
RUN PROGRAM(DSNTIAUL) PLAN(DSNTIB91) PARMS('SQL') -
      LIB('DSN910.RUNLIB.LOAD')
//SYSPRINT DD  SYSOUT=*
//SYSPUNCH DD  SYSOUT=*
//SYSRECOO DD  DSN=&&PKGEXTR,
//              DISP=(NEW,PASS),UNIT=SYSDA,SPACE=(CYL,(1,1))
//SYSIN      DD  *
SELECT
  CHAR(
    'BIND PACKAGE(TEST_' || STRIP(COLLID) || ' ' ||
    ' COPY(' || STRIP(COLLID) || ' ' || STRIP(NAME) || ' ' - \
    || CASE WHEN (VERSION <> '')
      THEN ' COPYVER(' || STRIP(VERSION) || ' ' - \
      ELSE '' END
    || ' QUALIFIER(' || STRIP(QUALIFIER) || ' ' - \
    || ' EXPLAIN(YES) ACTION(ADD) OWNER(sqlid) \
    || 'FREE PACKAGE(TEST_' || STRIP(COLLID) || ' ' || STRIP(NAME) ||
    | '.' || STRIP(VERSION) || ' '))'
    ,240)

FROM SYSIBM.SYSPACKAGE AS PKG
WHERE
  VALID <> 'N'
AND COLLID = 'collid'
ORDER BY
  1
;
/*
/*-----*
/*  REXX ROUTINE TO BREAK LINES BY '\ '
/*-----*
//REXXLIB EXEC  PGM=IEBUPDTE,PARM=NEW
//SYSPRINT DD  SYSOUT=*
//SYSUT2     DD  DSN=&&REXXLIB,
//              DISP=(NEW,PASS),UNIT=SYSDA,SPACE=(TRK,(3,1,1))
//SYSIN      DD  *
./ ADD NAME=LINEBRK
/**** REXX EXEC TO BREAK LINES AT '\ ' ****/
"EXECIO * DISKR INPUT (STEM INPUT."
DO I = 1 TO INPUT.0
  DO UNTIL (OUTPUT = '')
    PARSE VAR INPUT.I OUTPUT '\ ' INPUT.I
    IF (OUTPUT ^= '') THEN
      QUEUE OUTPUT
```

```

        END
    END
    "EXECIO" QUEUED() "DISKW OUTPUT (FINIS"
/*
/*-----*
/*  BREAK LINES BY '\ ' TO GENERATE BIND COPY / FREE COMMANDS
/*-----*
//LINEBRK EXEC PGM=IRXJCL,PARM=LINEBRK
//SYSEXEC DD DSN=&&REXXLIB,DISP=(OLD,DELETE)
//SYSTSPRT DD SYSOUT=*
//SYSTSIN DD DUMMY
//INPUT DD DSN=&&PKGEXTR,DISP=(OLD,DELETE)
//OUTPUT DD DSN=&&BINDFREE,
//          DISP=(,PASS),UNIT=SYSDA,SPACE=(CYL,(1,1))
/*
/*-----*
/*  BIND COPY / FREE
/*-----*
//BINDFREE EXEC PGM=IKJEFT01
//SYSTSPRT DD SYSOUT=*
//SYSTSIN DD *
        DSN SYSTEM(ssid)
//          DD DSN=&&BINDFREE,DISP=(OLD,DELETE)
/*

```

---

## A.1.2 Reporting column expressions

The query in Example A-2 reports on all column expressions that appear on the left-hand-side of predicates.

*Example: A-2 Reporting all column expressions used as predicates*

```

SELECT
    CHAR(PROGNAME,8) AS PROGNAME
    ,QUERYNO
    ,SUBSTR(TEXT,1,100) AS TEXT
FROM ssid.DSN_PREDICAT_TABLE
WHERE TYPE = 'EQUAL'
    AND LEFT_HAND_SIDE = 'NONCOL'

```

---

Typical results are listed in Example A-3.

*Example: A-3 Report of column expressions*

PROGNAME	QUERYNO	TEXT
program1	887	YEAR(tb11.DATE1)=(EXPR)
program2	745	SUBSTR(DIGITS(tb12.MM_DD),3,2)=(EXPR)
program2	1958	SUBSTR(tb13.COL3,1,7)='0000000'
program3	1282	CHAR(tb14.DATE2)=(EXPR)
program3	3695	VALUE(tb15.COL4,' ')=(EXPR)
program4	809	YEAR(tb11.DATE1)=(EXPR)

---

Note that the TEXT expresses the predicate as:



Tablename.columnname

Grouping by the column expressions and ordering by descending count provides a prioritized list of the column expressions that are reasonable tuning targets. To remediate:

- ▶ Rewrite individual SQL to a Stage 1 alternative, where possible, replace `YEAR(date) = year` with `date BETWEEN first-date-in-year AND last-date-in-year`.
- ▶ Create an index directly over a heavily-referenced column expression.

## A.2 Automated RUNSTATS column lists

Compare with the commonly employed RUNSTATS specification:

```
RUNSTATS TABLESPACE dbname.tsname  
TABLE(ALL) INDEX(ALL)
```

This collects COLUMN statistics for all columns. Our procedure targets COLUMN statistics only for those columns used in the predicates in the static SQL population. Statistics for the other columns have no optimization benefit and just add to the RUNSTATS cost.

This does not collect FREQVAL for ANY column (other than the leading column in each index). This procedure targets FREQVAL statistics for columns used in predicates with literal values, whose frequencies (potentially skewed) can help optimizer costs.

These tips employ a batch alternative to the Workload Statistics Advisor, to obtain a set of reasonable first-cut recommendations where the scale of the workloads (thousands or tens of thousands of SQL statements) prevents convenient processing by the Optimization Expert.

After these are run, run the Statistics Advisor to refine the estimates further. See Chapter 9, “Statistics Advisor” on page 223.

### A.2.1 Mass-explain procedure

Use the same procedure as described in Appendix A.1.1, “Mass-explain procedure” on page 398.

In fact, run this procedure only once to provide information for both analyses in this appendix.

### A.2.2 Columns in predicates (need COLUMN statistics)

The following SQL (Example A-4) returns all columns that appear:

- ▶ On the left-hand-side of predicates.
- ▶ As join columns (left-hand-side or right-hand-side of joins).

They appear along with reference counts for the package population processed according to Appendix A.1.1, “Mass-explain procedure” on page 398.

*Example: A-4 Columns used in predicates - Target COLUMN statistics*

---

```
SELECT  
  TBCREATOR  
  ,TBNAME  
  ,COLNAME  
  ,COUNT(*) AS REFERENCE_COUNT
```

```

FROM
(
  SELECT
    PL.CREATOR      AS TBCREATOR
    ,PL.TNAME       AS TBNAME
    ,CHAR(PR.LEFT_HAND_SIDE,30) AS COLNAME
  FROM DSN_PREDICAT_TABLE AS PR
  JOIN PLAN_TABLE        AS PL
  ON (PR.PROGNAME      = PL.PROGNAME
  AND PR.EXPLAIN_TIME = PL.BIND_TIME
  AND PR.QUERYNO       = PL.QUERYNO
  AND PR.LHS_QBNO      = PL.QBLOCKNO
  AND PR.LHS_TABNO     = PL.TABNO
  AND PR.LHS_TABNO     > 0
  AND PL.MIXOPSEQ      = 0)

  UNION ALL

  SELECT
    PL.CREATOR      AS TBCREATOR
    ,PL.TNAME       AS TBNAME
    ,CHAR(PR.RIGHT_HAND_SIDE,30) AS COLNAME
  FROM DSN_PREDICAT_TABLE AS PR
  JOIN PLAN_TABLE        AS PL
  ON (PR.PROGNAME      = PL.PROGNAME
  AND PR.EXPLAIN_TIME = PL.BIND_TIME
  AND PR.QUERYNO       = PL.QUERYNO
  AND PR.RHS_QBNO      = PL.QBLOCKNO
  AND PR.RHS_TABNO     = PL.TABNO
  AND PR.LHS_TABNO     > 0
  AND PR.RHS_TABNO     > 0)
) AS T

GROUP BY
  TBCREATOR
  ,TBNAME
  ,COLNAME
ORDER BY
  TBCREATOR
  ,TBNAME
;

```

---

A program can easily process the result set, breaking by table name, to generate the appropriate COLUMN list:

```

RUNSTATS TABLESPACE dbname.tsname
TABLE(tbcreator.tbname) INDEX(ALL)
COLUMN(column1,column2,...)

```

### A.2.3 Columns and literal values (need FREQVAL)

The following SQL (Example A-5) returns all columns that appear as “column op literal-value”, along with reference counts, for the package population processed according to Appendix A.1.1, “Mass-explain procedure” on page 398.

*Example: A-5 Columns used with literal values*

---

```
SELECT
  TBCREATOR
  ,TBNAME
  ,COLNAME
  ,COUNT(*) AS REFERENCE_COUNT
FROM
  (
    SELECT
      PL.CREATOR      AS TBCREATOR
      ,PL.TNAME       AS TBNAME
      ,CHAR(PR.LEFT_HAND_SIDE,30) AS COLNAME
    FROM DSN_PREDICAT_TABLE AS PR
    JOIN PLAN_TABLE        AS PL
    ON (PR.PROGNAME        = PL.PROGNAME
    AND PR.EXPLAIN_TIME    = PL.BIND_TIME
    AND PR.QUERYNO         = PL.QUERYNO
    AND PR.LHS_QBNO        = PL.QBLOCKNO
    AND PR.LHS_TABNO       = PL.TABNO
    AND PR.LHS_TABNO       > 0
    AND PL.MIXOPSEQ        = 0)
    WHERE (PR.TYPE IN ('EQUAL','IN')
    AND (PR.MARKER = 'N'))
  ) AS T
GROUP BY
  TBCREATOR
  ,TBNAME
  ,COLNAME
  ;
```

---

A program can easily process the result set, breaking by table name, to generate the appropriate COLGROUP list:

```
RUNSTATS TABLESPACE dbname.tsname
TABLE(tbcreeator.tbname) INDEX(ALL)
COLGROUP(column1) FREQVAL COUNT 10
COLGROUP(column2) FREQVAL COUNT 10
...
```

Archived

## Administrative Scheduler

This appendix discusses the Administrative (Admin) Scheduler. It contains the following topics:

- ▶ Purpose of the Administrative Scheduler
- ▶ Setup of the Administrative Scheduler
- ▶ Security for the Administrative Scheduler
- ▶ Sample SELECT to display the task list
- ▶ Additional information

## B.1 Purpose of the Administrative Scheduler

The Administrative Scheduler runs tasks that are defined in the Administrative Scheduler task list, according to a requested schedule. Tasks are stored procedures or JCL jobs.

You manage the Administrative Scheduler task list through DB2 stored procedures that add and remove tasks.

## B.2 Setup of the Administrative Scheduler

Starting with DB2 9, each DB2 subsystem or data sharing member now has a new started task for the Admin Scheduler. The Admin Scheduler uses a Virtual Storage Access Method (VSAM) relative record data set (RRDS) to house its task list. All members of a data sharing group share the same task list data set. However, each member has a separate unique started task name. For example, if we have a data sharing group with members DSN1 and DSN2, both members share the same VSAM RRDS task list and the started task for the Admin Scheduler is DSN1ADMT and DSN2ADMT. Having multiple members on the same LPAR does not change this requirement.

DB2 uses the DSNZPARM value for ADMTPROC to start the Admin Scheduler started task. The Admin Scheduler is brought up with DB2. However, it does not shutdown when DB2 shuts down.

There are a number of reasons why the Admin Scheduler start task might not start. One reason is a JCL error, in which case you see the message in Example B-1 at DB2 startup in the MSTR started task.

*Example: B-1 Example of a failed Admin Scheduler startup in the MSTR started task*

---

```
DSNI002I  -DB9A DSNADMTI UNABLE TO START
          OBJECT-TYPE 0001
          OBJECT-NAME DB9AADMT
          REASON 00000004
          TYPE
          NAME
```

---

When the Admin Scheduler is up, but DB2 is brought down, the Admin Scheduler started task displays the following message once every minute:

```
DSNA679I  DSNA6BUF THE ADMIN SCHEDULER DB9AADMT CANNOT ACCESS TASK LIST
SYSIBM.ADMIN_TASKS, REASON=-981
```

## B.3 Security for the Administrative Scheduler

The Administrative Scheduler started task requires a started task user ID and a default user ID. The default use ID is used in the Scheduler when a scheduled task does not have an execution user specified. In this case, we want to limit the authority of the task instead of providing a more powerful user, such as the one used by the started task itself.

A RACF security template for the Admin Scheduler is created during the base DB2 install and is found in the DSNTIJRA member of your SDSNSAMP data set.

## B.4 Sample SELECT to display the task list

Optimization Service Center or Optimization Expert does not provide a window to display the contents of the task list. One option to display the contents of the task list is to run the SELECT in Example B-2.

*Example: B-2 Task list display*

---

```
SELECT
  T.TASKID
  ,T.SCHEDULED_TASKID
  ,T.WLID
  ,W.NAME WLNAME
  ,T.SRCID
  ,S.NAME SRCNAME
  ,CASE(T.TYPE)
    WHEN 1 THEN 'CAPTURE'
    WHEN 2 THEN 'CONSOLIDATE_LITERALS'
    WHEN 3 THEN 'CONSOLIDATE_ACCESSPLAN'
    WHEN 4 THEN 'EXPLAIN'
    WHEN 5 THEN 'ANALYZE'
    WHEN 6 THEN 'MONITOR'
    WHEN 7 THEN 'SNAPSHOT'
  END AS TYPE
  ,CASE(T.STATUS)
    WHEN 'S' THEN 'SCHEDULED'
    WHEN 'F' THEN 'FINISHED'
    WHEN 'C' THEN 'CANCELLED'
    WHEN 'A' THEN 'ABEND'
    WHEN 'R' THEN 'RUNNING'
    WHEN 'W' THEN 'SLEEPING'
    WHEN 'T' THEN 'CANCELLING'
    WHEN 'N' THEN 'FRESH'
  END AS STATUS
  ,T.CREATOR
  ,T.START_TIME
  ,T.END_TIME
  ,T.INTERVAL
  ,T.CONSOLIDATE_RTINFO
  ,CASE(T.CONSOLIDATE_EPINFO)
    WHEN 1 THEN 'NONE'
    WHEN 2 THEN 'REPLACE'
    WHEN 3 THEN 'KEEP'
    WHEN 4 THEN 'CONSOLIDATE'
  END AS CONSOLIDATE_ACCESS_PLAN
  ,T.CONSOLIDATE_LITERA
  ,T.KEEP_STATEMENTS
  ,T.CONSOLIDATION_TIME
  ,T.START_TRACE
  ,T.STOP_TRACE
  ,T.WARMUP_TIME
  ,T.LAST_FIRE_TIME
  ,T.LAST_UPDATE_TS
FROM
  DB2OSC.DSN_WCC_TASKS T,
```

```
DB2OSC.DSN_WCC_WORKLOADS W,  
DB2OSC.DSN_WCC_WL_SOURCES S  
WHERE  
    T.WLID=W.WLID  
AND  S.WLID = W.WLID;
```

---

**Notes:**

- ▶ When the start time is complete, the task is triggered.
- ▶ If the task is scheduled in the client, there is a thread to check the start time repeatedly.
- ▶ If the task is scheduled using the Admin Scheduler and the start time is later than the current time, the task is triggered by the Admin Scheduler. T.SCHEDULED\_TASKID saves the task ID in the Admin Scheduler.

## B.5 Additional information

Additional information is found in:

- ▶ Chapter 14, “Scheduling administrative tasks”, *DB2 Version 9.1 for z/OS Administration Guide*, SC18-9840
- ▶ Chapter 43, “MODIFY rmtproc,APPL=SHUTDOWN”, *DB2 Version 9.1 for z/OS Command Reference*, SC18-9844
- ▶ Chapter 44, “MODIFY admtproc,APPL=TRACE”, *DB2 Version 9.1 for z/OS Command Reference*, SC18-9844



## Relevant DB2 objects

This appendix lists the DB2 objects (table spaces, tables, packages) created for supporting Optimization Service Center or Optimization Expert tools. It contains the following topics:

- ▶ “Security requirements for Optimization Service Center or Optimization Expert” on page 410
- ▶ “Job DSNTIJOS” on page 412
- ▶ “Job AOCDDL” on page 417
- ▶ Packages required by Optimization Service Center
- ▶ “Additional packages required by Optimization Expert” on page 423

## C.1 Security requirements for Optimization Service Center or Optimization Expert

This appendix includes considerations on privilege and authority requirements when you configure Optimization Service Center or Optimization Expert on the workstation in your environment. Because the steps are the same for both Optimization Service Center and Optimization Expert, the terms are interchangeable in this section:

- ▶ Bind Packages:
  - SYSADM or DBADM authority.
  - BINDADD privilege if the Optimization Expert packages do not exist, and CREATEIN privilege on the schema.
  - ALTERIN privilege on the schema if the Optimization Expert packages exist.
  - BIND privilege on the Optimization Expert packages if they exist.
- ▶ Free Package:
  - Ownership of the packages.
  - BINDAGENT privilege granted by the owner of the packages.
  - SYSADM or SYSCTRL authority.
  - PACKADM authority for the collection or for all collections.
- ▶ Grant/ Revoke EXECUTE authority on the required packages:
  - SYSADM authority.
  - Ownership of the packages.
- ▶ Create table spaces and tables:
  - SYSADM or DBADM authority.
- ▶ Use existing EXPLAIN tables
  - SET CURRENT SQLID = your specified Qualifier.
- ▶ Creating aliases for existing EXPLAIN tables:
  - The CREATEALIAS privilege.
  - SYSADM or SYSCTRL authority.
  - DBADM or DBCTRL authority on the database that contains the table, if the alias is for a table and the value of field DBADM CREATE AUTH on installation panel DSNTIPP is YES.
- ▶ Granting access to existing EXPLAIN tables:
  - Ownership of the tables.
  - SYSADM authority.
- ▶ Workload Control Center (WCC):
  - Create or Drop the WCC tables requires SYSADM authority.
  - Add or Remove Users to or from the WCC requires SYSADM authority.
  - Grant WCC users Read-only or Update privilege on this workload requires SYSADM user or the workload's owner.
  - Start or stop or explain a monitor profile requires SYSADM authority.
  - WCC Read-Only user:
    - Can do:
      - Check Workload list/Users/History/Tasks/Statement.

- Run Workload Statistics Advisor) and Workload Query Advisor on the explained workload. The Workload Index Advisor requires SET CURRENT SQLID = 'DB2OE').
- Cannot do:
  - Remove a statement.
  - Edit statement information.
  - Update the source.
  - Add a source.
  - Schedule a task (Explain or Consolidate literal value, access path).
  - Schedule an analyze task -WSA/WIA/WQA.
  - Capture from cache.
- WCC update user:
  - Can do:
    - Check Workload list/Users/History/Tasks/Statement.
    - Run WSA, WQA on the explained workload. (WIA requires SET CURRENT SQLID = 'DB2OE').
    - Remove a statement.
    - Edit statement information.
    - Update the source.
    - Add a source.
    - Schedule a task (consolidate access path).
    - Schedule an analyze task -WSA/WIA/WQA.
  - Cannot do:
    - Schedule a task (Explain/Consolidate literal value).
    - Capture from cache.
- Monitor:
  - Read only users:
    - Can do:
      - Check monitor/user/history/task/statement list
    - Cannot do:
      - Edit a monitor profile.
      - Remove a monitor profile.
      - Enable or disable a monitor profile.
      - Start or stop or explain the monitor profile.
      - Snapshot a monitor.
  - Update users:
    - Can do:
      - Check monitor/user/history/task/statement list.
      - Edit a monitor profile.
      - Remove a monitor profile.

- Enable/disable a monitor profile.
- Cannot do:
  - Start or stop or explain the monitor profile.
  - Snapshot a monitor.
- WCC stored procedures:
  - Users in the group DB2OSCA (also has the privilege to execute the stored procedure SYSPROC.OSC\_RUNSQL and SYSPROC.OSC\_EXECUTE\_TASK).
  - Can do the following via stored procedures:
    - Capture workload from cache.
    - Explain workload.
    - Consolidate literal values.
    - Check snapshot.
  - Cannot do the following via stored procedures:
    - Check monitor
  - Users not in the group DB2OSCA:
    - Cannot do the following via stored procedures:
      - Capture workload from cache
      - Explain workload
      - Consolidate literal values
      - Check snapshot
      - Check monitor

**Notes:**

- Only users with SYSADM authority can check the monitor.
- Any workload update user can consolidate the access path by grouping the client or stored procedure.

## C.2 Job DSNTIJOS

This is the install job that enables Optimization Service Center. It includes the definition of the objects listed in Table C-1.

Table C-1 DSNTIJOS objects

DB2 object	Components	Notes
Stogroup	DSNOSCSG	
Databases	DSNOSCD DB2OSC	

DB2 object	Components	Notes
Table space DSNOSCTS and LOB table space DSNOSCL1	SYSIBM.DSN_PROFILE_TABLE SYSIBM.DSN_PROFILE_TABLE_IX_ALL SYSIBM.DSN_PROFILE_TABLE_IX2_ALL SYSIBM.DSN_PROFILE_HISTORY SYSIBM.DSN_PROFILE_ATTRIBUTES SYSIBM.DSN_PROFILE_ATTRIBUTES_IX_ALL SYSIBM.DSN_PROFILE_ATTRIBUTES_HISTORY	Optimization Service Center profile tables
	SYSIBM.DSN_STATEMENT_RUNTIME_INFO SYSIBM.DSN_STATEMENT_RUNTIME_INFO_IDX1 AUX TABLE SYSIBM.DSN_STATEMENT_RUNTIME_INFO_AUX SYSIBM.DSN_STATEMENT_RUNTIME_INFO_AUXINX SYSIBM.DSN_OBJECT_RUNTIME_INFO	Optimization Service Center runtime information tables
	SYSIBM.PLAN_TABLE SYSIBM.PLAN_TABLE_IDX1 SYSIBM.DSN_STATEMNT_TABLE SYSIBM.DSN_STATEMNT_TABLE_IDX1	Optimization Service Center EXPLAIN tables
Table Space DSNOSCT8	SYSIBM.DSN_FUNCTION_TABLE SYSIBM.FUNC_EXPIDX1	Optimization Service Center EXPLAIN tables
Table space DSNMPLTS and LOB table space DSNMQLTS	SYSIBM.DSN_PREDICAT_TABLE SYSIBM.DSN_PREDICAT_TABLE_IDX1 SYSIBM.DSN_STRUCT_TABLE SYSIBM.DSN_STRUCT_TABLE_IDX1 SYSIBM.DSN_PGROUP_TABLE SYSIBM.DSN_PGROUP_TABLE_IDX1 SYSIBM.DSN_PTASK_TABLE SYSIBM.DSN_PTASK_TABLE_IDX1 SYSIBM.DSN_FILTER_TABLE SYSIBM.DSN_FILTER_TABLE_IDX1 SYSIBM.DSN_DETCOST_TABLE SYSIBM.DSN_DETCOST_TABLE_IDX1 SYSIBM.DSN_SORT_TABLE SYSIBM.DSN_SORT_TABLE_IDX1 SYSIBM.DSN_SORTKEY_TABLE SYSIBM.DSN_SORTKEY_TABLE_IDX1 SYSIBM.DSN_PGRANGE_TABLE SYSIBM.DSN_PGRANGE_TABLE_IDX1 SYSIBM.DSN_VIEWREF_TABLE SYSIBM.DSN_VIEWREF_TABLE_IDX1 SYSIBM.DSN_QUERY_TABLE SYSIBM.DSN_QUERY_TABLE_IDX1 SYSIBM.DSN_QUERY_TABLE_IDX2 AUX TABLE SYSIBM.DSN_QUERY_AUX SYSIBM.DSN_QUERY_AUXINX	Optimization Service Center EXPLAIN tables
Table space DSNSUMTS and LOB table space DSNLOBC1	DB2OSC.DSN_STATEMENT_CACHE_TABLE DB2OSC.DSN_STATEMENT_CACHE_IDX1 DB2OSC.DSN_STATEMENT_CACHE_IDX2 DB2OSC.DSN_STATEMENT_CACHE_IDX3 DB2OSC.DSN_STATEMENT_CACHE_IDX4 DB2OSC.DSN_STATEMENT_CACHE_IDX5 AUX TABLE DB2OSC.DSN_STATEMENT_CACHE_AUX DB2OSC.DSN_STATEMENT_CACHE_AUXINX	Optimization Service Center statement cache database

DB2 object	Components	Notes
Table spaces DSNWDFTS, DSNWLGTS, DSNWTXTS, DSNWISTS, DSNWTKTS, DSNWRTTS, DSNWWITS, DSNWEITS, DSNWTMTS, DSNWRPTS and LOB table spaces DSNTXLTS, DSNTKLTS, DSNWILTS, DSNEILTS, DSNTMLTS	DB2OSC.DSN_WCC_WORKLOADS DB2OSC.DSN_WCC_WL_IDX1 DB2OSC.DSN_WCC_WL_IDX2 DB2OSC.DSN_WCC_WL_IDX3 DB2OSC.DSN_WCC_WL_SOURCES DB2OSC.DSN_WCC_WLS_IDX1 DB2OSC.DSN_WCC_WLS_IDX2 DB2OSC.DSN_WCC_WLS_IDX3 DB2OSC.DSN_WCC_WLS_IDX4 DB2OSC.DSN_WCC_WLS_IDX5 DB2OSC.DSN_WCC_WLS_IDX6 DB2OSC.DSN_WCC_SOURCE_DTL DB2OSC.DSN_WCC_SD_IDX1 DB2OSC.DSN_WCC_SD_IDX2 DB2OSC.DSN_WCC_WL_AUTHIDS DB2OSC.DSN_WCC_WOLA_IDX1 DB2OSC.DSN_WCC_WLA_IDX2 DB2OSC.DSN_WCC_EV_HISTORY DB2OSC.DSN_WCC_EVH_IDX1 DB2OSC.DSN_WCC_MESSAGES DB2OSC.DSN_WCC_MGS_IDX1 DB2OSC.DSN_WCC_EP_HISTORY DB2OSC.DSN_WCC_EPH_IDX1 DB2OSC.DSN_WCC_STMT_TEXTS DB2OSC.DSN_WCC_TEXTS_IDX1 DB2OSC.DSN_WCC_TEXTS_IDX2 AUX TABLE DB2OSC.DSN_WCC_SMTXTS_AUX DB2OSC.DSN_WCC_T_AUXIDX DB2OSC.DSN_WCC_STMT_INSTS DB2OSC.DSN_WCC_INSTS_IDX1 DB2OSC.DSN_WCC_INSTS_IDX2 DB2OSC.DSN_WCC_INSTS_IDX3 DB2OSC.DSN_WCC_INSTS_IDX4 DB2OSC.DSN_WCC_INSTS_IDX5 DB2OSC.DSN_WCC_INSTS_IDX6 DB2OSC.DSN_WCC_INST_SUMMY	WCC objects

DB2 object	Components	Notes
	DB2OSC.DSN_WCC_SUMMY_IDX1 DB2OSC.DSN_WCC_SUMMY_IDX2 DB2OSC.DSN_WCC_SUMMY_IDX3 DB2OSC.DSN_WCC_STMT_RUNTM DB2OSC.DSN_WCC_STRT_IDX1 DB2OSC.DSN_WCC_STRT_IDX2 DB2OSC.DSN_WCC_STRT_IDX3 DB2OSC.DSN_WCC_STRT_IDX4 DB2OSC.DSN_WCC_OBJ_RUNTM DB2OSC.DSN_WCC_OBJRT_IDX1 DB2OSC.DSN_WCC_CAP_TMP_RS AUX TABLE DB2OSC.DSN_WCC_CAP_AUX DB2OSC.DSN_WCC_C_AUX_IDX DB2OSC.DSN_WCC_CTR_IDX1 DB2OSC.DSN_WCC_CTR_IDX2 DB2OSC.DSN_WCC_CTR_IDX3 DB2OSC.DSN_WCC_CTR_IDX4 DB2OSC.DSN_WCC_CTR_IDX5 DB2OSC.DSN_WCC_CTR_IDX6 DB2OSC.DSN_WCC_TASKS AUX TABLE DB2OSC.DSN_WCC_TASKS_AUX DB2OSC.DSN_WCC_T_AUX_IDX DB2OSC.DSN_WCC_TASKS_IDX1 DB2OSC.DSN_WCC_TASKS_IDX2 DB2OSC.DSN_WCC_TASKS_IDX3 DB2OSC.DSN_WCC_TB_STATUS DB2OSC.DSN_WCC_TBS_IDX1 DB2OSC.DSN_WCC_WL_INFO AUX TABLE DB2OSC.DSN_WCC_WLINFO_AUX DB2OSC.DSN_WCC_WIAUX_IDX DB2OSC.DSN_WCC_WLIF_IDX1 DB2OSC.DSN_WCC_STMT_INFO AUX TABLE DB2OSC.DSN_WCC_SMINFO_AUX DB2OSC.DSN_WCC_SI_AUX_IDX DB2OSC.DSN_WCC_SMIF_IDX1 DB2OSC.DSN_WCC_RP_TBS DB2OSC.DSN_WCC_RPT_IDX1 DB2OSC.DSN_WCC_RP_TBS_HIS DB2OSC.DSN_WCC_RPTH_IDX1 DB2OSC.DSN_WCC_RP_IDXES DB2OSC.DSN_WCC_RPI_IDX1 DB2OSC.DSN_WCC_RP_IDX_HIS DB2OSC.DSN_WCC_RPIH_IDX1 DB2OSC.DSN_WCC_RP_STMTS DB2OSC.DSN_WCC_RPINT_IDX1	
Table space DSNPFTS and LOB table space DSNPFLTS	DB2OSC.DSN_STATEMENT_RUNTIME_INFO DB2OSC.DSN_STATEMENT_RUNTIME_INFO_IDX1 DB2OSC.DSN_STATEMENT_RUNTIME_INFO_IDX2 AUX TABLE DB2OSC.DSN_STATEMENT_RUNTIME_INFO_AUX DB2OSC.DSN_STATEMENT_RUNTIME_INFO_AUXINX DB2OSC.DSN_OBJECT_RUNTIME_INFO	RUNTIME information tables

DB2 object	Components	Notes
Table spaces DSNFXNTS, DSNPGRTS, DSNPTKTS, DSNEXPTS, DSNQRYTS, and LOB table space DSNLOBT2	DB2OSC.PLAN_TABLE DB2OSC.PLAN_TABLE_IDX2 DB2OSC.PLAN_TABLE_IDX1 DB2OSC.PLAN_TABLE_IDX3 DB2OSC.PLAN_TABLE_IDX4 DB2OSC.DSN_STATEMNT_TABLE DB2OSC.DSN_STATEMNT_TABLE_IDX1 DB2OSC.DSN_FUNCTION_TABLE DB2OSC.DSN_FUNCTION_TABLE_IDX1 DB2OSC.DSN_PREDICAT_TABLE DB2OSC.DSN_PREDICAT_TABLE_IDX1 DB2OSC.DSN_STRUCT_TABLE DB2OSC.DSN_STRUCT_TABLE_IDX1 DB2OSC.DSN_PGROUP_TABLE DB2OSC.DSN_PGROUP_TABLE_IDX1 DB2OSC.DSN_PTASK_TABLE DB2OSC.DSN_PTASK_TABLE_IDX1 DB2OSC.DSN_FILTER_TABLE DB2OSC.DSN_FILTER_TABLE_IDX1 DB2OSC.DSN_DETCOST_TABLE DB2OSC.DSN_DETCOST_TABLE_IDX1 DB2OSC.DSN_SORT_TABLE DB2OSC.DSN_SORT_TABLE_IDX1 DB2OSC.DSN_SORTKEY_TABLE DB2OSC.DSN_SORTKEY_TABLE_IDX1 DB2OSC.DSN_PGRANGE_TABLE DB2OSC.DSN_PGRANGE_TABLE_IDX1 DB2OSC.DSN_VIEWREF_TABLE DB2OSC.DSN_VIEWREF_TABLE_IDX1 DB2OSC.DSN_QUERY_TABLE DB2OSC.DSN_QUERY_TABLE_IDX1 DB2OSC.DSN_QUERY_TABLE_IDX2 DB2OSC.DSN_QUERY_TABLE_IDX3	EXPLAIN tables
Table spaces DSNFXNTS, DSNPGRTS, DSNPTKTS, DSNEXPTS, DSNQRYTS, and LOB table space DSNLOBT2	AUX TABLE DB2OSC.DSN_QUERY_AUX DB2OSC.DSN_QUERY_AUXINX	



DB2 object	Components	Notes
Table space DSNWSATS	DB2OSC.DSN_WSA_SESSIONS DB2OSC.DSN_WSA_IDX_SSNS DB2OSC.DSN_WSA_DATABASES DB2OSC.DSN_WSA_IDX_DBS DB2OSC.DSN_WSA_TSS DB2OSC.DSN_WSA_IDX_TSS DB2OSC.DSN_WSA_TABLES DB2OSC.DSN_WSA_IDX_TABLES DB2OSC.DSN_WSA_COLUMNS DB2OSC.DSN_WSA_IDX_COLS DB2OSC.DSN_WSA_COLGROUPS DB2OSC.DSN_WSA_IDX_CGS DB2OSC.DSN_WSA_CGFREQS DB2OSC.DSN_WSA_IDX_CGFS DB2OSC.DSN_WSA_CGHISTS DB2OSC.DSN_WSA_IDX_CGHS DB2OSC.DSN_WSA_INDEXES DB2OSC.DSN_WSA_IDX_IDXS DB2OSC.DSN_WSA_KEYS DB2OSC.DSN_WSA_IDX_KEYS DB2OSC.DSN_WSA_KEYTARGETS DB2OSC.DSN_WSA_IDX_KTS DB2OSC.DSN_WSA_KTGS DB2OSC.DSN_WSA_IDX_KTGS DB2OSC.DSN_WSA_KTGFREQS DB2OSC.DSN_WSA_IDX_KTGFS DB2OSC.DSN_WSA_KTGHISTS DB2OSC.DSN_WSA_IDX_KTGHS DB2OSC.DSN_WSA_LITERALS DB2OSC.DSN_WSA_IDX_LTRS DB2OSC.DSN_WSA_ADVICE DB2OSC.DSN_WSA_IDX_ADVICE	WSA tables
Stored procedures SYSPROC.OSC_RUNSQL and SYSPROC.OSC_EXECUTE_TASK		

**Note:** Aliases are created for some of the above objects as part of job DSNTIJOS.

## C.3 Job AOCDDL

This is the install job that enables Optimization Expert. It includes the definition of the objects listed in Table C-2.

Table C-2 AOCDDL objects

DB2 object	Components	Notes
Stogroup	AOCOESG	EXPLAIN tables
Database	AOCOEDB	

Table space AOCOETS and LOB table space DSNSTLTS	"DB2OE"."DSN_WIA_SESSION" DB2OE.DSN_WIA_SESSN_IDX1 "DB2OE"."DSN_WIA_TABLES" DB2OE.DSN_WIA_TABLE_IDX1 DB2OE.DSN_WIA_TABLE_IDX2 "DB2OE"."DSN_WIA_INDEXES" DB2OE.DSN_WIA_INDEX_IDX1 DB2OE.DSN_WIA_INDEX_IDX2 "DB2OE"."DSN_WIA_COLUMNS" DB2OE.DSN_WIA_COL_IDX1 DB2OE.DSN_WIA_COL_IDX2 "DB2OE"."DSN_WIA_KEYS" DB2OE.DSN_WIA_KEYS_IDX1 "DB2OE"."DSN_WIA_COL_SEQ" DB2OE.DSN_WIA_COLSQ_IDX1 DB2OE.DSN_WIA_COLSQ_IDX2 "DB2OE"."DSN_WIA_STMT" DB2OE.DSN_WIA_STMT_IDX1 DB2OE.DSN_WIA_STMT_IDX2 AUX TABLE DB2OE.DSN_WIA_STMT_AUX DB2OE.DSN_WIA_STMAUX_IDX "DB2OE"."DSN_WIA_QBLOCK" DB2OE.DSN_WIA_QBLOK_IDX1 DB2OE.DSN_WIA_QBLOK_IDX2 "DB2OE"."DSN_WIA_TAB_REF" DB2OE.DSN_WIA_TABRF_IDX1 DB2OE.DSN_WIA_TABRF_IDX2 DB2OE.DSN_WIA_TABRF_IDX3 "DB2OE"."DSN_WIA_PREDICATE" DB2OE.DSN_WIA_PRED_IDX1 DB2OE.DSN_WIA_PRED_IDX2 "DB2OE"."DSN_WIA_GBOBDIST" DB2OE.DSN_WIA_GBOBD_IDX1 DB2OE.DSN_WIA_GBOBD_IDX2 DB2OE.DSN_WIA_GBOBD_IDX3 "DB2OE"."DSN_WIA_COL_REF" DB2OE.DSN_WIA_COLRF_IDX1 DB2OE.DSN_WIA_COLRF_IDX2 DB2OE.DSN_WIA_COLRF_IDX3 DB2OE.DSN_WIA_COLRF_IDX4 "DB2OE"."DSN_WIA_COLSEQ_KEY" DB2OE.DSN_WIA_CSKEY_IDX1 "DB2OE"."DSN_WIA_TABREF_IDX" DB2OE.DSN_WIA_TRIDX_IDX1 DB2OE.DSN_WIA_TRIDX_IDX2 "DB2OE"."DSN_WIA_SESION_HIS" DB2OE.DSN_WIA_SHIST_IDX1 "DB2OE"."DSN_WIA_ERROR_HIST" DB2OE.DSN_WIA_EHIST_IDX1 DB2OE.PLAN_TABLE DB2OE.DSN_STATEMNT_TABLE DB2OE.DSN_VIRTUAL_INDEXES	Workload-based Index Advisor
-----------------------------------------------------	-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------	---------------------------------

Table space AOCOETS and LOB table spaces DSNSTLTS, WQALBTS1, and WQALBTS2	"DB2OE"."DSN_WQA_SESSION" "DB2OE"."DSN_WQA_SIN_PRIX" "DB2OE"."DSN_WQA_STMT" "DB2OE"."DSN_WQA_STMT_PRIX" AUX TABLE "DB2OE"."DSN_WQA_AUXT1" "DB2OE"."DSN_WQA_AUX1" "DB2OE"."STMT_ROWIDIX" "DB2OE"."DSN_WQA_WARNING" "DB2OE"."DSN_WQA_WNG_PRIX" AUX TABLE "DB2OE"."DSN_WQA_AUXT2" "DB2OE"."DSN_WQA_AUX2" "DB2OE"."WARNING_ROWIDIX" "DB2OE"."DSN_WQA_EXCEPTION" "DB2OE"."DSN_WQA_SR_REF" "DB2OE"."DSN_WQA_WL_REF"	Workload-base Query Advisor
---------------------------------------------------------------------------	---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------	-----------------------------

## C.4 Packages required by Optimization Service Center

The Optimization Service Center packages are:

► DSN5OPKG:

This accesses catalog tables to check the existence of the package, and retrieves the privileges that are held by users over packages and tables. This group of packages accesses the following catalog tables:

- SYSIBM.SYSPACKAGE
- SYSIBM.SYSPACKAUTH
- SYSIBM.SYSTABAUTH
- SYSIBM.SYSROUTINES

► DSN5OADM:

This accesses catalog tables to check the existence of the objects, such as tables and databases used by the Administrative Scheduler. It checks the format of the objects, such as tables and columns. This group of packages accesses the following catalog tables:

- SYSIBM.SYSDATABASE
- SYSIBM.SYSTABLESPACE
- SYSIBM.SYSTABLES
- SYSIBM.SYSINDEXES
- SYSIBM.SYSCOLUMNS

► DSN5OEP:

This accesses catalog information for any table on which the EXPLAIN statement was issued. The catalog information includes information about columns, indexes, table spaces, and their statistics. This group of packages accesses the following tables:

- SYSIBM.SYSTABLES
- SYSIBM.SYSINDEXES
- SYSIBM.SYSKEYS
- SYSIBM.SYSCOLUMNS
- SYSIBM.SYSCOLDIST
- SYSIBM.SYSTABLESPACE
- SYSIBM.SYSTABLEPART
- SYSIBM.SYSTABSTATS
- SYSIBM.SYSROUTINES
- SYSIBM.SYSXMLRELS

- SYSIBM.SYSKEYTARGETS
- SYSIBM.SYSKEYTGTDIST
- SYSIBM.SYSINDEXSTATS

► DSN5ONPT:

This accesses the QMF control tables in case the current user does not have SYSADM privileges. This group of packages accesses the following catalog tables:

- Q.OBJECT\_DIRECTORY
- Q.OBJECT\_DATA

Query packages:

► DSN5OFM:

This accesses the catalog table SYSIBM.SYSVIEWS and obtains the DDL for any view or MQT.

Design Center packages:

► DSN5OWCC:

This is used by WCC client implementation to access WCC tables, catalog tables, profile tables, and explain tables under SYSIBM and DB2OSC. It is used by WCC client implementation to access WCC tables, catalog tables, profile tables, and explain tables under SYSIBM and DB2OSC. This group of packages accesses the following tables:

- DB2OSC.DSN\_WCC\_WORKLOADS
- DB2OSC.DSN\_WCC\_WL\_SOURCES
- DB2OSC.DSN\_WCC\_SOURCE\_DTL
- DB2OSC.DSN\_WCC\_WL\_AUTHIDS
- DB2OSC.DSN\_WCC\_EV\_HISTORY
- DB2OSC.DSN\_WCC\_MESSAGES
- DB2OSC.DSN\_WCC\_EP\_HISTORY
- DB2OSC.DSN\_WCC\_STMT\_TEXTS
- DB2OSC.DSN\_WCC\_STMT\_INSTS
- DB2OSC.DSN\_WCC\_INST\_SUMMY
- DB2OSC.DSN\_WCC\_STMT\_RUNTM
- DB2OSC.DSN\_WCC\_OBJ\_RUNTM
- DB2OSC.DSN\_WCC\_CAP\_TMP\_RS
- DB2OSC.DSN\_WCC\_TASKS
- DB2OSC.DSN\_WCC\_TB\_STATUS
- DB2OSC.DSN\_WCC\_WL\_INFO
- DB2OSC.DSN\_WCC\_STMT\_INFO
- DB2OSC.DSN\_WCC\_RP\_TBS
- DB2OSC.DSN\_WCC\_RP\_TBS\_HIS
- DB2OSC.DSN\_WCC\_RP\_IDXES
- DB2OSC.DSN\_WCC\_RP\_IDX\_HIS
- DB2OSC.DSN\_WCC\_RP\_STMTS
- DB2OSC.DSN\_FUNCTION\_TABLE
- DB2OSC.DSN\_PGROUPTABLE
- DB2OSC.DSN\_PTASK\_TABLE
- DB2OSC.PLAN\_TABLE
- DB2OSC.DSN\_STATEMNT\_TABLE
- DB2OSC.DSN\_PREDICAT\_TABLE
- DB2OSC.DSN\_STRUCT\_TABLE
- DB2OSC.DSN\_FILTER\_TABLE
- DB2OSC.DSN\_DETCOST\_TABLE
- DB2OSC.DSN\_SORT\_TABLE

- DB2OSC.DSN\_SORTKEY\_TABLE
- DB2OSC.DSN\_PGRANGE\_TABLE
- DB2OSC.DSN\_VIEWREF\_TABLE
- DB2OSC.DSN\_QUERY\_TABLE
- DB2OSC.DSN\_QUERY\_AUX
- DB2OSC.DSN\_STATEMENT\_CACHE\_TABLE
- DB2OSC.DSN\_STATEMENT\_CACHE\_AUX
- DB2OSC.DSN\_STATEMENT\_RUNTIME\_INFO
- DB2OSC.DSN\_OBJECT\_RUNTIME\_INFO
- DB2OSC.DSN\_STATEMENT\_RUNTIME\_INFO\_AUX
- SYSIBM.DSN\_PROFILE\_TABLE
- SYSIBM.DSN\_PROFILE\_HISTORY
- SYSIBM.DSN\_PROFILE\_ATTRIBUTES
- SYSIBM.DSN\_PROFILE\_ATTRIBUTES\_HISTORY
- SYSIBM.DSN\_STATEMENT\_RUNTIME\_INFO
- SYSIBM.DSN\_OBJECT\_RUNTIME\_INFO
- SYSIBM.DSN\_STATEMENT\_RUNTIME\_INFO\_AUX
- SYSIBM.PLAN\_TABLE
- SYSIBM.DSN\_STATEMNT\_TABLE
- SYSIBM.DSN\_PREDICAT\_TABLE
- SYSIBM.DSN\_STRUCT\_TABLE
- SYSIBM.DSN\_FILTER\_TABLE
- SYSIBM.DSN\_PGROUPTABLE
- SYSIBM.DSN\_PTASK\_TABLE
- SYSIBM.DSN\_DETCOST\_TABLE
- SYSIBM.DSN\_SORT\_TABLE
- SYSIBM.DSN\_SORTKEY\_TABLE
- SYSIBM.DSN\_PGRANGE\_TABLE
- SYSIBM.DSN\_VIEWREF\_TABLE
- SYSIBM.DSN\_QUERY\_TABLE
- SYSIBM.DSN\_QUERY\_AUX
- SYSIBM.DSN\_FUNCTION\_TABLE

► DSN5OWSP:

This is set by the WCC stored procedure implementation to access WCC tables, catalog tables, profile tables, and EXPLAIN tables under SYSIBM and DB2OSC. This group of packages accesses the following tables:

- DB2OSC.DSN\_WCC\_WORKLOADS
- DB2OSC.DSN\_WCC\_WORKLOAD\_SOURCES
- DB2OSC.DSN\_WCC\_SOURCE\_DETAIL
- DB2OSC.DSN\_WCC\_WORKLOAD\_AUTHIDS
- DB2OSC.DSN\_WCC\_TABLE\_STATUS
- DB2OSC.DSN\_WCC\_EVENT\_HISTORY
- DB2OSC.DSN\_WCC\_EXPLAIN\_HISTORY
- DB2OSC.DSN\_WCC\_MESSAGES
- DB2OSC.DSN\_WCC\_STATEMENT\_TEXTS
- DB2OSC.DSN\_WCC\_STATEMENT\_TEXTS\_AUX
- DB2OSC.DSN\_WCC\_STATEMENT\_INSTANCES
- DB2OSC.DSN\_WCC\_STATEMENT\_INSTANCES\_SUMMARY
- DB2OSC.DSN\_WCC\_TASKS
- DB2OSC.DSN\_WCC\_TASKS\_AUX
- DB2OSC.DSN\_WCC\_STATEMENT\_RUNTIME
- DB2OSC.DSN\_WCC\_OBJECT\_RUNTIME
- DB2OSC.DSN\_WCC\_WORKLOAD\_INFO
- DB2OSC.DSN\_WCC\_WORKLOADINFO\_AUX

- DB2OSC.DSN\_WCC\_STATEMENT\_INFO
- DB2OSC.DSN\_WCC\_STATEMENT\_INFO\_AUX
- DB2OSC.DSN\_WCC\_CAPTURE\_TEMP\_RESULT
- DB2OSC.DSN\_WCC\_CAPTURE\_AUX
- DB2OSC.DSN\_WCC\_REPORT\_TABLES
- DB2OSC.DSN\_WCC\_REPORT\_TABLES\_HISTORY
- DB2OSC.DSN\_WCC\_REPORT\_INDEXES
- DB2OSC.DSN\_WCC\_REPORT\_INDEXES\_HISTORY
- DB2OSC.DSN\_WCC\_REPORT\_STATEMENT\_INSTANCES
- DB2OSC.DSN\_FUNCTION\_TABLE
- DB2OSC.DSN\_PGROUPTABLE
- DB2OSC.DSN\_PTASK\_TABLE
- DB2OSC.PLAN\_TABLE
- DB2OSC.DSN\_STATEMENT\_TABLE
- DB2OSC.DSN\_PREDICAT\_TABLE
- DB2OSC.DSN\_STRUCT\_TABLE
- DB2OSC.DSN\_FILTER\_TABLE
- DB2OSC.DSN\_DETCOST\_TABLE
- DB2OSC.DSN\_SORT\_TABLE
- DB2OSC.DSN\_SORTKEY\_TABLE
- DB2OSC.DSN\_PGRANGE\_TABLE
- DB2OSC.DSN\_VIEWREF\_TABLE
- DB2OSC.DSN\_QUERY\_TABLE
- DB2OSC.DSN\_QUERY\_AUX
- DB2OSC.DSN\_STATEMENT\_CACHE\_TABLE
- DB2OSC.DSN\_STATEMENT\_CACHE\_AUX
- DB2OSC.DSN\_STATEMENT\_RUNTIME\_INFO
- DB2OSC.DSN\_OBJECT\_RUNTIME\_INFO
- DB2OSC.DSN\_STATEMENT\_RUNTIME\_INFO\_AUX
- SYSIBM.DSN\_PROFILE\_TABLE
- SYSIBM.DSN\_PROFILE\_HISTORY
- SYSIBM.DSN\_PROFILE\_ATTRIBUTES
- SYSIBM.DSN\_PROFILE\_ATTRIBUTES\_HISTORY
- SYSIBM.DSN\_STATEMENT\_RUNTIME\_INFO
- SYSIBM.DSN\_OBJECT\_RUNTIME\_INFO
- SYSIBM.DSN\_STATEMENT\_RUNTIME\_INFO\_AUX
- SYSIBM.PLAN\_TABLE
- SYSIBM.DSN\_STATEMENT\_TABLE
- SYSIBM.DSN\_PREDICAT\_TABLE
- SYSIBM.DSN\_STRUCT\_TABLE
- SYSIBM.DSN\_FILTER\_TABLE
- SYSIBM.DSN\_PGROUPTABLE
- SYSIBM.DSN\_PTASK\_TABLE
- SYSIBM.DSN\_DETCOST\_TABLE
- SYSIBM.DSN\_SORT\_TABLE
- SYSIBM.DSN\_SORTKEY\_TABLE
- SYSIBM.DSN\_PGRANGE\_TABLE
- SYSIBM.DSN\_VIEWREF\_TABLE
- SYSIBM.DSN\_QUERY\_TABLE
- SYSIBM.DSN\_QUERY\_AUX
- SYSIBM.DSN\_FUNCTION\_TABLE

Workload Advisor packages are:

- DSN5OWSA:

This accesses the following tables used by the Workload Statistics Advisor to manage the definition and content of workloads:

- DB2OSC.DSN\_WSA\_SESSIONS
- DB2OSC.DSN\_WSA\_DATABASES
- DB2OSC.DSN\_WSA\_TABLESPACES
- DB2OSC.DSN\_WSA\_TABLES
- DB2OSC.DSN\_WSA\_COLUMNS
- DB2OSC.DSN\_WSA\_COLGROUPS
- DB2OSC.DSN\_WSA\_CGFREQUENCIES
- DB2OSC.DSN\_WSA\_CGHISTOGRAMS
- DB2OSC.DSN\_WSA\_INDEXES
- DB2OSC.DSN\_WSA\_KEYS
- DB2OSC.DSN\_WSA\_KEYTARGETS
- DB2OSC.DSN\_WSA\_KEYTARGETGROUPS
- DB2OSC.DSN\_WSA\_KEYTGFREQUENCIES
- DB2OSC.DSN\_WSA\_KEYTGHISTOGRAMS
- DB2OSC.DSN\_WSA\_LITERALS
- DB2OSC.DSN\_WSA\_ADVICE

## C.5 Additional packages required by Optimization Expert

The Query packages are:

► AOC5OQA:

This accesses catalog information for referential constraints between tables. This group of packages accesses the following catalog tables:

- SYSIBM.SYSINDEXES
- SYSIBM.SYSKEYS
- SYSIBM.SYSRELS
- SYSIBM.SYSFOREIGNKEYS

► AOC5OQIA:

This accesses the catalog tables to get the referential constraints, table definition, and statistics. This group of packages accesses the following catalog tables:

- SYSIBM.SYSTABCONST
- SYSIBM.SYSKEYCOLUSE
- SYSIBM.SYSRELS
- SYSIBM.SYSFOREIGNKEYS
- SYSIBM.SYSTABLES
- SYSIBM.SYSCOLUMNS
- SYSIBM.SYSINDEXES
- SYSIBM.SYSKEYS
- SYSIBM.SYSCOLDIST
- SYSIBM.SYSINDEXPART
- SYSIBM.SYSTABLESPACE

The Workload Index Advisor packages are:

► AOC5OIA $n$ :

Where  $n=\{1-9, A\}$ . This accesses the Workload Index Advisor tables to perform workload-based Index Advisor process, and accesses the catalog tables to get referential constraints, table definitions, and statistics. This group of packages accesses the following tables:

- DB2OE.DSN\_WIA\_SESSION
- DB2OE.DSN\_WIA\_KEYS
- DB2OE.DSN\_WIA\_INDEXES
- DB2OE.DSN\_WIA\_STMT\_AUX
- DB2OE.DSN\_WIA\_TAB\_REF
- DB2OE.DSN\_WIA\_QBLOCK
- DB2OE.DSN\_WIA\_COL\_REF
- DB2OE.DSN\_WIA\_TABLES
- DB2OE.DSN\_WIA\_PREDICATE
- DB2OE.DSN\_WIA\_ERROR\_HIST
- DB2OE.DSN\_WIA\_TABREF\_IDX
- DB2OE.DSN\_WIA\_SESSION\_HIST
- DB2OE.DSN\_WIA\_COL\_SEQ\_KEY
- DB2OE.DSN\_WIA\_STMT
- DB2OE.DSN\_WIA\_COL\_SEQ
- DB2OE.DSN\_WIA\_GBOBDIST
- DB2OE.DSN\_WIA\_COLUMNS
- DB2OE.PLAN\_TABLE
- DB2OE.DSN\_STATEMNT\_TABLE
- DB2OE.DSN\_VIRTUAL\_INDEXES

Catalog tables:

- SYSIBM.SYSTABCONST
- SYSIBM.SYSKEYCOLUSE
- SYSIBM.SYSRELS
- SYSIBM.SYSFOREIGNKEYS
- SYSIBM.SYSTABLES
- SYSIBM.SYSCOLUMNS
- SYSIBM.SYSINDEXES
- SYSIBM.SYSKEYS
- SYSIBM.SYSCOLDIST
- SYSIBM.SYSINDEXPART
- SYSIBM.SYSTABLESPACE

► AOC5OWQA:

This accesses the following tables used by Workload Query Advisor to manage the Workload Query Advisor analysis session and corresponding operation:

- DSN\_WQA\_SESSION
- DSN\_WQA\_STMT
- DSN\_WQA\_WARNING
- DSN\_WQA\_EXCEPTION
- DSN\_WQA\_SESSION\_RULE\_REF
- DSN\_WQA\_WARNING\_LINE\_REF



# Related publications

The publications listed in this section contain more detailed discussion of the topics covered in this book.

## IBM Redbooks publications

For information about ordering these publications, see “How to get IBM Redbooks publications” on page 426. Note that some of the documents referenced here are available in softcopy only.

- ▶ *DB2 for z/OS and OS/390: Ready for Java*, SG24-6435
- ▶ *DB2 9 for z/OS Technical Overview*, SG24-7330
- ▶ *DB2 for z/OS Stored Procedures: Through the CALL and Beyond*, SG24-7083

## Other publications

The following publications are also relevant:

- ▶ *The Evolution of DB2 for z/OS Optimization* by Jay Bruce, IBM Systems Magazine, May-June 2006 issue
- ▶ *IBM DB2 Optimization Expert for z/OS Version 1.1 Getting Started*, GC19-1144-01
- ▶ *IBM Optimization Service Center for DB2 for z/OS Version 1.1 Getting Started*, GC19-1143-01
- ▶ *DB2 Version 9.1 for z/OS Administration Guide*, SC18-9840-00
- ▶ *DB2 Version 9.1 for z/OS Application Programming and SQL Guide*, SC18-9841-00
- ▶ *DB2 Version 9.1 for z/OS Application Programming Guide and Reference for JAVA*, SC18-9842-00
- ▶ *DB2 Version 9.1 for z/OS Codes*, GC18-9843-00
- ▶ *DB2 Version 9.1 for z/OS Command Reference*, SC18-9844-00
- ▶ *DB2 Version 9.1 for z/OS Data Sharing: Planning and Administration*, SC18-9845-00
- ▶ *DB2 Version 9.1 for z/OS Diagnosis Guide and Reference*, LY37-3218-00
- ▶ *DB2 Version 9.1 for z/OS Diagnostic Quick Reference*, LY37-3219-00
- ▶ *DB2 Version 9.1 for z/OS Installation Guide*, GC18-9846-00
- ▶ *DB2 Version 9.1 for z/OS Introduction to DB2*, SC18-9847-00
- ▶ *DB2 Version 9.1 for z/OS Licensed Program Specifications*, GC18-9848-00
- ▶ *DB2 Version 9.1 for z/OS Messages*, GC18-9849-00
- ▶ *DB2 Version 9.1 for z/OS ODBC Guide and Reference*, SC18-9850-00
- ▶ *DB2 Version 9.1 for z/OS Performance Monitoring and Tuning Guide*, SC18-9851-00
- ▶ *DB2 Version 9.1 for z/OS RACF Access Control Module Guide*, SC18-9852-00
- ▶ *DB2 Version 9.1 for z/OS Reference for Remote DRDA Requesters and Servers*, SC18-9853-00

- ▶ *DB2 Version 9.1 for z/OS SQL Reference*, SC18-9854-00
- ▶ *DB2 Version 9.1 for z/OS Reference Summary*, SX26-3854-00
- ▶ *DB2 Version 9.1 for z/OS Utility Guide and Reference*, SC18-9855-00
- ▶ *DB2 Version 9.1 for z/OS What's New*, GC18-9856-00
- ▶ *DB2 Version 9.1 for z/OS XML Extender Administration and Programming*, SC18-9857-00
- ▶ *DB2 Version 9.1 for z/OS XML Guide*, SC18-9858-00
- ▶ *Spatial Support for DB2 for z/OS User's Guide and Reference*, GC19-1145
- ▶ *Program Directory for IBM DB2 9 for z/OS*, GI10-8737-00
- ▶ *Program Directory for IBM DB2 V9.1 for z/OS DB2 Management Clients Package*, GI10-8738-00
- ▶ *Program Directory for IBM DB2 Accessories Suite for z/OS*, GI10-8749-00
- ▶ *Program Directory for DB2 Optimization Expert for z/OS*, GI10-8755-00
- ▶ *z/Architecture Principles of Operations*, SA22-7832-04

## Online resources

DB2 Tools for z/OS and IMS Tools Web site:

<http://www.ibm.com/software/data/db2imstools/overview.html>

## How to get IBM Redbooks publications

You can search for, view, or download IBM Redbooks publications, IBM Redpapers, Technotes, draft publications, additional materials, and order hardcopy IBM Redbooks publications at this Web site:

[ibm.com/redbooks](http://ibm.com/redbooks)

## Help from IBM

IBM Support and downloads:

[ibm.com/support](http://ibm.com/support)

IBM Global Services:

[ibm.com/services](http://ibm.com/services)

# Index

## Numerics

5655-S19 10

## A

A.CLOSEDATE 187  
access path xxiii, 4, 43, 159, 164, 223, 271, 292, 323, 344, 380, 411  
    graphical representation 324  
    table access part 340  
Access Plan Graph 171  
    detailed explanation 45  
access plan graph 10, 55, 245, 359, 385  
ACCESSTYPE 212  
Admin Scheduler 406  
    started task 406  
administrative scheduler 83, 97, 137, 352, 406  
ADMTPROC 406  
Advisors tab 254, 298, 356  
    existing project 307  
alias 151, 410  
ALL 136, 225, 336, 401  
annotated query  
    show examples 385  
APIs 5  
application 5, 84, 98, 166, 305, 321, 326, 345, 379  
Architecture 132  
attribute 51  
ATTRVALUE table 367  
authorization 128, 168, 346, 383  
autonomic computing 3  
autonomic computing blueprint 4

## B

BIND 115, 150, 398, 410  
bind 84, 98, 204  
boolean term 197  
BOTH 226  
buffer pool 96, 134, 203, 372  
buffer pools 82, 96

## C

C.STATUS 187  
CALL 83, 98  
candidate index 277  
    overall estimated performance benefit 277  
CARDF 184, 226, 325  
cardinality 50, 184, 224, 326, 362, 385  
CASE 146, 399, 407  
catalog table 420  
CO.TBCREATOR 237  
COLGROUP 226, 335, 363, 403  
collection-id 218

COLLID 132, 150, 218, 398  
COLTYPE 237  
column cardinality 51, 195, 225, 336, 362  
column expression 401  
column group  
    cardinality 362  
    detail 358  
Comments xxvi  
COMPLETE RUNSTATS 231  
COMPLETE STATISTICS 228  
components xxiv, 10, 161, 283, 323, 381  
compression 224  
connected DB2 system  
    DB2 commands 83, 97  
connection 26, 32, 81, 95, 113  
constraint 270, 304  
Control Center 29, 87, 101, 281, 410  
COPY 398  
COUNT 226, 284, 327, 363, 401  
CPU Time 158  
CPU time  
    substantial percentage 204  
CREATEIN 410  
CREATOR 150, 237, 402, 407  
CURRENT 168, 225, 318, 398, 410  
CURRENT SQLID 168, 411  
customer adaptation 7

## D

DASD space 269  
    requirement 270  
    restriction 269  
data sharing xxiii, 135  
DATE 225, 316, 398  
DB2 9 xxiii, 97, 333, 344, 406  
DB2 9 Command Reference SC18-9844 97  
DB2 command 83, 97, 111, 135  
DB2 commands 97  
DB2 Connect 11, 86, 100  
DB2 object 412, 417  
DB2 Optimization Expert for z/OS 10, 102  
DB2 Optimization Service Center xxiii, 3, 5, 9, 81  
DB2 privileges 22, 251  
DB2 query  
    transformation 182  
DB2 subsystem 14, 32, 135, 153, 301  
DB2 system xxiv, 82, 96, 167, 344  
    following tasks 82, 96  
DB2 V9  
    execution 131, 228  
    Profile Monitor 131  
    table DSN\_STATEMNT\_TABLE 271  
DB2 Version  
    6 204

9.1 82, 96, 145, 150, 224, 397, 408

DBA 5, 187, 382

DBADM 410

DBCTRL 410

DD DISP 85, 99

DDF 17, 82, 96, 111

DDL 200, 248, 371, 420

DEFAULT 230

default userid 406

DELETE 133, 400

delete 145, 159, 203, 257, 363, 381

DEPT table 45, 248, 304

    MGRNO column 305

    new index 249

DEPTNO column 50

DISTARRANG table 359

    FIELD3 predicates 359

DRDA 346

drop-down menu 64, 245, 301

    Access Plan Graph 70

    Run Statistics Advisor 73

DSN1ADMT 406

DSN2ADMT 406

DSN8.DSN8EXP 83, 97

DSNA679I 406

DSNDB06 236, 257

DSNJU004 111

DSNTEJXP 97

DSNTEP2 218

DSNTIAUL 399

DSNTIJCC 97

DSNTIJMS 83, 98

DSNTIJOS 82, 97

DSNTIJRA 82, 406

DSNTIJSG 83, 97

DSNTIJUZ 111

DSNTIPP 410

DSNWZP 83, 202

DSNZPARM 111, 202, 236, 387, 406

Dynamic SQL 131

dynamic SQL

    statement 363

dynamic statement cache

    captures workloads 83, 97

## E

E.FIRSTNME 301

EDM pool 133

EMP table 45, 252, 305

    WORKDEPT column 252

EMPLOYEE\_CNT column 265

    single index 267

environment 4, 79, 83, 97, 109, 131, 166, 379, 410

EXPLAIN 83, 97, 150, 167, 228, 247, 355, 397, 407, 410

Explain xxiii, 4, 45, 152, 170, 223, 246, 297, 380, 411

expression 225, 305, 397

external 139, 225

extract 5, 56, 201, 239

## F

fetch 146, 196

Firstkey cardinality 239

fixpack version 87, 101, 128

FMID 9

foreign key 268, 304, 385

function xxiii, 82, 97, 136, 152, 167, 268, 355, 383

## G

GA version 87, 101, 115

    full code replacement 87

given SQL

    first exception 135

Graph reports 220

GROUP BY 188, 284, 402

## H

H2AG110 9

handle 310

HFS 85, 100

HFS directory 88, 102

Hierarchical File System 87

HISTOGRAM 226, 363

Histogram statistics 236

host variable 225, 326

host variables 197, 327

HTML 177, 328

## I

IBM DB2 Accessories Suite 87

IBM DB2 Optimization Expert 95

IBM DB2 Version 9.1 86, 100

IBM Redbooks xxvi, 83, 324

IFCID 0318 150, 383

IFCID 318 139

IMS xxiii

IN 153, 229, 296, 398

index xxvi, 10, 52, 131, 150, 179, 226, 248, 316, 325, 358, 381, 398

Index Advisor 7, 151, 243, 297, 343, 381, 411

    Index statements 266

    not enough data 260

    Options button 255, 370

index list 265, 381

    tick boxes 271

Index reports 195

index scan 252, 325

indexable 315

indexable predicate 316

index-only access 249

inner table 54, 212, 296

    table space scan 212, 297

INSERT 133

insert 209

installation xxiii, 7, 81, 95, 167, 223, 250, 410

installation wizard 87, 101

internal xxiii, 135, 203

IP address 133, 346, 383

IS 230  
IT requirements 4

## J

Java xxiv, 83, 97  
JAVA SC18-9842 83, 98  
Java stored procedures 83, 98  
JCC 84, 98  
JDBC 84, 98  
join graph 206

## K

KB 23, 255  
key column 53  
keyword 145, 151, 336

## L

LEAST 226  
LEFT OUTER JOIN 305  
let 324  
LIKE 153, 230, 315, 362  
list 14, 32, 117, 132, 154, 164, 228, 286, 346, 381, 397, 406, 410  
LOAD 399  
LOB 23, 413  
LOB table 25, 413  
LOB table space 413  
    DSNSTLTS 418  
local predicate 216  
LOCATE 318  
log 111, 202

## M

maintenance 271  
Mass-explain procedure 401  
Matching Index Scan  
    matching column 249  
materialized query tables 168  
MAX\_FREQ 184, 325, 359  
MCARD statistic 336  
MGRNO column 305  
MODIFY 82, 97, 408  
Monitor Profile  
    List window 351  
Monitor profile 410  
monitor profile 134, 344, 382, 411  
    specification 347  
MQT 168, 420  
multi-column cardinality 195, 238, 339

## N

nested loop join 45, 296  
New Profile Monitor  
    Set 142  
next screen 89, 103  
NLJOIN object 54  
node 47, 173, 325

Node Descriptor 47, 172, 325  
Normal Monitor 133, 382  
    aggregated results 136  
normal monitoring  
    complete reporting 383  
    medium reporting 383  
NOT PADDED 251  
NULL 230

## O

Object 150, 265  
OPHTINT 218  
optimization xxiii, 1, 3, 31, 204, 225, 276, 321  
Optimization Expert 72, 81, 95, 110, 131, 149, 163, 223, 243, 283, 343, 397  
    convenient processing 401  
    other work 264  
optimization hint 204  
Optimization hints 204  
Optimization Service Center 31, 81, 95, 163, 344  
Optimizer estimate 325  
options 22, 42, 86, 101, 121, 132, 150, 165, 233, 247, 300, 350, 383  
ORDER 56, 195, 284, 329, 399  
ORDER BY 284, 402  
Ordering information 53  
OSC client 83  
outer table 193, 296  
    qualifying row 297

## P

package 10, 56, 87, 102, 138, 154, 218, 287, 383, 398, 419  
PADDED 251  
page size 194, 255  
parallelism 168  
parameter marker 198, 225  
PART 284  
partial RUNSTATS 354  
partitions 49, 193  
PATH 85, 99, 219  
Performance xxiv, 29, 145, 150, 224, 333, 383, 397  
performance xxiii, 4, 81, 95, 131, 149, 187, 223, 244, 292, 325, 343, 380, 397  
PL.CREATOR 402  
PL.MIXOPSEQ 402  
PL.QBLOCKNO 402  
PL.QUERYNO 402  
plan 10, 45, 84, 98, 134, 151, 245, 287, 324, 359, 380  
PLAN\_TABLE 22, 150, 172, 297, 324, 369, 398, 413  
populate 372  
PR.EXPLAIN\_TIME 402  
predicate 10, 54, 182, 228, 249, 304, 362, 385, 398  
predicate information 399  
Predicate reports 196  
PRIMARY\_ACESSTYPE 204  
privilege 410  
problem query 26, 198, 234, 321, 323, 379  
Profile Monitor 131, 149, 240, 382

- function 140
- performance graph 383
- progress indicator 62, 247, 289
- pull-down menu 57, 250, 292

## Q

- QMF 6, 149, 230, 250, 388, 420
- QoS 5
- QUALIFIER 399
- Quality of Service 5
- query xxiii, 1, 3, 31, 129, 151, 164, 227, 250, 295, 321, 323, 344, 379, 397
- Query Advisor
  - Preferences panel 307
- Query Annotation 163, 180, 325, 358, 380
- query annotation 189
- query optimization xxiii, 3
- query performance 5, 366, 387
- query project 36, 164, 324
  - current SQL statement 198
  - separate directory 177
- Query reports 191
- query reports 191
- Query tool 169
- Query Warehouse 6
- QUERYNO 150, 218, 400

## R

- RACF 82, 96, 406
- Real 150
- REBIND 204
- rebind 363
- recovery xxiii
- referential constraint 304
- referential integrity 264
- REGION 85, 99
- REOPT 204, 229
- REPAIR 228
- REPAIR RUNSTATS 231
- REPORT 135, 336, 363
- repository xxiii, 151
- requirements 4, 82, 96, 128, 410
- restart 136
- result set 133, 339, 402
- return 35, 220, 252, 316, 325, 351, 380
- REXX 399
- RI 305
- RID 249
- RIDs 334
- RUNSTATS 83, 97, 195, 223, 336, 354, 381, 401
- RUNSTATS cost 233, 401
- RUNSTATS statement 195, 236, 336, 363
- RUNSTATS syntax 224
- RUNSTATS TABLESPACE
  - DSNDB06.SYSDBASE 236
  - SW01DB1.DISTARRA 363
  - TPTEST.TPTSTTS1 336
- Runstats utility 50

## S

- SCHEMA 168
- Schema 168, 310
- schema 7, 128, 151, 236, 410
- scroll bar 39, 164, 265, 302
- SDSNEXIT 85, 99
- select count 236
- SELECT L\_ORDERKEY 189
- SELECT N\_NAME 284
- Server 4, 32, 86, 100, 350
- Service SQL 198, 384
  - data 199
  - info 199
  - information 201, 384
  - stream 199
  - tool 388
  - window 198
- Service SQL information 199
- SET 219, 239, 410
- setup.exe 87, 101
- SG247421.CUSTOMERS 187, 325
  - 192960 194, 329
  - C 196
- SHRLEVEL 237, 363
- SHUTDOWN 82, 97, 408
- side 39, 137, 198, 228, 313, 384, 400
- single query 8, 37, 177, 228
- single SQL statement 10, 37, 292
  - Index Advisor 27, 244
  - Query Advisor operating 300
- SMP/E 86, 96, 115
- special register 168, 225
- SPUFI 250
- SQL 10, 31, 81, 95, 113, 131, 149, 164, 223, 284, 324, 346, 397
- SQL query
  - parallelism 201
  - window 170
- SQL statement 10, 37, 133, 164, 225, 244, 292, 324, 358
  - actual operation 167
  - Catalog information 264
  - estimated total cost 270
  - portion 198
  - relevant information 175
  - text 39, 143, 175, 244, 296, 375
  - text box 39, 245
  - total weight 276
- SQL statement text 164, 245, 294, 324, 375
- SQL text 133, 149, 169, 240, 302
  - direct text entry window 151
- SQLCODE 211, 250
- SQLJ 84, 98
- stage 1 185, 316, 339
- stage 1 predicate 197, 316
- Stage 2 305, 398
- Started-task job 85, 99
- statement 6, 31, 83, 97, 133, 150, 164, 225, 291, 324, 344, 385, 411
- statement cache 6, 66, 83, 97, 154, 175, 275, 388, 413
- Static SQL 131, 150

- static SQL
  - population 401
  - project 71
- statistics xxiii, 6, 45, 131, 150, 181, 223, 252, 297, 333, 343, 381, 397, 419
- Statistics Advisor xxiii, 4, 71, 151, 187, 223, 251, 297, 343, 381, 397, 411
  - individual query 381
- statistics status 238
- STATUS 135, 187, 325, 407
- STOP DB2 140
- stored procedure 83, 97, 167, 234, 281, 412
- subquery 244, 318, 370
- subselect 181, 359
- SUBSTR 400
- Subsystem-wide Monitor 135
- SYSADM 23, 83, 97, 410
- SYSADM authority 410
- SYSCOLUMNS 225, 419
- SYSCTRL 410
- SYSIBM 84, 98, 139, 203, 225, 257, 335, 398, 406, 413
- SYSIBM.SQLCAMESSAGE 84, 98
- SYSIBM.SQLFOREIGNKEYS 84, 98
- SYSIBM.SQLGETTYPEINFO 84, 98
- SYSIBM.SQLPRIMARYKEYS 84, 98
- SYSIBM.SQLPROCEDURECOLS 84, 98
- SYSIBM.SQLPROCEDURES 84, 98
- SYSIBM.SQLSPECIALCOLUMNS 84, 98
- SYSIBM.SQLTABLEPRIVILEGES 84, 98
- SYSIBM.SQLTABLES 84, 98
- SYSIBM.SQLTABLETYPES 84, 98
- SYSIBM.SQLTTYPEINFO 84, 98
- SYSIBM.SQLUUDTS 84, 98
- SYSIBM.SQTCOLPRIVILEGES 84, 98
- SYSIBM.SQTFORIGNKEYS 84, 98
- SYSIBM.SQTPRIMARYKEYS 84, 98
- SYSIBM.SQTPROCEDURECOLS 84, 98
- SYSIBM.SQTPROCEDURES 84, 98
- SYSIBM.SQTSPECIALCOLUMNS 84, 98
- SYSIBM.SQTSTATISTICS 84, 98
- SYSIBM.SQTTABLEPRIVILEGES 84, 98
- SYSIBM.SQTUDTS 84, 98
- SYSIBM.SYSCOLDIST 419
- SYSIBM.SYSCOLUMNS 230, 419
  - COLGROUP 238
  - index SYSIBM.DSNDX01 KEYCARD 238
- SYSIBM.SYSDUMMY1 257
- SYSIBM.SYSFOREIGNKEYS 423
- SYSIBM.SYSINDEXES 239, 419
- SYSIBM.SYSKEYCOLUSE 423
- SYSIBM.SYSKEYS 419
- SYSIBM.SYSRELS 423
- SYSIBM.SYSROUTINES 419
- SYSIBM.SYSTABCONST 423
- SYSIBM.SYSTABLEPART 419
- SYSIBM.SYSTABLES 236, 419
- SYSIBM.SYSTABLESPACE 419
- SYSIBM.UINDEXES 84, 98
- SYSIN 234, 399

- SYSPACKAGE 150, 398, 419
- SYSPACKSTMT 150
- SYSPRINT 85, 99, 399
- SYSPROC 83, 97, 412
- SYSPROC.ADMIN\_COMMAND\_DB2 97
- SYSPROC.ADMIN\_TASK\_REMOVE 97
- SYSPROC.ADMIN\_TASK\_SCHEDULE 97
- SYSPROC.DSNACCMD 83, 97
- SYSPROC.DSNUTILU 97
- SYSPROC.DSNWZP 97
- SYSPROC.OSC\_EXECUTE\_TASK 97
- SYSPROC.OSC\_RUNSQL 97
- SYSTABAUTH 419
- SYSTABLEPART 419

## T

- T.SCHEDULED\_TASKID 407
- table 10, 45, 82, 96, 120, 137, 150, 167, 225, 248, 287, 325, 348, 398, 409
- table cardinality 362
- table GLWTDPT 265
  - EMPLOYEE\_CNT column 267
- table PAOLOR4.DEPT 304
  - columns DEPTNO 304
- table PAOLOR4.EMP 304
- table report 196
- Table space
  - AOCOETS 418
  - DSNMPLTS 413
  - DSNOSCTS 413
  - DSNPFTS 415
  - DSNSUMTS 413
  - DSNWSATS 417
- table space 23, 45, 194, 250, 297, 413
  - creation 25
  - name 24
  - scan 45, 212, 252, 297
- tables 19, 40, 82, 96, 134, 154, 224, 269, 287, 325, 354, 382, 398, 409
- TB.CREATOR 237
- TBNAME 237, 401
- TCO 5
- thread 138, 408
- TIME 85, 99
- TIMESTAMP 135, 398
- Total Cost of Ownership 5
- TRACE 82, 97, 326, 408
- traces 134
- transitive closure 182
- tree structure 249
- trigger 137, 168, 324
- triggers 164
- TYPE 146, 151, 197, 226, 400, 406

## U

- UNION 402
- UNION ALL 189
- UNIQUE 228
- universal table space 194

UNLOAD 399  
UPDATE 133, 239

## V

V91C.JAVAENV 85, 99  
V91CWLMJ PROC DB2SSN 85, 99  
VALUE 197, 400  
VARCHAR 251  
variable 84, 98, 225, 326  
vendor packaged solutions 4  
VERSION 146, 218, 399  
Version xxiv, 4, 82, 96, 145, 150, 200, 224, 397, 408  
versions 87, 101, 159, 271  
view 50, 83, 111, 135, 151, 176, 305, 333, 347, 420  
View SQL Statement 145, 151  
views 50  
Visual Explain 4, 68, 171  
Visual plan hint 204  
VPH tab 206  
    graph area 221  
VPH window 213  
VSAM RRDS 406

## W

Web site 86, 100  
What-If Analysis 268  
WHEN 146, 399, 407  
WIA 344  
Windows desktop 87, 101, 110  
WLM 83, 97  
workfile 45  
Workload Index Advisor 30, 257, 344, 381, 424  
    default 271  
    option 260  
    Recommendations panel 265  
Workload Project  
    Statements tab 155  
workload project 144, 153, 164, 284, 353  
    query tools 164  
    Statement tab 368  
Workload Statistics Advisor 29, 227, 344, 381, 401, 423  
    convenient processing 241  
WSA 344

## Z

z/OS xxiii, 4, 81, 95, 110, 135, 150, 220, 223, 243, 283,  
333, 344, 383, 397, 408  
zIIP 383  
ZIPCODE 187, 324  
zSeries 102





# IBM DB2 9 for z/OS: New Tools for Query Optimization

(1.0" spine)

0.875" x 1.498"

460 <-> 788 pages







**Redbooks®**

# IBM DB2 9 for z/OS: New Tools for Query Optimization

**Install and customize  
Optimization Service  
Center and  
Optimization Expert**

**Collect workload data  
and identify issues**

**Tune workloads and  
individual critical  
queries**

The cost-based optimizer of IBM DB2 for z/OS has continually evolved since its initial inception. Support has included new optimization algorithms, join methods, complex relational data structures, such as star schemas. DB2 for z/OS V8 has provided access path visualization and a Statistics Advisor function through a new Visual Explain.

DB2 9 for z/OS starts addressing the challenge of reducing cost of ownership by extending query optimization through expert-based query and workload analysis, server enhancements, and an Index Advisory function. These functions are delivered with two tools: DB2 Optimization Service Center and DB2 Optimization Expert.

This book helps you understand the installation, customization, and usage aspects of the tools. You are guided through scenarios of gradually increasing complexity where the functions of the tools are exploited for query optimization.

**INTERNATIONAL  
TECHNICAL  
SUPPORT  
ORGANIZATION**

**BUILDING TECHNICAL  
INFORMATION BASED ON  
PRACTICAL EXPERIENCE**

IBM Redbooks are developed by the IBM International Technical Support Organization. Experts from IBM, Customers and Partners from around the world create timely technical information based on realistic scenarios. Specific recommendations are provided to help you implement IT solutions more effectively in your environment.

**For more information:  
[ibm.com/redbooks](http://ibm.com/redbooks)**

SG24-7421-00

ISBN 0738488615