Redbooks
ibm.com/redbooks

Modernize Your IBM DB2 for IBM z/OS Maintenance with Utility Autonomics

Dean Brown
Carlos Gomes
Arthur Marais
Hennie Mynhardt

Information Management

IBM

Redbooks

International Technical Support Organization

Modernize Your IBM DB2 for IBM z/OS Maintenance with Utility Autonomics

August 2015

SG24-8304-00

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

First Edition (August 2015)

This edition applies to Version 11 and Version 10 of DB2 for z/OS.

© Copyright International Business Machines Corporation 2015. 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

Notices vii
Trademarks viii

IBM Redbooks promotions ix

Preface xi
Authors xi
Now you can become a published author, too! xii
Comments welcome xiii
Stay connected to IBM Redbooks xiii

Chapter 1. The need for autonomics in database management 1
1.1 Overview 2
1.2 Why autonomics and modernization 2
1.2.1 The mainframe skills gap 3
1.2.2 What is autonomics and how it can solve business problems 3
1.3 DB2 autonomics 4
1.3.1 Autonomics Director for DB2 5
1.3.2 Maintenance windows 6

Chapter 2. Product overview: Management Console and IBM DB2 Utilities Solution Pack 9
2.1 IBM Management Console for IMS and DB2 for z/OS 10
2.1.1 Architecture 11
2.1.2 Benefits 12
2.1.3 Plug-ins 13
2.2 DB2 Utilities Solution Pack 13
2.2.1 Why use Solution Packs? 14
2.2.2 Products overview 15
2.2.3 DB2 Automation Tool for z/OS 15
2.2.4 DB2 Utilities Enhancement Tool for z/OS 16

Chapter 3. Business scenario and technical challenges 17
3.1 Obstacles to autonomic computing 18
3.2 The solution 18
3.3 A typical business case for DB2 Utility Autonomics 19
3.3.1 Challenges 19
3.3.2 Meeting these challenges 20
3.4 A typical traditional DB2 maintenance scenario: passive autonomics 21
3.5 The implementation process 22
3.5.1 Implementing an ACTIVE strategy for your DB2 maintenance tasks 22
3.6 Conclusion 26

Chapter 4. Installing and configuring the DB2 Management Console 27
4.1 IBM software components required for the installation 28
4.1.1 IBM Installation Manager for z/OS V1.4.0 (5655-IMZ) 28
4.1.2 IBM Management Console for IMS and DB2 for z/OS, V1.1 (5655-TAC) 28
4.2 Using the IBM Tools Customizer to install the DB2 metadata 28
4.3 Installation and customization of the web server on z/OS 35

© Copyright IBM Corp. 2015. All rights reserved.

4.3.1 Setting up WLM for our installation 35
4.3.2 Creating the Unix System Services environment 37

Chapter 5. IBM DB2 Utilities Solution Pack for z/OS configuration 41
5.1 Configuration overview 42
5.2 DB2 Automation Tool Customization 42
5.2.1 Tools Customizer steps for DB2 Automation Tool 42
5.2.2 More Automation Tool configuration steps 55
5.3 DB2 Utilities Enhancement Tool Customization configuration 56
5.3.1 Tools Customizer steps for DB2 Utilities Enhancement Tool 56
5.3.2 More Utilities Enhancement Tool configuration steps 63

Chapter 6. Creating the autonomics infrastructure 65
6.1 Profiles 66
6.1.1 Object Profiles 66
6.1.2 Utility Profiles 75
6.1.3 Exception Profiles 76
6.1.4 Job Profiles 77

Chapter 7. Configuring maintenance windows with Management Console and Autonomics Director 81
7.1 Introduction 82
7.2 Management Console for DB2 83
7.2.1 Defining a new maintenance window using Management Console 84
7.2.2 Adding a maintenance window to the schedule 86
7.2.3 Scheduling a maintenance window 87
7.3 Autonomics Console 89
7.3.1 Creating a maintenance window using Autonomics Console 90
7.3.2 Creating a time period using Autonomics Console 92
7.3.3 Scheduling a maintenance window using Autonomics Console 96
7.3.4 Define the base period 97
7.3.5 Define Job Profiles 98

Chapter 8. Running it all together: Automation Tool, UET, Management Console, Autonomics Director 101
8.1 Synergy with the DB2 Automation Tool 102
8.1.1 Building the process 102
8.1.2 Building the Build Job 102
8.2 Execution 105
8.2.1 Executing the Build Job 105
8.2.2 Executing the Autonomics Director 106
8.2.3 Autonomics Director and autonomics leading practices 109

Chapter 9. Benefits and results of a fully autonomic solution 111
9.1 Business challenges 112
9.2 First steps 112
9.3 Second step 112
9.4 A more general problem: The aging workforce 113
9.5 Autonomics benefits 113

Appendix A. Hints and tips 115
DB2 Stored Procedures 116
APF Authorization Failure in Automation Tool 116
The Management Console and JCC packages 116
Automation Tool and Management Console widgets 116

What constitutes the Autonomics Director 117
Management Console terms 117
Management Console steps towards Active Autonomics 117

Appendix B. DB2 Autonomics Director tables 119
Concepts 120
List of tables and their descriptions 120

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 grant 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 websites are provided for convenience only and do not in any manner serve as an endorsement of those websites. The materials at those websites are not part of the materials for this IBM product and use of those websites 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.

Any performance data contained herein was determined in a controlled environment. Therefore, the results obtained in other operating environments may vary significantly. Some measurements may have been made on development-level systems and there is no guarantee that these measurements will be the same on generally available systems. Furthermore, some measurements may have been estimated through extrapolation. Actual results may vary. Users of this document should verify the applicable data for their specific environment.

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.

© Copyright IBM Corp. 2015. All rights reserved.

Trademarks

IBM, the IBM logo, and ibm.com are trademarks or registered trademarks of International Business Machines Corporation in the United States, other countries, or both. These and other IBM trademarked terms are marked on their first occurrence in this information with the appropriate symbol (® or ™), indicating US registered or common law trademarks owned by IBM at the time this information was published. Such trademarks may also be registered or common law trademarks in other countries. A current list of IBM trademarks is available on the Web at http://www.ibm.com/legal/copytrade.shtml

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

DB2®
IBM®
IBM z™
IBM z Systems™
IMS™
RACF®
Redbooks®
Redbooks (logo) ®
z Systems™
z/OS®

The following terms are trademarks of other companies:

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

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

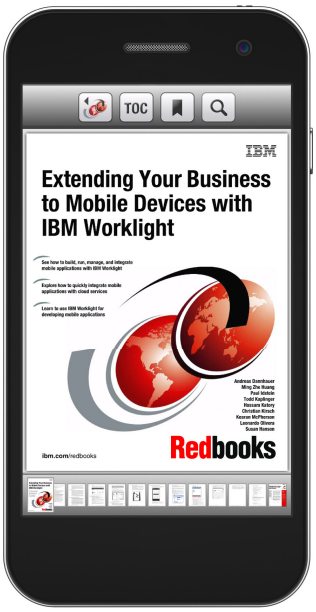
Other company, product, or service names may be trademarks or service marks of others.

Find and read thousands of IBM Redbooks publications

- Search, bookmark, save and organize favorites
- Get up-to-the-minute Redbooks news and announcements
- Link to the latest Redbooks blogs and videos

Get the latest version of the Redbooks Mobile App

iOS **Download Now** Android



Promote your business in an IBM Redbooks publication

Place a Sponsorship Promotion in an IBM® Redbooks® publication, featuring your business or solution with a link to your web site.

Qualified IBM Business Partners may place a full page promotion in the most popular Redbooks publications. Imagine the power of being seen by users who download millions of Redbooks publications each year!

ibm.com/Redbooks
About Redbooks → Business Partner Programs

THIS PAGE INTENTIONALLY LEFT BLANK

Preface

IBM® DB2® for IBM z/OS® helps lower the cost of managing data by automating administration, increasing storage efficiency, improving performance, and simplifying the deployment of virtual appliances. By automating tasks such as memory allocation, storage management, and business policy maintenance, DB2 is able to perform many management tasks itself, freeing up Database Administrators to focus on new projects.

This IBM Redbooks® publication introduces autonomics for DB2 for z/OS. IBM provides several different components that, when combined, can create an autonomic database environment. All these respective components cover certain aspects of autonomics, which can collaborate into one coherent solution. In our evolution of autonomics and the need to move to smarter systems there has been a bigger drive to the concept of “Active” versus “Passive” autonomics. With the inclusion of the IBM Management Console for IMS™ and DB2 for z/OS and the Autonomics Director, it is now easier than ever to make that transition by leveraging the strength of the DB2 Utilities Solution Pack for z/OS all in one standardized and centralized interface.

This publication guides you through the business reasons for adopting autonomic solutions, and provides step-by-step guidance to implement these capabilities in your DB2 for z/OS configuration. This publication is of interest primarily to DB2 Database Administrators and DB2 Systems Programmers, and for anyone looking to understand the benefits of DB2 autonomic solutions.

Authors

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

Dean Brown is a Lead Database Administrator in IBM US. He has 30 years of IT experience and 15 years of database experience. Dean holds a degree in Computer Science from the University of San Francisco.

Carlos Gomes is a zAnalytics Technical Sales Specialist in IBM Brazil. He is certified as an IBM IT Specialist in Information Management and IBM DB2 for z/OS. He has 10 years of experience in IT with 8 years of experience with DB2 for z/OS. He holds a degree in Business Management from FATEC. Carlos co-authored *Managing DB2 for z/OS Utilities with DB2 Tools Solution Packs*, SG24-8046.

© Copyright IBM Corp. 2015. All rights reserved.

Arthur Marais is a DB2 for z/OS Tools Specialist working with DB2 in IBM US. He has held roles working with DB2 as a Database Administrator, Systems Programmers, and Developer for the past 25 years.

Hennie Mynhardt is an IBM Senior Certified Consulting IT Specialist with the IBM Software Group. He has led and worked on various technical projects for database clients in the US and other countries. His special interests are systems performance tuning and backup and recovery. He currently provides technical consulting and pre-sales and post-sales support for DB2 for z/OS Engine and Tools. Hennie has co-authored multiple IBM Redbooks publications about DB2.

This project was led by:

- Martin Keen

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

- Robert Haimowitz
- Christopher Holtz
- Terrie Jacopi
- Badri Krishnamoorthy
- Jim Reed

Now you can become a published author, too!

Here's an opportunity to spotlight your skills, grow your career, and become a published author—all at the same time! Join an ITSO residency project and help write a book in your area of expertise, while honing your experience using leading-edge technologies. Your efforts will help to increase product acceptance and customer satisfaction, as you expand your network of technical contacts and relationships. Residencies run from two to six weeks in length, and you can participate either in person or as a remote resident working from your home base.

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

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** review Redbooks form found at:

  ibm.com/redbooks

- Send your comments in an email to:

  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

Stay connected to IBM Redbooks

- Find us on Facebook:

  http://www.facebook.com/IBMRedbooks

- Follow us on Twitter:

  https://twitter.com/ibmredbooks

- Look for us on LinkedIn:

  http://www.linkedin.com/groups?home=&gid=2130806

- Explore new Redbooks publications, residencies, and workshops with the IBM Redbooks weekly newsletter:

  https://www.redbooks.ibm.com/Redbooks.nsf/subscribe?OpenForm

- Stay current on recent Redbooks publications with RSS Feeds:

  http://www.redbooks.ibm.com/rss.html

1

The need for autonomics in database management

In this chapter, we discuss the technical challenges that today's DB2 System and Database Administrators face to meet their business requirements. These include a growing size of DB2 table space objects and how to perform maintenance in environments with shrinking batch windows and to do more with fewer resources.

This chapter contains the following sections:

- Overview
- Why autonomics and modernization
- DB2 autonomics

© Copyright IBM Corp. 2015. All rights reserved.

1.1 Overview

In April 2014, IBM celebrated the 50th anniversary of the mainframe. IBM unveiled the System/360 mainframe, a groundbreaking computer that allowed new levels of compatibility between systems and helped NASA send astronauts to the moon.

The IBM System/360 ushered in an era of computer compatibility and for the first time, allowing machines across a product line to work with each other. It was the first product family that allowed business data-processing operations to grow from the smallest machine to the largest without the enormous expense of rewriting vital programs.

Code written for the smallest member of the family was upwardly compatible with each of the family's larger processors. Peripherals such as printers, communications devices, storage, and input-output devices were compatible across the family.

While Moore's Law and competition decrease the per-unit cost of hardware and software, the shortage of skilled professionals that can manage the growing complexity of information technology (IT) systems, increasingly hampers the growth of IT systems and dominates their total cost of ownership today.

The continued growth of the IT industry demands that we automate many of the functions that are now performed by humans, both to save cost and to penetrate new markets.

The current economic environment and budget reductions are also important industry challenges. Pressure is constant to analyze the IT budget to lower costs and invest only in technologies that provide a good return on investment.

Companies around the world and the respective CIOs are being questioned about metrics such as CPU utilization and the workload running in the environment needs to be justified in an era of limited resources.

Autonomics can help IT staff reduce think time for repetitive tasks and also analyze the environment in order to run only what is needed when it is needed, prioritizing critical applications, and reducing the CPU utilization for maintenance jobs that do not need to run in a defined maintenance window.

1.2 Why autonomics and modernization

We live in the age where data is one of an organization's most important assets. Companies want the ability to deliver the right information at the right time to the right people. Being agile and applying predictive analytics is key to the insights needed to grow business so that it remains at a peak competitive edge.

Today, there is more alignment between IT and the business side of an organization. This alignment is placing greater complexity on the IT organization. There is an enormous explosion of data that is predicted to grow exponentially. There has been a rapid growth in the volume, variety, and complexity of data due to the explosion of smart devices, mobile applications, cloud computing, and social media. Consider DB2 11 for z/OS and how it has changed to become the database for *big data* and critical business analytics.

The constant change in the type of applications and growth of data is also met with the ongoing demands for more traditional DBMS administrative tasks, such as executing IBM DB2 Utilities that manage the data stored in DB2 for z/OS. The greater amount of data combined with a reduced staff has led to greater demands on the time of the existing IT

database administrators and business requirements have reduced the amount of allowable downtime. In this scenario, DBAs need to do more tasks, learn more things, and be more efficient within a smaller downtime window.

Another interesting situation in many IT environments is that most DBAs do not spend their time setting up batch schedules for new applications or modifying existing ones. The existing batch schedules can be years or decades old with many changes having taken place to utility jobs, hardware, and the application itself (new DB2 tables) that have not been factored in to the existing set of batch jobs. Those changes include faster DASD, larger DB2 buffer pools, RTS tables, UTS TS with greater capacity to store data and new utility functionality (that is, online reorg) that make the original environment that an application was developed in totally different from today's environment. Is the maintenance work that was set up years ago still relevant today? Autonomics can help answer the question of whether the existing batch cycle is performing the work that should be done now based on how the application and hardware function now. The tools described in this book can be used to implement an autonomic solution that not only answers the question of what utility work should be done but help you get that work done in the right window.

In your current environment, there is an existing set of jobs that perform utility functions and why bother changing those jobs now? Batch cycles that were set up years ago are likely performing utility functions that may have been relevant back then but are no longer relevant. Reduction of utility executions directly lead to cost savings and fewer abends. Also, application releases introduce new tables and the maintenance requirements passed along may have changed or were not accurate from the start. There may be DB2 tables that need more frequent maintenance than was originally requested. Autonomics helps in both of these cases by allowing you to reduce unnecessary jobs and to add work that is needed but not currently scheduled.

1.2.1 The mainframe skills gap

According to recent IBM research, on average, 70% of the data that feeds data warehousing and business analytics solutions originates on the IBM z™ Systems platform (financial information, customer lists, personal records, manufacturing, and so forth). A *ComputerWeekly.com* article from March 2014 states that "Today, 80% of the world's corporate data is still managed by mainframes." Find the article at this link:

http://www.computerweekly.com/feature/Can-the-mainframe-remain-relevant-the-cloud-and-mobile-era

What does that mean to us? It means that the mainframe still is the most important platform to major companies and the transformation era that IT is experiencing.

Obviously, to support all of this growing data and complex environments, skilled mainframe people are needed. However, experienced DBAs and System Programmers continue to retire, while new DBAs and System Programmers take years to become fully experienced and employees from Y and Z generations spend less time in a single job.

DBAs and System Programmers must become more efficient to keep up with the demands of today's work environment.

1.2.2 What is autonomics and how it can solve business problems

The concept of automating routine database tasks is not new. It has been theorized, discussed, and written about for several decades. But, true autonomic capabilities have been slow to be developed and adopted and put into practical use. There are a number of factors

helping to drive the need for a solution today and some resistance. By definition, an Autonomic Database Management System (ADBMS) should be able to self-regulate, including the ability to self-configure and self-optimize, as well as self-protect and self-heal without human intervention.

The IBM DB2 Utilities Solution Pack for z/OS, which is the focus of this publication, contains autonomic capabilities in DB2 utilities execution in a timed rollout that focuses on the following actions:

- Automate the routine collection of data.
- Automate the simple analysis of data.
- Automate the obvious decisions based on this analysis.
- Automate the straightforward execution of decisions.

With autonomics in place, the expected approach is to define all the objects, utilities, and maintenance windows once while the autonomics itself would deliver better system behavior, choosing and prioritizing what to run, analyzing maintenance windows and workloads, and deciding when to run. DBA reviews are needed just when significant changes occur. This means that DBAs spend less time in repetitive and administrative tasks, increasing efficiency and reducing costs (Figure 1-1).

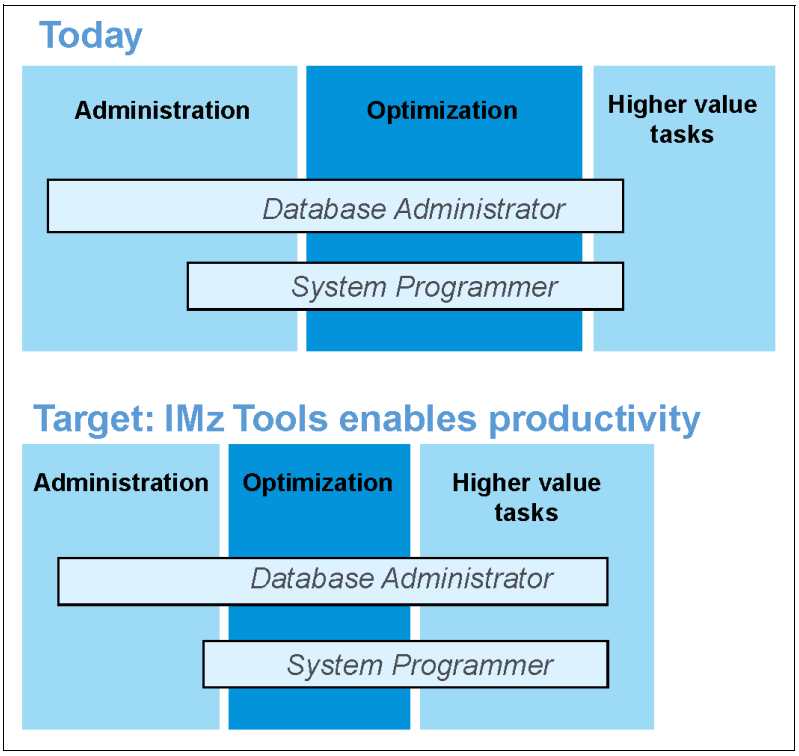


Figure 1-1 The value of autonomics

1.3 DB2 autonomics

DB2 autonomics is the IBM solution to solve the business problems presented previously and leverage a smarter way to manage DB2 utilities and maintenance windows.

To provide a complete DB2 autonomics solution, the components presented in Figure 1-2 on page 5 are needed.

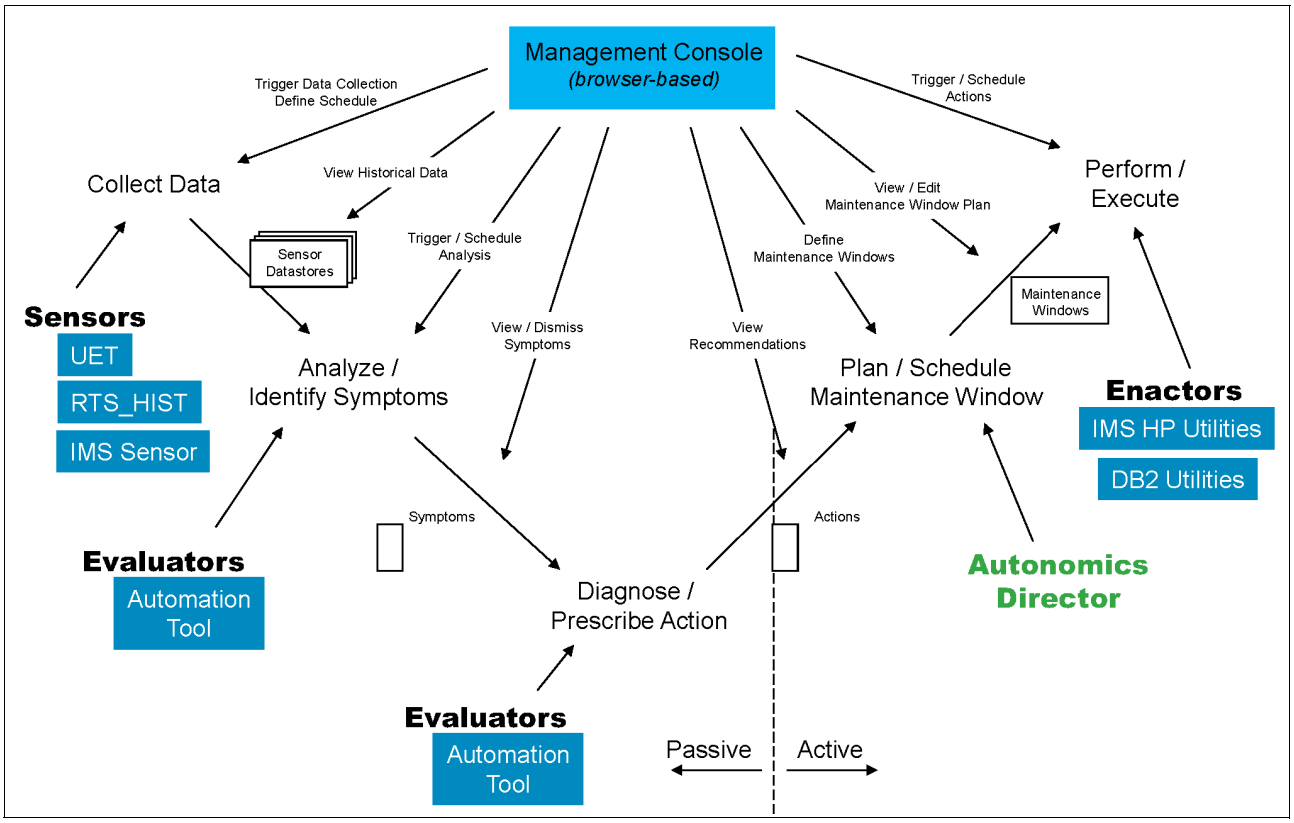


Figure 1-2 DB2 autonomics components

1.3.1 Autonomics Director for DB2

Autonomics Director for DB2 provides the framework for a comprehensive DB2 autonomics environment. With Autonomics Director for DB2, DBAs can schedule the utilities to run autonomically in a maintenance window. You can define the maintenance window and the priority of the tasks to run in that window. When you build a job profile, instead of generating JCL, DB2 Automation Tool generates tasks, called *actions* that are stored in a data repository. The Autonomics Director autonomically runs the actions that are in the repository when the maintenance window opens. In addition, you can specify the application objects that are most important and that are to be run first in the next maintenance window.

Autonomics Director for DB2 consists of stored procedures that gather DB2 real-time statistics and historical data about utility execution and controls the tasks that execute in the maintenance window. Specifically, Autonomics Director for DB2 provides the following stored procedures:

- RTS_SNAPSHOT creates a new snapshot of the current DB2 real-time statistics for a specific database object or objects. You can call this stored procedure before or after a maintenance window.
- Active Autonomics Director Stored Procedure evaluates a list of possible actions, prioritizes the actions, and determines which utilities can be run on which objects considering the amount of time remaining in the maintenance window.

The data that is collected by these stored procedures is stored in a series of DB2 tables that begin with the prefix SYSAUTO.

Figure 1-3 shows more details about Autonomics Director for DB2 architecture, including DB2 tables and the DB2 Automation Tool profiles involved.

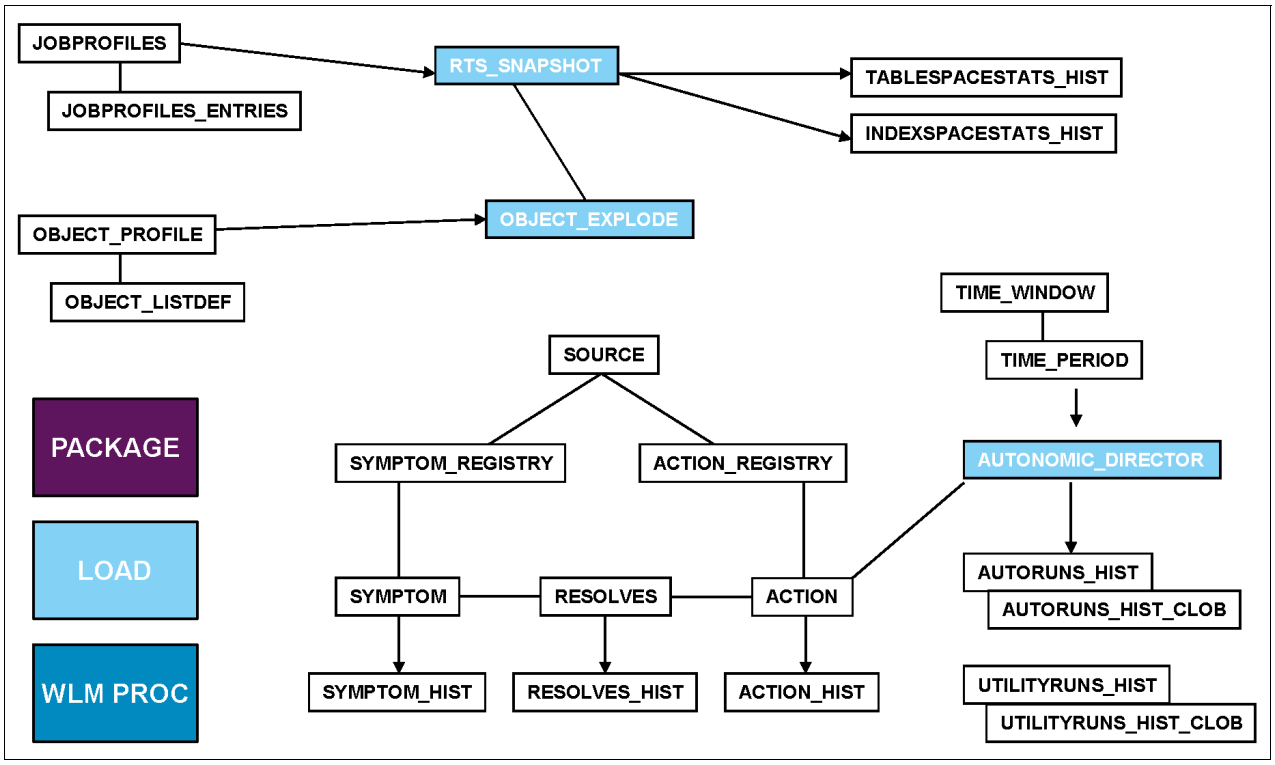


Figure 1-3 Autonomics Director overview

1.3.2 Maintenance windows

A *maintenance window* is a specific time period that allows the actions to be run on your objects. Once a maintenance window is associated with the job profile, an autonomic build can be run.

Historically, companies already have your maintenance windows defined and batch cycles schedules sometimes are years or decades old with many changes having taken place to utility jobs that do not consider new objects, new features to DB2 software, and even new hardware improvements that could benefit DB2 Utilities and change the batch processing behavior.

Maintenance windows and batch processing are concepts linked together because batch processing occurs in defined maintenance windows. Batch processes typically have the following characteristics:

- Large amounts of input data are processed and stored (perhaps terabytes or more), large numbers of records are accessed, and a large volume of output is produced.
- Immediate response time is usually not a requirement. However, batch jobs often must complete within a batch window, a period of less-intensive online activity, as prescribed by a service level agreement (SLA).
- Information is generated about large numbers of users or data entities (for example, customer orders or a retailer's stock on hand).
- A scheduled batch process can consist of the execution of hundreds or thousands of jobs in a pre-established sequence.

Of course not all batch processing during maintenance windows are just DB2 related, but it is important that DBAs review DB2 work and use tools as described in this book to ensure that only the needed jobs run and maintenance windows are not impacted.

2

Product overview: Management Console and IBM DB2 Utilities Solution Pack

Several new business challenges are a reality in IT nowadays. Data explosion, the mainframe skills gap, and the pressure for cost reduction are some of these challenges that DB2 Autonomics and its related products help solve by improving productivity and allowing employees to spend more time problem solving higher-level tasks.

In this chapter, we provide an overview of the software products that work together as individual pieces of a jigsaw puzzle to provide this solution. The products are IBM Management Console for IMS and DB2 for z/OS, Autonomics Director, and IBM DB2 Utilities Solution Pack for z/OS, which consists of DB2 Automation Tool, DB2 Sort, DB2 High Performance Unload, and DB2 Utilities Enhancement Tool.

© Copyright IBM Corp. 2015. All rights reserved.

2.1 IBM Management Console for IMS and DB2 for z/OS

IBM Management Console for IMS and DB2 for z/OS modernizes mainframe administration by providing a single, intuitive, and modern web-based interface that configures, administers, automates, and optimizes the performance of your DB2 environment. It consolidates data from multiple DB2 tools, including tools from non IBM vendors, and presents a more complete picture of your DB2 subsystems and databases. You can use the comprehensive context-sensitive help system included with Management Console as a teaching aid for new DBAs and system programmers. It is a no-charge extensible product that is growing the number of products and solution packs supported.

Following are the main advantages:

- Zero-install for the web-based interface.
- Consolidated information from IMS, DB2, and tools to from across the entire enterprise.
- Reduced time for problem identification and resolution through tight integration with IMS and DB2 Autonomics.
- Embedded help throughout the interface for educating users with IMS and DB2 for z/OS details and concepts, helping to reduce the learning curve, and building the essential foundation for new IMS and DB2 for z/OS System Programmers and DBAs.
- The ability to progressively drill down from the enterprise or system level down to individual objects. You can view, understand, and access identified IMS and DB2 for z/OS symptoms and recommended actions.
- Support for setting up DB2 real-time statistics (RTS) snapshot job profiles and scheduling them through the DB2 administrative task scheduler to regularly collect RTS.
- Graphical charts and graphs showing historical DB2 RTS and IMS sensor data statistics, allowing users to validate autonomic symptoms and recommendations and identify overall trends.
- Graphical calendaring support for defining and optionally scheduling active autonomic maintenance windows to automatically perform recommended actions.

Today's global enterprise must manage larger and more complex information management environments. With this increased complexity comes more data and larger tables; more subsystems, objects, and buffer pools; and new utility and metadata options. As the number of experienced DBAs in the workplace declines, your business must focus on efficiency, choose the right options, and avoid unnecessary work. Management Console provides a comprehensive view of your databases and tools (Figure 2-1 on page 11) that builds upon the existing capabilities of DB2 Automation Tool to maximize efficiency and availability.

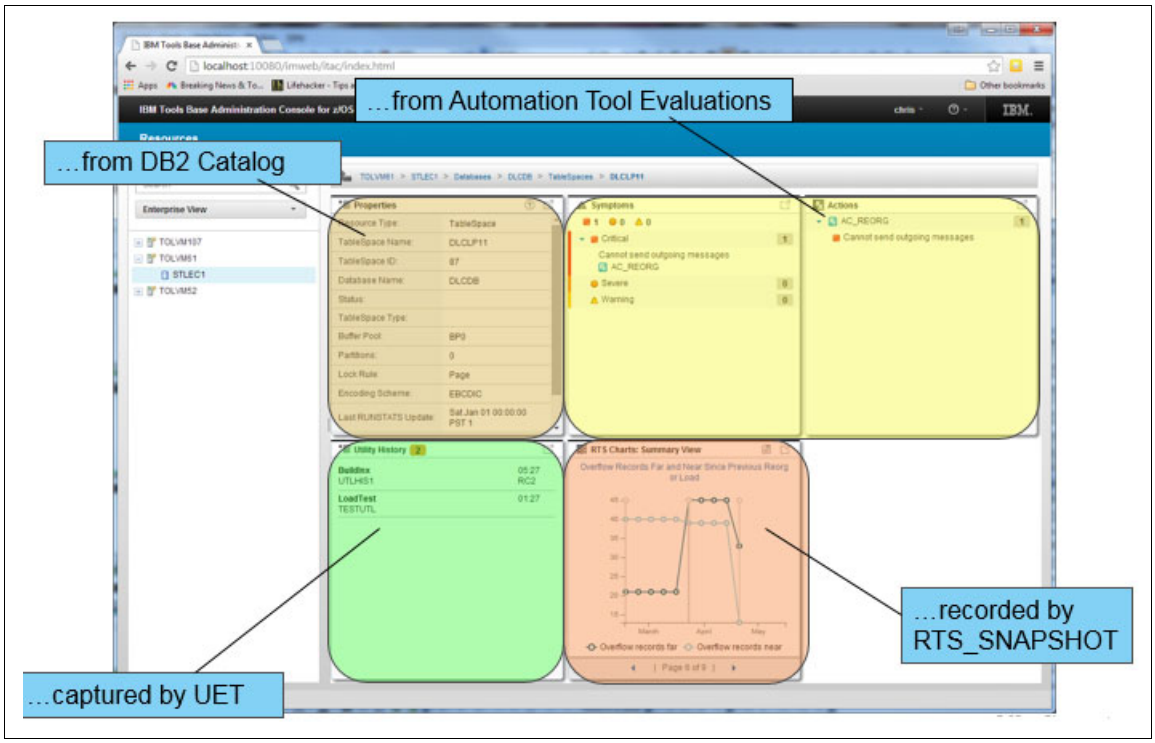


Figure 2-1 Holistic Dashboard

2.1.1 Architecture

The application server for IBM Management Console for IMS and DB2 for z/OS (Management Console) can be installed on Microsoft Windows or z/OS. Management Console can then connect to z/OS through Secure Sockets Layer (SSL) TCP/IP connections.

The Management Console architecture (Figure 2-2 on page 12) consists of the following elements:

- A client machine with Internet connectivity and a supported web browser. The client connects to the Management Console server by using HTTP or HTTPS.
- The Management Console web server, which can be installed on a Microsoft Windows workstation, a Microsoft Windows server, or z/OS.
- A connection interface that provides access to z/OS resources, such as the DB2 distributed data facility (DDF), which is part of DB2 for z/OS. DB2 DDF provides TCP/IP access to DB2 data.

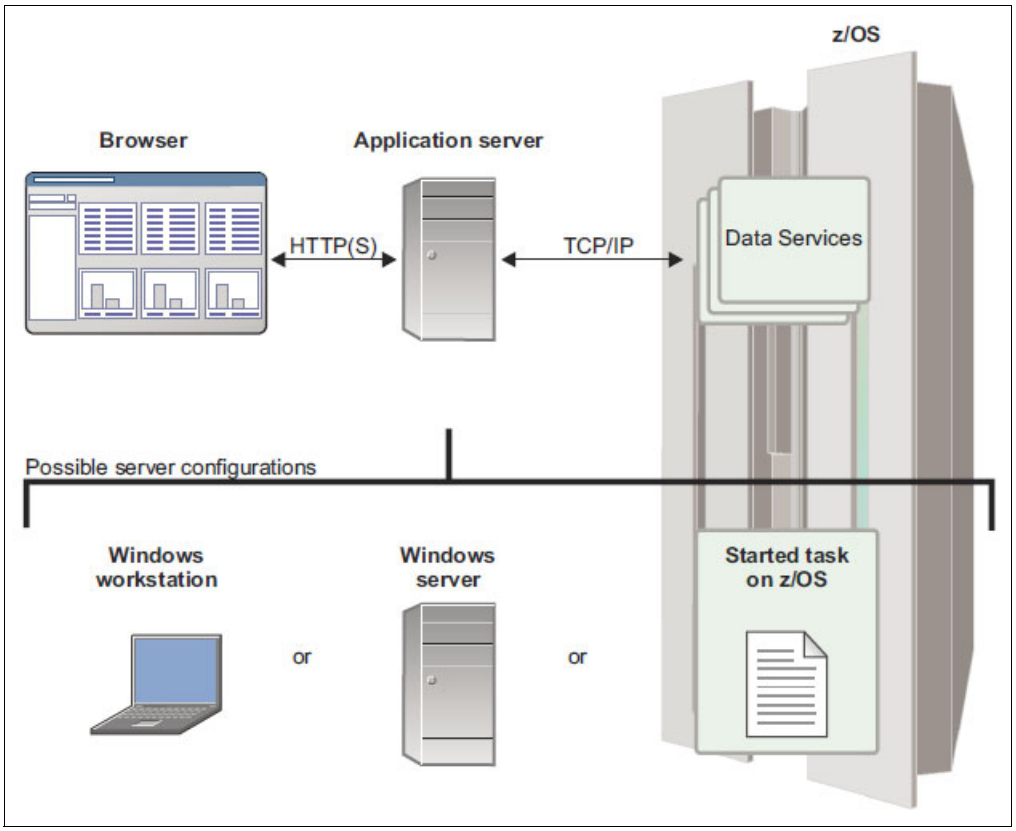


Figure 2-2 Management Console architecture

2.1.2 Benefits

Imagine that you arrive at the office in the morning, grab one coffee, and open a Web Console that provides a consolidated view of your DB2 subsystems and their weekend workload. From this console, you can check the status of your objects, all the jobs that ran during the weekend and their return codes and also see if there are outstanding actions that you need to address. Management Console can give you this scenario.

Using a user-friendly interface that includes graphical and historical information, in a centralized place for all DB2 subsystems, IMS, and tools, Management Console can help DBAs to reduce time for problem identification and resolution through tight integration with DB2 Autonomics and also reduce the time spent in administrative and repetitive tasks.

Another important benefit from Management Console is the integrated help (Figure 2-3 on page 13), which can educate new and experienced DBAs on database concepts and how to interpret charts.

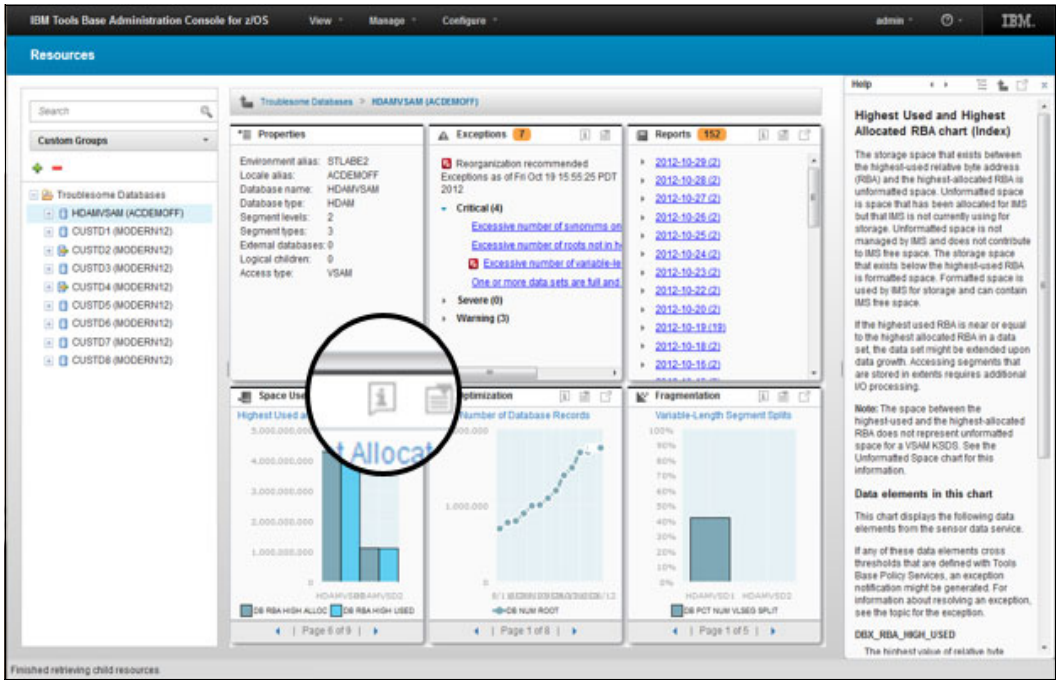


Figure 2-3 Integrated help

2.1.3 Plug-ins

IBM Management Console for IMS and DB2 for z/OS has connections to the following plug-ins:

- IMS
- IBM Tools Base Autonomics Director for z/OS
- IBM Tools Base Policy Services for z/OS
- IBM Tools Base IMS Tools Knowledge Base for z/OS
- IBM IMS Database Solution Pack for z/OS
- IBM IMS Fast Path Solution Pack for z/OS
- DB2 for z/OS
- DB2 Utilities Solution Pack

2.2 DB2 Utilities Solution Pack

IBM DB2 Utilities Solution Pack for z/OS is a product that combines a number of IBM components into a consolidated solution that enables enterprises to efficiently and intelligently run utilities, while optimizing the performance of regular utility management activities.

IBM DB2 Utilities Solution Pack for z/OS is focused on four main areas (Figure 2-4 on page 14).

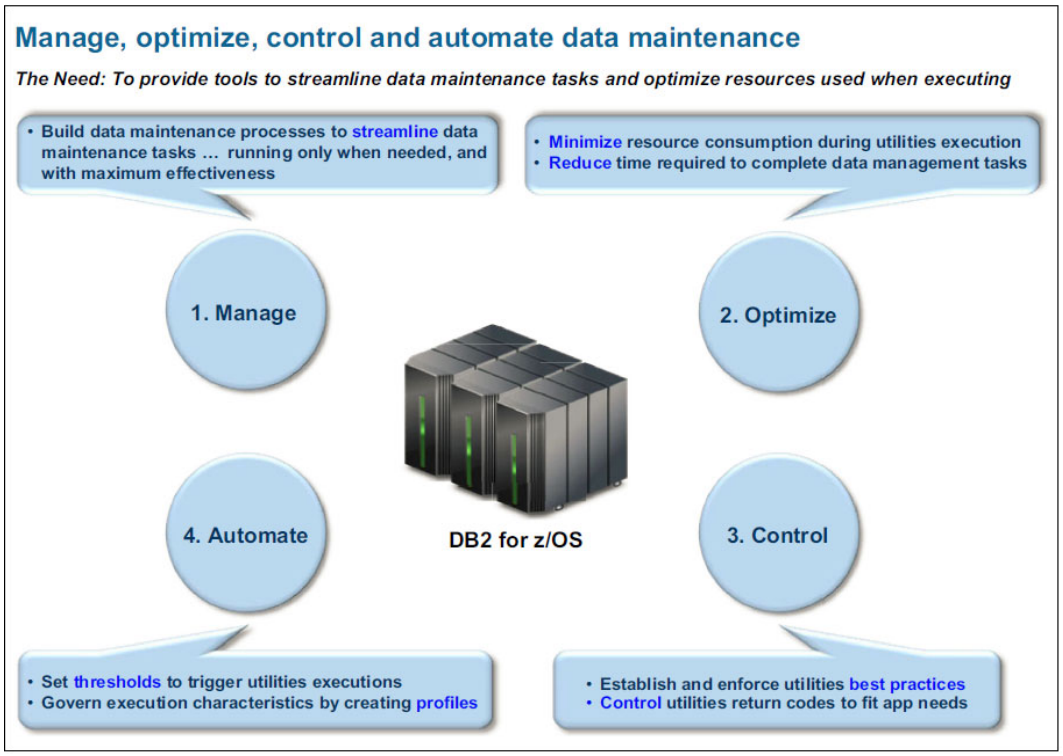


Figure 2-4 IBM DB2 Utilities Solution Pack

2.2.1 Why use Solution Packs?

Imagine a situation that your company purchased a product to automate your production DB2 Utilities (DB2 Automation Tool), but you want to implement the leading practices in the test environment or for utilities running outside of a batch scheduler. Wouldn't it be better to have utility automation and leading practices assured? Or imagine that you have both utilities automation and leading practices in place but you are still facing problems to meeting your service level agreements (SLAs) and your manager is still complaining about CPU utilization?

One of the main advantages of using Solution Packs is the integration between the individual products and the possibility to solve a set of business challenges. IBM DB2 Utilities Solution Pack is designed to solve the business challenges presented above by providing a comprehensive solution that ties together utility automation, leading practices, and reducing resource consumption to provide your environment with the stability of leading practices and the efficiency of conditional reorgs combined with more efficient tools for sort and unload.

IBM DB2 Utilities Solution Packs is the building block of an autonomics solution that helps you to anticipate and avoid emergency situations and high-severity problems before they occur. DB2 Utilities Solution Pack also helps you to maintain availability, integrity, security, and comply with service level agreements. The ability to optimize performance is also an essential component of the solution. In combination with Autonomics Director for DB2 and Management Console, the DB2 Utilities Solution Pack provides the precise level of automation and scheduled maintenance that your business requires to supplement manual approaches. Thanks to a robust set of configuration options based on policies and profiles, you are in complete control. You decide which actions to automate and which to perform manually.

The Solution Pack offers easy-to-configure policy services that build intelligence into utilities and tools management. In addition to the real-time statistics (RTS) that DB2 generates,

including DB2 sensor data that any tool can access, Autonomics Director for DB2 adds an additional layer of features that enable you to define policies. These policies recommend or automate recurring DB2 monitoring and maintenance tasks, including the ability to schedule the collection and evaluation of the sensor data.

Autonomics Director for DB2 collects these point-in-time statistics about database health and stores them in a central repository. It evaluates the sensor data and then compares the values to thresholds that you define in the policies. If DB2 activities exceed a threshold that you specify, Autonomics Director for DB2 generates an exception. The user has the option to define actions, such as triggering notifications or launching processes, that Autonomics Director for DB2 takes in response to the exceptions.

2.2.2 Products overview

DB2 Utilities Solution Pack includes the following components:

- **DB2 Automation Tool for z/OS** (also referred to as *DB2 Automation Tool*). You can use this component to set up recurring utility jobs for conditional and routine DB2 maintenance tasks. You can also automate common DB2 maintenance tasks and generate JCL for more complex tasks on one or more objects.
- **DB2 High Performance Unload for z/OS** (also referred to as *DB2 High Performance Unload*). This component offers performance options for unloading and extracting large amounts of data across the enterprise. Because High Performance Unload (HPU) works outside of the DB2 address space, it does not compete for the same DB2 resources.
- **DB2 Sort for z/OS** (also referred to as *DB2 Sort*). Sort processing of large volumes of data, large table spaces, or large indexes typically requires intensive processor usage and heavy I/O activity. Large amounts of memory and disk space can be required for this sort processing. DB2 Sort meets these demands by optimizing the use of memory, using IBM z Systems™ Integrated Information Processor (zIIP) technology, and optimizing the management of disk space that is used for sort work.
- **DB2 Utilities Enhancement Tool for z/OS** (also referred to as *DB2 Utilities Enhancement Tool*). This component allows you to define and enforce company standards for utility processing. The standards enforced allow you to implement leading practices giving you the best performance in your utility jobs. DB2 Utilities Enhancement Tool can block and cancel threads on DB2 objects for other DB2 online utilities and for any programs that use a DB2 database.

In this book, we focus on the IBM DB2 Automation Tool for z/OS and DB2 Utilities Enhancement Tool for z/OS that are part of the autonomics solution for DB2 z/OS.

2.2.3 DB2 Automation Tool for z/OS

IBM DB2 Automation Tool for z/OS automates the running of utilities against a specified set of objects.

Using DB2 Automation Tool, database administrators can oversee the most routine database administration tasks, without time-consuming manual interventions. With DB2 Automation Tool, you can set up recurring utility jobs for conditional and routine DB2 maintenance tasks.

DB2 Automation Tool can help you reduce manual routine tasks and focus on more complex job responsibilities that add more value to your company. It can automate common DB2 maintenance tasks as well as generate JCL for more complex tasks on one or more objects.

Using an online Interactive System Productivity Facility (ISPF) interface, you select the objects that you want and place them into an object profile. You then select utilities to run and place them into a utility profile. Once you create the object profile and the utility profile, you combine the two into a job profile. When you build the job profile, it produces a z/OS batch job containing the job control language (JCL) to execute the utilities. The job that is produced can be executed immediately or inserted into a job scheduler.

DB2 Automation Tool's exception processing generates maintenance JCL for only objects that need it. Therefore, maintenance procedures consume less resources by only running when needed. For example, DB2 Automation Tool can generate a REORG utility if a table space has too much embedded fragmented space.

2.2.4 DB2 Utilities Enhancement Tool for z/OS

DB2 Utilities Enhancement Tool for z/OS provides several distinct features that can help DBAs, DB2 System Programmers, and other IT professionals manage their DB2 resources. The product provides the following features:

- A Utility Syntax Monitor feature that allows you to establish and maintain company-wide DB2 utility syntax policies. This feature can be also useful for auditing purposes.
- After the policy is created, an element called PRACTICE allows the DSNUTILB intercept feature of DB2 Utility Enhancement Tool (UET) to search the provided utility statements and do one or more of the following functions:
  - Remove a text string from the utility statement if the specified string is found in the utility statement.
  - Substitute a text string to replace the text string found in the utility statement.
  - Journal the occurrence of a monitored event to a DB2 table for subsequent review or analysis.
  - Fail the utility job step with a specified return code if a text string is found in the utility statement.
  - Change the return code of a job based on the presence of a DB2 message.
- Can block and cancel threads on DB2 objects so that the objects can be accessed by DB2 utilities or other applications and programs in your environment that use DB2 databases.

DB2 Utilities Enhancement Tool runs as a started task on a z/OS system. A single product configuration can handle requests from multiple users concurrently and perform product processing across multiple DB2 subsystems on the same z/OS image or in the same data sharing group. The product provides three interfaces: the DSNUTILB intercept, the ISPF interface, and the batch interface. Your choice of interface depends on the task that you need to perform.

3

Business scenario and technical challenges

In this chapter, we discuss the technical challenges that today's DB2 system administrators and database administrators (DBAs) face to meet their business requirements. These include a growing size of DB2 table space objects and how to perform maintenance in environments with shrinking batch windows and to do more with fewer resources.

There have been many changes in the past 25 years in our IT world that have led to the need for autonomics in our environment. We'll look at a hypothetical company, Company H, and the changes that they have gone through to illustrate some of the problems that have arisen based on the changes in business and IT that are challenges faced by all of us right now.

Company H had a very successful growth period, 1990 - 2000. This was a very dynamic time for the business and IT with much new DB2 application development and that development helped support growth for the company in revenue and new business opportunities. The IT department expanded with new DBAs. Application teams implemented many new applications in DB2 and there was significant growth in the knowledge required to develop and implement new DB2 applications in production. During the years from 2000 - 2008, there was a slowdown in that initial burst of new DB2 application development, but the existing applications continued to be enhanced with new features and capabilities. After the financial crisis in 2008, there was a change in IT where costs were scrutinized more closely and budgets were flat or reduced and there was a loss of expertise due to restructuring and retirement of skilled workforce.

© Copyright IBM Corp. 2015. All rights reserved.

3.1 Obstacles to autonomic computing

One of the challenges to adopting autonomic computing is the element of change. There are many adages relating to the concept of changing. For an IT organization, such as our fictitious Company H, change may be driven by business needs imposing on IT to come up with a solution that supports the goals and needs of the business are expected. As mentioned, our Company H is asked to do more with less.

There are fewer experienced DBAs and those that remain, are being tasked with managing more than the traditional role of the DBA to support changing IT needs.

There is a greater emphasis on efficiency for the DBA and IT. It is necessary to choose the right options the first time to prevent unnecessary work. DBAs are pressured to maximize availability of key applications while reducing systems costs such as resources and CPU usage.

One of the biggest obstacles is also trust, especially from a database administrator who wants to implement an autonomic solution. By definition, an *Autonomic* DB2 Management process system should be able to self-regulate, including the ability to self-configure and self-optimize, as well as self-protect and self-heal without human intervention. For the DBA, allowing tooling to perform self-regulating tasks is a leap of technological faith that has to be proven over stages in smaller increments. There's a factor of being used to doing database management and knowing that, while time consuming, is successful and delivers the needed results. There's also trust that should the technology fail, it would still be up to the database professional to correct any missteps without losing availability or incurring any negative customer reaction.

3.2 The solution

What is needed is technology that combines successful results and the DBA comfort level of trust and implementation. The IBM DB2 Utilities Solution Pack for z/OS, which is the focus of this publication, is introducing autonomic capabilities in DB2 Utilities execution in a timed rollout. This allows the implementation of the level of autonomics that meets the needs of the organization and the DBA. For example, a user can choose to implement passive autonomics and then move to active autonomics when ready.

The goal then of DB2 autonomics is to make computer systems and software capable of managing themselves. This in turn requires an awareness of the DB2 environment, the operating systems environment, and knowledge of changing usage patterns and resources, and the ability to adapt to shifting requirements.

In this chapter, we provide an overview of how Company H evolved from a *passive* implementation (those administration tasks that they have implemented many years ago), to an *active* (autonomics) approach:

- **Passive autonomics:** This may be the traditional approach, which has been in place for many years. The implementation relies heavily on experience of highly skilled DBAs and the techniques used are based on application knowledge and some home-grown methodologies.
- **Active autonomics:** Now corrective actions are taken automatically by the system: that is, monitoring and analyzing the related metrics to proactively make recommendations and even execute them. It depends on the accuracy of said metrics and using trusted advisors to make decisions that are typically done by a DBA.

Before, the DBA evaluated the recommendations and made the ultimate decision to implement a solution. Now, the system evaluates the recommendations and resolves the situation.

3.3 A typical business case for DB2 Utility Autonomics

Business reasons are driving the need for autonomics in today's IT. The DBA plays a central role in those business decisions. In this section, we focus on the database administrator and their need for autonomics. We often hear or read about the lack of mainframe skills. This is a combination of lack of mainframe skills being taught at the university level, on-the-job training that requires commitment and investment, and a mainframe skilled workforce that is taking all the skills, knowledge, and training with them upon retirement.

If we focus only on the area of DB2 utilities management, there is enough complexity and growth to warrant a need for autonomics. There are many factors that contribute to the frequency and amount of DB2 Utility maintenance that a DBA performs regularly. A DBA must also consider when and how to perform maintenance. It wasn't so many years ago when a DBA would set up nightly REORGs and Image Copy jobs on objects as well as RUNSTATS, whether they were needed or not. Some DBAs applied maintenance on an 80/20% rule where utility maintenance was applied to 20% of the objects, trying to estimate which objects fit into which part of the equation.

Now coming back to Company H. The changes, as mentioned before, increased the complexity for some of the DB2 staff at Company H. The production batch application and utility jobs that were created in the 1990s execute DB2 utilities unconditionally. The expertise that was around when these applications were initially developed is no longer there. The DBA team believes that there are reorgs being done on DB2 tables that do not need to be done but no one has the time to do the analysis to prove that this is the case. Also, some of the new DB2 tables that were added during the 2000 - 2008 period are not being REORged as frequently as some of the original DB2 tables, and the DBA team is concerned that some of them should be reorged more frequently. The DBA manager is asking whether some of the workload, such as RUNSTATS, can be moved out of the peak CPU period in order to help other batch jobs complete faster and to reduce costs.

Finally, during the 1990s, Company H developed a home-grown DB2 utility and control card generator, which has not kept up with changes to some of the utilities and the support person who developed this home-grown solution and is planning to retire without anyone else having the know-how to maintain is planning to retire soon.

3.3.1 Challenges

Company H faces some real world challenges that are common to many companies. DB2 autonomics can address the problems that Company H and other companies face and we cover how that works in general terms here. Let's look at some of the challenges facing Company H and how autonomics can help:

- DB2 batch utilities are run unconditionally without intelligence. DB2 Automation Tool can help apply intelligence to the execution of DB2 utilities. This covers both the situation where DB2 tables may be reorged too often or DB2 tables may not be reorged often enough.

- Some utilities, like RUNSTATS, are being executed during peak CPU workloads and slowing down the execution of the production batch cycle and increasing costs. By having DB2 real-time statistics (RTS) externalized to a history table using the function provided with the IBM DB2 Management Console for IMS and DB2 (DB2 MC) will save CPU (versus running expensive RUNSTATS).
- DB2 Utility standards were first developed in the 1990s and have not kept up with the changes introduced in the past few releases of DB2 tools. Also, the expert at Company H on the home-grown utility generator is retiring soon. DB2 Utilities Enhancement Tool (UET) can help implement and enforce best practices for utility jobs and allow Company H to not have to rely on one DBA for support of the home-grown tool.

These challenges are explored in more detail throughout this publication as we try to explain how autonomics can help with the business challenges that you face.

3.3.2 Meeting these challenges

DB2 for z/OS has evolved over several releases to provide more information about the health of the objects. RUNSTATS and real-time statistics can provide invaluable information that can be mined to help determine the need for utility maintenance. In addition, IBM DB2 tools have allowed users to set up a utility base upon predefined conditions that can trigger what maintenance is needed on which objects based on which criteria has been met or exceeded.

The size of the DB2 for z/OS environment has grown exponentially and so has the size of the DB2 utility maintenance window. There is more competition to share the batch window as well as the cycles during this time. There are also more demands for accessibility and availability of business critical data. The DBA has to know when the best time is to perform utility maintenance and negotiate with the systems programmer to share the window and resources available.

All of these factors help to drive the need for applying autonomics when it comes to DB2 utilities. The ideal setting would be to allow the DBA to set up their environment and then allow the “autonomics” to determine when, where, and how maintenance is applied. The autonomic utilities environment should also handle dynamic changes, such as the ability to prioritize a utility job or to estimate the length that a utility job requires.

Today, the DB2 DBA may have many steps and the related tasks to automate object maintenance. Yet, in order to maintain and implement efficient production on DB2 for z/OS systems, the DBA is required to periodically monitor the DB2 objects that make up those systems.

This type of monitoring is a very essential component of post-implementation duties because the production environment is dynamic in nature, and the amount of data is growing. DBAs also do not have the time to perform monitoring. In addition, the sizes of the DB2 objects are expanding, causing DBAs to rethink their traditional sizing of the maintenance window.

In today’s analytical world, data access patterns and mobile computing are also forcing the DBA to pay more attention to all administrative needs. Systems performance and availability are not getting easier or going away. Effective strategies for monitoring DB2 objects in the production environment will catch and forestall problems before they affect performance.

Traditionally, DBAs monitored database objects via SQL queries against the DB2 Catalog tables. Also, they may have written home-grown routines to capture space and extend information. This information was ad hoc and not always up to date.

IBM provides a rich portfolio of functions, utilities, and tooling to help today's modern DB2 DBA and system administrator manage utility processing against application and DB2 Catalog objects. We discuss these in the following sections as they are related to the objectives of this publication.

3.4 A typical traditional DB2 maintenance scenario: passive autonomics

DB2 has been around for more than 30 years. During this time, DB2 made many changes and enhancements to assist in object maintenance. In order to maintain efficient production DB2 systems, DBAs had to periodically monitor the DB2 objects that make up their DB2 systems.

Database object monitoring was and still is an essential component of post-implementation duties due to:

- The dynamic nature of a typical production environment
- Business changes and requirements (such as, schema changes without impacting online availability, shrinking batch windows) are forcing the DBA to make changes to their maintenance strategy
- New applications and related requirements cause changes in data access patterns

Today's DBAs are required to do more with less: less skills, less experienced DB2 DBAs, and expectations of having DB2 for z/OS DBAs also administer DB2 on other platforms. This contributes to a possible lack of attention to administrative needs and can cause a system to perform inadequately or meet service level agreements (SLAs).

The objective for any effective strategy for monitoring DB2 objects in the production environment is to catch and prevent problems before they impact performance in a negative way. Preventive maintenance is the IBM recommended leading practice today. Running any utility without any benefits is a waste of CPU and other system resources.

DB2 provides many ways to help the DBA make decisions when to perform maintenance. Many features and enhancements are available. Here are some examples:

- DB2 Utilities: With every new DB2 version, more functions and features are provided to not only support new object types and sizes (for example, support for EF (Extended Format) data sets). As each new z/OS version also brings forth new features, the DB2 Utility Suite supports and exploits them. In addition, new enhancements are developed and become available throughout the maintenance stream.
- DB2 statistics: Traditionally, DBAs relied on RUNSTATS or using SQL queries to obtain information to make decisions to perform maintenance. Today, DB2 provides metrics using RTS.
- DB2 Catalog tables provide information via SQL queries.
- DB2 Tools: DB2 Tools, such as the DB2 Automation Tool for z/OS, DB2 Utilities Enhancement Tool, DB2 Sort, DB2 High Performance Unload, and DB2 Sort for z/OS are all part of the DB2 Utilities Solution Pack. These tools provide even more and unique functions to reduce resources and implement best practices for your maintenance requirements.
- The IBM DB2 Management Console for IMS and DB2.

We use the DB2 tools to perform a number of backup scenarios and demonstrate the flexibility that the tools provide in generating a comprehensive automated REORG strategy: replacing the manual effort that we have today. We demonstrate how DB2 Automation Tool can improve maintenance throughput by selectively determining which objects need to have a COPY, RUNSTATS, or REORG utility run via exception. One of the most useful features of the DB2 Automation Tool is its interface with the DB2 Administrative Task Scheduler. This feature enables us to set up maintenance windows where we can run the job profile defined with DB2 Automation Tool. That is what we refer to as *passive autonomics*. Using the DB2 Management Console and its Autonomics Director, we show how to implement an active maintenance schedule.

3.5 The implementation process

Company H has many challenges and this publication provides the jigsaw pieces that solve the individual problems outlined above in a way that allows for the pieces of the puzzle to be put together over time. Each company can have different priorities, which dictate the order of the implementation steps. Company A may need to apply intelligence to their reorg utilities as their top priority, while Company H may need to address utility standards because of the impending retirement of the support person for their home-grown DB2 utility generator. Regardless of the order in which you start, IBM provides the software for a comprehensive autonomic environment that addresses the business problems that most companies face: limited expertise, greater application availability, the need to control costs by moving work to off-peak hours.

Company H has to install the software to implement autonomics and determine how to implement autonomics in their environment. This publication covers both the software installation process and information about implementation. While the implementation varies from company to company based on the different challenges they face, this publication describes how to implement autonomics in a phased approach that most companies would follow.

3.5.1 Implementing an ACTIVE strategy for your DB2 maintenance tasks

In this section, we describe a scenario about how Company H can modernize their traditional and passive implementation (as described previously) steps to an active autonomic strategy using the IBM DB2 Utilities Solution Pack. We show how to move from these traditional steps into a modernized autonomic strategy. This entails how your maintenance window may now be more optimized with running your most important maintenance jobs on a predefined, priority-based method.

The maintenance window can be managed by the IBM DB2 Utilities Enhancement Tool (DB2 UET) and now with the IBM DB2 Management Console for IMS and DB2. DB2 UET provides utility governance and transparent thread control, ensuring that the maintenance window is not compromised by unwanted or unnecessary thread delays. This is done through an interface with DSNUTILB, called the *DSNUTILB Intercept*. DB2 UET can also enforce company-wide standards in utility syntax. This helps DBAs support multiple or diverse environments. The Utility Monitor provides the needs of DBAs with confidence that the utility jobs conform to company syntax standards, providing a secure execution environment for junior staff. Having syntax standards also reduces errors and provides tighter auditing on utility execution.

DB2 UET also provides additional functionality to some of the DB2 utilities. For example, DB2 UET can automatically size and create mapping tables and indexes for use by the REORG utility and can work with DB2 11 mapping table creation.

The IBM Management Console for IMS and DB2 now takes the automation a step further with a feature called *Autonomics Director*. The autonomics implementation provides an open framework upon which any number of IBM individual tools can participate, collectively providing the autonomic functions to allow the system to manage itself. The Autonomics Framework is primarily made up of two pieces:

- A collection of tables and stored procedures to access and modify those tables
- An Autonomics Director procedure for planning and driving the autonomics maintenance window

The Autonomics functions consist of:

- **Evaluators:** Analyze the state of the system and identify symptoms that they register in the SYMPTOMS table and make recommendations by registering them in the ACTIONS table.
- **Enactors:** Perform a specified action out of the ACTIONS table as instructed by the Autonomics Director. Enactors are also responsible for providing projections of time and CPU for actions.

Step 1: Collection of metrics and related statistics for utility maintenance

A collection of statistical data on your DB2 objects is the first step towards a maintenance strategy. This data can be used to filter out objects that are physically disorganized. In addition, you can use the metrics to help you group your objects, discussed in the next step, by update activity and size as well.

The main goal of this collection process is to decrease the amount of maintenance, especially REORG, jobs that needed to be run. Your object is to execute run maintenance by exception and filter out wasting of resources attributed to running of utilities.

Traditionally, running RUNSTATS was the one way to collect statistical information about DB2 objects and their underlying Virtual Storage Access Method (VSAM) data sets. Customers also used the HISTORY option to collect statistics over time. Consider using RUNSTATS primarily for your optimizer needs.

Today, there is no RTS history as part of DB2 for z/OS itself. RTS history can be extremely valuable for capacity planning, update activity trends, and for index usage and growth. One example is to consider backing up your active index spaces that are based on update activity: fast log apply available for index spaces.

The DB2 Management Console provides RTS history. This is achieved via a stored procedure called *RTS_SNAPSHOT*.

IBM provides two DB2 stored procedures to deal with autonomic statistics:

- Monitor stored procedure **ADMIN_UTL_MONITOR:** The MONITOR collects statistics about objects we have defined via a profile. This procedure generates an alert if the statistics exceed criteria, placing the alert into a table.
- Execution procedure **ADMIN_UTL_EXECUTE:** The EXECUTE stored procedure performs the RUNSTATS.

They use the DB2 Admin Task Scheduler to capture metrics for our test scenario. We set up the Workload Manager (WLM) to be used here. In order to verify that the WLM environment is working properly, we used the DB2 installation verification procedure (IVP) jobs to run the DSNWZP stored procedure. This ensured that everything was in place.

For our purposes here, we assume Company H to not use a Job Scheduler for running RUNSTATS, but instead uses the DB2 Admin Task Scheduler. They also use the DB2 Automation Tool, which provides the capability to add batch builds and utility execution jobs to the administrative task scheduler. In addition, DB2 Automation Tool provides an interface that allows the following functionality:

- *Create* new tasks
- *Update* and view existing tasks
- *Delete* tasks
- And *View* the status and output of executed tasks

This DB2 administrative task scheduler interface can be used to schedule Job Profiles for either batch or online generations. It can also be used when running stand-alone utilities through the Automation Tool ISPF interface. Another DB2 Automation Tool feature that can use DB2 administrative task scheduler is running RUNSTATS when needed by using DB2 autonomic statistics.

Step 2: Group your objects

Grouping your DB2 objects can be achieved in several ways. Traditionally, DBAs grouped their objects by application or even by Plan/Package name (and associated and related objects). It allows you to create reusable lists of objects for maintenance, backups, and collecting of statistics. DB2 provides the LISTDEF utility as a means for grouping. DB2 Automation Tool provides a function called *Object Profiles*, which provides more flexibility and functions for object grouping, called *Object Profile*. Using these Object Profiles, you can include objects on which you want to run utilities, as well as exclude objects that you want the utilities to ignore. Object Profiles are similar to DB2 TEMPLATEs. They allow table spaces and index spaces to be chosen for processing in much the same way.

Some considerations for grouping your objects can be by activity (using DB2 real-time statistics SYSTABLESPACESTATS columns COPYUPDATESPAGES and COPYCHANGES). These two columns are very useful when you have RTS history, like the history provided with RTS_SNAPSHOT SP.

Another consideration is to use SYSTABLESPACESTATS columns NPAGES, SPACE, and DATASIZE to group the objects by size.

Step 3: Create exceptions and thresholds for utilities

The DB2 Automation Tool provides a function called the *Exception Profile*. This definition contains the conditions under which users want to run utilities. When combined with Object Profiles and Utility Profiles, the Exception Profiles act as a filter against the objects specified in the Object Profile. The utilities specified in the Utility Profiles are only generated on the objects in the Object Profiles that satisfy the “conditions (exceptions)” specified in the Exception Profiles.

Part of implementing an Active Autonomic strategy for maintenance is to run all your maintenance by exception filtering. This filtering has many benefits, including:

- Reduce the waste that is associated by running unnecessary utilities
- Optimize your maintenance window, which might be shrinking

Step 4: Building optimized Utility JCL and jobs

Job Profiles are used to connect the different profiles, which are created in DB2 Automation Tool. A Job Profile is the master profile and associated with all the other profiles: Utility Profiles, Object Profiles, and Exception Profiles together. The combined profiles, which are headed by the Job Profile, form the basis of a DB2 Automation Tool task.

We can submit this task manually or schedule it by using the DB2 administration task.

Step 5: Execute the jobs in a predefined maintenance window

As mentioned before, a typical maintenance strategy today has predefined jobs in a job scheduler. These jobs are run in maintenance widows weekly, monthly, and quarterly. The maintenance window may be shrinking due to:

- DB2 Object size increase: REORGS take longer to run
- Competition, demand, and sharing shrinking window with other processes: unplanned changes
- Changes to applications and metadata, forcing REORGS to materialize such changes: planned changes

With Company H's new strategy discussed here, there is direct dependency on any specific scheduler within our Autonomics Framework. This allows you to leverage your own batch scheduler for creating evaluation jobs as well as starting the Autonomics Director procedure at any time during your maintenance window.

The Evaluators and the Autonomics Director Procedure can typically be driven by one of the following:

- Your current and existing Batch Scheduler
- The DB2 for z/OS Admin Scheduler
- Admin Console On Demand (or scheduled through DB2 for z/OS Admin Console Scheduler)

The result is that the evaluators and the autonomics maintenance windows provide both to start through a Stored Procedure.

Autonomics Director

The Autonomics Director manages and controls the projecting and executing of the maintenance window. By scheduling and invoking a stored procedure, the Autonomics Director examines an Actions list, prioritizes the required actions, and drives registered enactors to complete the highest priority actions that fit within the maintenance window.

The DB2 Autonomics Director lets you schedule utilities to run in an active/autonomic way in a maintenance window. This maintenance window can be set up by you based on utility history. Company H is not able to let DB2 Utilities like REORG and RUNSTATS be scheduled automatically.

Considering first a passive autonomics environment, they can automate the gathering of DB2 real-time statistics and information about the results of your DB2 utilities execution, which is stored in a utility history repository. This data is used as input to identify potential problem situations and to recommend actions that allow them to correct the problem situations and symptoms. Based on their own analysis, they can then decide whether to follow the recommendations provided.

For a consideration of an active autonomics strategy, the utilities are run for you automatically based on a number of factors, including the priority of the job, how many tasks are already

running, and how much time remains in your maintenance window. Company H can specify the application objects that are most critical in order to run in which maintenance window.

3.6 Conclusion

Company H, and the combination of autonomics and the IBM Management Console for IMS and DB2, implemented a central management of their database maintenance.

By using a modern single web interface, they now can administer DB2 from a single console. In addition, they do more important maintenance tasks utilizing their maintenance windows much more efficiently. In addition, they automated basic administration tasks to give DBAs freedom to work on higher business value tasks. And more importantly, they do not rely on old, home-grown processes, which are difficult to maintain and keep up with new DB2 versions.

4

Installing and configuring the DB2 Management Console

In this chapter, we describe the steps and related tasks to install and configure the IBM Management Console for IMS and DB2 (Management Console). At the end of this chapter, you will have learned how to install the DB2 for z/OS metadata for the DB2 Management Console objects, and how to install a server environment on z/OS for the web server in order to have a fully functional DB2 Management Console.

The Management Console server is a central web server. We configure a single instance of the server, which provides a single access point for all users. You can install the server on Microsoft Windows and on z/OS.

For our purposes, we install the server on z/OS. We assume the SMP/E on z/OS tasks has been completed. The installation repository used by Installation Manager to install Management Console on Microsoft Windows or z/OS is only available on shopZ as an SMP/E part. Regardless of whether you are installing the Management Console on Microsoft Windows or z/OS, you have to complete the SMP/E installation of Management Console first in order for the installation repository to be installed.

© Copyright IBM Corp. 2015. All rights reserved.

4.1 IBM software components required for the installation

The following software components are required and were installed to accomplish our objective in this chapter.

4.1.1 IBM Installation Manager for z/OS V1.4.0 (5655-IMZ)

The IBM Installation Manager is a separately orderable product at no cost, which has the following benefits:

- A modern installation process for a growing number of z/OS products
- Simplified installation logic for z/OS products
- Reduced installation redundancy on client systems
- Alternative delivery mechanisms, such as z/OS ServerPac

Publications can be found in the IBM Knowledge Center at this link:

`http://www.ibm.com/support/knowledgecenter`

4.1.2 IBM Management Console for IMS and DB2 for z/OS, V1.1 (5655-TAC)

The IBM Management Console for IMS and DB2 for z/OS, V1.1 (5655-TAC), is a lightweight web server that consolidates and simplifies information from IMS and DB2 for z/OS into a single, holistic, web-based interface that is accessible from a standard web browser. Benefits of the Management Console for IMS and DB2 for z/OS include:

- Simplified and consolidated presentation of complex information that is gathered across all of the IMS and DB2 systems in your enterprise
- Enterprise-wide presentation of system health data, including autonomic symptoms, exceptions, and recommended actions
- Reduced learning curve for users of both IMS and DB2 for z/OS

Publications can be found at the following site:

`http://www.ibm.com/support/docview.wss?uid=swg27020910`

4.2 Using the IBM Tools Customizer to install the DB2 metadata

We use the IBM Tools Customizer product to install the DB2 for z/OS components. The Tools Customizer load libraries are already APF-authorized and we invoke the REXX EXEC by using the CCQTCZ member of the EXEC library.

This REXX EXEC is in the *yourhlq*.SCCQEXEC library. Perform the following steps:

1. In Figure 4-1 on page 29, we invoke the Tools Customizer REXX EXEC and entered the product library. The product library, DBTLSP.SDYXDENU, is the one containing the metadata. The code “SDYX” is the name for the DB2 Autonomics Director that we install.

   To find the DB2 subsystem where we want to connect to, we use the **ASSOCIATE** command, either by typing in `ASSOCIATE` on the command line or putting our cursor on the **ASSOCIATE** command, as shown in Figure 4-1 on page 29.

```
TCUSTMZR          Customizer Workplace: DB2 Autonomics
Command ===> _____________________________________________ Scroll ===> PAGE

Use the Generate jobs line command to select the DB2 entries on which to
customize the product, and press Enter to generate the customization jobs.

Commands:  ASSOCIATE  DISCOVER  GENERATEALL  JOBLIST

Product to Customize
  Product metadata library . : DBTLSP.SDYXDENU          > LPAR  . . : SC63
  Product name . . . . . . . : Autonomics Director for  > Version . : 1.5.0


Product and LPAR Parameter Status
  Line commands: E - Edit  B - Browse
  _ Product parameters . : Verify Values
  _ LPAR parameters  . . : Verify Values

Associated DB2 Entries and Parameter Status
  Line commands: G - Generate jobs  E - Edit  B - Browse  C - Copy  R - Remove
  Cmd SSID Grpattch Lvl Mode User ID Date       Status              Message
------------------------------ End of DB2 entries -------------------------------
```

Figure 4-1 Tools Customizer main panel

2. Next, we find a DB2 subsystem to connect to. We use the associate panel (Figure 4-2) to specify which DB2 subsystem to connect to by entering `create` to attach to a specific DB2 subsystem.

```
TCUSTMZR                Associate DB2 Entry for Product                10:53:16
Command ===> create_____________________________________ Scroll ===> PAGE

Select any of the following DB2 entries to add them to the Customizer
Workplace panel. You use the Customizer Workplace panel to choose the DB2
subsystems, data sharing members, and group attach names on which to
customize the product.

Commands:  CREATE - Create new DB2 entries

DB2 Entries
  Line commands: A - Associate  C - Copy  D - Delete
  Cmd SSID Grpattch
------------------------------ End of DB2 entries -------------------------------
```

Figure 4-2 Choose a DB2 subsystem

3. After finding the target DB2 subsystem for our installation, we connect to this DB2 by entering `a` (to associate) the installation process to it (Figure 4-3 on page 30).

```
TCUSTMZR                Associate DB2 Entry for Product       DB2 entry created
Command ===> _____________________________________________ Scroll ===> PAGE

Select any of the following DB2 entries to add them to the Customizer
Workplace panel. You use the Customizer Workplace panel to choose the DB2
subsystems, data sharing members, and group attach names on which to
customize the product.

Commands:  CREATE - Create new DB2 entries

DB2 Entries
  Line commands: A - Associate  C - Copy  D - Delete
  Cmd SSID Grpattch
  a   D1C1  --
------------------ End of DB2 entries -----------------------------
```

Figure 4-3 Using "Associate" to attach the DB2 target subsystem

4. After we connect to a target DB2 (where we install our tool), we see in the status column (see Figure 4-4) that the installation is "Incomplete", which denotes that we have connected to the DB2 subsystem but the installation has not yet been started or finished.

```
TCUSTMZR          Customizer Workplace: DB2 Autonomics     DB2 entry associated
Command ===> _____________________________________________ Scroll ===> PAGE

Use the Generate jobs line command to select the DB2 entries on which to
customize the product, and press Enter to generate the customization jobs.

Commands:  ASSOCIATE  DISCOVER  GENERATEALL  JOBLIST

Product to Customize
  Product metadata library . : DBTLSP.SDYXDENU          > LPAR  . . : SC63
  Product name . . . . . . . : Autonomics Director for  > Version . : 1.5.0


Product and LPAR Parameter Status
  Line commands: E - Edit  B - Browse
  _ Product parameters . : Verify Values
  _ LPAR parameters  . . : Verify Values

Associated DB2 Entries and Parameter Status
  Line commands: G - Generate jobs  E - Edit  B - Browse  C - Copy  R - Remove
  Cmd SSID Grpattch Lvl Mode User ID Date       Status              Message
      D1C1  --          NFM                     Incomplete
------------------------------ End of DB2 entries ----------------------------
```

Figure 4-4 Status of the installation

5. Now we edit and verify the product parameters for our DB2 metadata installation (Figure 4-5 on page 31). During this step, you have the ability to change the default settings to your standards. These parameters are used to build the installation jobs.

```
TCUSTMZR          Customizer Workplace: DB2 Autonomics      DB2 entry associated
Command ===> _____________________________________________ Scroll ===> PAGE

Use the Generate jobs line command to select the DB2 entries on which to
customize the product, and press Enter to generate the customization jobs.

Commands:  ASSOCIATE  DISCOVER  GENERATEALL  JOBLIST

Product to Customize
  Product metadata library . : DBTLSP.SDYXDENU           > LPAR  . . : SC63
  Product name . . . . . . . : Autonomics Director for   > Version . : 1.5.0


Product and LPAR Parameter Status
  Line commands: E - Edit  B - Browse
  e Product parameters . : Verify Values
  _ LPAR parameters  . . : Verify Values

Associated DB2 Entries and Parameter Status
  Line commands: G - Generate jobs  E - Edit  B - Browse  C - Copy  R - Remove
  Cmd SSID Grpattch Lvl Mode User ID Date       Status              Message
  _   D1C1 --           NFM                   Incomplete
------------------------------ End of DB2 entries -------------------------------
```

Figure 4-5 Selection of the product parameters

6. We now enter our high-level qualifier for our target libraries where the installation jobs are built. As seen in Figure 4-6, there are options to change the defaults if required.

```
TCUSTMZR            Product Parameters: DB2 Autonomics                  11:01:51
Command ===> _____________________________________________ Scroll ===> csr

Complete the following tasks to customize the products. The required tasks,
required steps within a required or selected task, and required parameters
are preceded by an asterisk (*). Ensure that values are specified for the
required parameters. Press End to save and exit.


Commands: SAVE  VERIFYOFF
Line Commands: / - Select

Product customization library  : DB2TOOLS.PRODUCT.CUST.$SC63$.DYX150
                                                                  More:      +
Common parameters
 *DB2 Autonomics HLQ . .  DBtlsp ____________________________

* Create or Drop DB2 Autonomics Director tables

  _ Drop DB2 Autonomics tables

  * Create DB2 Autonomics Director tables

* Bind and free packages and plans

  _ Free packages and plans

  * Bind packages and plans

* Grant EXECUTE privilege
```

Figure 4-6 Customizing the product option panel

7. In the example in Figure 4-7, we entered a PROCLIB where the installation jobs add the started task for the Management Console.

```
TCUSTMZR            Product Parameters: DB2 Autonomics                  11:01:51
Command ===> _____________________________________________________ Scroll ===> csr

Complete the following tasks to customize the products. The required tasks,
required steps within a required or selected task, and required parameters
are preceded by an asterisk (*). Ensure that values are specified for the
required parameters. Press End to save and exit.


Commands: SAVE  VERIFYOFF
Line Commands: / - Select

Product customization library  : DB2TOOLS.PRODUCT.CUST.$SC63$.DYX150
                                                                  More:    -
  * Create DB2 Autonomics Director tables

* Bind and free packages and plans

  _ Free packages and plans

  * Bind packages and plans

* Grant EXECUTE privilege

  * Grant EXECUTE privilege on the plans

* Set up WLM Enviroment for DB2 Autonomics Director

  * Define DB2 Autonomics Director WLM address space
   *Started Task PROCLIB . db2tools.proclib ____________________________________
```

Figure 4-7 Customize panel: Target PROCLIB for the started task

8. We now edit the DB2 target subsystem environment. As seen in Figure 4-8, we want to change DB2 subsystem D1C1.

```
TCUSTMZR          Customizer Workplace: DB2 Autonomics          Row 1 to 1 of 1
Command ===> _____________________________________________________ Scroll ===> CSR

Use the Generate jobs line command to select the DB2 entries on which to
customize the product, and press Enter to generate the customization jobs.

Commands:  ASSOCIATE  DISCOVER  GENERATEALL  JOBLIST

Product to Customize
  Product metadata library . : DBTLSP.SDYXDENU          > LPAR  . . : SC63
  Product name . . . . . . . : Autonomics Director for  > Version . : 1.5.0


Product and LPAR Parameter Status
  Line commands: E - Edit  B - Browse
  _ Product parameters . : Ready to Customize
  _ LPAR parameters  . . : Ready to Customize

Associated DB2 Entries and Parameter Status
  Line commands: G - Generate jobs  E - Edit  B - Browse  C - Copy  R - Remove
  Cmd SSID Grpattch Lvl Mode User ID Date       Status              Message
  e   D1C1  --          NFM                     Incomplete
------------------------------ End of DB2 entries -------------------------------
```

Figure 4-8 Updating the DB2 subsystem parameters

9. In Figure 4-9, we updated all the DB2 parameters, which are used by the Tools Customizer to build the JCL with the correct information.

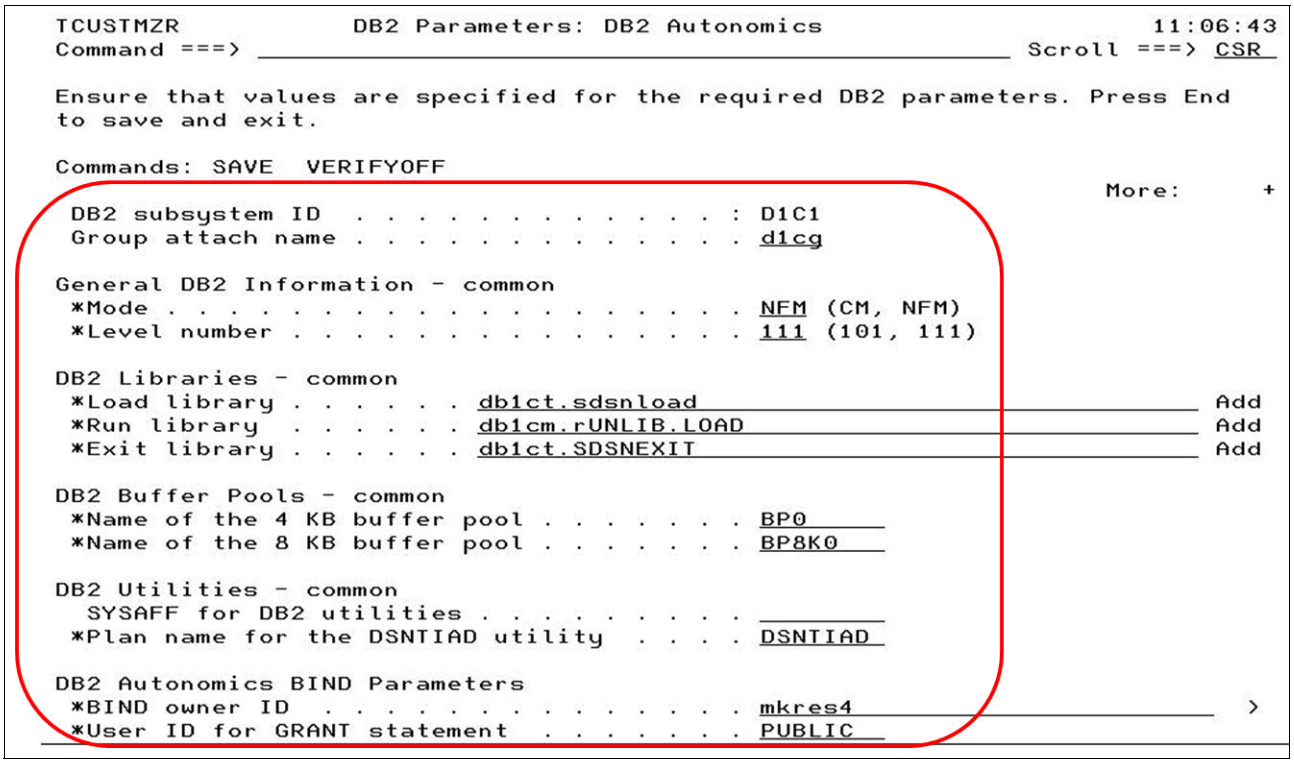


Figure 4-9 DB2 parameters update and verify

10. In Figure 4-10 on page 34, we saved our updates and the Tools Customizer now is ready to build the JCL and related installation jobs to execute.

```
TCUSTMZR          Customizer Workplace: DB2 Autonomics          Row 1 to 1 of 1
Command ===> _____________________________________________________ Scroll ===> CSR

Use the Generate jobs line command to select the DB2 entries on which to
customize the product, and press Enter to generate the customization jobs.

Commands:  ASSOCIATE  DISCOVER  GENERATEALL  JOBLIST

Product to Customize
  Product metadata library . : DBTLSP.SDYXDENU          > LPAR  . . : SC63
  Product name . . . . . . . : Autonomics Director for  > Version . : 1.5.0


Product and LPAR Parameter Status
  Line commands: E - Edit  B - Browse
  _ Product parameters . : Ready to Customize
  _ LPAR parameters  . . : Ready to Customize

Associated DB2 Entries and Parameter Status
  Line commands: G - Generate jobs  E - Edit  B - Browse  C - Copy  R - Remove
  Cmd SSID Grpattch Lvl Mode User ID Date       Status           Message
  _   D1C1 D1CG     111 NFM  MKRES4  2015/05/14 Ready to Customize Edited
------------------------------ End of DB2 entries ------------------------------


        CCQB001I The DB2 parameter data was saved in the data store.
```

Figure 4-10 Updates saved and installation ready for build

11. In Figure 4-11, we see the four jobs what were created by the Tools Customizer. We run these four jobs in the order it was built.

```
TCUSTMZR                Finish Product Customization            Row 1 to 4 of 4
Command ===> _____________________________________________________ Scroll ===> CSR

For a first-time customization, submit the jobs in the members in the order
in which they apply to the DB2 entries. Otherwise, submit only the necessary
jobs that were generated after changes were made. To submit jobs, browse
the members and issue the TSO SUBMIT command.

Line Commands: E - Edit  B - Browse

 _  Product customization library:   DB2TOOLS.PRODUCT.CUST.$SC63$.DYX150

 Cmd Member   SSID Grpattch Template Date       Description
 -   -------- ---- ------- -------- ---------- ------------------------------>
 _   A1DDL1AA D1C1 D1CG     DYXDDL1  2015/05/14 Create DB2 Autonomics Objects
 _   A3BINDAA D1C1 D1CG     DYXBIND  2015/05/14 Binds packages and plans
 _   A4GRANAB  --  D1CG     DYXGRANT 2015/05/14 Grants EXECUTE authority
 _   A5WLMAA  D1C1 D1CG     DYXWLM   2015/05/14 Autonomics WLM address space
-------------------------- End of customized jobs ---------------------------


        CCQC000I The jobs were generated on the selected DB2 entries.
```

Figure 4-11 Installation jobs generated by the Tools Customizer

We have now completed the DB2 Metadata installation and customization steps.

4.3 Installation and customization of the web server on z/OS

Next, we install the web server on z/OS. This particular installation and customization step, and related tasks, are typically not performed by the DB2 systems programmer, systems administrators, or database administrators. We show some of the steps we completed to install a z/OS web server.

You can install the IBM Management Console for IMS and DB2 for z/OS (Management Console) on z/OS by modifying and running the jobs provided in the SAMPLIB. These jobs use Installation Manager to install Management Console in a z/OS UNIX System Services file system.

4.3.1 Setting up WLM for our installation

Following are some of the steps to verify that we have a WLM environment ready for our usage (Figure 4-12):

1. In Choose Service Definition select option 2.

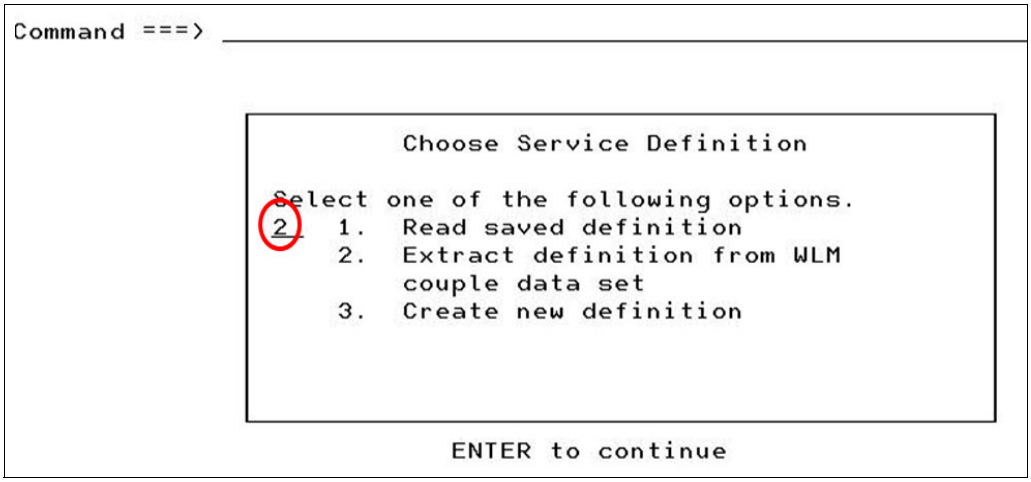


Figure 4-12 Extract WLM data set information

2. In the next panel (Figure 4-13 on page 36), we show how to locate an environment for our testing.

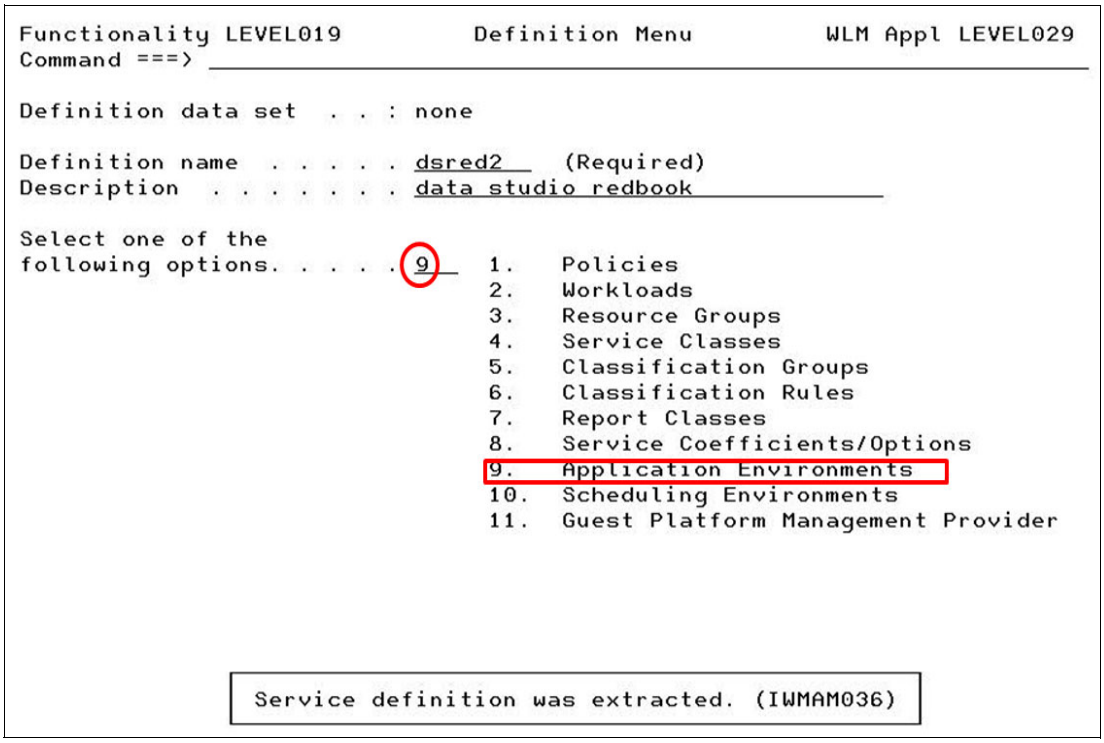


Figure 4-13 Selecting the WLM application environment

3. Next, we copied an existing definition and cloned our own environment (Figure 4-14).

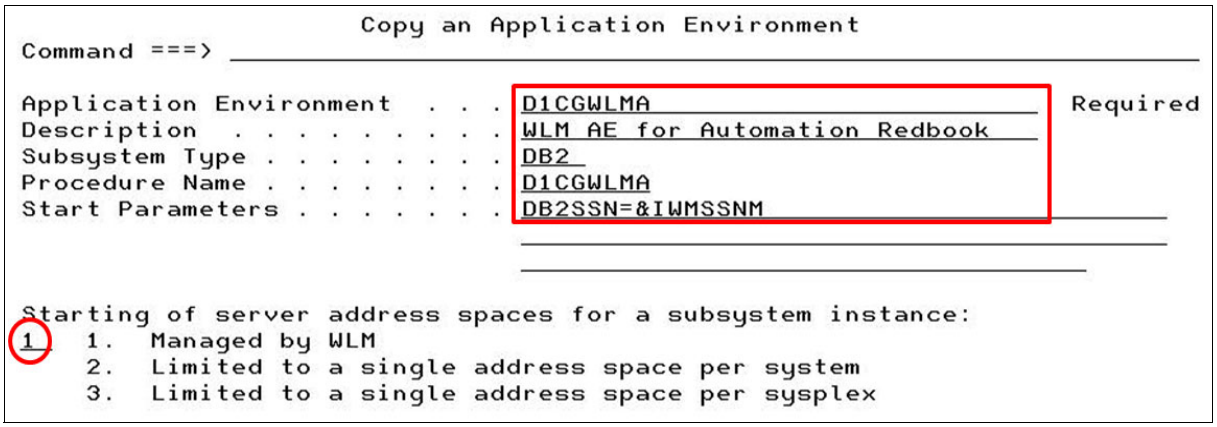


Figure 4-14 Cloning an existing WLM application environment

4. We now verify that WLM is up and running. As mentioned, we did not show all the steps, but some basic ones to create our WLM environment. Next, we start and display our WLM address space. We display our PROC using System Display and Search Facility (SDSF) (Figure 4-15 on page 37).

```
/d wlm,applenv=d1cgwlma
```

```
SDSF ULOG  CONSOLE MKRES4$                          LINE  COMMAND ISSUED
COMMAND INPUT ===>                                        SCROLL ===> PAGE
RESPONSE=SC63
 IWM029I  13.35.27  WLM DISPLAY 345
   APPLICATION ENVIRONMENT NAME      STATE     STATE DATA
   D1CGWLMA                          AVAILABLE
   ATTRIBUTES: PROC=D1CGWLMA SUBSYSTEM TYPE: DB2
```

Figure 4-15 Display of our WLM

4.3.2 Creating the Unix System Services environment

Next, we demonstrate how to set up the UNIX System Services environment for the Management Console. In Figure 4-16, we select the mounted File System, which we use for our installation.

```
                    Work with Mounted File Systems

Select one or more file systems with / or action codes.
U=Unmount   A=Attributes   M=Modify   R=Reset unmount or quiesce
  File system name                          Status          Row 25 of 234
_ DBTLSP.SGINKIT.D150425                    Available
_ DB2TOOLS.IMTOOLS.ZFS                      Available
_ EGF.V1R1M0.SEGFHFS                        Available
_ HBR.SHBRZFS                               Available
_ HERING.TEST.ASCII.ZFS                     Available
_ HERING.TEST.ZFS                           Available
  HFS.SC63.DEV                              Available
```

Figure 4-16 UNIX System Services file system selection

Allocating the z/OS File System and installation repository

Next, we run five jobs to install and run the Management Console server on z/OS. These jobs are found in the data set *yourhlq*.**SDYW**SAMP. “SDYW” is the product code for the Management Console.

Job one is using member DYWZCOPY. This job allocates and mounts a z/OS File System, which we used during our installation (Figure 4-17 on page 38). You can also use Hierarchical File System (HFS). It contains the Management Console installation repository. This job copies the installation repository from the data set that it is placed in by SMP/E into the zFS file system and sets the permissions on that repository. The repository is then accessible by Installation Manager for installing the Management Console.

```
//*****************************************************************
//*   The following step allocates a new filesystem for the      *
//*   Installation Manager repository for                        *
//*   IBM Management Console for IMS and DB2 for zOS.            *
//*****************************************************************
//CREATE   EXEC PGM=IKJEFT01,REGION=0M
//SYSTSPRT DD  SYSOUT=*
//BPXOUT   DD  SYSOUT=*
//STDOUT   DD  SYSOUT=*
//SYSTSIN  DD  *
 BPXBATCH SH +
 /usr/lpp/InstallationManager/bin/eclipse/tools/+
   zCreateFileSystem.sh +
    -name DB2TOOLS.HDYW.REPOS +
    -type ZFS +
    -cylinders 1500 250 +
    -mountpoint /usr/lpp/InstallationManagerRepository/HDYW110 +
    -owner #userid +
    -group #groupid +
    -perm 775
/*
..

//*********************************************************************
//*  Use OPUT command to copy the Management Console repository from  *
//*  the MVS PDS or PDSE where it was installed by SMP/E to the root  *
//*  of the HFS or ZFS filesystem created in the above step.          *
//*********************************************************************
//COPY     EXEC PGM=IKJEFT01
//SYSTSPRT DD   SYSOUT=*
//SYSPRINT DD   SYSOUT=*
//SYSTSIN  DD   *
  OPUT  'DBTLSP.SDYWBIN(DYWZREP0)' +
  '/usr/lpp/InstallationManagerRepository/HDYW110/mgmtconsole.zip' +
  binary
/*
```

Figure 4-17 Job to allocate and mount z/OS File System

Allocate and mount a new zFS file system

The next job we run is in member DYWZCFS. It allocates and mounts our new zFS file system into which Management Console is installed by Installation Manager (Figure 4-18 on page 39).

```
//CREATE   EXEC PGM=IKJEFT01,REGION=0M
//SYSTSPRT DD  SYSOUT=*
//BPXOUT   DD  SYSOUT=*
//STDOUT   DD  SYSOUT=*
//SYSTSIN  DD  *
 BPXBATCH SH +
 /usr/lpp/InstallationManager/bin/eclipse/tools/+
   zCreateFileSystem.sh +
    -name #mcdsn +
    -type ZFS +
    -volume #tvolfs +
    -cylinders 1500 250 +
    -mountpoint /usr/lpp/IBM/mgmtconsole/v110 +
    -owner #userid +
    -group #groupid +
    -perm 775
/*
```

Figure 4-18 Allocating and mounting the zFS system

Installing Management Console by using Installation Manager

The next job installs the Management Console by using the Installation Manager that we created previously (Figure 4-19). The job is in member DYWZINS.

```
//INSTALL  EXEC PGM=IKJEFT01,REGION=0M
//SYSTSPRT DD  SYSOUT=*
//BPXOUT   DD  SYSOUT=*
//STDOUT   DD  SYSOUT=*
//SYSTSIN  DD  *
 BPXBATCH SH +
  /usr/lpp/InstallationManager/V1R4/tools/imcl +
    install 'com.ibm.imac',imsbase.feature,db2.feature +
    -installationDirectory /IMTools/IBM/mgmtconsole/v110 +
    -sharedResourcesDirectory /IMTools/InstallationManager/+
     sharedResources +
    -repositories /IMTools/InstallationManagerRepository/HDYW110/+
     mgmtconsole.zip +
    -preferences com.ibm.cic.common.core.preferences.preserveDownlo+
     adedArtifacts=false +
    -properties +
     user.enableHttpPortCheckButtonUserData,,com.ibm.imac=false,+
     user.httpPortPropertyUserData,,com.ibm.imac=9080,+
     user.enableHttpsPortCheckButtonUserData,,com.ibm.imac=true,+
     user.httpsPortPropertyUserData,,com.ibm.imac=9443,+
     user.zosJavaHomeUserData,,com.ibm.imac=/usr/lpp/java/J7.0_64 +
    -acceptLicense +
    -sVP
/*
```

Figure 4-19 Installation of the IBM Installation Manager

Authorizing the Management Console to run as an MVS started task

Next, we run the job which will issue IBM RACF® commands to authorize the Management Console as a started task (Figure 4-20).

```
//     EXEC PGM=IKJEFT01
//SYSTSPRT DD SYSOUT=*
//SYSTSIN  DD *
    RDEF STARTED DYWZSRV.* UACC(NONE) STDATA(USER(wlpuser) +
      GROUP(wlpgroup) PRIVILEGED(NO) TRUSTED(NO) TRACE(YES))
    SETROPTS RACLIST(STARTED) GENERIC(STARTED) REFRESH
/*
```

Figure 4-20 Authorizing the Management Console as a started task

Copy the Management Console started task into a PROCLIB and start it

Once the previous steps completed successfully, we copied our started task PROC into a system PROCLIB and then started the Management Console.

You have now completed the installation and customization of the IBM Management Console for IMS and DB2 for z/OS (Figure 4-21).

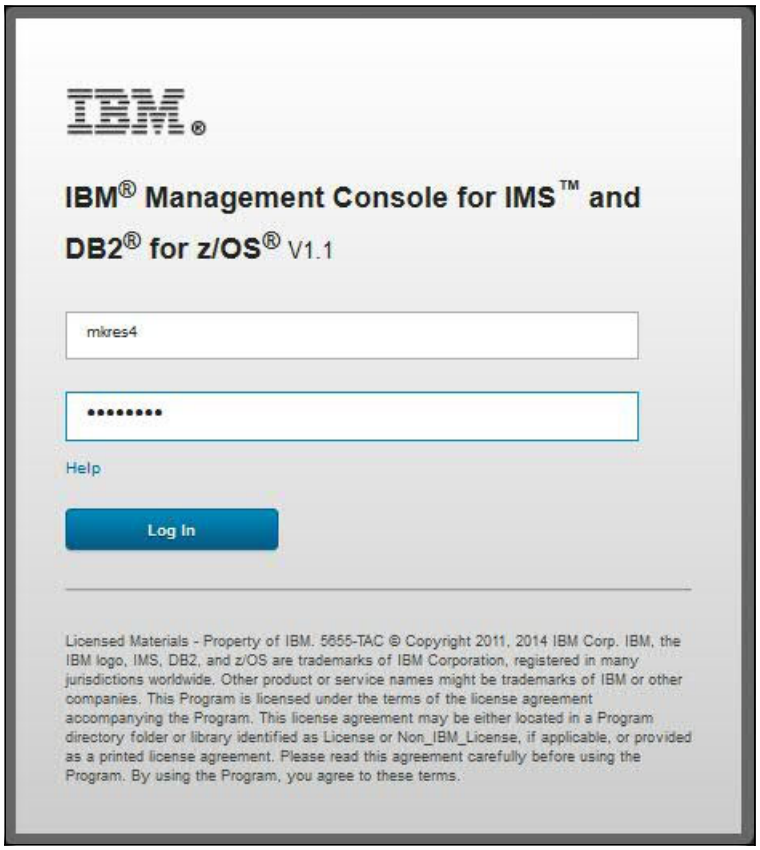


Figure 4-21 Log In to Management Console

5

IBM DB2 Utilities Solution Pack for z/OS configuration

The IBM DB2 Utilities Solution Pack for z/OS combines powerful DB2 tools that, when used together, can improve utility performance and simplify utility management. This chapter contains the following sections:

- Configuration overview
- DB2 Automation Tool Customization
- DB2 Utilities Enhancement Tool Customization configuration

© Copyright IBM Corp. 2015. All rights reserved.

5.1 Configuration overview

After completing the SMP/E steps to install the products, several customization steps are required for the products involved in the DB2 Autonomics solution. The SMP/E steps are not covered in this publication because the necessary information can be found on the respective program directories.

Every product from IBM DB2 Utilities Solution Pack for z/OS is customized via IBM Tools Customizer.

Because DB2 Automation Tool and DB2 Utilities Enhancement Tool have their respective User's Guide, we provide information related to configuration steps needed to register both tools to Autonomics Director and Management Console.

5.2 DB2 Automation Tool Customization

Next, we describe the configuration steps needed to customize DB2 Automation Tool and register it to Autonomics Director.

5.2.1 Tools Customizer steps for DB2 Automation Tool

The IBM Tools Customizer is used to configure DB2 Automation Tool. Use the following steps for configuration:

1. Every component installed through Tools Customizer requires a metadata library. This is normally shipped as a DENU-type library. In our environment, DBTLSP.SHAADENU is the metadata library for DB2 Automation Tool (Figure 5-1 on page 43).

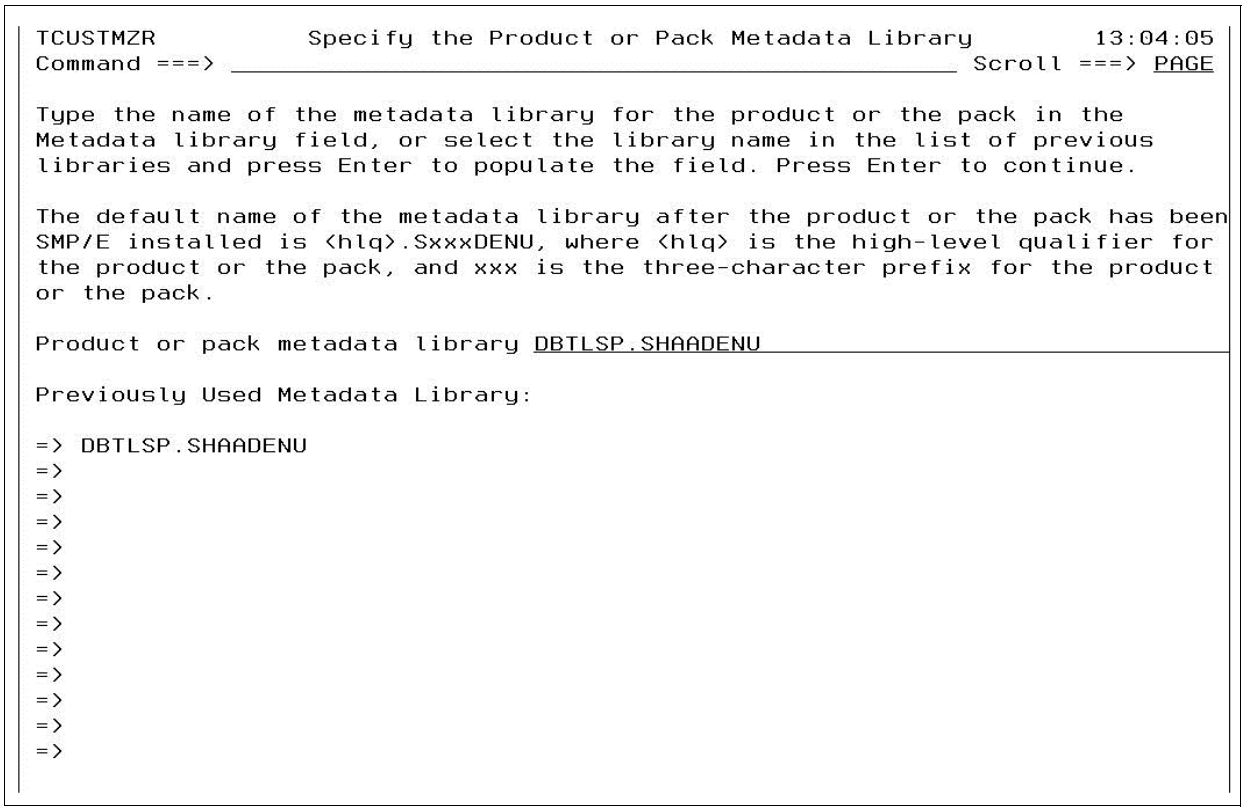


Figure 5-1 SHAADENU metadata selection

2. A Discover panel is displayed to trigger options used in previous configurations. We skipped this step because it is the first DB2 Automation Tool installation in this environment, but are showing the panel here for documentation purposes (Figure 5-2 on page 44).

```
TCUSTMZR         Discover Customized Product Information          End of data
Command ===> _____________________________________________ Scroll ===> PAGE

For the product you are customizing, the Discover EXEC retrieves product
information from an already customized product. Specify the required
information. To save your information and run the Discover EXEC, issue the RUN
command. To save your information and stay on this panel, issue the SAVE
command. To verify the syntax of your information without saving it, press
Enter. To save and exit, press  End.

Commands: RUN  SAVE


Discover EXEC for Extracting Information from an Already Customized product
 *Discover EXEC library  . . . DBTLSP.SHAADENU
  Discover EXEC name . . . . : HAADISC
 *Discover output data set . . DB2TOOLS.SBBY22.DISCOVER

Information for Discover EXEC
 *New Automation Tool load library
    DBTLSP.SHAALOAD
  New FEC load library . . DBTLSP.SFECLOAD
 *Previous control file  . HAA.DB2.CONTROL
 *Configuration ID . . . . . . . . . . . . . HAA
 *Previous Startup CLIST data set
    DB2TOOL.HAA410.SHAASAMP
 *Previous startup CLIST 2 . . . . . . . . . HAAV41C
```

Figure 5-2 DB2 Automation Tool discover panel

3. The next step is to associate the customization to a DB2 subsystem. We used D1C1, the same subsystem that was used for Management Console. After the D1C1 association, the next step is to go to “product parameters”, as shown in Figure 5-3.

```
TCUSTMZR         Customizer Workplace: DB2 Automation     DB2 entry associated
Command ===> ____________________________________________ Scroll ===> PAGE

Use the Generate jobs line command to select the DB2 entries on which to
customize the product, and press Enter to generate the customization jobs.

Commands:  ASSOCIATE  DISCOVER  GENERATEALL  JOBLIST

Product to Customize
  Product metadata library . : DBTLSP.SHAADENU          > LPAR  . . : SC63
  Product name . . . . . . . : IBM DB2 Automation Tool  > Version . : 4.2.0


Product and LPAR Parameter Status
  Line commands: E - Edit  B - Browse
  E Product parameters . : Verify Values
  _ LPAR parameters  . . : Verify Values

Associated DB2 Entries and Parameter Status
  Line commands: G - Generate jobs  E - Edit  B - Browse  C - Copy  R - Remove
  Cmd SSID Grpattch Lvl Mode User ID Date       Status              Message
  _   D1C1 D1CG     111 NFM  MKRES4  2015/05/14 Verify Values
------------------------------ End of DB2 entries ------------------------------
```

Figure 5-3 DB2 Automation Tool product parameters selection

4. On the first Product Parameter panels for Automation Tool (Figure 5-4), one important field is the IBM DB2 Utilities Solution Pack load library because it is crucial for the DB2 Autonomics Solution proper execution.

```
TCUSTMZR           Product Parameters: DB2 Automation Tool            13:21:41
Command ===> _____________________________________________ Scroll ===> CSR

Complete the following tasks to customize the products. The required tasks,
required steps within a required or selected task, and required parameters
are preceded by an asterisk (*). Ensure that values are specified for the
required parameters. Press End to save and exit.


Commands: SAVE  VERIFYOFF
Line Commands: / - Select

Product customization library  : DB2TOOLS.PRODUCT.CUST.$SC63$.HAA420
                                                                More:     +
Usage Notes:
 - If you edited the parameters on this panel before you edited the DB2
   parameters, some of the parameters on this panel might have moved from
   their original sections to the Required Parameters section. To avoid this
   situation, create and edit the DB2 entries before you edit the parameters
   on this panel. After you select or change tasks and steps on this panel,
   review the DB2 parameters, if necessary.

Common parameters
 *Startup CLIST library  . DB2TOOLS.CLIST_____________________________
 *Automation Tool load library
    DBTLSP.SHAALOADAD_____________________________
  Solution Pack load library
    DBTLSP.SBBYLOAD_____________________________
```

Figure 5-4 DB2 Automation Tool product parameters panel 1

5. On the second panel, some DB2 Automation Tool libraries are needed as shown in Figure 5-5.

```
TCUSTMZR           Product Parameters: DB2 Automation Tool            13:21:41
Command ===> _____________________________________________ Scroll ===> CSR

Complete the following tasks to customize the products. The required tasks,
required steps within a required or selected task, and required parameters
are preceded by an asterisk (*). Ensure that values are specified for the
required parameters. Press End to save and exit.


Commands: SAVE  VERIFYOFF
Line Commands: / - Select

Product customization library  : DB2TOOLS.PRODUCT.CUST.$SC63$.HAA420
                                                                More:   - +
 *Automation Tool panel library
    DBTLSP.SHAAPENU_____________________________
  User skeleton library  . _____________________________________________
 *Automation Tool skeleton library
    DBTLSP.SHAASLIB_____________________________
  FEC common code load library
    DBTLSP.SFECLOAD_____________________________
 *Control file . . . . . DB2TOOLS.HAA.DB2.CONTROL_____________
 *Configuration ID . . . . . HAA
 *Automation Tool SAMPLIB data set
    DBTLSP.SHAASAMP_____________________________
 *Owner of plans and packages  . . . . . MKRES1_____________________  >
 *Qualifier for repository object names  . . DLC_____________________  >
 *Automation Tool DBRM library
    DBTLSP.SHAADBRM_____________________________
```

Figure 5-5 DB2 Automation Tool product parameters panel 2

6. On the next panel (Figure 5-6), DB2 Autonomics Director load library and DBRM library are required because TCZ generates some Management Console-related jobs also.

```
TCUSTMZR           Product Parameters: DB2 Automation Tool            13:21:41
Command ===> _____________________________________________ Scroll ===> CSR

Complete the following tasks to customize the products. The required tasks,
required steps within a required or selected task, and required parameters
are preceded by an asterisk (*). Ensure that values are specified for the
required parameters. Press End to save and exit.


Commands: SAVE  VERIFYOFF
Line Commands: / - Select

Product customization library  : DB2TOOLS.PRODUCT.CUST.$SC63$.HAA420
                                                                More:    - +
 *Automation Tool customization package name HAA42TMK
 *Automation Tool customization plan name  . HAA42TMP
  DB2 Autonomics Director load library
    DBTLSP.SDYXLOAD
  DB2 Autonomics Director DBRM library
    DBTLSP.SDYXDBRM

* Configure the product CLISTs

  * Configure Startup CLISTs
   *Startup CLIST 1  . . . . . . . . . . . . HAAV42
   *Startup CLIST 2  . . . . . . . . . . . . HAAV42C
   *Automation Tool message library
      DBTLSP.SHAAMENU
    FEC message library  . DBTLSP.SFECMENU
```

Figure 5-6 DB2 Automation Tool product parameters panel 3

7. On the next panel (Figure 5-7), some FEC libraries are required.

```
TCUSTMZR           Product Parameters: DB2 Automation Tool            13:21:41
Command ===> _____________________________________________ Scroll ===> CSR

Complete the following tasks to customize the products. The required tasks,
required steps within a required or selected task, and required parameters
are preceded by an asterisk (*). Ensure that values are specified for the
required parameters. Press End to save and exit.


Commands: SAVE  VERIFYOFF
Line Commands: / - Select

Product customization library  : DB2TOOLS.PRODUCT.CUST.$SC63$.HAA420
                                                                More:    - +
    FEC panel library  . . DBTLSP.SFECPENU
    EMC load libraries . . ____________________________________________ Add

  * Copy the CLISTs
    FEC SAMPLIB  . . . . . DBTLSP.SFECSAMP

S Create the control file

    Create a new control file
    Volume serial number for control file  :

* Update the Control File

  * SSID-specific information
```

Figure 5-7 DB2 Automation Tool product parameters panel 4

8. On the next panel (Figure 5-8), we selected **DB2 Utilities Solution Pack Setup** in order to register DB2 Automation Tool with Autonomics Director.

```
TCUSTMZR           Product Parameters: DB2 Automation Tool            13:21:41
Command ===> ____________________________________________________ Scroll ===> CSR

Complete the following tasks to customize the products. The required tasks,
required steps within a required or selected task, and required parameters
are preceded by an asterisk (*). Ensure that values are specified for the
required parameters. Press End to save and exit.


Commands: SAVE  VERIFYOFF
Line Commands: / - Select

Product customization library  : DB2TOOLS.PRODUCT.CUST.$SC63$.HAA420
                                                                More:    - +
  _ DB2 Recovery Expert information
    DB2 Recovery Expert control file

    DB2 Recovery Expert load libraries
      DB2TOOL.ARY310.SARYLOAD                                     Add

  _ DB2 HPU information
    DB2 HPU load libraries DB2TOOL.HPU420.SHPULOAD                Add
    DB2 HPU ver/rel/mod  . . . . . . . . . : 420 (420)

/ DB2 Utilities Solution Pack Setup

    Register with Autonomics Director
```

Figure 5-8 DB2 Automation Tool product parameters panel 5

9. On the next panel (Figure 5-9), there was nothing to complete because “Create or update DB2 objects” was already selected.

```
TCUSTMZR           Product Parameters: DB2 Automation Tool            13:21:41
Command ===> ____________________________________________________ Scroll ===> CSR

Complete the following tasks to customize the products. The required tasks,
required steps within a required or selected task, and required parameters
are preceded by an asterisk (*). Ensure that values are specified for the
required parameters. Press End to save and exit.


Commands: SAVE  VERIFYOFF
Line Commands: / - Select

Product customization library  : DB2TOOLS.PRODUCT.CUST.$SC63$.HAA420
                                                                More:    - +
* Create or update DB2 Objects

  _ Drop repository

  * Preview repository changes

  * Execute repository changes

  _ Create optional user-managed index files
    Volume serial number for SYSCOPY index :

  _ Create optional indexes

  _ Copy Automation Tool repository
```

Figure 5-9 DB2 Automation Tool product parameters panel 6

10. On the next panel (Figure 5-10), the **Management Console Stored Procedures** option should be selected in order to have TCz generate the jobs needed to integrate DB2 Automation Tool and Management Console.

```
TCUSTMZR           Product Parameters: DB2 Automation Tool            13:21:41
Command ===> _____________________________________________ Scroll ===> CSR

Complete the following tasks to customize the products. The required tasks,
required steps within a required or selected task, and required parameters
are preceded by an asterisk (*). Ensure that values are specified for the
required parameters. Press End to save and exit.


Commands: SAVE  VERIFYOFF
Line Commands: / - Select

Product customization library  : DB2TOOLS.PRODUCT.CUST.$SC63$.HAA420
                                                                More:    - +
    Qualifier for previous repository object names
      DLC                                                              >
    From ver/rel/mod . . . . . . . . . . . : 410 (310, 410, 420)

* Automation Tool Plans and Packages

  _ FREE

  * BIND

  * GRANT execute authority

/ Management Console Stored Procedures

    Free Management Console packages
```

Figure 5-10 DB2 Automation Tool product parameters panel 7

11. On the next panel (Figure 5-11 on page 49), the **Job Tracking Started Task** and **Install the sample profiles** options were selected.

```
TCUSTMZR           Product Parameters: DB2 Automation Tool            13:21:41
Command ===> _____________________________________________ Scroll ===> CSR

Complete the following tasks to customize the products. The required tasks,
required steps within a required or selected task, and required parameters
are preceded by an asterisk (*). Ensure that values are specified for the
required parameters. Press End to save and exit.


Commands: SAVE  VERIFYOFF
Line Commands: / - Select

Product customization library  : DB2TOOLS.PRODUCT.CUST.$SC63$.HAA420
                                                                More:    - +
    Drop Management Console stored procedures

    Create Management Console stored procedures

    Management Console BIND

    Create Management Console WLM address space

/ Job Tracking Started Task

    Create the job tracking started task

/ Install the sample profiles

    Install sample profiles
```

Figure 5-11 DB2 Automation Tool product parameters panel 8

12. On the last panel (Figure 5-12), we completed the Creator Name for the sample profiles.

```
TCUSTMZR           Product Parameters: DB2 Automation Tool          End of data
Command ===> _____________________________________________ Scroll ===> CSR

Complete the following tasks to customize the products. The required tasks,
required steps within a required or selected task, and required parameters
are preceded by an asterisk (*). Ensure that values are specified for the
required parameters. Press End to save and exit.


Commands: SAVE  VERIFYOFF
Line Commands: / - Select

Product customization library  : DB2TOOLS.PRODUCT.CUST.$SC63$.HAA420
                                                                More:    -

/ Install the sample profiles

    Install sample profiles
    Sample profile creator name  . . . . . : HAAUSER
    Sample profile update option . . . . . : V (U, V, N)

_ Add Automation Tool to the DB2 Admin Launchpad

    Create the REXX to add Automation Tool to the Launchpad
    DB2 Administration Tool high-level qualifier
      ADBHILVL
    ADBDMTI EXEC data set  ADBHILVL.SADBEXEC
    DB2 Admin version  . . . . . : 10
```

Figure 5-12 DB2 Automation Tool product parameters panel 9

13. After changing and selecting the options needed to customize DB2 Automation Tool on all Product Parameters panels, we saved and pressed PF3 to return to the main Customizer Workplace panel.

14. On the Customizer Workplace panel (Figure 5-13), we typed `E` on the “LPAR parameters” line.

```
TCUSTMZR         Customizer Workplace: DB2 Automation Tool     Row 1 to 1 of 1
Command ===> ____________________________________________________ Scroll ===> CSR

Use the Generate jobs line command to select the DB2 entries on which to
customize the product, and press Enter to generate the customization jobs.

Commands:  ASSOCIATE  DISCOVER  GENERATEALL  JOBLIST

Product to Customize
  Product metadata library . : DBTLSP.SHAADENU           > LPAR  . . : SC63
  Product name . . . . . . . : IBM DB2 Automation Tool   > Version . : 4.2.0


Product and LPAR Parameter Status
  Line commands: E - Edit  B - Browse
  _ Product parameters . . : Ready to Customize
  E LPAR parameters  . . . : Ready to Customize

Associated DB2 Entries and Parameter Status
  Line commands: G - Generate jobs  E - Edit  B - Browse  C - Copy  R - Remove
  Cmd SSID Grpattch Lvl Mode User ID Date       Status              Message
  _   D1C1 D1CG     111 NFM  MKRES4  2015/05/14 Verify Values
------------------------------ End of DB2 entries -----------------------------
```

Figure 5-13 LPAR parameters selection

15. On the LPAR parameters panel, several fields should be completed as shown in Figure 5-14.

```
TCUSTMZR           LPAR Parameters: DB2 Automation Tool              14:59:21
Command ===> ____________________________________________________ Scroll ===> CSR

Ensure that values are specified for the required LPAR parameters. Press End
to save and exit.

Commands: SAVE  VERIFYOFF


Job Entry Subsystem (JES) - common
 *JES version  . . . . . . . . . . . . . . . 2

ISPF Libraries - common
 *Message library  . . . . ISP.SISPMENU                                  Add
 *ISPF table input library ISP.SISPTENU                                  Add

Language Environment Libraries -common
 *Load library . . . . . . CEE.SCEELLIB                                  Add

Automation Tool LPAR Parameters
 *Automation Tool PROCLIB  DB1CM.PROCLIB
  Job tracking started task PROC member  . . DC1GHAA
 *Job tracking started task Parms data set
   DB2CM.PARMLIB(HAAPARM)
 *Job tracking started task ID . . . . . . . HAAT
 *Job tracking started task AUTHID . . . . . STC
```

Figure 5-14 LPAR parameters panel

16. After saving the options for LPAR parameters, we press PF3 and returned to the main DB2 Automation Tool Workplace panel.

17. On the Customizer Workplace panel (Figure 5-15), we typed `E` in front of D1C1, the DB2 subsystem associated with this customization.

```
TCUSTMZR          Customizer Workplace: DB2 Automation Tool      Row 1 to 1 of 1
Command ===> _____________________________________________________ Scroll ===> CSR

Use the Generate jobs line command to select the DB2 entries on which to
customize the product, and press Enter to generate the customization jobs.

Commands:  ASSOCIATE  DISCOVER  GENERATEALL  JOBLIST

Product to Customize
  Product metadata library . : DBTLSP.SHAADENU           > LPAR  . . : SC63
  Product name . . . . . . . : IBM DB2 Automation Tool   > Version . : 4.2.0


Product and LPAR Parameter Status
  Line commands: E - Edit  B - Browse
  _ Product parameters . : Ready to Customize
  _ LPAR parameters  . . : Ready to Customize

Associated DB2 Entries and Parameter Status
  Line commands  G - Generate jobs  E - Edit  B - Browse  C - Copy  R - Remove
  Cmd SSID Grpatch Lvl Mode User ID Date       Status               Message
  E   D1C1 D1CG     111 NFM  MKRES4  2015/05/14 Verify Values
  ---------------------------- End of DB2 entries -----------------------------
```

Figure 5-15 DB2 parameters selection

18. On the first DB2 parameters panel (Figure 5-16), some DB2 libraries, buffer pools, and other general information should be completed.

```
TCUSTMZR            DB2 Parameters: DB2 Automation Tool                14:07:17
Command ===> _____________________________________________________ Scroll ===> CSR

Ensure that values are specified for the required DB2 parameters. Press End
to save and exit.

Commands: SAVE  VERIFYOFF
                                                                    More:     +
 DB2 subsystem ID  . . . . . . . . . . . . : D1C1
 Group attach name . . . . . . . . . . . . . D1CG

General DB2 Information - common
 *Mode . . . . . . . . . . . . . . . . . . . NFM (CM, CM8, CM9, NFM)
 *Level number . . . . . . . . . . . . . . . 111 (101, 111)

DB2 Libraries - common
 *Load library . . . . . . DB1CT.SDSNLOAD                                  Add
 *Run library  . . . . . . DB1CM.RUNLIB.LOAD                               Add
 *Exit library . . . . . . DB1CT.SDSNEXIT                                  Add
 *Bootstrap data set . . . DB1CB.D1C1.BSDS01                               Add

DB2 Buffer Pools - common
 *Name of the 4 KB buffer pool . . . . . . . BP0
 *Name of the 16 KB buffer pool  . . . . . . BP16K0
 *Name of the 32 KB buffer pool  . . . . . . BP32K

DB2 Utilities - common
  SYSAFF for DB2 utilities . . . . . . . . .
 *Plan name for the DSNTIAD utility  . . . . DSNTIA11
```

Figure 5-16 DB2 parameters panel 1

19. On the next DB2 parameters panel (Figure 5-17), some general information is required, including the default WLM environment that should be used. It is crucial for DB2 Autonomics usage that the same WLM environment created on IBM Management Console configuration is used, in our case, D1CGWLMA.

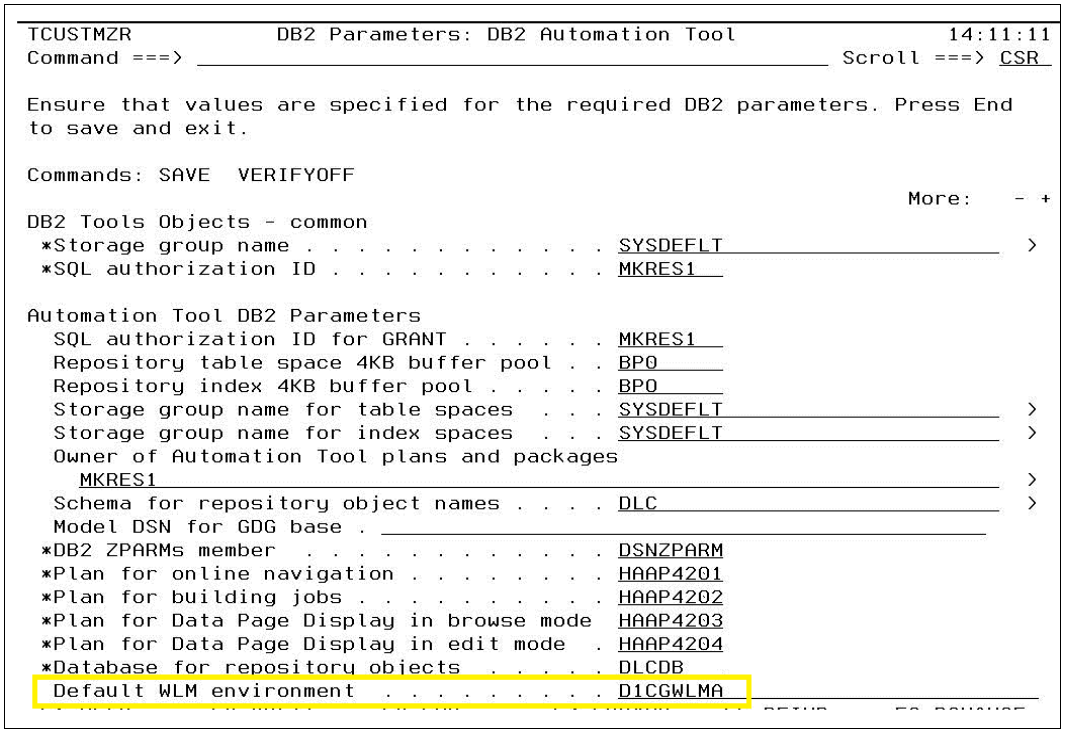


Figure 5-17 DB2 parameters panel 2

20. On the next panel (Figure 5-18), we had to complete more general information.

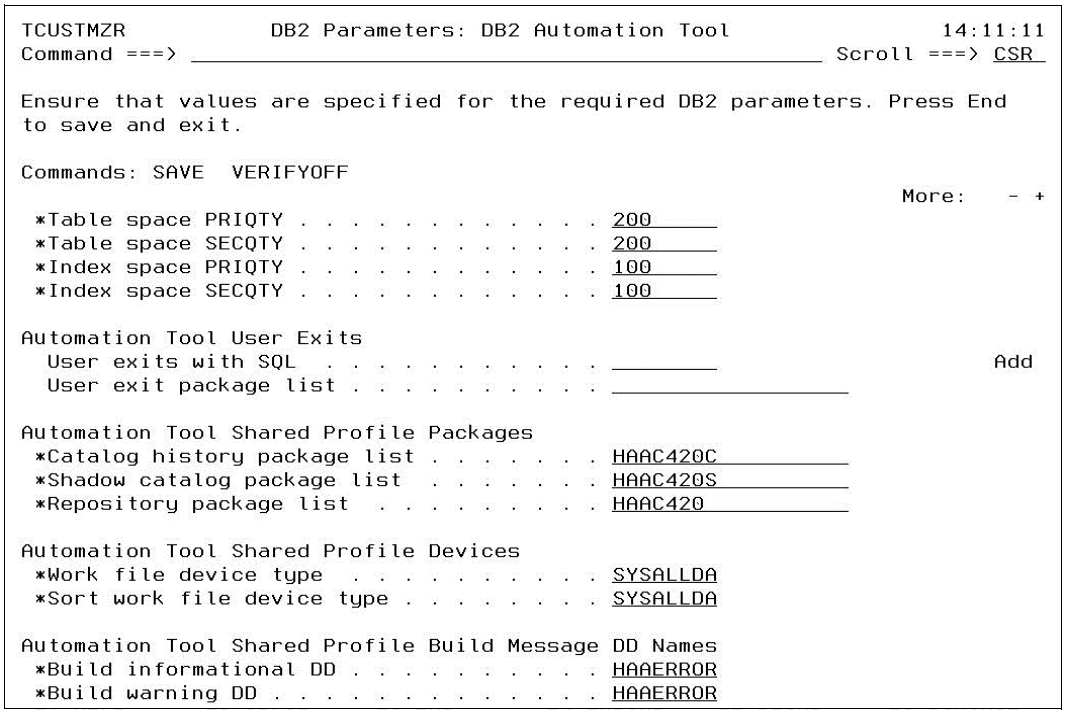


Figure 5-18 DB2 parameters panel 3

21. On the next panel (Figure 5-19), more general information is required, including the option to `Enable administrative task scheduler support`. Because we use Administrative Task Scheduler, we enabled this option.

```
TCUSTMZR           DB2 Parameters: DB2 Automation Tool                14:11:11
Command ===> _____________________________________________________ Scroll ===> CSR

Ensure that values are specified for the required DB2 parameters. Press End
to save and exit.

Commands: SAVE  VERIFYOFF
                                                                    More:    - +
 *Build error DD . . . . . . . . . . . . . . HAAERROR

Automation Tool Shared Profile Miscellaneous
  Job tracking started task name . . . . . . ____
 *Max primary space allocation . . . . . . . 999999
 *Size unit for max primary space allocation C (T, C, M)
 *Secondary allocation percentage  . . . . . 050
 *Utility region size  . . . . . . . . . . . 1024
 *DB2 fetch buffer size  . . . . . . . . . . 0004
 *Parallel MVS catalog locates . . . . . . . 10
 *Terminate utility if an abend occurs . . . N (Y, N)
 *Generate STEPLIB DDs . . . . . . . . . . . Y (Y, N)
 *Gen image copy DSNs in GMT . . . . . . . . Y (Y, N)
 *Indexes of DEFINE NO table spaces  . . . . N (Y, N)
 *Enable administrative task scheduler support
    Y (Y, N)
 *Administrative task scheduler max history  0100
 *Segment size for repository table spaces . 4                          List
 *Event notifications  . . . . . . . . . . . Y (Y, N)
 *Sysout Class for SMTP output . . . . . . . A
 *z/OS Host Name . . . . . . . . . . . . . . WTSCPLX2
```

Figure 5-19 DB2 parameters panel 4

22. On the last panel (Figure 5-20), it is important to use the same PROC member that was used in the Management Console installation. The NUMTCB value should be the same as used previously for the Management Console.

```
TCUSTMZR           DB2 Parameters: DB2 Automation Tool                End of data
Command ===> _____________________________________________________ Scroll ===> CSR

Ensure that values are specified for the required DB2 parameters. Press End
to save and exit.

Commands: SAVE  VERIFYOFF
                                                                    More:    -
 *Indexes of DEFINE NO table spaces  . . . . N (Y, N)
 *Enable administrative task scheduler support
    Y (Y, N)
 *Administrative task scheduler max history  0100
 *Segment size for repository table spaces . 4                          List
 *Event notifications  . . . . . . . . . . . Y (Y, N)
 *Sysout Class for SMTP output . . . . . . . A
 *z/OS Host Name . . . . . . . . . . . . . . WTSCPLX2
 *SMTP task name . . . . . . . . . . . . . . SMPTTASK

Automation Tool Shared Profile Sort Parameters
  Primary sort work space  . . . . . . . . . _____
  Secondary sort work space  . . . . . . . . _____
  Number of sort work DDs  . . . . . . . . . __

Automation Tool Stored Procedure Parameters
 *Management Console package list  . . . . . MCCOLL1
 *Management Console PROC member . . . . . . D1CGWLMA
  Management Console WLM environment . . . . ________________________________
 *Management Console WLM NUMTCB  . . . . . . 8
```

Figure 5-20 DB2 parameters panel 5

23. After this last panel, we saved and pressed PF3 to return to the main Customizer Workplace panel.

24. On the Customizer Workplace panel (Figure 5-21), we typed `G` in front of D1C1 to generate the customization jobs.

```
TCUSTMZR         Customizer Workplace: DB2 Automation Tool     Row 1 to 1 of 1
Command ===> _____________________________________________________ Scroll ===> CSR

Use the Generate jobs line command to select the DB2 entries on which to
customize the product, and press Enter to generate the customization jobs.

Commands:  ASSOCIATE  DISCOVER  GENERATEALL  JOBLIST

Product to Customize
  Product metadata library . : DBTLSP.SHAADENU          > LPAR  . . : SC63
  Product name . . . . . . . : IBM DB2 Automation Tool  > Version . : 4.2.0


Product and LPAR Parameter Status
  Line commands: E - Edit  B - Browse
  _ Product parameters . : Ready to Customize
  _ LPAR parameters  . . : Ready to Customize

Associated DB2 Entries and Parameter Status
  Line commands: G - Generate jobs  E - Edit  B - Browse  C - Copy  R - Remove
  Cmd SSID Grpattch Lvl Mode User ID Date       Status           Message
  G   D1C1 D1CG     111 NFM  MKRES1  2015/05/14 Ready to Customize  Edited
------------------------------ End of DB2 entries -----------------------------
```

Figure 5-21 Job generation selection

25. The next panel (Figure 5-22 on page 55) shows the customization jobs generated by TCz and the library that these jobs were stored in. From this panel it is possible to browse, edit, and even submit the jobs. We reviewed and ran all of the customization jobs.

```
TCUSTMZR                Finish Product Customization         Row 1 to 15 of 17
Command ===> _____________________________________________ Scroll ===> CSR

For a first-time customization, submit the jobs in the members in the order
in which they apply to the DB2 entries. Otherwise, submit only the necessary
jobs that were generated after changes were made. To submit jobs, browse
the members and issue the TSO SUBMIT command.

Line Commands: E - Edit  B - Browse

 _  Product customization library:   DB2TOOLS.PRODUCT.CUST.$SC63$.HAA420

 Cmd Member   SSID Grpattch Template Date       Description
 -   -------- ---- ----     -------- ---------- ------------------------------>
 _   A0V42     --   --       HAAV42   2015/05/14 Configures Startup CLIST 1
 _   A1V42C    --   --       HAAV42C  2015/05/14 Configures Startup CLIST 2
 _   A2EXECS   --   --       HAAEXECS 2015/05/14 Copies the CLISTs
 _   A3CNTFL   --   --       HAACNTFL 2015/05/14 Creates control file
 _   A4CF2AAA D1C1 D1CG     HAACF2A  2015/05/14 Update Control File SSID specif
 _   A5CF2BAB  --  D1CG     HAACF2B  2015/05/14 Updates datasharing group infor
 _   A8USLAA  D1C1 D1CG     HAAUSL   2015/05/14 Utilities Solution Pack Setup
 _   B0DDLYAA D1C1 D1CG     HAADDLY  2015/05/14 Creates preview DDL
 _   B1DDLXAA D1C1 D1CG     HAADDLX  2015/05/14 Executes update DDL
 _   B6#BNDAA D1C1 D1CG     HAA#BND  2015/05/14 Binds plans and packages
 _   B7GRANAA D1C1 D1CG     HAAGRANT 2015/05/14 Grant Execute Authority
 _   C0DDLSAA D1C1 D1CG     HAADDLS  2015/05/14 Management Console stored proce
 _   C1BND2AA D1C1 D1CG     HAABND2  2015/05/14 Binds packages for Management C
 _   C2WLM1AA D1C1 D1CG     HAAWLM1  2015/05/14 Management Console WLM address
 _   C3PROC    --   --       HAAPROC  2015/05/14 Creates the started task
```

Figure 5-22 Customization jobs

5.2.2 More Automation Tool configuration steps

Some additional Automation Tool configuration steps need to be done before using the product. Probably, your z/OS system programmer is required for most of these:

- Add the program FEC$TSOC to the AUTHPGM and AUTHTSF sections of member IKJTSO00 in SYS1.PARMLIB.IKJTSO02 modification in order to add FEC$TSOC to AUTHPGM and AUTHTSF.
- Include SHAALOAD libraries as part of your authorized list. If you installed FEC into a separate library from DB2 Automation Tool, APF authorizes the FEC load library (usually, SFECLOAD).
- Add the IEFACTRT exit to the SMFPRMxx member of SYS1.PARMLIB. The IEFACTRT exit is required for successful operation of the execution reports facility. If the exit is not currently defined, add the IEFACTRT exit at either the SYS or SUBSYS level to the appropriate JES subsystem in the SMFPRMxx member of your site's SYS1.PARMLIB. The HAAPROC member checks for IEFACTRT at the SUBSYS.JES level or SYS level upon start. If the IEFACTRT entry is not found, DB2 Automation Tool job tracking will not work. If the IEFACTRT entry is found, the CSVDYNEX service dynamically defines the exit for DB2 Automation Tool job tracking. In addition, ensure that type 30 SMF records are included in the SMF record types specified in SMFPRMxx.

Another important point to mention is that for Automation Tool integration with Management Console, the repository file (in SHAABIN(HAA$REP0)) should be copied into the file system that was created for Management Console.

The jobs that are needed to install the Automation extension are as follows:

- HAAZCOPY allocates and mounts a new ZFS or HFS file system to be used for the Management Console product repositories, and then copies the DB2 Automation Tool extension repository file (in SHAABIN(HAA$REP0)) into the file system.
- HAAZINS installs the DB2 Automation Tool extension into the Management Console using IBM Installation Manager.
- HAAZUNI uninstalls the DB2 Automation Tool extension using IBM Installation Manager.

5.3 DB2 Utilities Enhancement Tool Customization configuration

Next, we describe the configuration steps that are needed to customize DB2 Utilities Enhancement Tool.

5.3.1 Tools Customizer steps for DB2 Utilities Enhancement Tool

We use IBM Tools Customizer to configure DB2 Utilities Enhancement Tool. Follow these steps:

1. Every component installed through Tools Customizer requires a metadata library. This is normally shipped as a DENU-type library. In our environment, DBTLSP.SABPDENU is the metadata library for DB2 Utilities Enhancement Tool (Figure 5-23).

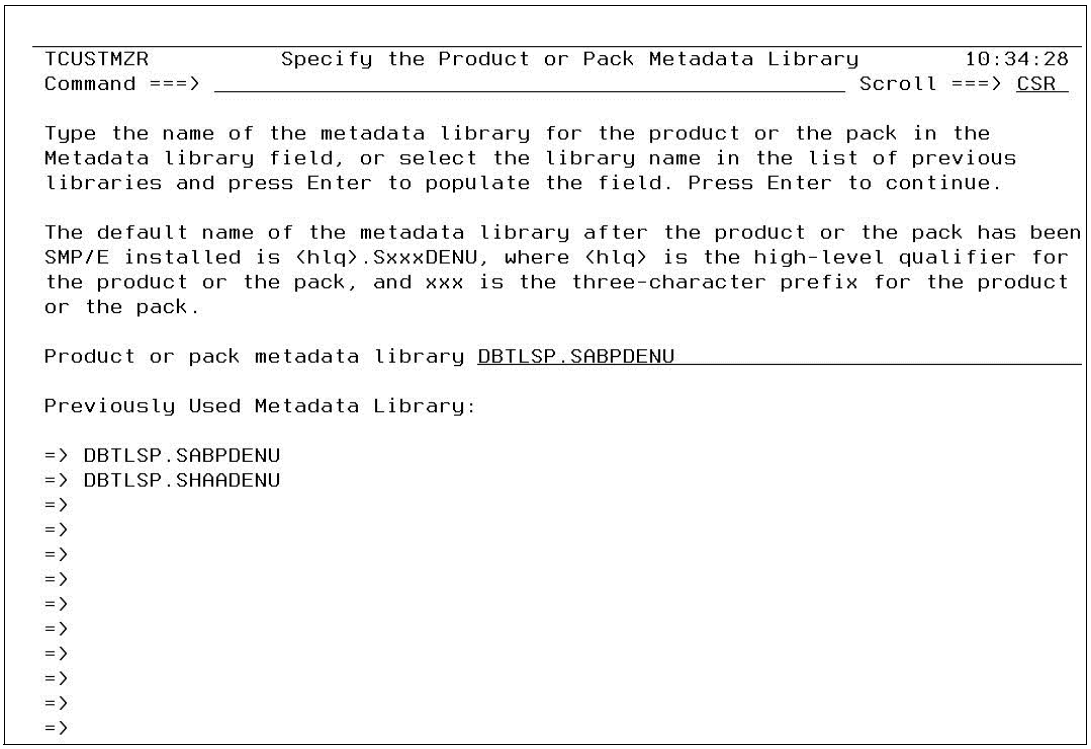


Figure 5-23 SABPDENU selection

2. On the main Customizer Workplace panel (Figure 5-24), we typed E in front of “Product parameters”.

```
TCUSTMZR         Customizer Workplace: DB2 UET                  Row 1 to 1 of 1
Command ===> ____________________________________________________ Scroll ===> CSR

Use the Generate jobs line command to select the DB2 entries on which to
customize the product, and press Enter to generate the customization jobs.

Commands:  ASSOCIATE  DISCOVER  GENERATEALL  JOBLIST  CONFIGURATION

Product to Customize
  Product metadata library . : DBTLSP.SABPDENU           > LPAR  . . : SC63
  Product name . . . . . . . : DB2 Utilities Enhancemen  > Version . : 2.2.0
  Configuration ID: ABP1  >  Description: DB2 Utilities Enhancement Tool fo  >

Product and LPAR Parameter Status
  Line commands: E - Edit  B - Browse
  E Product parameters . : Ready to Customize
  _ LPAR parameters  . . : Not Required

Associated DB2 Entries and Parameter Status
  Line commands: G - Generate jobs  E - Edit  B - Browse  C - Copy  R - Remove
  Cmd SSID Grpattch Lvl Mode User ID Date       Status               Message
  _   D1C1 D1CG     111 NFM  MKRES2  2015/05/18 Ready to Customize
------------------------------ End of DB2 entries ------------------------------
```

Figure 5-24 DB2 Utilities Enhancement Tool product parameters selection

3. On the first product parameters panel for DB2 Utilities Enhancement Tool (Figure 5-25), some general information is required, including the Load Library for the IBM DB2 Utilities Solution Pack.

```
TCUSTMZR           Product Parameters: DB2 UET                        11:29:14
Command ===> ____________________________________________________ Scroll ===> CSR

Complete the following tasks to customize the products. The required tasks,
required steps within a required or selected task, and required parameters
are preceded by an asterisk (*). Ensure that values are specified for the
required parameters. Press End to save and exit.


Commands: SAVE  VERIFYOFF
Line Commands: / - Select

Product customization library  : DB2TOOLS.PRODUCT.CUST.$SC63$.ABP220
                                                                  More:      +
Usage Notes:
 If you did not run the Discover EXEC to use values from a previous version
   of DB2 Utilities Enhancement Tool, you must define a primary DB2 subsystem
   before you edit the parameters on this panel.

Common parameters
 *DB2 UET data sets HLQ  . DBTLSP_____________________________
 *DB2 UET plan qualifier . . . . . . . . . . ABP22PLN
 *DB2 UET started task user ID . . . . . . . MKRES1__
  Solution Pack load library
    DBTLSP.SBBYLOAD_____________________________
 *DB2 UET plan name  . . . . . . . . . . . . ABPPLAN_
 *DB2 UET configuration name . . . . . . . . ABP1
 *Thread cancel member name  . . . . . . . . ABP@BCAN
 *Global parameters member name  . . . . . . ABP@BGLB
```

Figure 5-25 DB2 Utilities Enhancement Tool product parameters panel 1

4. On the next panel (Figure 5-26), some more general information needs to be completed, such as the work database name that will be created for the DB2 Utilities Enhancement Tool.

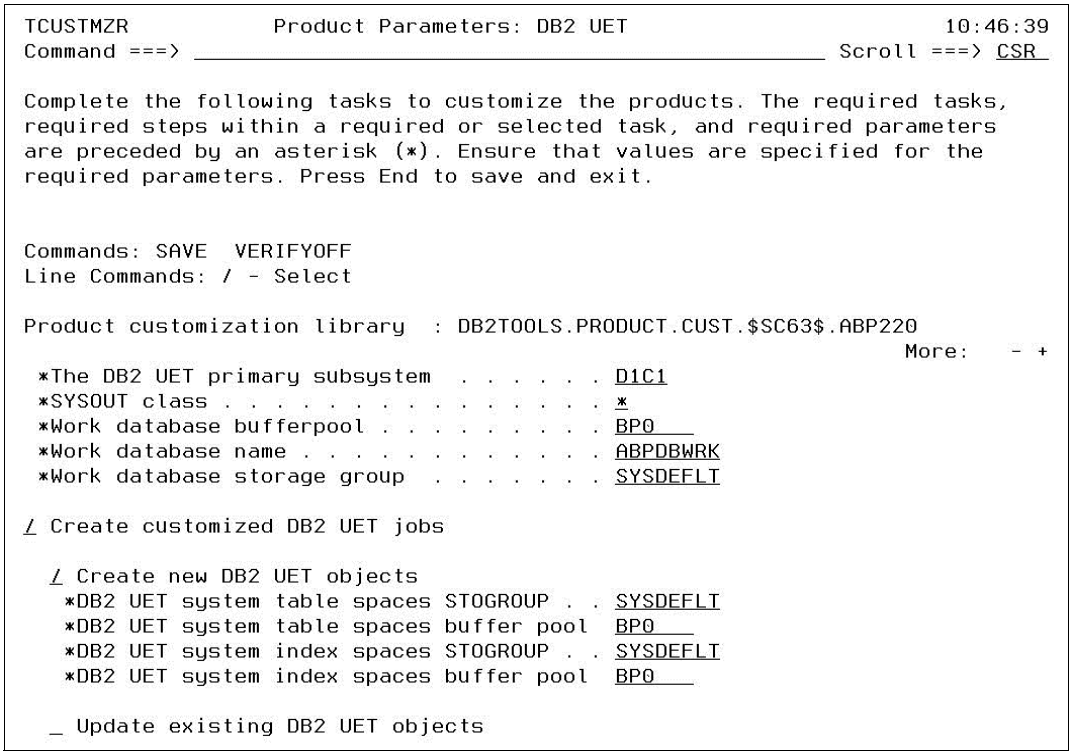


Figure 5-26 DB2 Utilities Enhancement Tool product parameters panel 2

5. On the next panel (Figure 5-27 on page 59), some options can be selected in a DB2 Utilities Enhancement Tool version migration, which is not our case.

```
TCUSTMZR           Product Parameters: DB2 UET                      10:46:39
Command ===> _____________________________________________ Scroll ===> CSR

Complete the following tasks to customize the products. The required tasks,
required steps within a required or selected task, and required parameters
are preceded by an asterisk (*). Ensure that values are specified for the
required parameters. Press End to save and exit.


Commands: SAVE  VERIFYOFF
Line Commands: / - Select

Product customization library  : DB2TOOLS.PRODUCT.CUST.$SC63$.ABP220
                                                                More:    - +
  _ Update existing DB2 UET objects
    DB2 UET system table spaces STOGROUP . : SYSDEFLT
    DB2 UET system table spaces buffer pool  BP0
    DB2 UET system index spaces STOGROUP . : SYSDEFLT
    DB2 UET system index spaces buffer pool  BP0

  _ Create additional indexes (optional)

  _ Create LOG/AUDIT Migration job (optional)
    V2.1 LOG and AUDIT tables CREATOR name : ABP21

  _ Create JOURNAL Migration job (optional)
    V2.1 JOURNAL tables CREATOR name . . . : ABP21

  _ Create RUNSTATS job (optional)
```

Figure 5-27 DB2 Utilities Enhancement Tool product parameters panel 3

6. On the next panel (Figure 5-28), some options related to the DB2 Utilities Enhancement Tool options module need to be completed. We just used the defaults.

```
TCUSTMZR           Product Parameters: DB2 UET                      10:46:39
Command ===> _____________________________________________ Scroll ===> CSR

Complete the following tasks to customize the products. The required tasks,
required steps within a required or selected task, and required parameters
are preceded by an asterisk (*). Ensure that values are specified for the
required parameters. Press End to save and exit.


Commands: SAVE  VERIFYOFF
Line Commands: / - Select

Product customization library  : DB2TOOLS.PRODUCT.CUST.$SC63$.ABP220
                                                                More:    - +
  * Create BIND job

  _ Create cleanup job for a worklist table (optional)

* Create the Options Module

  * Specify the parameters
   *STC audit active . . . . . YES (YES, NO)
   *Maximum STC audit age  . . . . . 0
   *Escalated cancel active  . . . . . NO  (YES, NO)
   *Connect to all DB2 subsystems? . . . . . YES (YES, NO)
   *Connection idle timeout  . . . . . 300
   *DB2 tasks count  . . . . . 2
   *DB2 task idle timeout  . . . . . 900
   *STC logging active . . . . . YES (YES, NO)
```

Figure 5-28 DB2 Utilities Enhancement Tool product parameters panel 4

7. On the next panel (Figure 5-29), we just changed the SVC number to 254, which was the available one in our environment.

```
TCUSTMZR           Product Parameters: DB2 UET                        10:46:39
Command ===> _____________________________________________ Scroll ===> CSR

Complete the following tasks to customize the products. The required tasks,
required steps within a required or selected task, and required parameters
are preceded by an asterisk (*). Ensure that values are specified for the
required parameters. Press End to save and exit.


Commands: SAVE  VERIFYOFF
Line Commands: / - Select

Product customization library  : DB2TOOLS.PRODUCT.CUST.$SC63$.ABP220
                                                                More:    - +
   *Maximum STC log age  . . . . . . . . . . 0
   *Override mapping table . . . . . . . . . NO  (YES, NO)
   *Post-cancel user exit  . . . . . . . . . NONE
   *Pre-cancel user exit . . . . . . . . . . NONE
   *Security-cancel exit . . . . . . . . . . NONE
   *SVC number . . . . . . . . . . . . . . . 254
   *Shadow database prefix . . . . . . . . . ABPS
   *Shadow schema  . . . . . . . . . . . . . ABPSTC
   *DB2 UET trace active . . . . . . . . . . YES (YES, NO)
   *Size of trace table  . . . . . . . . . . 1
   *Workfile data class  . . . . . . . . . . NONE
   *Workfile management class  . . . . . . . NONE
   *Workfile storage class . . . . . . . . . NONE
   *Workfile unit  . . . . . . . . . . . . . SYSALLDA
   *Maximum worklist table age . . . . . . . 0
```

Figure 5-29 DB2 Utilities Enhancement Tool product parameters panel 5

8. On the next panel (Figure 5-30), no changes were made and we used the defaults.

```
TCUSTMZR           Product Parameters: DB2 UET                        10:46:39
Command ===> _____________________________________________ Scroll ===> CSR

Complete the following tasks to customize the products. The required tasks,
required steps within a required or selected task, and required parameters
are preceded by an asterisk (*). Ensure that values are specified for the
required parameters. Press End to save and exit.


Commands: SAVE  VERIFYOFF
Line Commands: / - Select

Product customization library  : DB2TOOLS.PRODUCT.CUST.$SC63$.ABP220
                                                                More:    - +
   *WTO ROUTCDE  . . . . . . . . . . . . . . 11

* Create SAMPLIB members

  * Create BCAN, BGLB, PROC, and PLCY members
   *Reuse UET V2.1 policy  . . . . . . . . . : N
   *Cancellation processing  . . . . . . . . BACKOUT_______              List
   *Escalate thread cancel . . . . . . . . . NO  (YES, NO)
   *Check thread termination retry count . . 20
   *Check thread termination interval  . . . 3
   *Escape character . . . . . . . . . . . . +
   *Execution mode . . . . . . . . . . . . . EXECUTE                     List
   *On thread cancel failure
      CONTINUE_____________________________                              List
   *Report type  . . . . . . . . . . . . . . SUMMARY (SUMMARY, DETAIL)
 F1=HELP      F2=SPLIT     F3=END       F4=EXPAND    F5=RFIND     F6=RCHANGE
 F7=UP        F8=DOWN      F9=SWAP      F10=LEFT     F11=RIGHT    F12=RETRIEVE
```

Figure 5-30 DB2 Utilities Enhancement Tool product parameters panel 6

9. On the last product parameters panel (Figure 5-31), we selected the option to **Create maintenance members**, but this is optional and does not influence DB2 Autonomics Solution behavior.

```
TCUSTMZR           Product Parameters: DB2 UET                      End of data
Command ===> ____________________________________________________ Scroll ===> CSR

Complete the following tasks to customize the products. The required tasks,
required steps within a required or selected task, and required parameters
are preceded by an asterisk (*). Ensure that values are specified for the
required parameters. Press End to save and exit.


Commands: SAVE  VERIFYOFF
Line Commands: / - Select

Product customization library  : DB2TOOLS.PRODUCT.CUST.$SC63$.ABP220
                                                                  More:   -
   *Cancellation processing  . . . . . . . . BACKOUT                  List
   *Escalate thread cancel . . . . . . . . . NO  (YES, NO)
   *Check thread termination retry count . . 20
   *Check thread termination interval  . . . 3
   *Escape character . . . . . . . . . . . . +
   *Execution mode . . . . . . . . . . . . . EXECUTE                  List
   *On thread cancel failure
      CONTINUE                                                        List
   *Report type  . . . . . . . . . . . . . . SUMMARY (SUMMARY, DETAIL)
   *Thread quiesce time  . . . . . . . . . . 0

  / Create maintenance members

  _ Create product CLISTs

 F1=HELP      F2=SPLIT     F3=END       F4=EXPAND    F5=RFIND     F6=RCHANGE
```

Figure 5-31 DB2 Utilities Enhancement Tool product parameters panel 7

10. After saving the options for LPAR parameters, we press PF3 and return to the main workplace panel. We customized the DB2 parameters, but we skip the screen captures because only defaults were used.

11. We typed `G` in front of D1C1 to generate the customization jobs (Figure 5-32 on page 62).

```
TCUSTMZR         Customizer Workplace: DB2 UET                  Row 1 to 1 of 1
Command ===> _____________________________________________ Scroll ===> CSR

Use the Generate jobs line command to select the DB2 entries on which to
customize the product, and press Enter to generate the customization jobs.

Commands:  ASSOCIATE  DISCOVER  GENERATEALL  JOBLIST  CONFIGURATION

Product to Customize
  Product metadata library . : DBTLSP.SABPDENU           > LPAR  . . : SC63
  Product name . . . . . . . : DB2 Utilities Enhancemen  > Version . : 2.2.0
  Configuration ID: ABP1  >  Description: DB2 Utilities Enhancement Tool fo  >

Product and LPAR Parameter Status
  Line commands: E - Edit  B - Browse
  _ Product parameters . : Ready to Customize
  _ LPAR parameters  . . : Not Required

Associated DB2 Entries and Parameter Status
  Line commands: G - Generate jobs  E - Edit  B - Browse  C - Copy  R - Remove
  Cmd SSID Grpattch Lvl Mode User ID Date       Status             Message
  G   D1C1 D1CG     111 NFM  MKRES1  2015/05/22 Ready to Customize  Edited
```

Figure 5-32 DB2 Utilities Enhancement Tool customization jobs generation

12. The next panel (Figure 5-33) shows the customization jobs generated by TCz and the library that these jobs were stored. From this panel, it is possible to browse, edit, and even submit the jobs. We reviewed and ran all the customization jobs, except for the B0SMAAAA member because maintenance was not required in our environment.

```
TCUSTMZR                Finish Product Customization           Row 1 to 5 of 5
Command ===> _____________________________________________ Scroll ===> CSR

For a first-time customization, submit the jobs in the members in the order
in which they apply to the DB2 entries. Otherwise, submit only the necessary
jobs that were generated after changes were made. To submit jobs, browse
the members and issue the TSO SUBMIT command.

Line Commands: E - Edit  B - Browse

 _  Product customization library .: DB2TOOLS.PRODUCT.CUST.$SC63$.ABP220      >

 Cmd Member   SSID Grpattch Template Date       Description                    >
 -   -------- ---- ----     -------- ---------- ------------------------------
 _   A0OBAAAA D1C1 D1CG     ABPOBJCR 2015/05/15 Creates new DB2 UET DB2 objects
 _   A6BNAAAA D1C1 D1CG     ABPBNDCR 2015/05/15 Bind the DB2 UET plan
 _   A8OPAAAA D1C1 D1CG     ABPOPTCR 2015/05/15 Create the options module
 _   A9MOAAAA D1C1 D1CG     ABPMODCR 2015/05/15 Create BCAN, BGLB, PROC, PLCY m
 _   B0SMAAAA D1C1 D1CG     ABPSMPCR 2015/05/15 Create maintenance members
---------------------------- End of customized jobs ----------------------------
```

Figure 5-33 DB2 Utilities Enhancement Tool customization jobs

5.3.2 More Utilities Enhancement Tool configuration steps

Some additional Utilities Enhancement Tool configuration steps need to be done before using the product. Your z/OS system programmer is probably required for most of these:

1. Copy the DB2 Utilities Enhancement Tool started task PROC to your system PROCLIB to make the started task address space available to the user interfaces for the product.
2. Include SABPLOAD libraries as part of your authorized list.
3. Copy the DSNUTILF module into one of the APF-authorized libraries in the STEPLIB/JOBLIB concatenation. Optionally, you can copy the DSNUTILF module to a data set other than RUNLIB.LOAD, SDSNEXIT, or SDSNLOAD. If you choose this method, ensure that the data set is APF-authorized and is contained in the JOBLIB of your utility job or in the STEPLIB of the DSNUTILB job steps.
4. If you run DB2 Utilities Enhancement Tool in more than one LPAR, you need one started task for each LPAR.

6

Creating the autonomics infrastructure

Do you still create utility jobs manually to maintain several objects? Do you think that your maintenance jobs need to be run on a predefined frequency basis? IBM DB2 Automation Tool for z/OS V4.2 (DB2 Automation Tool) helps you with these challenges. Industry specialists know that data grows extremely fast. Databases, such as DB2 for z/OS, can be very useful to store all the data that requires data availability and security.

The current economic environment and budget reductions are also important industry challenges. Pressure is constant to analyze the IT budget to lower IT costs and invest only in technologies that provide a good return on investment. DB2 Automation Tool can help IT staff reduce IT think time to repetitive tasks and also to analyze the environment in order to run only what is needed when it is needed. This reduces the CPU utilization for maintenance jobs that do not really need to run in a defined maintenance window.

Combining Object, Utility, and Job Profiles, DB2 Automation Tool can reduce and facilitate manual routine tasks and focus on more complex job responsibilities that add more value to your company. It can automate common DB2 maintenance tasks, as well as generate JCL for more complex tasks on one or more objects.

Using Exception Profiles and DB2 Automation Tool, you can define in a Utility Profile when to run a utility against an object in an Object Profile. You select the conditions from a statistics list in the Exception Profile. For example, you can specify that the value in the PERCINDREF column in SYSTABLEPART for a table must be greater than a specified value in order to trigger a REORG, or you can set DB2 Automation Tool to run an image copy (IC) in case an object has changed. DB2 Automation Tool also has a feature that exploits DB2 10 autonomic statistics in order to analyze when a RUNSTATS needs to run. DB2 Automation Tools' statistics monitoring profiles enable you to define what objects to monitor and identify the criteria that define out-of-date statistics.

With the addition of the Management Console and the Autonomics Director, we now cannot only exercise “Passive” autonomics but we can start to move into “Active” autonomics. The Management Console makes monitoring the current symptoms and automating the suggested actions easy.

© Copyright IBM Corp. 2015. All rights reserved.

6.1 Profiles

This section describes the following profiles:

- Object Profiles
- Utility Profiles
- Exception Profiles
- Job Profiles

6.1.1 Object Profiles

Object Profiles allow you to create reusable lists of objects. You can group related objects into one profile, such as all objects for a particular application, objects with similar maintenance requirements, and so on. In an Object Profile, you can include objects on which you want to run utilities, as well as exclude objects that you want the utilities to ignore. Object Profiles are similar to DB2 TEMPLATEs. They allow table spaces and index spaces to be chosen for processing in much the same way.

You can select one or more of these objects to include in an Object Profile:

- Entire table spaces
- Selected partitions of a table space
- Entire indexes
- Selected partitions of an index
- Table spaces, index spaces, or both on a particular volume or set of volumes
- Wildcards that can automatically include table spaces or index spaces that match a specified mask

You can create Object Profiles using either the IBM Management Console for IMS and DB2 or by using the ISPF panels in the Automation Tool, but Object Profiles created using the IBM Management Console for IMS and DB2 can only be used by the Autonomics Director for RTS Collection.

Creating Object Profiles using the IBM Management Console for IMS and DB2

To create Object Profiles by using the IBM Management Console for IMS and DB2, please use the following steps:

1. After logging on the IBM Management Console for IMS and DB2, hover over the Autonomics tab, then select **DB2** (Figure 6-1 on page 67).

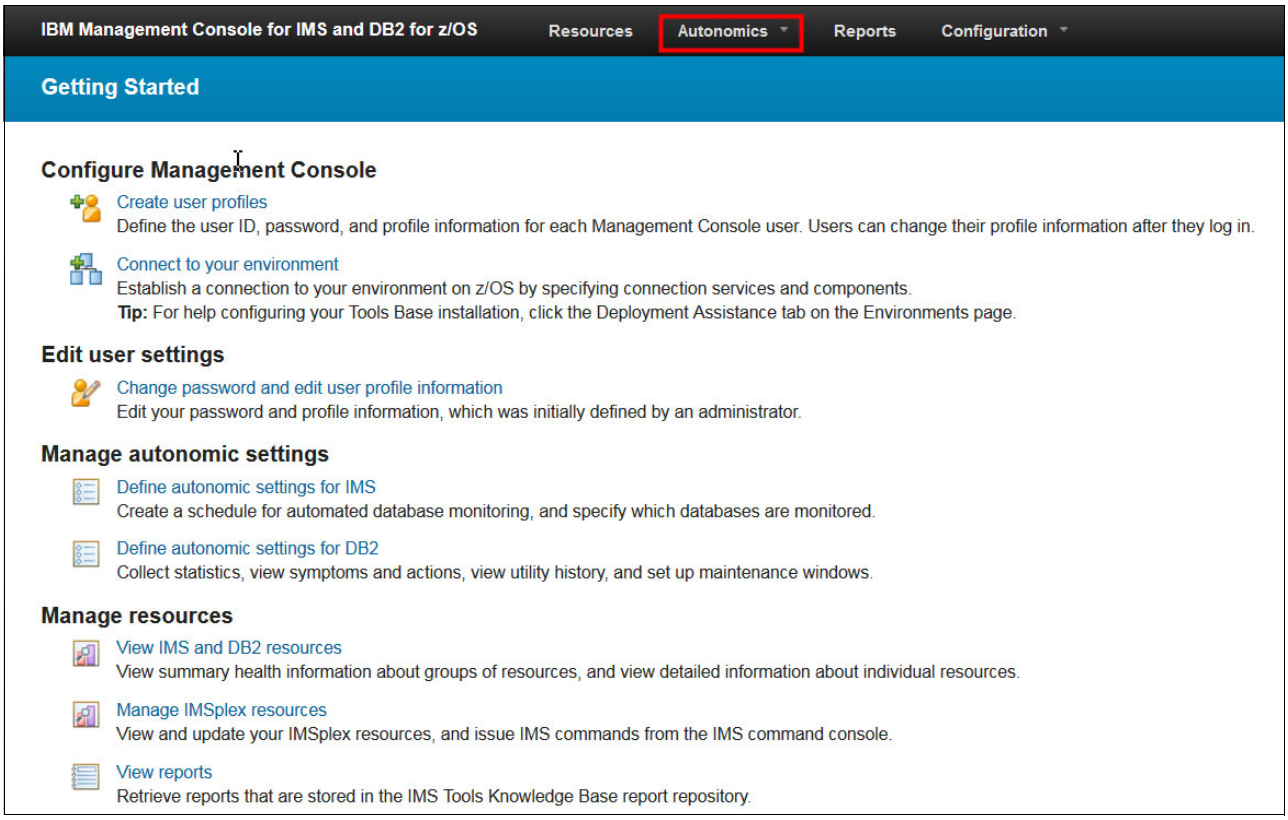


Figure 6-1 Autonomics → DB2

2. The next panels should default to the Object Profiles tab but if it does not, simply click the Object Profiles tab (Figure 6-2 on page 68).

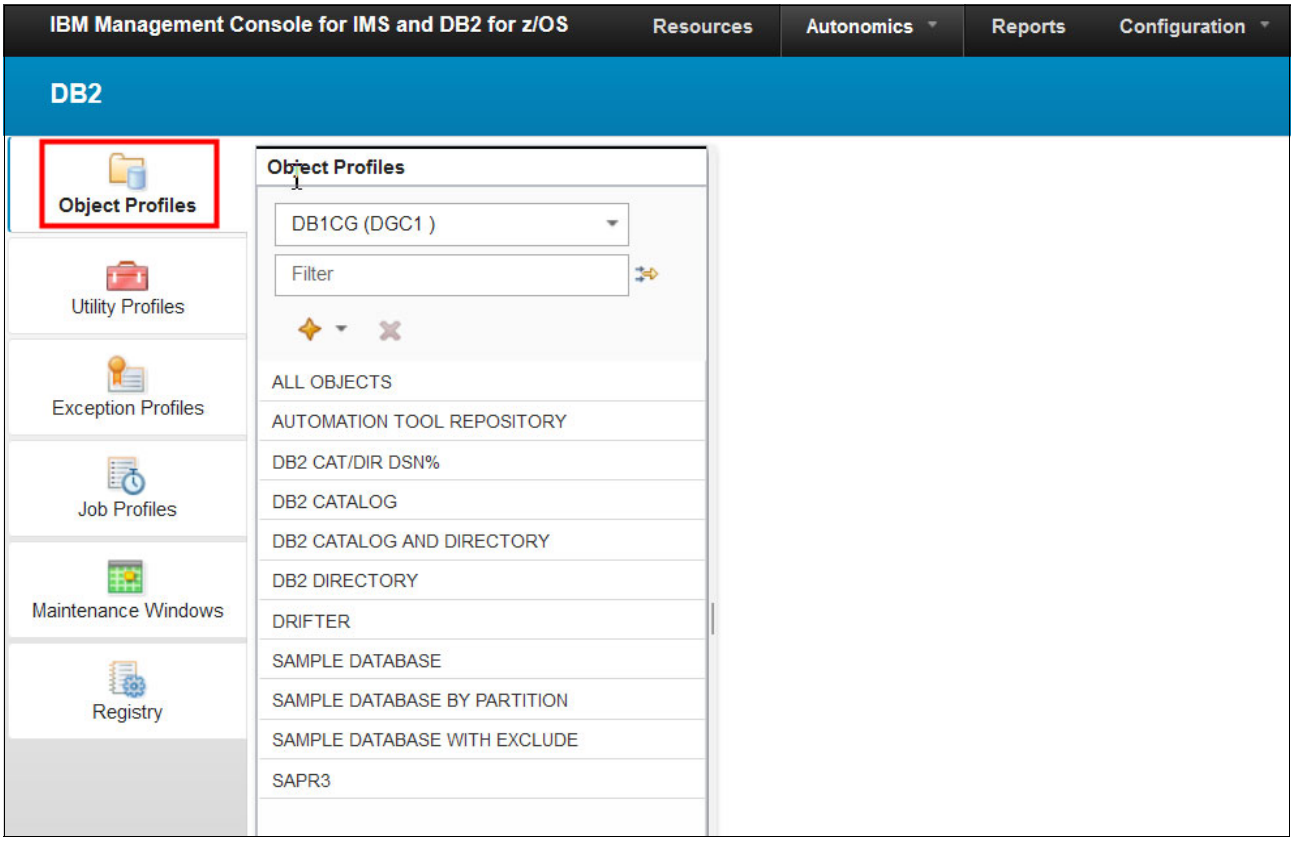


Figure 6-2 Object Profiles

3. This lists all the Object Profiles defined in your DB2 group. During the installation of the DB2 Automation Tool for z/OS, a set of default Object Profiles is created. To see only a subset of profiles, use a filter.
4. To create a new profile, click the **plus** symbol (Figure 6-3) below Filter, and select **New LISTDEF Object Profile**.

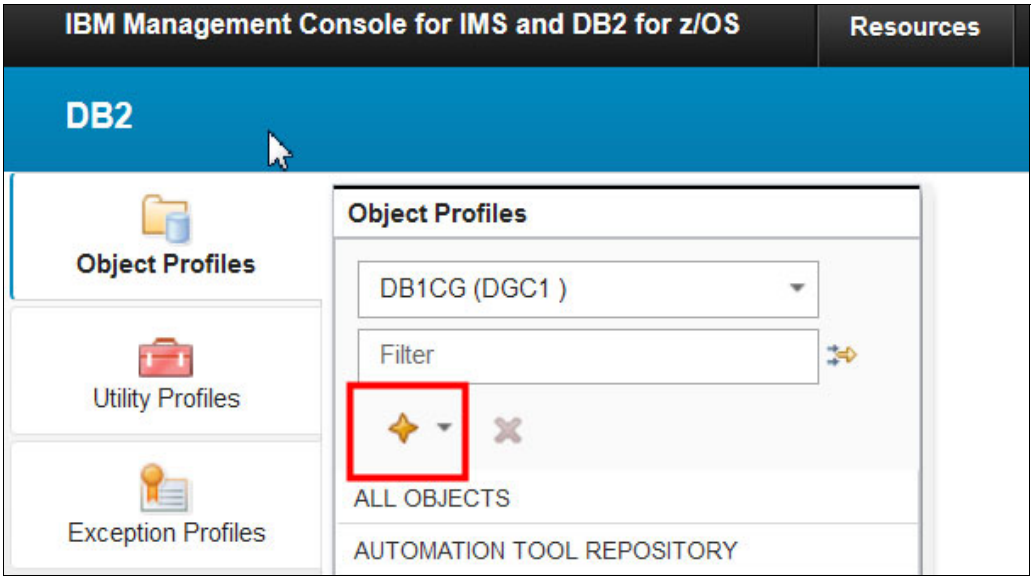


Figure 6-3 New LISTDEF Object Profile

5. Assign a name to the Object Profile and click **Create** (Figure 6-4).

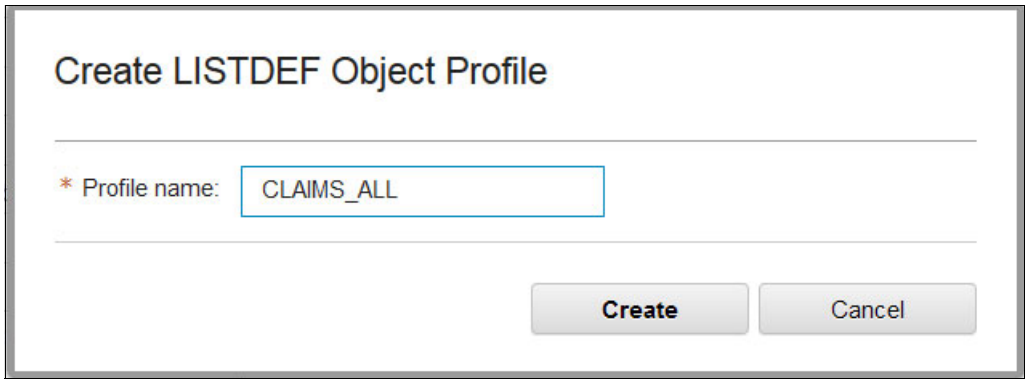


Figure 6-4 Object Profile name

6. Add a description and click **Save**. Click the **+** symbol to add an Object Profile Entry (Figure 6-5).

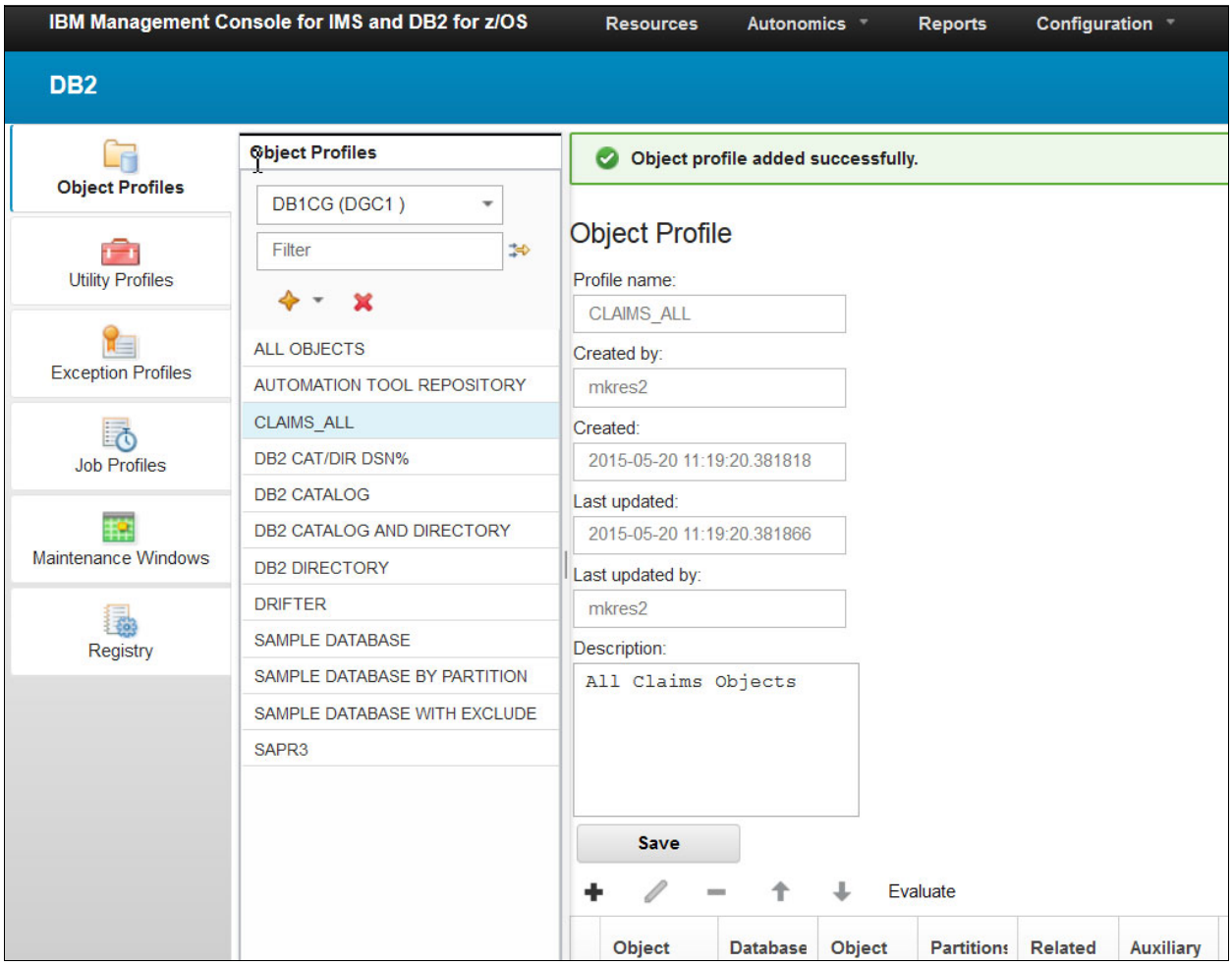


Figure 6-5 Create Object Profile

7. Complete the “Object Profile Entry” with the filter needed to define your Object Group (Figure 6-6). Using wildcards is allowed in the Name fields. Scroll to the bottom and click **Create**. You can add as many as needed to define your set of objects.

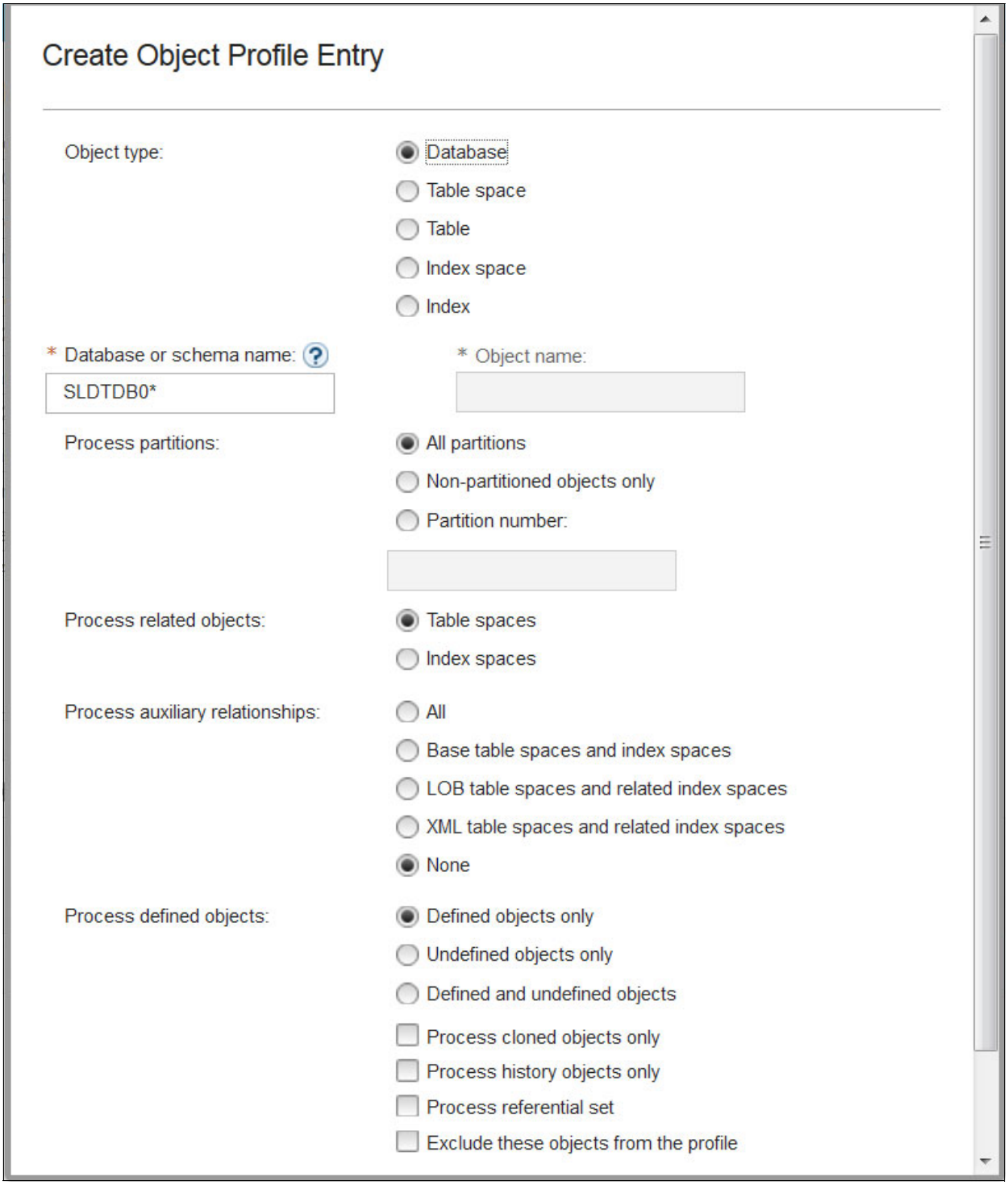


Figure 6-6 Create Object Profile Entry panel

8. Click the **Evaluate** button to execute the LISTDEF and display the contents in a pop-up window (Figure 6-7). Press Esc to close the window.

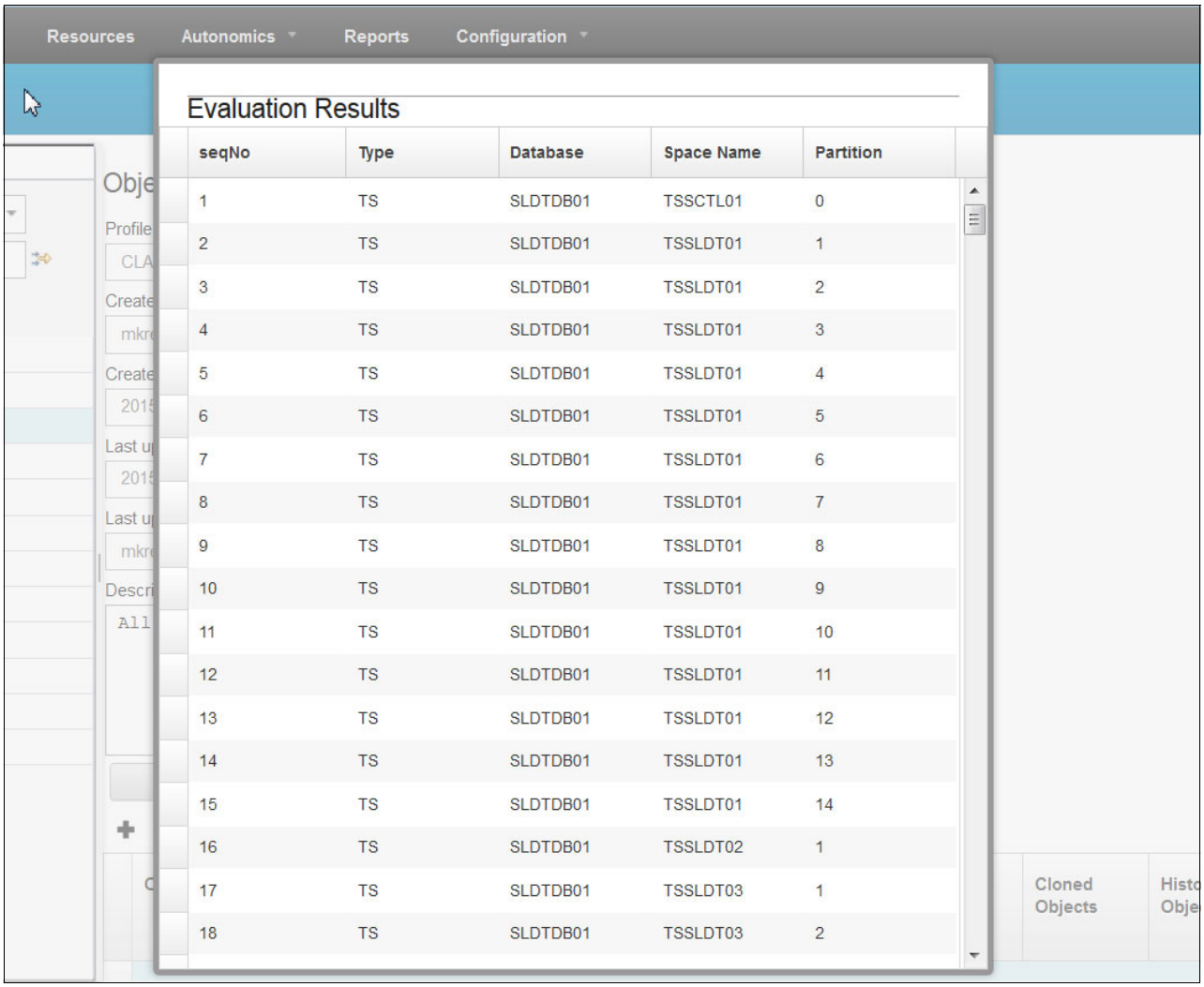


Figure 6-7 Display Evaluation Results

9. Use the **Edit** button to change an Object Profile to fit your needs.

Creating Object Profiles by using IBM DB2 Automation Tool for z/OS

In order to create an Object Profile from within IBM DB2 Automation Tool for z/OS, use the following steps:

1. Select option `1 – Object Profiles` from the main menu (Figure 6-8).

```
AUTOTOOL V4R2 --- IBM DB2 Automation Tool for z/OS    --- 2015/05/20  13:12:22
Option  ===> _____________________________________________________________________

------------------------------------------------------------------------------

       Options: 0 - Setup                    8 - Dataset Manager
                1 - Object Profiles          9 - Data Page Display
                2 - Utility Profiles        10 - Event Notifications
                3 - Exception Profiles      11 - Stand Alone Utilities
                4 - Job Profiles            12 - DB2 Admin Scheduler
                5 - Quick Build             13 - Autonomic Console
                6 - Execution Reports        X - Exit
                7 - DB2 Command Processor

------------------------------------------------------------------------------
 DB2 Subsystem ID: D1C1          (1-4 Character Subsystem ID or ? for list)
 Current SQLID:    ________     User: MKRES2  - Configuration ID: HAA
------------------------------------------------------------------------------
```

Figure 6-8 Select Option 1 – Object Profiles

2. Use Line Command `C` to initiate the creation process (Figure 6-9).

```
AUTOTOOL V4R2   -------- Objects Profile Display --------  2015/05/20  13:36:21
Option  ===> _____________________________________________ Scroll ===> CSR
------------------------------------------------------------------------------
Line Commands: C - Create  D - Delete  E - Export  I - Import
               Q - Quick   V - View    U - Update  J - Jobs    R - Rename
Profile Like  *_____________________________          DB2 Subsystem: D1C1
Creator Like  MKRES2__                          Row 1 of 1              >
------------------------------------------------------------------------------

Cmd  Name_____________________________  Creator   Updt
 C   Line Cmds: (Create,Import)
****************************** Bottom of Data ********************************
```

Figure 6-9 Create Object Profile

3. Enter the profile name, description, and update option (Figure 6-10 on page 73). The Update Option determines how the profile can be viewed or changed. Option U allows other users other than the creator of the profile to change the Object Profile. Option V allows other users to view the Object Profile and Option N prevents other users from viewing or updating the Object Profile.

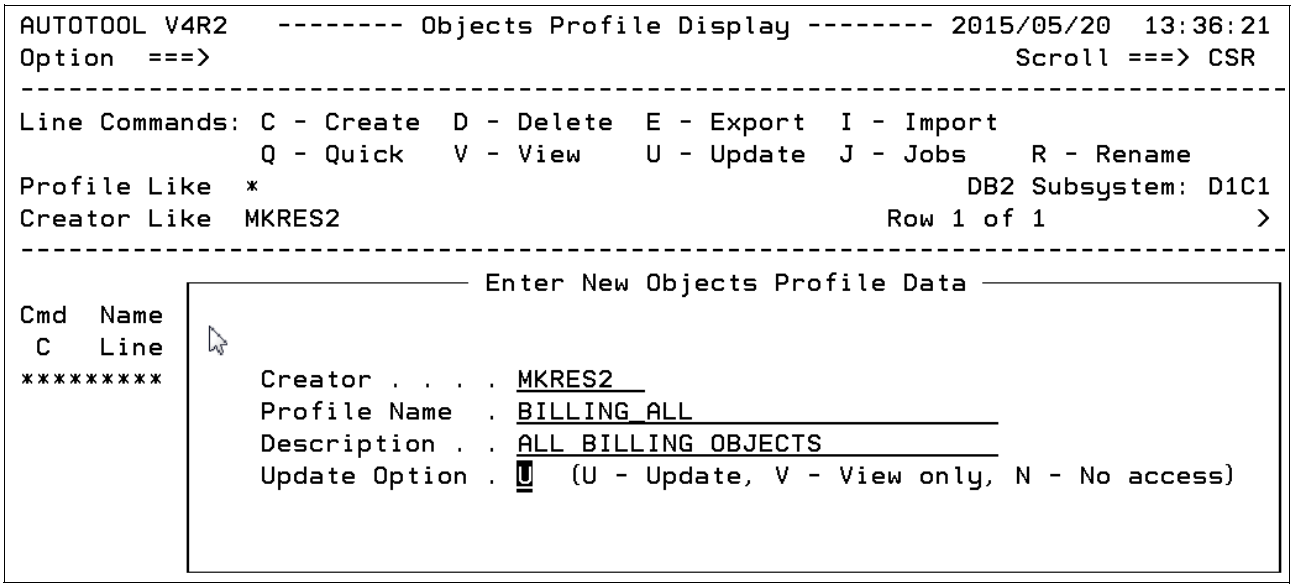


Figure 6-10 Enter Object Profile data

4. In the next panel (Figure 6-11), we selectively add objects to the profile. “Add Tablespaces” includes selected table spaces and their respective indexes. “Add Indexes” selectively adds indexes and “Add Volumes” selects table spaces and index spaces on a particular volume. You can select more than one option. Each option opens a selection tool.

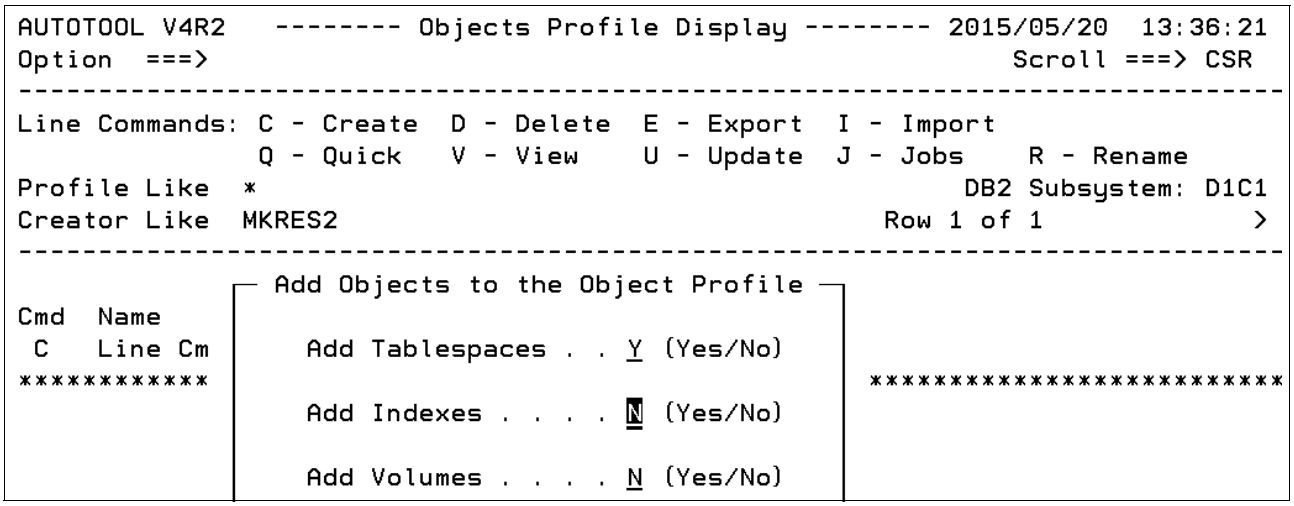


Figure 6-11 Enter Object Profile data

5. You can specify the “Wildcard” option to allow the inclusion of similarly named objects (Figure 6-12 on page 74). Without specifying this option, a list of objects are returned that can be individually selected as wanted. This is good for a limited number of objects but consider using wildcards for larger lists. Individual objects in the wildcard list can be excluded after the list is presented if needed.

```
AUTOTOOL V4R2   -------- Objects Profile Display -------- 2015/05/21  16:11:16
Option  ===>                                                Scroll ===> CSR
--------------------------------------------------------------------------------
Line Commands: C - Create  D - Delete  E - Export  I - Import
               Q - Quick   V - View    U - Update  J - Jobs    R - Rename
Profile Like  *                                        DB2 Subsystem: D1C1
Creator Like  *                                    Row 1 of 13           >
--------------------------------------------------------------------------------
                        Enter Tablespaces Like to Display

 Database Like. . SLDTDB*    Wildcard  Y (Yes/No)
 Tablespace Like. *          Exclude   I (E - Exclude, I - Include)
 Creator Like . . *          >

 Process Dependent Indexes . . . . . . . . . . N (Yes/No)
 Process Referentially Dependent Tablespaces . N (Y - Yes, N - No,
                                                  B - Build time Expansion,
                                                  R - Run time Expansion)
 Process Cloned Tables . . . . . . . . . . . . N (Yes/No)
 Advanced SQL . . . . N (Yes/No)    Update SQL N (Yes/No)

******************************* Bottom of Data ********************************
```

Figure 6-12 Wildcard objects

6. Based on your site preferences, standards, or leading practices you can choose to process individual partitions or objects as a whole (Figure 6-13). Although there can be performance improvements in certain utilities, careful consideration needs to be given to how your installation handles recoverability and disaster recovery.

```
                    Choose Partition Method

   Utilities can run against each partition or it can
   run against all partitions.  When HAA explodes wild
   card table and index spaces, which method would you
   like partitioned spaces exploded?

            Explode A  (A - All, P - Partitioned)
```

Figure 6-13 Partition method

7. The list that is returned can be “Exploded” to display the materialized list of objects (Figure 6-14). Once the list is materialized, objects can be excluded by selecting them until the wanted objects are in the profile.

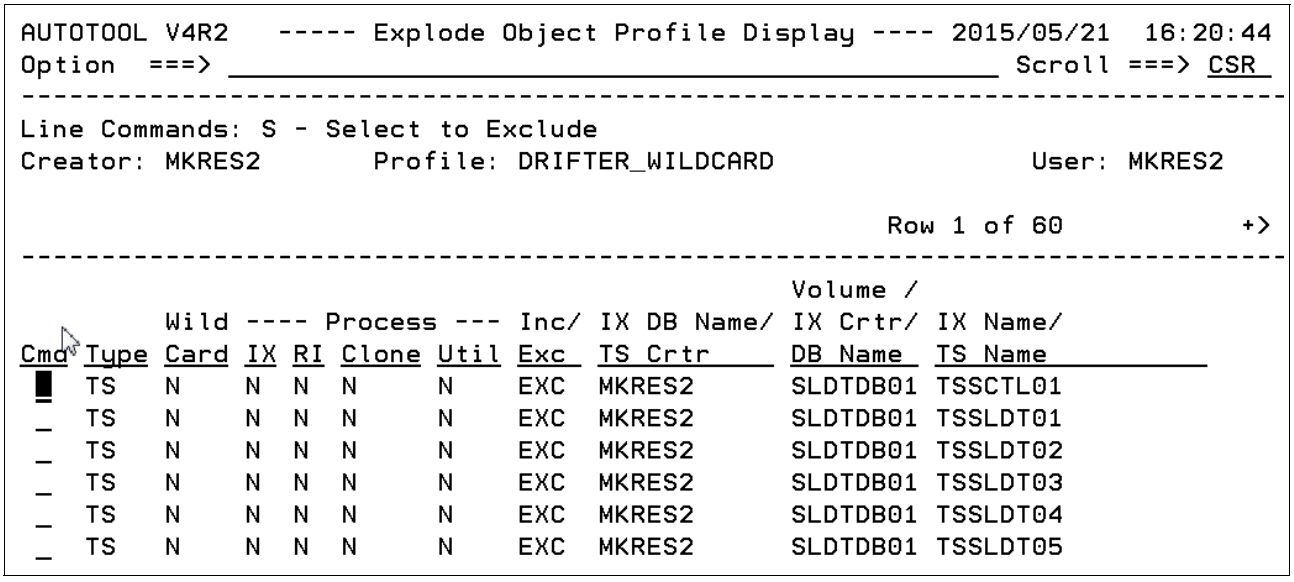


Figure 6-14 Select objects to exclude them

8. This process can be repeated for objects with different names until the profile contains all the objects required for this profile.

Once an Object Profile has been created, it can be updated at any time to add or remove more objects.

6.1.2 Utility Profiles

A *Utility Profile* is a collection of one or more utilities and their respective run time options. Using a similar technique to create Object Profiles, we can now create a Utility Profile to address any particular maintenance requirement.

The following list shows the utilities and functions that are available (Figure 6-15).

```
AUTOTOOL V4R2 --------- Utility Profile Options --------- 2015/05/21  17:59:45
Option   ===> _____________________________________________________________

  Creator: MKRES2     Profile: CLAIMS_COPY                      User: MKRES2
  Description  IMAGE COPY OF CLAIMS OBJECTS
  Share Option U (U - Update, V - View, N - No)


                                 Include      Update
                                 Utility      Utility
Data Page Verification Reporting N (Yes/No)   N (Yes/No)
Reallocation . . . . . . . . . . N (Yes/No)   N (Yes/No)
Recover  . . . . . . . . . . . . N (Yes/No)   N (Yes/No)
Image Copy . . . . . . . . . . . N (Yes/No)   N (Yes/No)
Recovery Expert Image Copy . . . N (Yes/No)   N (Yes/No)
Copy to Copy . . . . . . . . . . N (Yes/No)   N (Yes/No)
Runstats . . . . . . . . . . . . N (Yes/No)   N (Yes/No)
TS Reorg . . . . . . . . . . . . N (Yes/No)   N (Yes/No)
IX Reorg . . . . . . . . . . . . N (Yes/No)   N (Yes/No)
Quiesce  . . . . . . . . . . . . N (Yes/No)   N (Yes/No)
Modify . . . . . . . . . . . . . N (Yes/No)   N (Yes/No)
Repair . . . . . . . . . . . . . N (Yes/No)   N (Yes/No)
Check Data . . . . . . . . . . . N (Yes/No)   N (Yes/No)
Rebind . . . . . . . . . . . . . N (Yes/No)   N (Yes/No)
```

Figure 6-15 Select objects to exclude them

We can now select the utilities that we want to execute, and “Update Utility” allows us to specify the parameters that we want to specify for that given utility.

Once created, a Utility Profile can be updated at any time to include more utilities or to change the options for a given utility.

6.1.3 Exception Profiles

An *Exception Profile* contains the conditions under which users want to run utilities. When combined with Object Profiles and Utility Profiles, the Exception Profiles act as a filter against the objects specified in the Object Profile. The utilities specified in the Utility Profiles are only generated on the objects in the Object Profiles that satisfy the “conditions (exceptions)” specified in the Exception Profiles.

There are 184 available selection criteria that we can use to select candidate objects. Also, we can provide our own criteria through a user exit interface. This can be REXX CLIST, a program, or a stored procedure with pre, current, and post exit points. For a detailed description of the available selection criteria, see the following site:

- IBM DB2 Automation Tool for z/OS Users Guide

  http://www.ibm.com/support/docview.wss?uid=swg27020910

There are 10 supplied default profiles and viewing these give you a good idea about how to create and specify your own based on your site standards. Use the **V** (View) command to display the various options specified for a given Exception Profile (Figure 6-16).

```
AUTOTOOL V4R2   ------- Exceptions Profile Display ------ 2015/05/21  20:25:56
Option  ===> ____________________________________________________ Scroll ===> CSR
Line Commands: C - Create  D - Delete  E - Export  I - Import
               U - Update  V - View    J - Jobs    R - Rename
-------------------------------------------------------------------------------
 Profile Like  *____________________________              DB2 Subsystem: D1C1
 Creator Like  *________                                Row 1 of 10           >
-------------------------------------------------------------------------------
 Display Statistics Reports and/or Perform Statistics Maintenance? N (Yes/No)
-------------------------------------------------------------------------------

Cmd  Name                          Creator   Updt
 _   BEST PRACTICE IMAGE COPY FULL  HAAUSER    V
 _   BEST PRACTICE REORG AVOIDANCE  HAAUSER    V
 _   BEST PRACTICE RUNSTATS         HAAUSER    V
 _   CLUSTERRATIO VALUE RANGE       HAAUSER    V
 _   COPY CONDITIONS                HAAUSER    V
 _   LARGE OBJECTS                  HAAUSER    V
 _   NEVER COPIED                   HAAUSER    V
 _   REORG CONDITIONS               HAAUSER    V
 _   SATURDAY ONLY                  HAAUSER    V
 _   SMALL OBJECTS                  HAAUSER    V
***************************** Bottom of Data ********************************
```

Figure 6-16 Default Exception Profiles

The types of “Triggers” or selection criteria include object metadata, calendar events, object status, and Real Time Statistics values.

6.1.4 Job Profiles

A *Job Profile* is the master profile. It is used to connect all the other profiles. It is associated with one or more Utility Profiles, Object Profiles, and Exception Profiles. The combined profiles, which are headed by the Job Profile, form the basis of a DB2 Automation Tool task. We can submit this task manually or schedule it by using the DB2 administration task scheduler or your site’s scheduling software.

When creating a Job Profile, you select the Object, Exception, and Utility profiles created earlier. Most Job Profiles can be executed in Batch (JCL) or Online mode. The Job Profile evaluates the Exception Profile against the objects in the Object Profile and when a condition is met, will generate JCL and Utility statements to perform the tasks specified in the Utility Profiles against the objects that met the condition. If no conditions are met, the Job Profile can generate an IEFBR14 and no tasks are performed.

There are a number of supplied default Job Profiles. Viewing the default profiles can give more insight into the various uses and possibilities.

To create a Job Profile, use the C - Create command on the command line (Figure 6-17).

```
AUTOTOOL V4R2   ---------- Jobs Profile Display --------- 2015/05/22  09:41:52
Option  ===> ____________________________________________________ Scroll ===> CSR
--------------------------------------------------------------------------------
Line Commands: B - Build  C - Create  D - Delete  E - Export
               I - Import R - Rename  U - Update  V - View
--------------------------------------------------------------------------------
 Profile Like  *_____________________________            DB2 Subsystem: D1C1
 Creator Like  *_______                               Row 1 of 19           >
--------------------------------------------------------------------------------

Cmd  Name                              Creator   Updt
 c   BEST PRACTICE FULL INCR COPY      HAAUSER    V
 _   BEST PRACTICE REORG AVOIDANCE     HAAUSER    V
 _   BEST PRACTICE RUNSTATS            HAAUSER    V
```

Figure 6-17 Creating a Job Profile

The update option determines how users other than the Job Profile Creator can access the profile. Update, View, or No access (Figure 6-18).

```
                    Enter New Jobs Profile Data

 Creator . . . . MKRES2__
 Profile Name  . CLAIMS_RUNSTATS_____________
 Description . . Weekly Runstats for Claims___
 Update Option . U  (U - Update, V - View only, N - No access)
```

Figure 6-18 Job Profile Data

Specify the type of autonomics. Active Autonomics creates an Action item that will be executed the next time the Autonomics Director is invoked for a given maintenance window and that maintenance window is active. This can be done by selecting the option when creating the maintenance window or by running the supplied batch program when the maintenance window is active.

Passive adds an Action from the output of the build, but the action does not run when a maintenance window opens. However, you can still review the actions, objects, and symptoms that generated the actions on the autonomic panels. This is a good way to start off to ensure that the recommended action is correct. Once satisfied, change the Passive to Active to allow this task to be executed the next time the associated maintenance window opens.

If setting either Passive or Active, a Maintenance window must be selected (Figure 6-19).

```
AUTOTOOL V4R2   ---------- Jobs Profile Display --------- 2015/06/04  15:02:35
Option  ===>                                                Scroll ===> CSR
-------------------------------------------------------------------------------
Line Commands: B - Build  C - Create  D - Delete  E - Export
               I - Import R - Rename  U - Update  V - View
-------------------------------------------------------------------------------
 ------------------ Generation Options for MKRES2.CLAIMS_RUNSTATS ------------------

  Option  ===> ____________________________________________________

                                                                  More:     +
  Autonomic Director Options:
     Autonomic Type . . . . . . . A (A - Active, P - Passive, N - None)
     Select Maintenance Window  . Y (Yes/No, C - Clear)
     Selected Maintenance Window
  Update Setup Override Options  . . . . . N (Yes/No)
  Update Template/Listdef/Option parms . . N (Yes/No)
  Update Job Group Break Down Options  . . N (Yes/No)
  Update Notifications . . . . . . . . . . N (Yes/No)
  Automatically Gen GDG Base . . . . . . . 000 (0-255 Limit)
  Load Balance jobs by . . . . . . . . . . N (T - Time, D - Dasd, N - None)
  Capture run times for Load Balancing . . N (Yes/No)
  Start spaces in Utility/Read Only  . . . N (N - No, U - Utility,
                                              R - Read only)
  Prefix Utility ID with jobname . . . . . N (J - Job, S - Step, B - Both,
                                              N - No)
  Set JCL member equal to jobname  . . . . N (Yes/No)
  Generate Job when Errors encountered . . Y (Y - Yes, N - No, W - Warnings)
  Preview Exception Report . . . . . . . . N (Y - Yes, N - No)
```

Figure 6-19 Job Profile options

Complete the remaining options per your installation standards and procedures (Figure 6-20). Press Enter to select a maintenance window.

```
AUTOTOOL V4R2   ------ Autonomic Maintenance Window ----- 2015/06/04  15:17:24
Option  ===> ____________________________________________________ Scroll ===> CSR
-------------------------------------------------------------------------------
Line Commands: S - Select   V - View
-------------------------------------------------------------------------------
 Window Like  *____________________________            DB2 Subsystem: D1C1
                                                        Row 1 of 1         >
-------------------------------------------------------------------------------
     Window                      Window
Cmd  Name                        Description                 Source        SSID
 S   EACHDAY                                                 AUTODIRECTOR
***************************** Bottom of Data ********************************
```

Figure 6-20 Selecting a maintenance window

Proceed to add an Object, Exception, and Utility Profile as required. Once this process is complete, the Job Profile is created and ready to be executed. See Chapter 8, “Running it all together: Automation Tool, UET, Management Console, Autonomics Director ” on page 101 to create and execute a build of this Job Profile.

7

Configuring maintenance windows with Management Console and Autonomics Director

This chapter covers the creation and definition of DB2 utility maintenance windows using the Management Console. Maintenance windows are important because they are used by the Autonomics Director to execute the highest priority DB2 utility work that we have defined. Defining a proper maintenance window allows us to avoid contention with other workloads on the system and gives us the best chance to complete the needed utility work. Maintenance windows are required for an active autonomics build.

© Copyright IBM Corp. 2015. All rights reserved.

7.1 Introduction

This chapter covers two different ways of creating maintenance windows: Using the DB2 Automation Tool and the Autonomic Director function of Management Console. These two tools allow you to create a maintenance window. The main difference is the interface that the two functions use, DB2 Automation Tool, an Interactive System Productivity Facility (ISPF)-based tool, which provides you with standard ISPF-based windows to navigate through the various menus. DB2 Management Console for IMS and DB2 is a GUI-based web browser interface that provides a different way to accomplish the same thing for those who might not be familiar with ISPF or prefer a GUI interface.

In this chapter, we cover some of the basic functions related to maintenance windows and show you how to navigate the various windows related to those functions. For those unfamiliar with the GUI interface, we provide enough screen captures to help you learn how to navigate through this unfamiliar territory.

There are three distinct steps involved in working with maintenance windows: Creating the maintenance window (giving it a name), defining the window (when does it start and end, how many jobs can run during the window, and so on), and scheduling the window (which says this is available to be used by any tasks that need to be run during active autonomics). We cover each of these using the DB2 Automation Tool or Autonomics Director function of Management Console.

There are different ways to get utility batch jobs scheduled in a maintenance window and we cover using the DB2 Admin Task scheduler and using an existing batch scheduler together with the DB2 Automation Tool.

7.2 Management Console for DB2

The first thing that we cover is the use of Management Console to create a maintenance window. The Management Console is a web browser GUI interface, which is common in the IBM z/OS product suite. Learning how to navigate through the product is fairly intuitive, as described below:

1. We start at the main panel (Figure 7-1) and work our way to create a maintenance window.

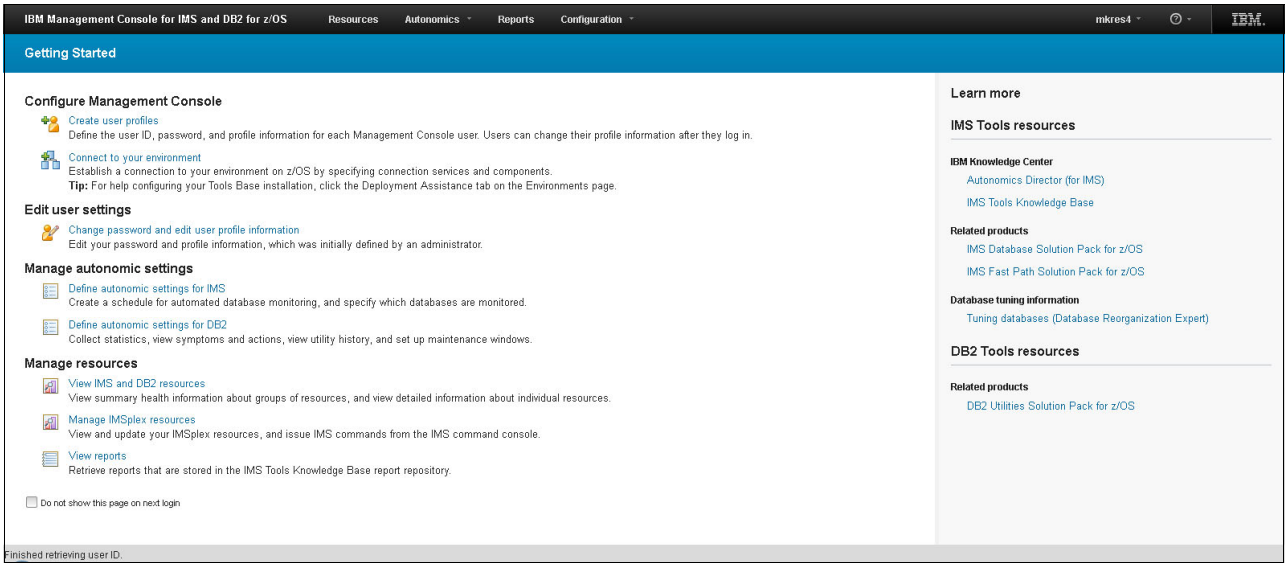


Figure 7-1 Management Console GUI window

2. At the next window, we click the **Autonomics** tab and we get a choice of IMS or DB2 (Figure 7-2). Click the **DB2** tab.

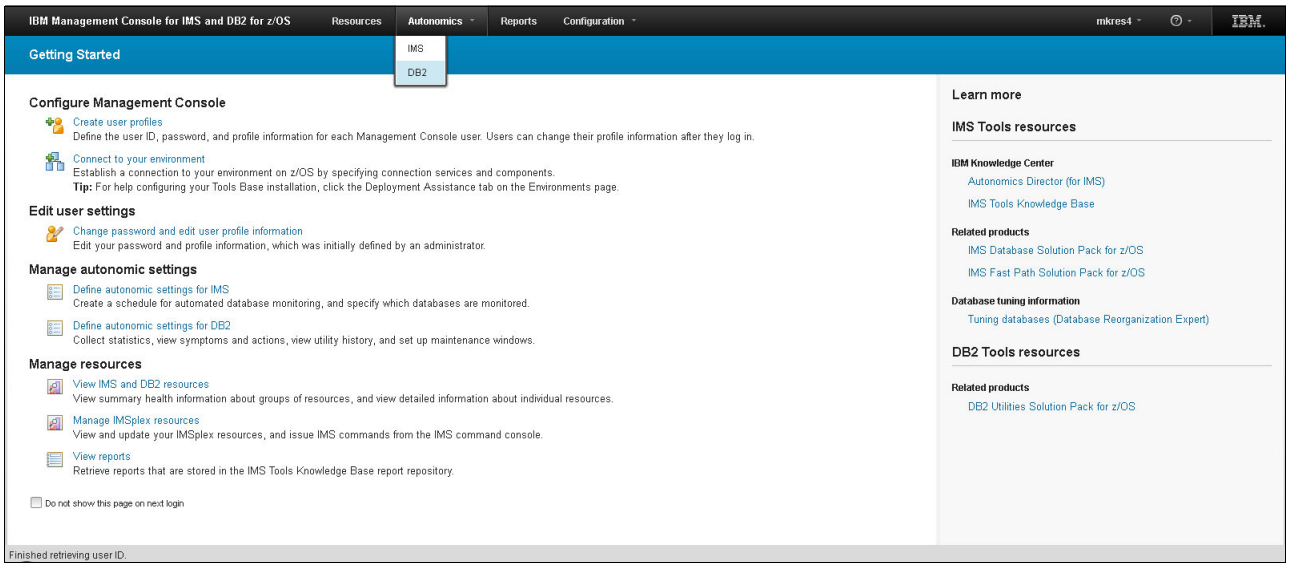


Figure 7-2 Management Console GUI with DB2 drop-down box

3. The next window is the main DB2 Management Console panel (Figure 7-3) and there are a number of options available to us on the left side. We select the **Maintenance Windows** tab, which opens a list of maintenance windows. Because we have not defined any maintenance windows, none are listed. The star icon is what is used to create a new maintenance window.

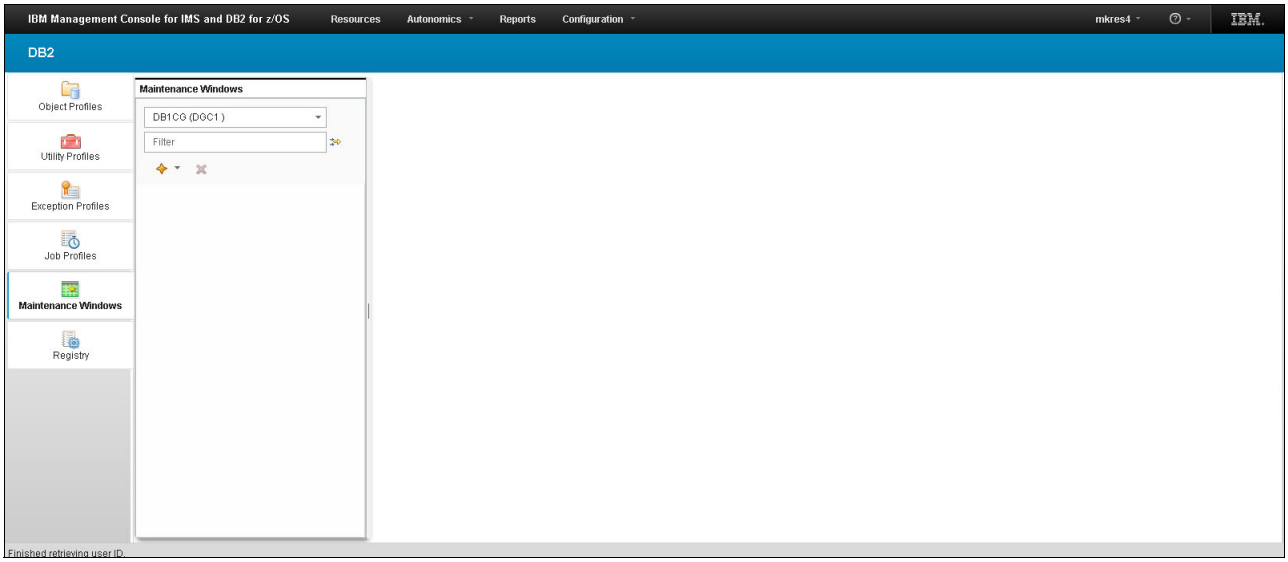


Figure 7-3 Management Console GUI maintenance windows

7.2.1 Defining a new maintenance window using Management Console

Perform the following steps to define a new maintenance window using Management Console:

1. The **star** icon is clicked and we see a box for “New Maintenance Window” (Figure 7-4).

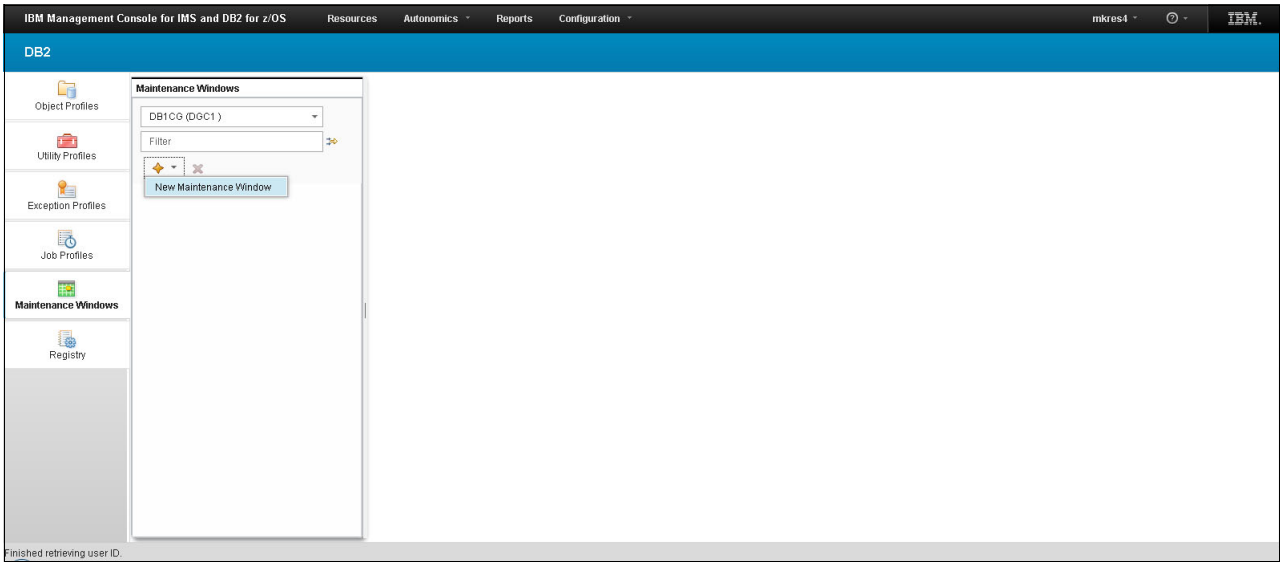


Figure 7-4 Management Console GUI: Add new maintenance window

2. The creation of a new maintenance window requires some specific information starting with a name and description (Figure 7-5).

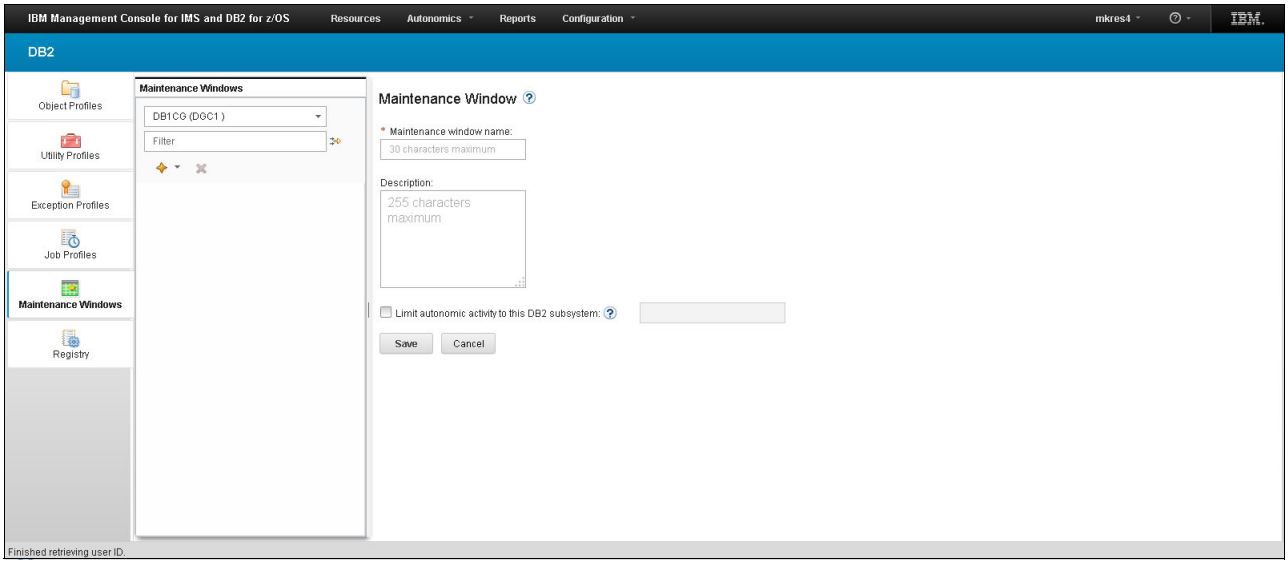


Figure 7-5 Maintenance window name and description

3. The window defined in Figure 7-6 is an example of a common maintenance window that exists for weekend batch utility jobs. We entered a name, description, and selected the **Limit autonomic activity to this DB2 subsystem** check box to limit this to one DB2 subsystem.

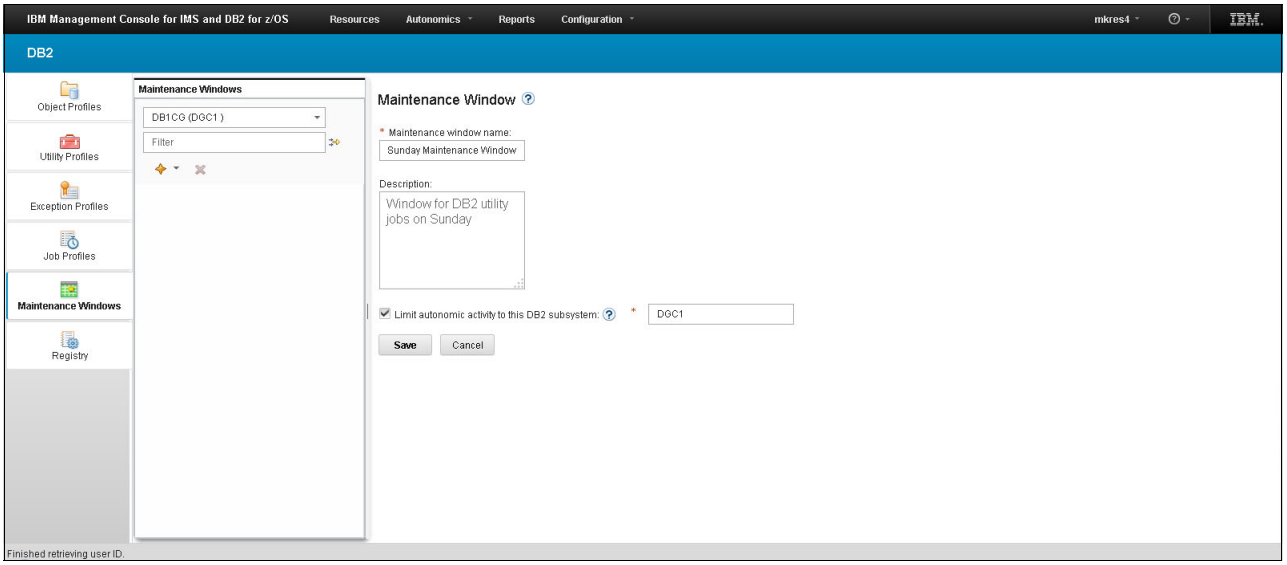


Figure 7-6 Management Console GUI panel: Maintenance window definition

7.2.2 Adding a maintenance window to the schedule

Perform the following steps to add a maintenance window to the schedule:

1. Clicking **Save** gets you to this panel (Figure 7-7). We named our new window the “Sunday Maintenance Window” but have not supplied any details about when the window starts, and so on.

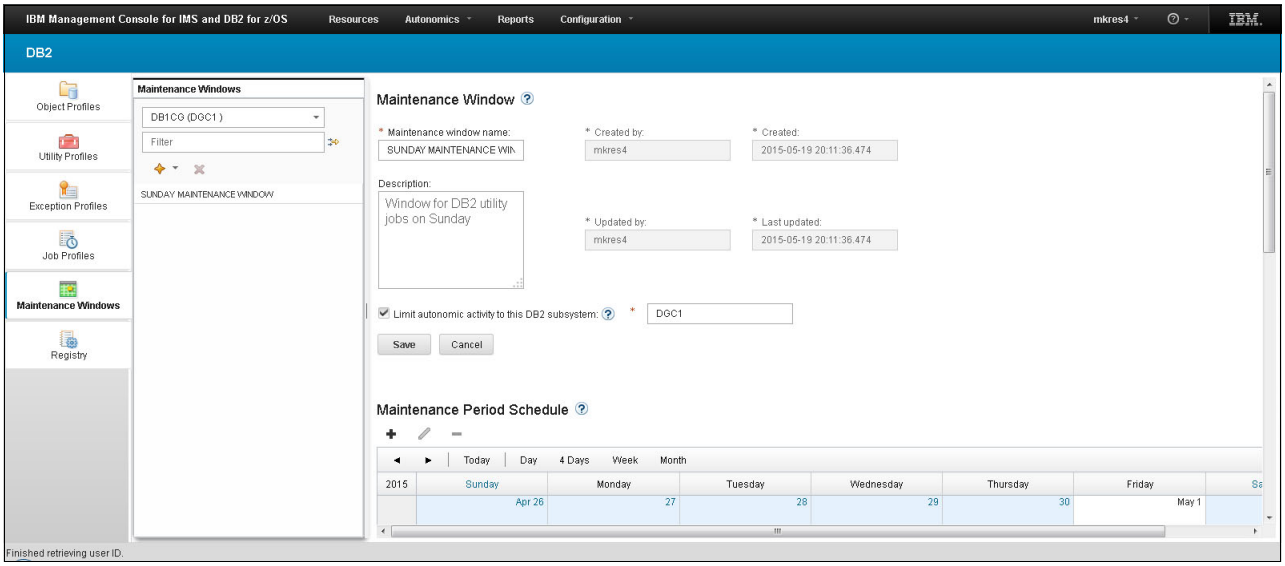


Figure 7-7 Management Console maintenance window saved

2. Details are supplied by expanding the options by clicking the **+** sign icon underneath “Add Maintenance Period Schedule” (Figure 7-8).

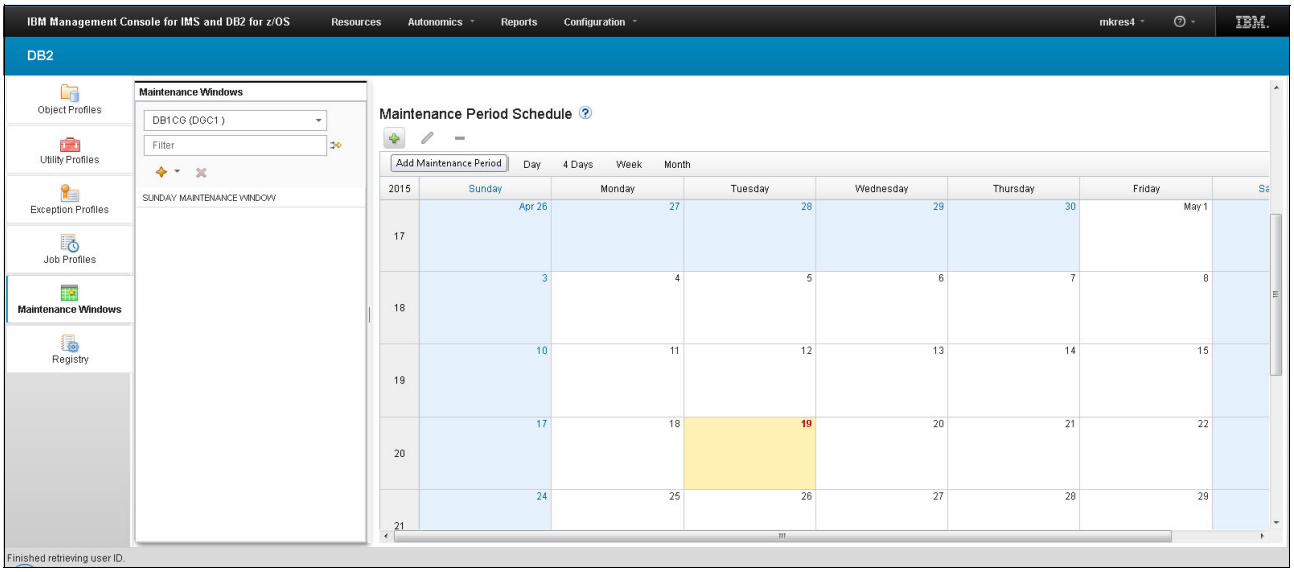


Figure 7-8 Management Console: Add details

3. Clicking **+** opens a box with the specifics of the maintenance schedule (Figure 7-9).

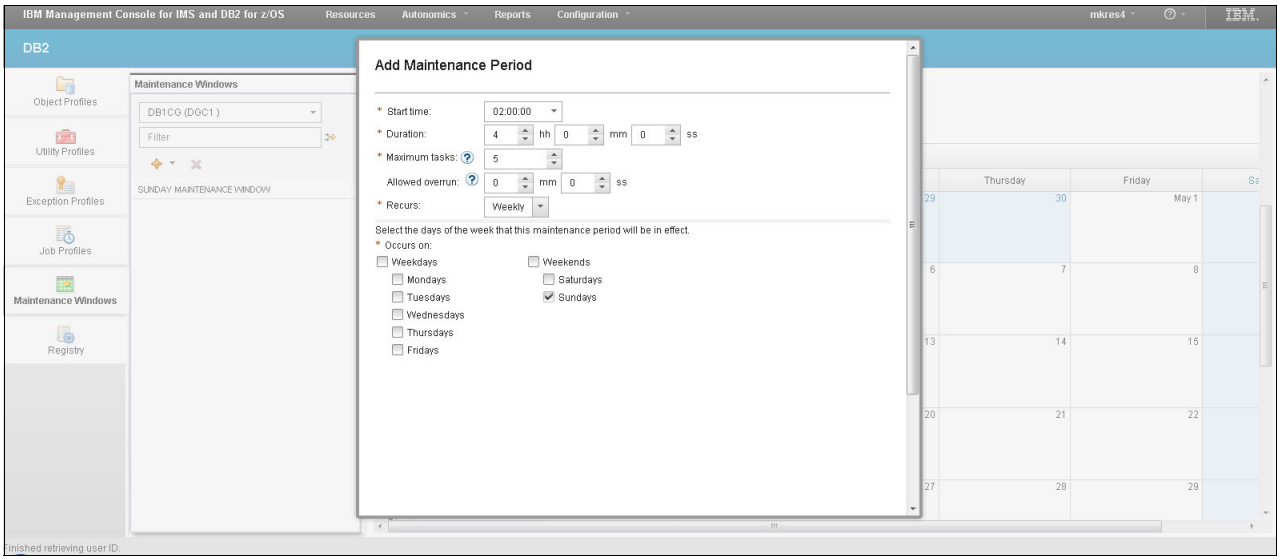


Figure 7-9 Management Console: Completing maintenance windows details

7.2.3 Scheduling a maintenance window

Perform the following steps to schedule a maintenance window:

1. Scrolling down brings you to the option window “Start Autonomics Director” (Figure 7-10). This box has significant meaning because it means schedule the maintenance window, which is required when you go from passive to active autonomics.

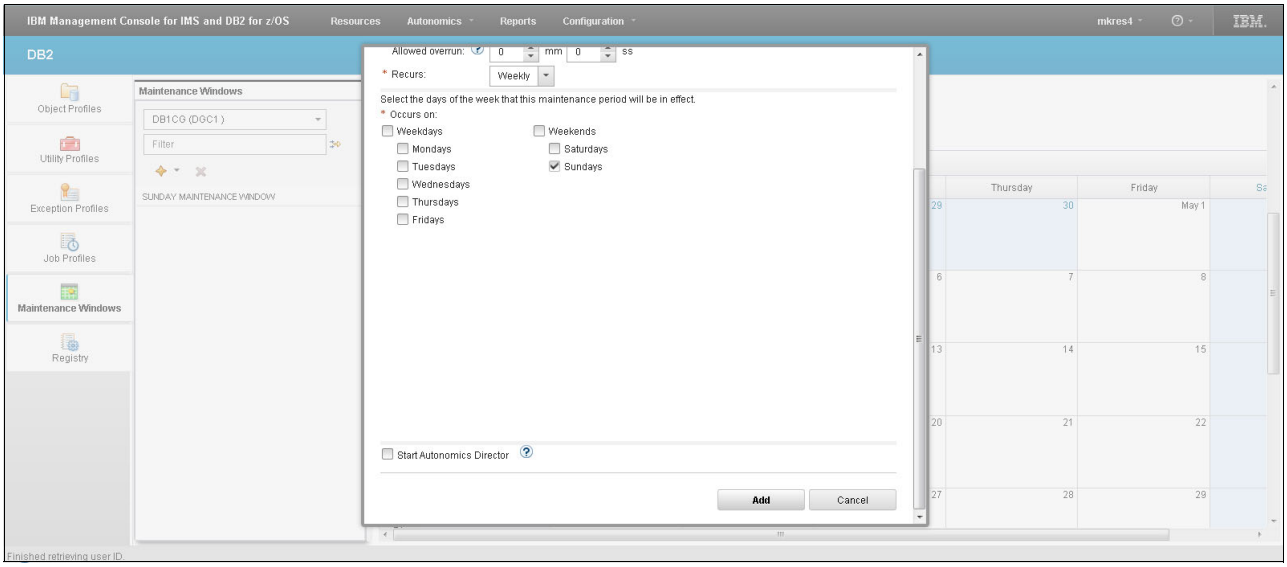


Figure 7-10 Adding maintenance windows details

2. This window shows the maintenance schedule on the calendar (Figure 7-11).

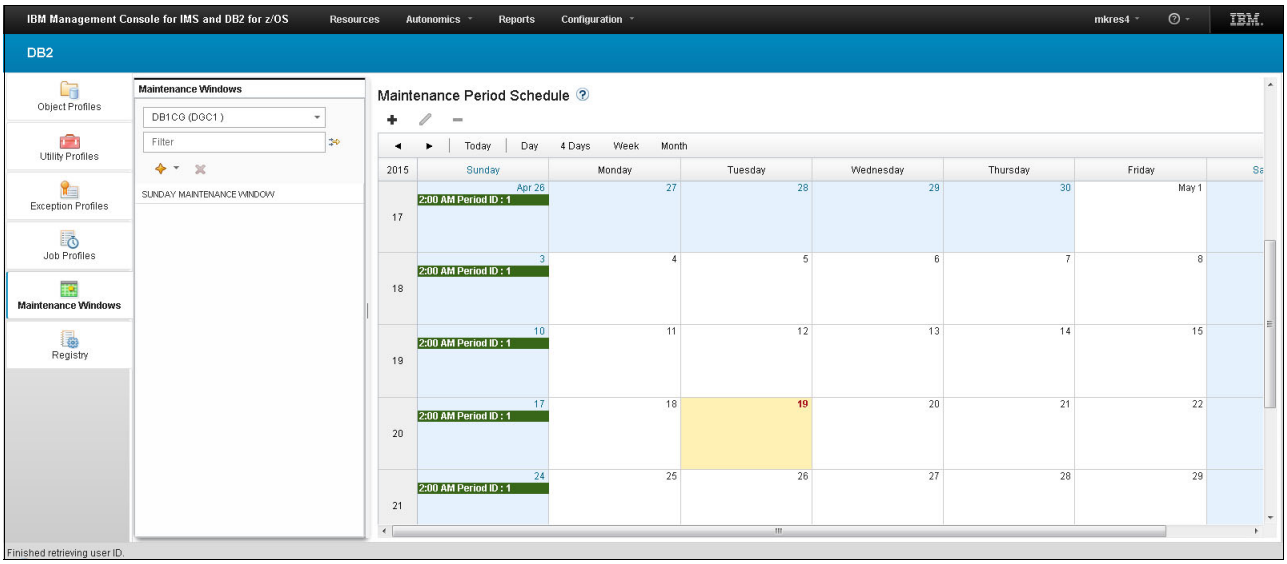


Figure 7-11 Management Console Maintenance: Window calendar

3. Scroll down a little and you can see some of the details from when the maintenance period was added to the calendar (Figure 7-12).

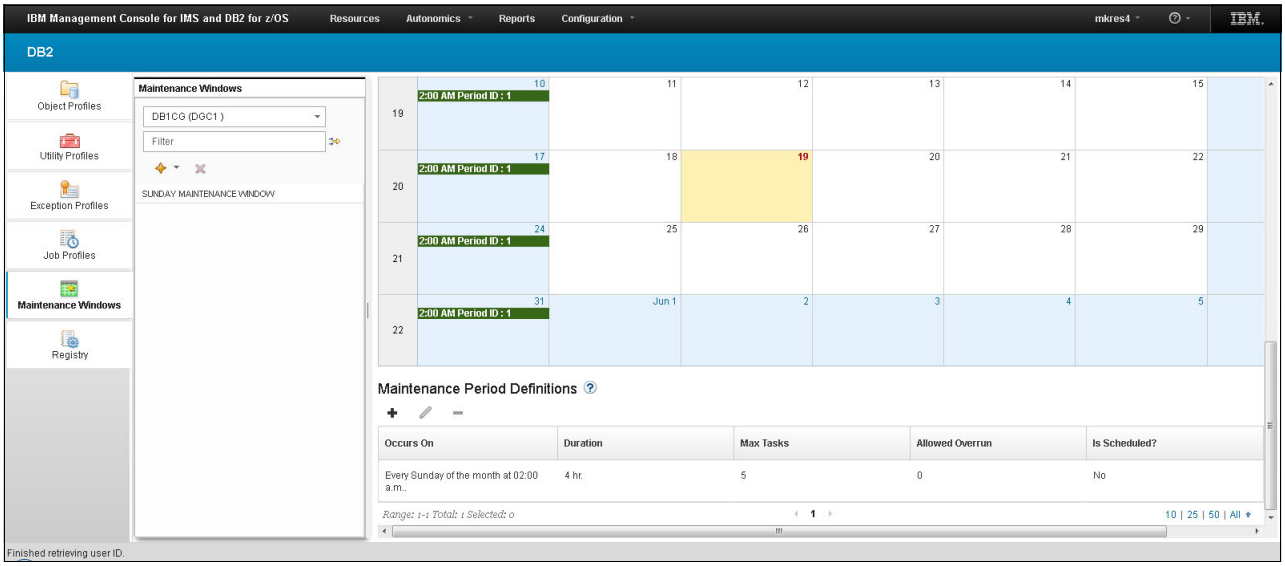


Figure 7-12 Management Console Windows: Details view

7.3 Autonomics Console

The Autonomics Console function is part of the DB2 Automation Tool, which is an ISPF-based function option that allows you to perform all of the actions that we just went through with Management Console. We cover the same functions with Autonomics Console so that you have a choice of which method to use when working with maintenance windows (Figure 7-13).

```
AUTOTOOL V4R2 ------------ Autonomic Console ------------ 2015/05/19
Option  ===> 2
-------------------------------------------------------------------------

       Options: 1 - Autonomic Object Prioritization
                2 - Autonomic Maintenance Windows
                3 - Autonomic Execution History
                4 - Autonomic Actions
                5 - Autonomic Statistics Monitor Profiles
```

Figure 7-13 Autonomic Console: Main menu

Selecting Option 2 brings you to the following panel (Figure 7-14).

```
AUTOTOOL V4R2   ------ Autonomic Maintenance Window ----- 2015/05/19  10:08:06
Option  ===> ____________________________________________________ Scroll ===> PAGE
-------------------------------------------------------------------------------
Line Commands: C - Create AutoDirector Window  D - Delete  R - Repeat
               T - Create AutoStats Window  U - Update  V - View
               S - Schedule Window  X - Unschedule Window  W - Workload
-------------------------------------------------------------------------------
 Window Like  *________________________________          DB2 Subsystem: D1C1
 Source       D (D - AutoDirector, S - AutoStats)
                                                        No rows to display
-------------------------------------------------------------------------------
     Window                      Window
Cmd  Name                        Description                Source        SSID

          HAAM714I - Specify selection criteria and press <Enter>.
```

Figure 7-14 Autonomic Maintenance Window panel

7.3.1 Creating a maintenance window using Autonomics Console

Perform the following steps to create a maintenance window using Autonomics Console:

1. Pressing Enter automatically populates your panel with the C for Create AutoDirector Window if this is your first time in this panel (Figure 7-15).

```
AUTOTOOL V4R2   ------ Autonomic Maintenance Window ----- 2015/05/19  10:10:22
Option  ===> ____________________________________________________ Scroll ===> PAGE
-------------------------------------------------------------------------------
Line Commands: C - Create AutoDirector Window
               T - Create AutoStats Window

-------------------------------------------------------------------------------
 Window Like  *_______________________________            DB2 Subsystem: D1C1
 Source       D (D - AutoDirector, S - AutoStats)
                                                          Row 1 of 1          >
-------------------------------------------------------------------------------
     Window                      Window
Cmd  Name_____________________ Description_______________ Source_______ SSID
 C
***************************** Bottom of Data ********************************

 HAA590I - No time windows were found. Press enter to create a time window.
```

Figure 7-15 Autonomics Maintenance Window creation

2. This panel (Figure 7-16) allows the user to create their maintenance windows. Here is the process for defining your maintenance window.

```
AUTOTOOL V4R2   ------- Create Maintenance Window  ------ 2015/05/19  10:14:59
Option  ===> ____________________________________________________ Scroll ===> PAGE
-------------------------------------------------------------------------------
Line Commands:  V - View  C - Create  D - Delete  R - Repeat  U - Update
                S - Schedule Timeperiod  X - Unschedule Timeperiod
-------------------------------------------------------------------------------
Window Name  . . . . _______________________________
Window Description   _______________________________
SSID . . . . . . . . D1C1       (Blank for any datasharing member)
                                                          No rows to display
-------------------------------------------------------------------------------
    Day of       Day of Time of
Cmd Week   Month Month  Day    Duration   Overrun    MaxTasks   Scheduled
```

Figure 7-16 Maintenance Window definition panel

3. On this panel (Figure 7-17), you must tab down to the Window Name line and enter your name for this window. After that, tab down to the Window Description line and enter a further description of your maintenance window. Press Enter and this panel is automatically populated with the **Create** command at the bottom.

```
AUTOTOOL V4R2   ------- Create Maintenance Window  ------ 2015/05/19  10:14:59
Option  ===> _____________________________________________________ Scroll ===> PAGE
--------------------------------------------------------------------------------
Line Commands:  V - View  C - Create  D - Delete  R - Repeat  U - Update
                S - Schedule Timeperiod  X - Unschedule Timeperiod
--------------------------------------------------------------------------------
Window Name  . . . . Sunday Maintenance Window
Window Description   Window for DB2 work on sundays
SSID . . . . . . . . D1C1       (Blank for any datasharing member)
                                                        No rows to display
--------------------------------------------------------------------------------
    Day of       Day of Time of
Cmd Week   Month Month  Day     Duration   Overrun    MaxTasks   Scheduled
```

Figure 7-17 Naming maintenance window panel

4. The maintenance window is populated after pressing Enter. Select Enter again and another window is displayed. Selecting Enter with `C` on the command line brings you to the window to supply the Maintenance Window details (Figure 7-18).

```
AUTOTOOL V4R2   ------- Create Maintenance Window  ------ 2015/05/19  10:19:49
Option  ===> _____________________________________________________ Scroll ===> PAGE
--------------------------------------------------------------------------------
Line Commands:  V - View  C - Create  D - Delete  R - Repeat  U - Update
                S - Schedule Timeperiod  X - Unschedule Timeperiod
--------------------------------------------------------------------------------
Window Name  . . . . SUNDAY MAINTENANCE WINDOW
Window Description   WINDOW FOR DB2 WORK ON SUNDAYS
SSID . . . . . . . . D1C1       (Blank for any datasharing member)
                                                        Row 1 of 1
--------------------------------------------------------------------------------
    Day of       Day of Time of
Cmd Week   Month Month  Day     Duration   Overrun    MaxTasks   Scheduled
 C  Line Cmds: (Create)
****************************** Bottom of Data ***********************************

 HAAM435I - No time periods were found. Press enter to create a time period.
```

Figure 7-18 Creating line command on Maintenance Window panel

7.3.2 Creating a time period using Autonomics Console

Perform the following to create a time period using Autonomics Console:

1. This window (Figure 7-19) is where you supply the details of the maintenance window time period. The method for supplying the day of the week is unusual with the day being supplied by a number `0 - 6`, as shown in Figure 7-19.

```
AUTOTOOL V4R2   ------- Create Maintenance Window  ------ 2015/05/19  10:19:49
Option  ===>                                                 Scroll ===> PAGE
-------------------------------------------------------------------------------
Line Commands:  V - View  C - Create  D - Delete  R - Repeat  U - Update
                S - Schedule Timeperiod  X - Unschedule Timeperiod
-------------------------------------------------------------------------------
Window Name  . . . . SUNDAY MAINTENANCE WINDOW
Window Description   WINDOW FOR DB2 WORK ON SUNDAYS
  ┌──────────────────────────── Create Timeperiod ────────────────────────────┐
  │                                                                           │
  │ Window Name  . . . . SUNDAY MAINTENANCE WINDOW                            │
  │ Window Description   WINDOW FOR DB2 WORK ON SUNDAYS                       │
  │ SSID . . . . . . . . D1C1                                                 │
  │ ------------------------------------------------------------------------- │
  │ Window Start:                                                             │
  │   Day of Week . . . . . ___   (0-6, Sun-Sat, *)                           │
  │   Month . . . . . . . . ___   (1-12, Jan-Dec, *)                          │
  │   Day of Month    . . . __    (1-31, *)                                   │
  │   Time of Day . . . . . 00:00 (HH:MM)                                     │
  │ Options:                                                                  │
  │   Duration    . . . . . 0001:00:00 (HHHH:MM:SS)                           │
  │   Overrun     . . . . . 0000:05:00 (HHHH:MM:SS)                           │
  │   Max Tasks   . . . . . 5          (Integer)                              │
  │                                                                           │
  │                                                                           │
  └───────────────────────────────────────────────────────────────────────────┘
```

Figure 7-19 Create time period window

2. This “create time period” window opens and the user gets to enter the specifics of the date and time information for the window that they are creating (Figure 7-20 on page 93). We populated a window with a common maintenance window that happens to overlap with the deepest sleep that always seems to be interrupted during an on-call week. Also, you can see the use of the asterisk in the time period definition.

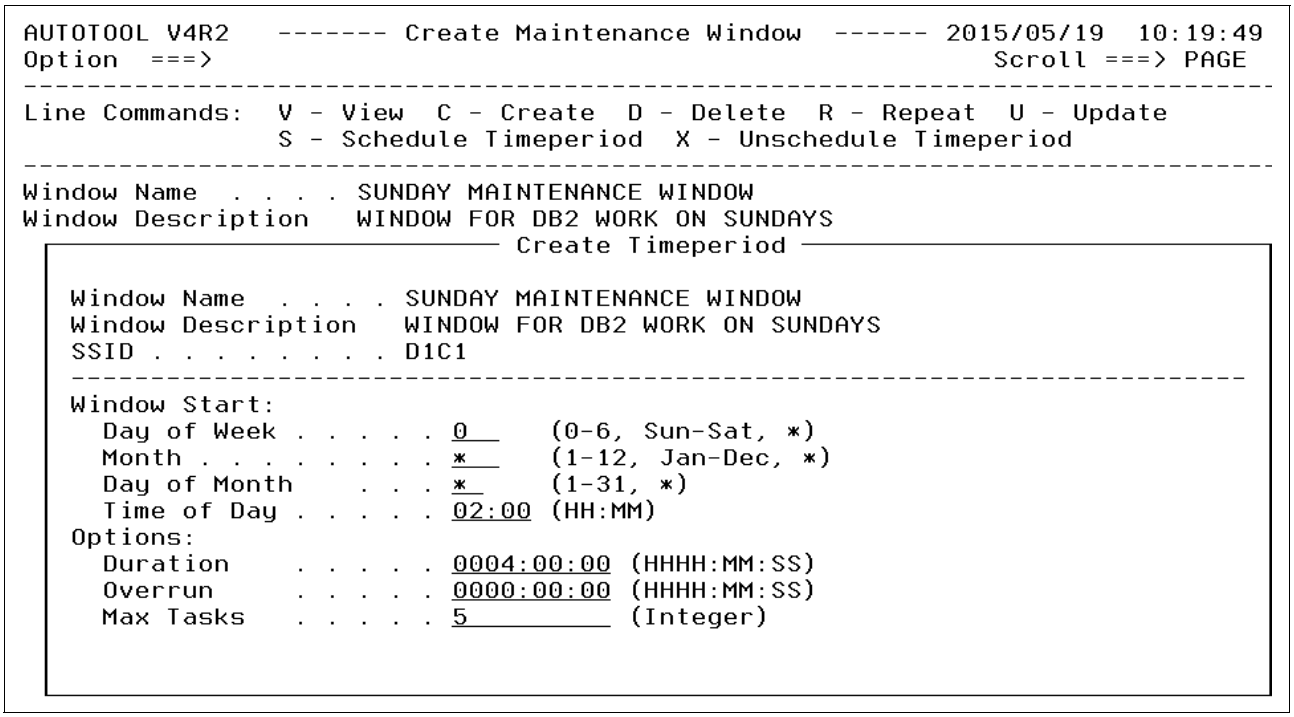


Figure 7-20 Maintenance window date and time information

3. Press Enter and you have the successful creation of an inconvenient maintenance window. Note the message at the bottom that the time period has been created. That is your confirmation of success (Figure 7-21 on page 94).

```
AUTOTOOL V4R2   ------- Create Maintenance Window  ------ 2015/05/19  10:31:17
Option  ===> ____________________________________________________ Scroll ===> PAGE
--------------------------------------------------------------------------------
Line Commands:  V - View  C - Create  D - Delete  R - Repeat  U - Update
                S - Schedule Timeperiod  X - Unschedule Timeperiod
--------------------------------------------------------------------------------
Window Name  . . . . SUNDAY MAINTENANCE WINDOW
Window Description   WINDOW FOR DB2 WORK ON SUNDAYS
SSID . . . . . . . . D1C1       (Blank for any datasharing member)
                                                        Row 1 of 1
--------------------------------------------------------------------------------
    Day of       Day of Time of
Cmd Week   Month Month  Day     Duration   Overrun    MaxTasks   Scheduled
 _  0      *     *      02:00   0004:00:00 0000:00:00          5 NO
***************************** Bottom of Data ********************************

                 HAA597I - Timeperiod has been successfully created.
```

Figure 7-21 Maintenance window created panel

4. You can enter a V on the command line next to your maintenance window to view (Figure 7-22).

```
AUTOTOOL V4R2   ------- Create Maintenance Window  ------ 2015/05/19  10:31:17
Option  ===> ____________________________________________________ Scroll ===> PAGE
--------------------------------------------------------------------------------
Line Commands:  V - View  C - Create  D - Delete  R - Repeat  U - Update
                S - Schedule Timeperiod  X - Unschedule Timeperiod
--------------------------------------------------------------------------------
Window Name  . . . . SUNDAY MAINTENANCE WINDOW
Window Description   WINDOW FOR DB2 WORK ON SUNDAYS
SSID . . . . . . . . D1C1       (Blank for any datasharing member)
                                                        Row 1 of 1
--------------------------------------------------------------------------------
    Day of       Day of Time of
Cmd Week   Month Month  Day     Duration   Overrun    MaxTasks   Scheduled
 v  0      *     *      02:00   0004:00:00 0000:00:00          5 NO
***************************** Bottom of Data ********************************
```

Figure 7-22 View option on Maintenance Window panel

5. While viewing the newly created maintenance window might not be the most useful action after you create your window, this option will be helpful after you have created multiple windows and might need to check something on a weekend with some special activity, such as hardware work or time change (Figure 7-23 on page 95).

```
AUTOTOOL V4R2   ------- Create Maintenance Window  ------ 2015/05/19  10:31:17
Option  ===>                                                Scroll ===> PAGE
-------------------------------------------------------------------------------
Line Commands:  V - View  C - Create  D - Delete  R - Repeat  U - Update
                S - Schedule Timeperiod  X - Unschedule Timeperiod
-------------------------------------------------------------------------------
Window Name  . . . . SUNDAY MAINTENANCE WINDOW
Window Description   WINDOW FOR DB2 WORK ON SUNDAYS
                              View Timeperiod

  Window Name  . . . . SUNDAY MAINTENANCE WINDOW
  Window Description   WINDOW FOR DB2 WORK ON SUNDAYS
  SSID . . . . . . . . D1C1
  ---------------------------------------------------------------------------
  Window Start:
    Day of Week . . . . . 0     (0-6, Sun-Sat, *)
    Month . . . . . . . . *     (1-12, Jan-Dec, *)
    Day of Month    . . . *     (1-31, *)
    Time of Day . . . . . 02:00 (HH:MM)
  Options:
    Duration    . . . . . 0004:00:00 (HHHH:MM:SS)
    Overrun     . . . . . 0000:00:00 (HHHH:MM:SS)
    Max Tasks   . . . . . 5          (Integer)
```

Figure 7-23 Maintenance Window view

6. Press PF3 to get back to the main Maintenance Window panel and you get another message about the creation of your window (Figure 7-24).

```
AUTOTOOL V4R2   ------ Autonomic Maintenance Window ----- 2015/05/19  10:41:36
Option  ===> ____________________________________________________ Scroll ===> PAGE
-------------------------------------------------------------------------------
Line Commands: C - Create AutoDirector Window  D - Delete  R - Repeat
               T - Create AutoStats Window  U - Update  V - View
               S - Schedule Window  X - Unschedule Window  W - Workload
-------------------------------------------------------------------------------
 Window Like  *_______________________________          DB2 Subsystem: D1C1
 Source       D (D - AutoDirector, S - AutoStats)
                                                        Row 1 of 1          >
-------------------------------------------------------------------------------
     Window                      Window
Cmd  Name                        Description                Source        SSID
 _   SUNDAY MAINTENANCE WINDOW   WINDOW FOR DB2 WORK ON SUN AUTODIRECTOR
***************************** Bottom of Data ********************************

  HAA597I - SUNDAY MAINTENANCE WINDOW has been successfully created.
```

Figure 7-24 Maintenance window created

7.3.3 Scheduling a maintenance window using Autonomics Console

Perform the following steps to schedule a maintenance window:

1. From here, you have many options and we schedule our window because that is why we created it. Type an S on the command line and press Enter (Figure 7-25).

```
AUTOTOOL V4R2   ------ Autonomic Maintenance Window ----- 2015/05/19  10:41:36
Option  ===> _____________________________________________________ Scroll ===> PAGE
--------------------------------------------------------------------------------
Line Commands: C - Create AutoDirector Window  D - Delete  R - Repeat
               T - Create AutoStats Window  U - Update  V - View
               S - Schedule Window  X - Unschedule Window  W - Workload
--------------------------------------------------------------------------------
 Window Like  *                                    DB2 Subsystem: D1C1
 Source       D (D - AutoDirector, S - AutoStats)
                                                   Row 1 of 1          >
--------------------------------------------------------------------------------
     Window                      Window
Cmd  Name                        Description                Source      SSID
 s   SUNDAY MAINTENANCE WINDOW   WINDOW FOR DB2 WORK ON SUN AUTODIRECTOR
****************************** Bottom of Data ***********************************
```

Figure 7-25 Autonomic Director Schedule panel

2. This panel for “Schedule DB2 Admin Task” is displayed (Figure 7-26).

```
AUTOTOOL V4R2   -------- Schedule DB2 Admin Task -------- 2015/05/19  11:10:09
Option   ===> ____________________________________________ Scroll ===> PAGE

Task Name  . . . . SUNDAY MAINTENANCE WINDOW                          >
Task Description   SYSAUTO.TIME PERIOD 3                             >

Begin Timestamp  . . _____________________________ (DB2 Timestamp)
End Timestamp  . . . _____________________________ (DB2 Timestamp)
Max Invocations  . . __________ (Integer, Blank)
SSID . . . . . . . . ____       (Blank for any datasharing member)
Job Wait . . . . . . Y          (Y - Yes, N - No, P - Purge)
Execution Threads. . 001        (Integer)

Invocation Options:
  Interval (minutes) __________ (Integer, Blank)
         -Or-
  Trigger:
    Task Name . . . ____________________________________________________ >
    Cond  . . . . . __          (GT,GE,EQ,LT,LE,NE)
    Code  . . . . . ____________ (Integer, Blank)
         -Or-
  Point in Time . . 00 02 * * 0                                         >
```

Figure 7-26 Schedule DB2 Admin Task panel

3. Press PF3 until you get the following message (Figure 7-27).

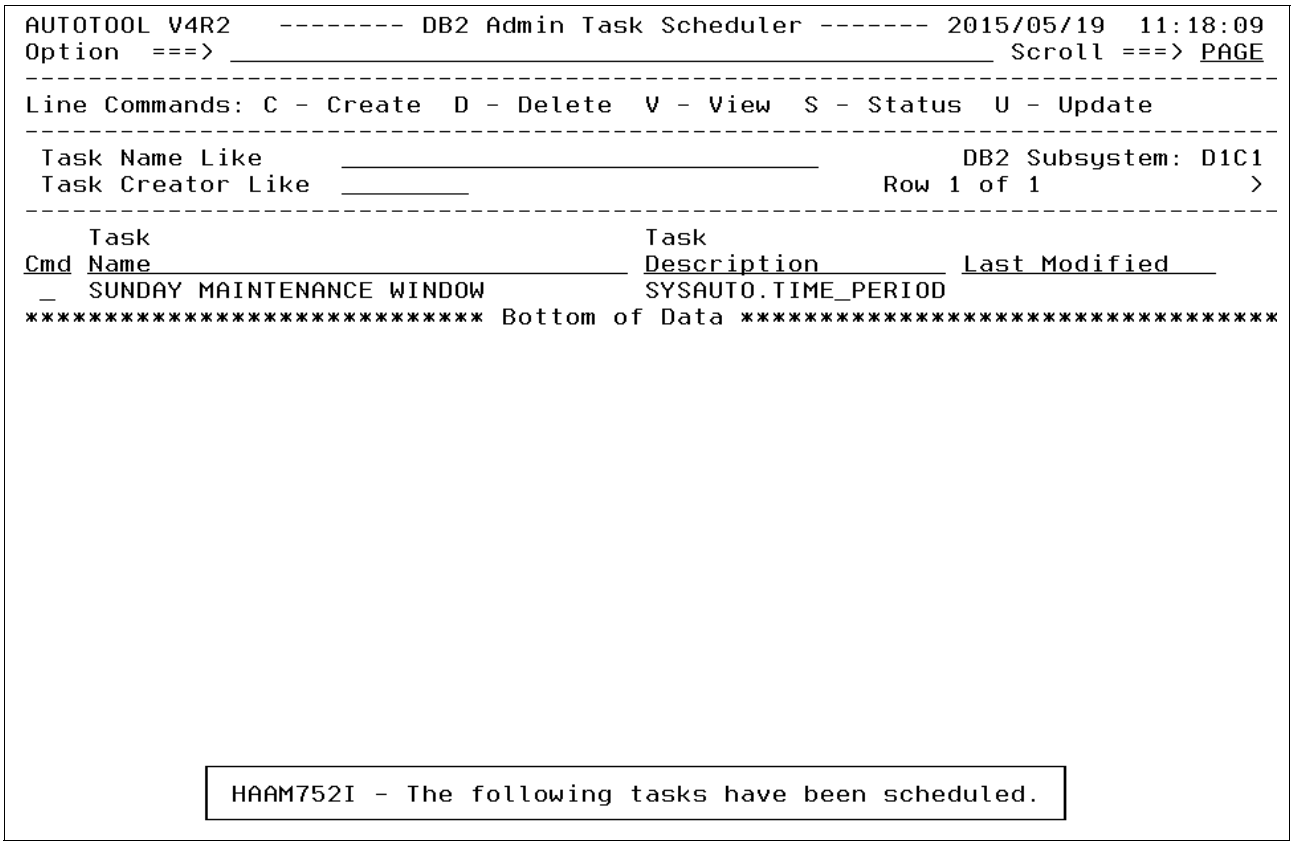


Figure 7-27 DB2 Admin task scheduler panel

Executing this sequence of panels has created and scheduled a maintenance window for the DB2 Admin Task Scheduler. The DB2 Admin Task Scheduler is part of DB2 for z/OS. There is a started task for the Admin Task Scheduler that may have ADMT as part of the name. The started task is required only if you are using the Admin Task Scheduler functionality so some configurations may not have the started task.

If you are using another batch job scheduler, there is an additional step that needs to be done to get your DB2 utility jobs to run in this maintenance window. The Automation Tool job profile has some fields that allow you to select the maintenance window. In addition, the job profile has a field for the autonomic type where you specify active or passive and specifying active means that a task is created when a build is run. The creating of a maintenance window is only one step toward getting actual DB2 utility jobs run in your production environment.

Every application has a unique batch schedule and in order to know when utility jobs can run, we need to understand when they should not run. The existing schedule should be understood so that when plans are made to convert to an autonomic batch environment, there will not be any surprises.

7.3.4 Define the base period

The *base period* is one or more time intervals when batch utility jobs will not be run. This period could be defined for different reasons: Online hours, maintenance window, or a peak CPU period that you want to avoid.

How to override the base period

There might be a base period for Monday - Friday of 8 am - 5 pm, but for the weekend, the workload on the system may be different and the application online hours may change. The Autonomics Director gives you the flexibility to make these changes by defining an override.

Procedure to edit the base period

You can edit existing base period definitions. To edit base period definitions, follow these steps:

1. From the Autonomics Director Primary Options Menu panel (IAVPRIM), choose option `1` System Administration and press Enter. The Autonomics Director System Administration Options Menu panel (IAVPSAM) is displayed.
2. Select option `1`: Manage scheduling calendar, and press Enter. The Autonomics Director Scheduler: Period Management panel (IAVPCLN) is displayed.
3. Select option `2`: Edit schedules and press Enter. The Edit Periods Filter panel (IAVPEOP) is displayed.
4. Apply a filter to narrow your search for operational definitions or press Enter to indicate no filtering. The Autonomics Director Maintenance Period Name List panel (IAVPEPD) is displayed.
5. Type an `S` into in the Act column of the period name to choose the period you want to edit. The Autonomics Director Scheduler: Edit Periods panel (IAVPTOP) is displayed.
6. Type an `S` into the Action column to select the period type to edit. Depending on the period type, the Specify Base Periods by Using Days of the Week (IAVPCWO), Specific Dates in a Month (IAVPBDT), or Generic Dates in a Month (IAVPBDY) panel is displayed.
7. Edit the overrides, type `SAVE`, and press Enter. For more information about how to specify the base period overrides, see *Defining overrides to the base period*:

   `https://www.ibm.com/support/knowledgecenter/SSS8US_1.4.0/com.ibm.imstools.aiiad.doc.ug/topics/aiad_usngoverdef.html`
8. Issue a **`REFRESH SCHEDULE`** server command to force the Autonomics Director server to use the updated definitions.

7.3.5 Define Job Profiles

Job Profiles are used to connect profiles. A Job Profile is the master profile. It is associated with one or more Utility Profiles, Object Profiles, and Exception Profiles. The combined profiles, which are headed by the Job Profile, form the basis of a DB2 Automation Tool task. We can submit this task manually or schedule it by using the DB2 administration task scheduler. Select Option `4`, Job Profiles, from the main DB2 Automation Tool menu. Here, we can list or create Job Profiles. For this scenario, we keep it simple and manually submit our image copy job. But first, we have to build the profile and associate it with the other profiles that we built. Similar to the other profile options, Option 4 presents us with a Job Profile list after we provide the selection criteria. In this case, we opted to use an asterisk for both the name and creator fields, which lists all Job Profiles. We have the installed sample profiles and the already created image copy Job Profile. From the selection list, we select the Job Profile that is created by ADMR2, which is called IMAGE_COPY.

Using Job Profiles in the maintenance window

While Job Profiles pull together the other profiles such as Object, Utility, and Exception Profiles, we need to be able to turn these profiles into some action. The action that we want is to execute a specific utility when a certain exception occurs and that is done in a two-step process.

A job is created in the DB2 Automation Tool that can execute out of a batch scheduler that produces one of two possible outputs. The first output produced in a situation where no exception has been triggered is writing JCL to an output data set that is an IEFBR14 job. The output data set would be used by the batch scheduler to submit the JCL and control cards created by the first job. The other possible output, in the situation where an exception is triggered, is a DB2 utility job with JCL and control cards built to the standards specified in the utility profile. This job would then be submitted by the batch scheduler to run in the maintenance window.

Validate current batch

In order to modify your current batch schedule to allow for utility exception processing, it is necessary to gather data on the performance of your current utility jobs. The Utilities Enhancement function of the DB2 Utilities Solution, intercepts the relevant data for you when active on your system. This data allows you to gain a better understanding of the existing batch utility jobs and how long they run and provide valuable input when determining whether they can fit into the maintenance window.

Modify existing batch cycle

Now that you have gathered data about the current production utility jobs, the existing batch schedule can be modified to create a larger maintenance window and allow for DB2 exception processing to occur.

The batch cycle should be examined to determine which utility jobs can be moved from the existing static schedule into the maintenance window for exception-based processing. By moving static jobs out of the batch schedule, you can enlarge your maintenance window giving your exception-based utilities a greater chance to run and complete. The removal of jobs from your batch cycle can be done concurrently with your giving Autonomics Director the ability to perform tasks in the maintenance window.

8

Running it all together: Automation Tool, UET, Management Console, Autonomics Director

In our evolution of autonomics and the need to move to smarter systems, there has been a bigger drive to the concept of “Active” versus “Passive” autonomics. With the inclusion of the Management Console and the Autonomics Director, it is now easier than ever to make that transition by leveraging the strength of the DB2 Utilities Solution Pack all in one standardized and centralized interface.

The Management Console streamlines the enterprise-wide monitoring of “Symptoms” and viewing the suggested “Actions” and allows for a phased approach toward Active Autonomics.

© Copyright IBM Corp. 2015. All rights reserved.

8.1 Synergy with the DB2 Automation Tool

The Management Console creates a window into the DB2 Automation Tool by allowing for the creation, modification, and viewing of the various DB2 Automation Tool profiles.

The Autonomics Director adds new capability to allow for a user view the various actions DB2 Automation Tool is recommending and as the user builds trust in the given actions, these can be altered from Passive to Active and be executed as needed. Once set to “Active”, the Autonomics Director executes these recommended “Actions” during the next maintenance window without human interaction.

8.1.1 Building the process

The DB2 Autonomics Director drives the “Active” automation process with the DB2 Automation Tool. By scheduling a “Build” job from within DB2 Automation Tool, any exceptions detected by the Build job are translated into Symptoms. Depending on the Job Profile used, these Symptoms can trigger Actions. The actions can be executed automatically at the beginning of the next maintenance window (Active) or they can be added to the queue as Passive and only set to Active once it has been reviewed. The following section guides you through the process of getting the process set up.

8.1.2 Building the Build Job

Build Jobs can be built for ad hoc functions, but in this instance we are looking at creating Build Jobs to be scheduled regularly by using the DB2 Administrative Task Scheduler. Perform the following steps to build the Build Job:

1. In order to create a regularly scheduled Build Job, you need to identify and create an Object Profile, Exception Profile, Utility Profile, and Job Profile.
2. From the DB2 Automation Tool main menu, choose option `4 – Job Profiles` (Figure 8-1).

```
AUTOTOOL V4R2 --- IBM DB2 Automation Tool for z/OS    --- 2015/06/02  16:20:04
Option  ===> _____________________________________________________________________
------------------------------------------------------------------------------

       Options: 0 - Setup                    8 - Dataset Manager
                1 - Object Profiles          9 - Data Page Display
                2 - Utility Profiles        10 - Event Notifications
                3 - Exception Profiles      11 - Stand Alone Utilities
                4 - Job Profiles            12 - DB2 Admin Scheduler
                5 - Quick Build             13 - Autonomic Console
                6 - Execution Reports        X - Exit
                7 - DB2 Command Processor

------------------------------------------------------------------------------
 DB2 Subsystem ID: D1C1          (1-4 Character Subsystem ID or ? for list)
 Current SQLID:    ________     User: MKRES2  - Configuration ID: HAA
------------------------------------------------------------------------------
```

Figure 8-1 Option 4 - Job Profiles

3. Once in the job profile window, press Enter again to display the list of defined job profiles. Then, tab to the profile that you want to schedule and use the **Build** command to start the process. Select how you want to build the jobs, either `O` – Online or `B` – Batch (Figure 8-2). This determines how the evaluation is done. This is a personal preference. Both can be added to the DB2 Administrative Task Scheduler. Batch provides the user with the JCL that can be added to an external scheduler or run manually as needed.

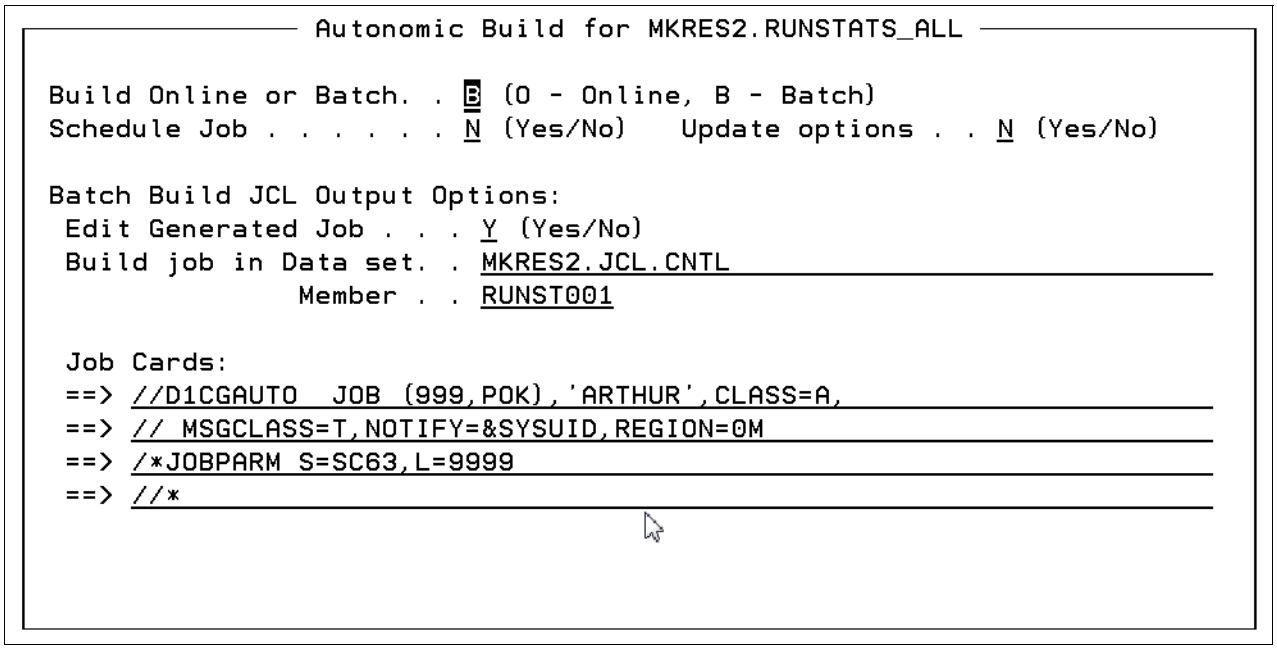


Figure 8-2 Autonomic Build panel

4. Then, determine if the job needs to be scheduled by the DB2 Administrative Scheduler (Figure 8-3 on page 104). This job will not be the job performing, in our case, the utility execution. This job runs to evaluate the listed objects in the Object Profile against the exceptions flagged in the Exception Profile. If an exception is tripped, the object along with the action are recorded. Specifying `Y` includes the panel for the DB2 Administrative Task Scheduler.
5. If you are automating this task using the DB2 Administrative Task Scheduler, schedule the task to run before the maintenance window that the Job Profile is attached to.
6. Specify a data set and member name for the JCL to reside and whether you want to edit the JCL after creation from this panel.
7. After all values are specified, press Enter.
8. Specify the options as needed for the DB2 Administrative Scheduler. Change the End Timestamp to the time stamp that you no longer want this job to be scheduled. Max Invocations can be set to blank to indicate this is a repetitive job.
9. You have the options to specify an Interval, Trigger, or Point in Time to schedule the job. Remember, this job needs to execute before the maintenance window in order to add the actions for that window to the Autonomics Director.
10. When all the options are completed, press PF3 to create the JCL. If you specified to schedule the job, the following panels are displayed (Figure 8-3 on page 104).

```
AUTOTOOL V4R2 ---------- Update DB2 Admin Task  --------- 2015/06/02  16:49:34
Option   ===> ____________________________________________________ Scroll ===> CSR

Task Name  . . . . AUTOMATION TOOL UTIL: &JOBNAME                          >
Task Description   ____________________________________________________   >

Begin Timestamp  . . &CURRENT_____________________ (DB2 Timestamp)
End Timestamp  . . . 2015-06-02-16.33.27.602293 (DB2 Timestamp)
Max Invocations  . . 1_________ (Integer, Blank)
SSID . . . . . . . . D1C1       (Blank for any datasharing member)

Invocation Options:
  Interval (minutes) __________ (Integer, Blank)
         -Or-
  Trigger:
    Task Name . . . ____________________________________________________   >
    Cond  . . . . . __          (GT,GE,EQ,LT,LE,NE)
    Code  . . . . . ____________ (Integer, Blank)
         -Or-
  Point in Time . . ____________________________________________________   >

Execution Source:
  JCL Data set. . .
  JCL Member  . . .              (If Partitioned Data set)
  Job Wait  . . . . Y            (Y - Yes, N - No, P - Purge)
         -Or-
  Procedure Schema                                                        >
  Procedure Name                                                          >
  Proce                                                                   >
         HAAM720E - End Timestamp must be later than current time.
```

Figure 8-3 DB2 Administrative Scheduler options

11. If you specified O - Online for the generation option, you see a similar panel (Figure 8-4).

```
          Messages Generated for MKRES2.RUNSTATS_ALL

      64 Informational,      226 Warning, and        0 Error
Messages were generated while building the job online for
Job Group . . . DEFAULT GROUP #1
of Jobs Profile MKRES2.RUNSTATS_ALL

View messages? . . . . Y (Yes/No)
View job summary?. . . N (Yes/No)
```

Figure 8-4 JCL creation messages

12. Press Enter to view any messages issued. Press PF3 to get out of the messages, and if you specified Y to edit the JCL, you are placed in edit mode for that member.

13. Make any or all changes needed. If changes are needed to the JCL, try to incorporate them in the build process, that is, jobcard options, and so on. It saves a tremendous amount of time if you can customize your installation to where any job that is generated can be executed without any customization.
14. If you chose not to add the job to the DB2 Administrative Scheduler, you can now either submit the job manually or schedule it in a different scheduler to execute before your maintenance window.

8.2 Execution

This section describes the following:

- Executing the Build Job
- Executing the Autonomics Director
- Autonomics Director and autonomics leading practices

8.2.1 Executing the Build Job

When the Build Job gets executed, it evaluates the objects in the Object Profile to the exceptions in the Exception Profile.

If a given object exceeds a specified threshold, the Build Job flags the symptom and adds the Recommended Action to the Actions List. This Actions List is what the Autonomics Director looks at when it is invoked. The symptoms and actions can be viewed in either the DB2 Management Console (Figure 8-5) or in the traditional Interactive System Productivity Facility (ISPF) DB2 Automation Tool interface. Perform the following steps:

1. To view the symptoms and actions in the DB2 Management Console, click **Resources**. Then, expand the DB2 Group and click the location name.

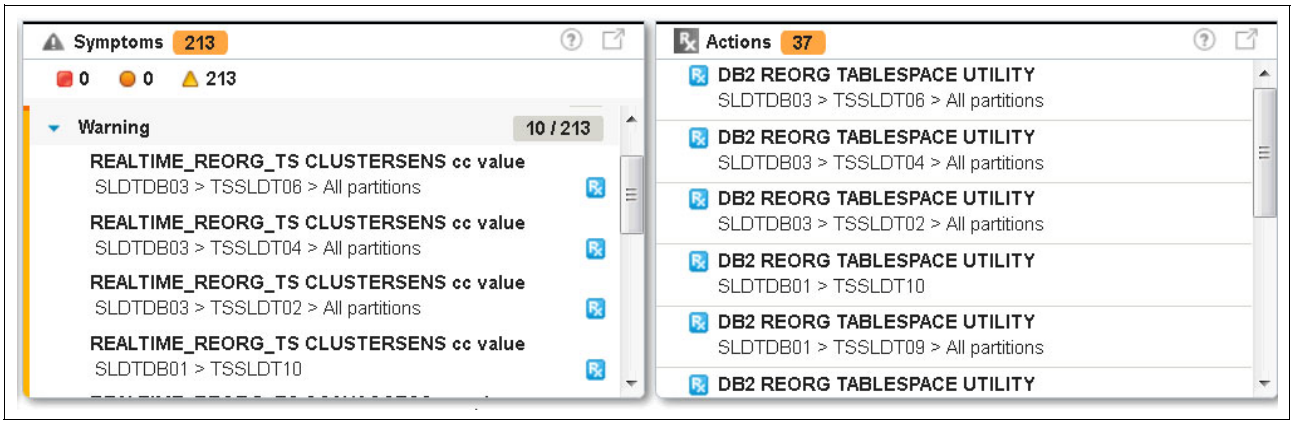


Figure 8-5 Symptoms and actions

2. The symptom and action window can be expanded by using the expand button. Expand the Actions panel to view all recommended actions (Figure 8-6).

Recommended Action	Status	Database Name	Resource Type	TS / IS Name	Partitior	Priority	Autonomic type	Window Name	Created On
DB2 REORG TABLESPACE UTILITY	OPEN	SLDTDB03	TS	TSSLDT06	All	-1	P	EACHDAY	2015-06-02 05:29:37
DB2 REORG TABLESPACE UTILITY	OPEN	SLDTDB03	TS	TSSLDT04	All	-1	P	EACHDAY	2015-06-02 05:29:36
DB2 REORG TABLESPACE UTILITY	OPEN	SLDTDB03	TS	TSSLDT02	All	-1	A	EACHDAY	2015-06-02 05:29:36
DB2 REORG TABLESPACE UTILITY	OPEN	SLDTDB01	TS	TSSLDT10	0	-1	A	EACHDAY	2015-06-02 05:29:36
DB2 REORG TABLESPACE UTILITY	OPEN	SLDTDB01	TS	TSSLDT09	All	-1	A	EACHDAY	2015-06-02 05:29:35

Figure 8-6 Recommended actions

3. The Actions can now be validated by the DBA to determine if this is the correct course of action. If the Job Profile was created using Passive Autonomics, the Action shows up with a “P” under autonomics type. This means that while the action is recommended, it will not be executed at the start of the next associated maintenance window. Any Action marked as “A” (Active) is executed at the start of the next associated maintenance window. Once executed, the Autonomics Director updates the action to indicate the status (“Completed”, “Error”, “Inprogress”, and “Open”).
4. The next execution of the Build Job resets any symptoms that no longer exist.
5. To view the Actions in the ISPF interface of DB2 Automations Tool, select the following options (Figure 8-7):
   - 13 - Autonomic Console
   - 4 - Autonomic Actions

```
AUTOTOOL V4R2   ----------- Autonomic Actions ----------- 2015/06/04  08:37:45
Option  ===> _                                              Scroll ===> CSR
--------------------------------------------------------------------------------
Line Commands: G - Group  O - Output  P - oPtions  S - Symptoms
--------------------------------------------------------------------------------
 Target Qualifier *
 Target Object    *
 Source I (S - AutoStats, T - Automation Tool)
 Status * (C - Completed, E - Error, I - Inprogress, O - Open, * - All)
 DB2 Subsystem: D1C1                                Row 19 of 37        -+>
--------------------------------------------------------------------------------
    Target           Target              Target
Cmd Qualifier        Object              Part  Priority  Action  Status
 _  SLDTDB01         TSSLDT06            0001            REORGTS OPEN
 _  SLDTDB01         TSSLDT07            0001            REORGTS OPEN
 _  SLDTDB01         TSSLDT08            0001            REORGTS OPEN
 _  SLDTDB01         TSSLDT09            0001            REORGTS OPEN
 _  SLDTDB01         TSSLDT10            ALL             REORGTS OPEN
 _  SLDTDB03         TSSLDT02            0001            REORGTS OPEN
```

Figure 8-7 Displaying Actions in Automation Tool

8.2.2 Executing the Autonomics Director

The Autonomics Director is a DB2 Stored Procedure. The Autonomics Director can be invoked in multiple ways.

If the option was selected to invoke the Autonomics Director when creating the maintenance window, a task is added to the DB2 Administrative Task Scheduler to execute the Autonomics Director at the start of that maintenance window.

If the option was not selected, the Autonomics Director can be invoked by a batch job that is supplied via APAR PI40921. This allows the Autonomics Director to be scheduled outside of the DB2 Administrative Task Scheduler. This JCL can be added to your installation's job scheduler as needed (Figure 8-8).

This is helpful if the available DB2 maintenance window may depend on other processes completing first.

```
000001 //D1CGAUTO  JOB (999,POK),'ARTHUR',CLASS=A,
000002 // MSGCLASS=T,NOTIFY=&SYSUID,REGION=0M
000003 /*JOBPARM S=SC63,L=9999
000004 //JOBLIB   DD DISP=SHR,
000005 //            DSN=DBTLSP.SDYXLOAD
000006 //         DD DISP=SHR,
000007 //            DSN=DBTLSP.SHAALOAD
000008 //         DD DISP=SHR,
000009 //            DSN=DB1CT.SDSNLOAD
000010 //         DD DISP=SHR,
000011 //            DSN=DB1CT.SDSNEXIT
000012 //DYX@ADRV EXEC PGM=DYX@ADRV,
000013 //         PARM='CALL=SP,EACHDAY,D1C1,DYXADSP'
000014 //UTPRINT  DD SYSOUT=*
000015 //SYSPRINT DD SYSOUT=*
000016 //STPRIN01 DD SYSOUT=*
000017 //SYSTSPRT DD SYSOUT=*
000018 //SYSOUT   DD SYSOUT=*
000019 //*
```

Figure 8-8 Sample Autonomics Director JCL

When the Autonomics Director gets invoked, it first checks to see if the maintenance window for which it is being executed, in this case, EACHDAY, is active. This depends on the Time Period specifications for the given maintenance window and the current time of day.

If the maintenance window is active, the Autonomics Director retrieves the list of "Actions" from the DB2 Automation Tool tables and begins to execute them. The number of tasks is determined by the definition in the maintenance window.

The Autonomics Director continues to execute the actions that are defined for the specified maintenance window until either all the Actions are completed, or the maintenance window's time expires. No new tasks are scheduled past the maintenance window, but running tasks are allowed to continue until they either complete or the amount of time that is specified for the "Allowed Overrun" expires. If the time expires the tasks are canceled.

The order of the tasks is determined by the Autonomic Object Prioritization in the DB2 Automation Tool. The priority values can range from 1 to the maximum for an integer (2.14 B) with 1 being the highest priority.

In the example shown in Figure 8-9, any table spaces in the database Europa* are processed before any table spaces in the database, SLDT*. Index spaces can also be prioritized.

```
AUTOTOOL V4R2   ---- Autonomic Object Prioritization ---- 2015/06/04  09:08:17
Option  ===> ____________________________________________ Scroll ===> CSR
-------------------------------------------------------------------------------
Line Commands: C - Create  D - Delete  U - Update  V - View
-------------------------------------------------------------------------------
                                                      DB2 Subsystem: D1C1
                                                  Row 1 of 3
-------------------------------------------------------------------------------
                                                     Obj
Cmd  Obj Database              Space Name            Type Obj Weight
 _   EUROPA*                   *                     IX          300
 _   EUROPA*                   *                     TS            5
 _   SLDT*                     *                     TS         5000
```

Figure 8-9 Autonomic Object Prioritization panel

As Autonomics Director tasks end, DB2 Utilities Enhancement Tool captures the utility output and stores it in the DB2 Automation Tool table, SYSAUTO.UTILITYRUNS_HISTORY. The output of the utilities can be viewed in the DB2 Management Console as follows:

1. Right-click the **Utilities** box on the resource display (Figure 8-10).

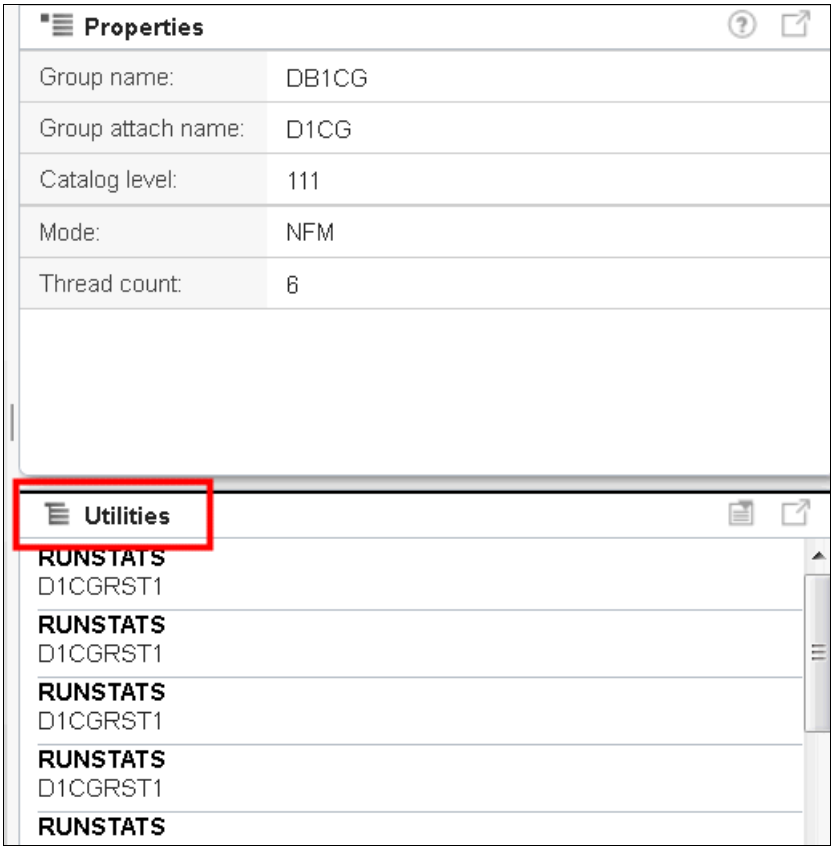


Figure 8-10 Utility History display

2. Use the expand button to list all the utilities. This displays a detailed view of the utility history (Figure 8-11).

Utility Name	Job Name	Step Name	Object Qualifier	Object Type	Object Name	Partition ID	Start Time	End Time	Output
TEMPLATE	D1CGWLMU	D1CGWLMU	UTLLDB01	TS	TSUTLL02	0	2015-06-02 14:05:30.41532	2015-06-02 14:05:30.536517	View...
TEMPLATE	D1CGWLMU	D1CGWLMU				0	2015-06-02 14:05:30.41532	2015-06-02 14:05:30.536517	View...
TEMPLATE	D1CGWLMU	D1CGWLMU	SLDTDB03	TS	TSSLDT06	0	2015-06-02 14:05:32.077467	2015-06-02 14:05:32.192541	View...
TEMPLATE	D1CGWLMU	D1CGWLMU				0	2015-06-02 14:05:32.077467	2015-06-02 14:05:32.192541	View...
TEMPLATE	D1CGWLMU	D1CGWLMU	SLDTDB03	TS	TSSLDT04	0	2015-06-02 14:05:33.549764	2015-06-02 14:05:33.665478	View...
TEMPLATE	D1CGWLMU	D1CGWLMU				0	2015-06-02 14:05:33.549764	2015-06-02 14:05:33.665478	View...
TEMPLATE	D1CGWLMU	D1CGWLMU	SLDTDB03	TS	TSSLDT02	0	2015-06-02 14:05:35.025083	2015-06-02 14:05:35.14123	View...

Figure 8-11 Expanded utility history

3. Click **View** to display the utility output (Figure 8-12).

```
                              DB2 UTILITY RUN HISTORY OUTPUT


JTILITY RUN ID                  JOB ID                  JOB NAME                START TIME

18                              JOB08727                D1CGRST1                2015-06-04 09:25:38.87377


)SNU000I    155 09:25:38.89 DSNUGUTC - OUTPUT START FOR UTILITY, UTILID = MKRES2.D1CGRST1
)SNU1044I   155 09:25:38.90 DSNUGTIS - PROCESSING SYSIN AS EBCDIC
)SNU050I    155 09:25:38.90 DSNUGUTC -  OPTIONS EVENT(ITEMERROR, SKIP)
)SNU1035I   155 09:25:38.90 DSNUZODR - OPTIONS STATEMENT PROCESSED SUCCESSFULLY
)SNU050I    155 09:25:38.90 DSNUGUTC -  LISTDEF RUNTS_LIST INCLUDE TABLESPACE DYXDB.DYXRHTS
)SNU1035I   155 09:25:38.91 DSNUILDR - LISTDEF STATEMENT PROCESSED SUCCESSFULLY
)SNU050I    155 09:25:38.91 DSNUGUTC -  RUNSTATS TABLESPACE LIST RUNTS_LIST TABLE(ALL) INDEX(ALL) KEYCARD SHRLEVEL
CHANGE UPDATE ALL REPORT NO HISTORY NONE
)SNU1033I   155 09:25:38.91 DSNUGULM - PROCESSING LIST ITEM: TABLESPACE DYXDB.DYXRHTS
)SNU610I  -D1C1 155 09:25:38.92 DSNUSUTP - SYSTABLEPART CATALOG UPDATE FOR DYXDB.DYXRHTS SUCCESSFUL
)SNU604I  -D1C1 155 09:25:38.92 DSNUSEF2 - TABLESPACE IS EMPTY
)SNU610I  -D1C1 155 09:25:38.92 DSNUSUTB - SYSTABLES CATALOG UPDATE FOR SYSAUTO.RESOLVES_HIST SUCCESSFUL
)SNU610I  -D1C1 155 09:25:38.92 DSNUSUCO - SYSCOLUMNS CATALOG UPDATE FOR SYSAUTO.RESOLVES_HIST SUCCESSFUL
)SNU610I  -D1C1 155 09:25:38.92 DSNUSUTS - SYSTABLESPACE CATALOG UPDATE FOR DYXDB.DYXRHTS SUCCESSFUL
)SNU610I  -D1C1 155 09:25:38.92 DSNUSUIP - SYSINDEXPART CATALOG UPDATE FOR SYSAUTO.DYXRHX1 SUCCESSFUL
)SNU604I  -D1C1 155 09:25:38.92 DSNUSEOF - INDEXSPACE IS EMPTY
)SNU610I  -D1C1 155 09:25:38.92 DSNUSUCO - SYSCOLUMNS CATALOG UPDATE FOR SYSAUTO.DYXRHX1 SUCCESSFUL
)SNU610I  -D1C1 155 09:25:38.92 DSNUSUIX - SYSINDEXES CATALOG UPDATE FOR SYSAUTO.DYXRHX1 SUCCESSFUL
)SNU620I  -D1C1 155 09:25:38.92 DSNUSEOF - RUNSTATS CATALOG TIMESTAMP = 2015-06-04-09.25.38.917083
)SNU010I    155 09:25:38.93 DSNUGBAC - UTILITY EXECUTION COMPLETE, HIGHEST RETURN CODE=0



                              DB2 Utilities Enhancement Tool intercept completed
```

Figure 8-12 DB2 Utility Run History Output panel

8.2.3 Autonomics Director and autonomics leading practices

Here are a few leading practices that can help you to maximize your value proposition from true DB2 Autonomics:

- Start out slow. Start off with Passive Autonomics. Then, by using the DB2 Management Console, validate the Actions and if you are satisfied, change the Action from Passive to Active. Tune the Job Profile as needed to ensure consistent and wanted results. After a given Job Profile is producing the required Symptoms and Actions, consider updating the Job Profile from Passive to Active.

- When to execute the Build Job. The Build Job does not perform any utilities rather, it evaluates and sets Symptoms and Actions. If you are looking to monitor the state of the objects, more frequent executions are needed. Once a day should be the minimum span between Build Executions. At a minimum, the Build Job needs to execute before the maintenance window for that given Job Profile.
- In order to build faith and report on what is operated on over a weekend, the last Build Job can be executed Friday and the output evaluated before the weekend. The ideal, when confidence levels allow, is to get to a point where you execute the build before the maintenance window. This would be the “Set it and forget it” approach.
- DB2 Management Console: Use the DB2 Management Console to view and take note of objects that are tripping exceptions. Use this as input to fine-tune exception profiles and maintenance windows.
- There are a number of supplied Best Practices Objects, Exception, Utility, and Job Profiles. View these to ensure that they meet your shop standard. If so, use these as initial starting points to get the basics out of the way, that is, RUNSTATS, COPY, and so on. You can later create your own if further fine-tuning is needed.

9

Benefits and results of a fully autonomic solution

This book described how to implement a comprehensive autonomics strategy using the tools and components shown in Figure 9-1. This chapter outlines the steps that our fictional company has taken to embrace this strategy and the benefits they are realizing from it.

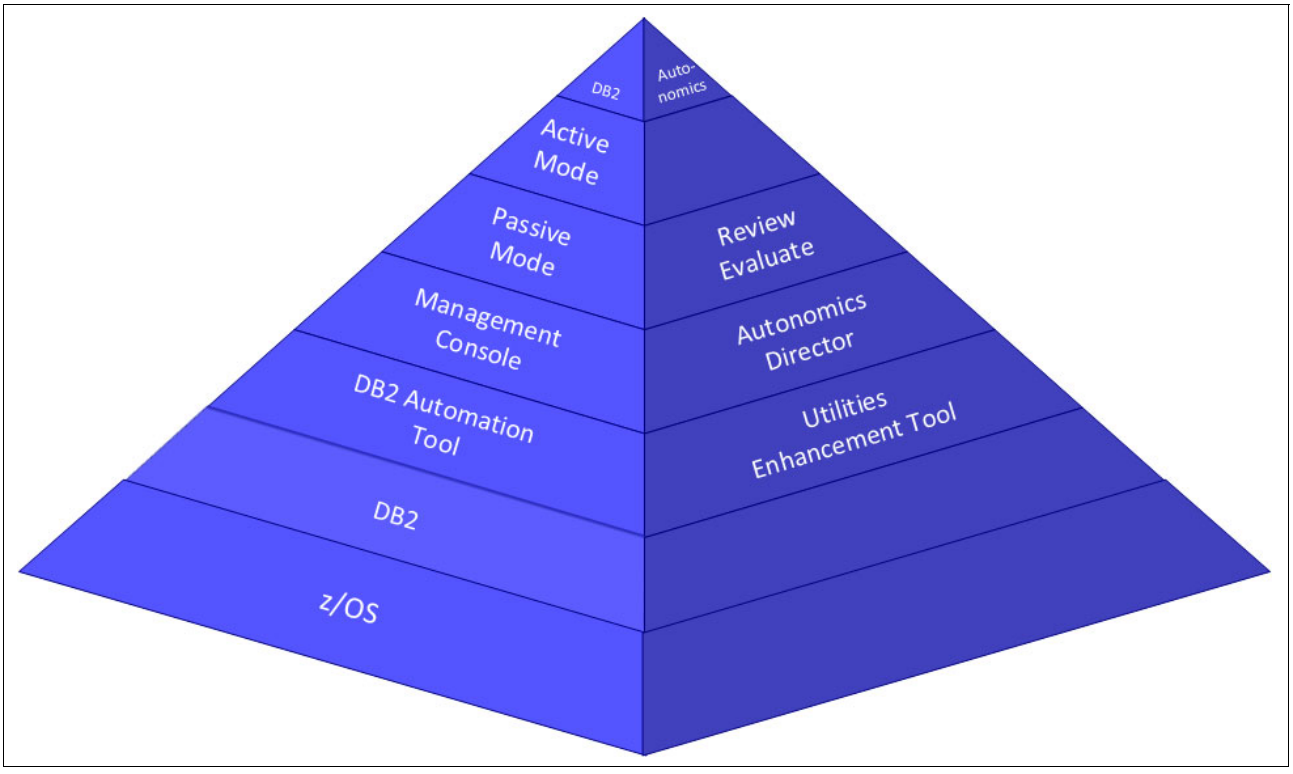


Figure 9-1 DB2 Autonomics

© Copyright IBM Corp. 2015. All rights reserved.

9.1 Business challenges

The challenges facing business and IT have accelerated in the past few years. Some challenges are ones faced by everyone in the business environment these days while others, such as the aging of the mainframe workforce, are more specific to our industry. These challenges that we discussed in this book require changes in how we work in order for us to perform effectively in this new working world.

For Company H, the overall number of database objects has grown in our environments while the support staff has not. Support responsibilities have increased where some mainframe DBAs are now supporting DB2 logical unit of work (LUW) in addition to DB2 for z/OS. Machines are faster, systems have been consolidated, buffer pools are larger, and application up-time demands have increased. DB2 subsystems come down less frequently and machines are IPLed less often. These changes caused the IT professionals at Company H to reevaluate how they do their jobs and what they spend their time and money on.

9.2 First steps

Every company has different challenges facing them, which determines the specific steps required for the appropriate response. Although each company takes a different path in terms of addressing these challenges, the problems outlined in this Redbooks publication are common to many companies and many of the actions they take are similar. Company H has decided they need to work smarter and use some of the new metrics that are available to make intelligent decisions about the maintenance jobs they run in production.

Company H is running utilities via the brute force method. Reorg and image copy everything weekly. RUNSTATS is part of the weekly maintenance cycle followed by a BIND job. There are inefficiencies built into this maintenance cycle but it has worked for the past 20 years. With real-time statistics (RTS) available, Company H has decided to take advantage of having that data available to make an intelligent decision about whether to reorg or not. The DB2 Automation Tool is implemented to reduce CPU wasted on unnecessary reorgs and improve throughput by not running some utility jobs. Company H should also save money because they incur charges based on busy CPU times, which are reduced with this change. That dollar savings can help justify the work that needs to be done to implement automation tools in your environment. Another benefit is reducing the overall costs of the mainframe environment making it more competitive when compared with midrange.

9.3 Second step

The next thing that they tackle is old utility standards that have not kept up to date with all of the changes in DB2 utilities and using a tool that enforces those standards in test and production. Every version of DB2 introduces new utility enhancements and Company H has not incorporated those changes into existing production jobs. They have application groups that are running LOAD utilities with JCL and control cards that have been passed down from someone who is long gone. Company H uses the Utilities Enhancement Tool function to create standard utility policies for all of their DB2 utilities, using current keywords and syntax. They also use this opportunity to phase out the JCL and control card generator that has been supported by the DBA who is planning to retire next year. Who supports those homegrown utilities when someone retires? Company H is protecting themselves by implementing UET.

9.4 A more general problem: The aging workforce

Company H faces an aging mainframe workforce but has a plan to address some of their problems related to this issue.

The Management Console is part of the transition taking place to provide a modern web-based GUI interface in addition to the trusted ISPF-based interface. While many of us in the mainframe world have grown up with TSO and ISPF, future employees come to the mainframe world already knowing how to navigate GUI-based tools. Company H decided to implement the Management Console because they know that the younger generation of mainframe technicians embrace modern GUI-based interfaces. In addition, the Management Console provides embedded help that the younger generation embraces because it allows them to get up to speed much faster.

Company H found that implementing these changes has allowed them to work their way up the Autonomic Hierarchy with Management Console and Autonomics Director. These tools and functions are helpful when you work in an environment with many DB2 subsystems, many applications, and many utility jobs. The goal is to work toward an environment where you can manage by exception and focus on the higher-level tasks, such as performance analysis or application tuning. Company H started by implementing passive autonomics for 6 months in one of their production systems before switching to active autonomics.

There is a need to have a common and consolidated presentation of all the complex information that is gathered today related to DB2 maintenance and performance metrics. The Management Console provides an enterprise-wide presentation of DB2 system health data, including autonomic symptoms, exceptions, and recommended actions.

IT organizations may suffer from generational shifts in technology, business pressures, as well as dwindling IT skills. The IBM autonomics strategy and the IBM Management Console for IMS and DB2 for z/OS can help close the gap between yesterday's IT implementation and tomorrow's future IT demands. While autonomics is aimed at eliminating the repetition of manual day-to-day tasks, the Management Console aims to increase user efficiency by consolidating information from IMS and DB2 for z/OS systems in a single interface that is easily accessible from a standard web browser.

The process of implementing these automation pieces does require time and planning but we have given examples throughout the book of how this can be done in a phased implementation. This is a requirement in production environments to develop a plan that provides a phased implementation with the ability to validate along the way. The best example of this documented in our book is the conversion from passive to active autonomics, which can be done by monitoring for a period of time while in passive mode. This is important to allow a period to develop trust in the autonomics software before turning over control when going to active mode. Before this conversion, verify before you trust.

9.5 Autonomics benefits

There have been many benefits for Company H since implementing a comprehensive autonomics strategy. Utility jobs are run when needed, not just when scheduled. CPU consumption by DB2 utility jobs has been reduced. Application groups no longer run jobs that use non-supported and deprecated DB2 syntax. DB2 Utility standards have been developed, published, and enforced. All of these changes have meant a more stable environment allowing for more time spent on projects and application tuning.

The DBA team is working on a project to identify and drop unused indexes, something they have not been able to find time for in the past. Also, they have been working with Application Performance Analyzer for z/OS to improve the elapsed and CPU time of some key online transactions.

Company H faced their problems and challenges, which are not new but have become more pressing in the past few years. These problems are not going away and require action on your part sooner rather than later. The best course of action is to analyze your environment and determine which problems you want to address first. Develop a plan of action based on the problem that you need to solve now but keep in mind that the immediate problem should be part of an overall plan to implement Active Autonomics in your environment. Autonomics is very strategic for all DB2 customers. We documented how *Passive Autonomics* collect, analyze, and recommend actions to resolve performance issues related to DB2 object maintenance or lack thereof. With *Active Autonomics*, you now have the ability to take real actions and get real benefits. Full implementation of Active Autonomics helps to reduce the learning curve for new users of IMS and DB2 for z/OS and also frees up valuable time for experienced users who benefit from these time-saving features.

A

Hints and tips

During our exercises in this book, we also found some items that we would like to document here as important for your planning and implementation steps towards a Passive Autonomics and Active Autonomics strategy.

© Copyright IBM Corp. 2015. All rights reserved.

DB2 Stored Procedures

DB2 provides two Stored Procedures: ADMIN_UTL_MONITOR and ADMIN_UTL_EXECUTE. Using these stored procedures with the DB2 Administrative Task Scheduler, we were able to:

- Detect which objects had out-of-date statistics due to lack of RUNSTATS utility and then were able to alert (and recommend that RUNSTATS needs to be run) the right utility for the object to a SYSAUTOALERTS table, using the ADMIN_UTL_MONITOR.
- By using the ADMIN_UTL_EXECUTE stored procedure, you can execute RUNSTATS to resolve said alerts. In addition, a specified maintenance window (defined in SYSAUTOTIMEWINDOWS) resolves as many alerts as it can by calling the RUNSTATS utility.
- It writes the results of the alert resolution to the SYSAUTORUNS_HIST history file.

APF Authorization Failure in Automation Tool

Typically the DB2 Automation Tool requires no extra security measures outside of standard DB2 security. However, if your site uses ACF2 to restrict the TSO command use, you may need to add the TSO commands that DB2 Automation Tool uses to the ACF2 Command Limiting table. The TSO commands that DB2 Automation Tool uses are:

- `HAA$MAIN`
- `HAA@BULD`
- `FECDBCX`

Also, authorize the FEC$TSOC program by adding it to the AUTHPGM and AUTHTSF sections of member IKJTSO00 in SYS1.PARMLIB.

The Management Console and JCC packages

In order to use the Management Console, ensure that the JCC packages are BINDed on the same DB2 or DB2 data sharing group, where you want to connect to.

Automation Tool and Management Console widgets

The Management Console uses widgets. The widgets are the Symptoms, Exceptions, and so on icons that you double-click from the dashboard.

In order to view the actions and symptoms in the Management Console, verify that the widgets are installed, otherwise, you will not be able to see them in the dashboard.

What constitutes the Autonomics Director

When you install the Autonomics Director, you install the following DB2 objects: a Stored Procedure, and DB2 tables:

- The Stored Procedure, DYXEXP1. This SP is used to explode the Automation Tool Profiles
- Three DB2 tables:
  - OBJECT_EXPLODE – Object explode for LISTDEF Profiles
  - RTS_SNAPSHOT
  - AUTONOMIC_DIRECTOR

Management Console terms

- A *Symptom* is an indicator of a problem with the system. The Symptom's severity should reflect the overall pain to the system.
- An *Action* is an action that the Autonomics Director can perform (for example, REORG or RUNSTATS Utility). The objective is to the highest priority of Actions that have the biggest impact to the overall system.

Management Console steps towards Active Autonomics

Using the modern Management Console, we were able to do the following actions:

- Access DB2 subsystems, run discovery, and explore DB2 objects, from a single view
- Automate the data collection by executing the stored procedure, RTS_SNAPSHOT, through the DB2 Administrative Task Scheduler to build a historical record of object statistics
- Enable the ability to automatically capture utility jobs and record the metrics in the Utility History Table
- Automate data analysis by driving utility evaluations through the DB2 Administrative Task Scheduler. You can use your own scheduler
- Enable Active Autonomics by defining a maintenance window. Set up Active Autonomics Director to regularly drive REORG, COPY, RUNSTATS on a set of objects

B

DB2 Autonomics Director tables

In this chapter, we describe the tables that are part of DB2 Autonomics Director and that are shipped with Tools Base. These tables must be created within the DB2 catalog for Autonomics to work.

© Copyright IBM Corp. 2015. All rights reserved.

Concepts

To fully understand DB2 Autonomics operation, we should clearly understand *Evaluators* and *Enactors* concepts because the actual work in DB2 Autonomics is left up to two classes of these pluggable functions:

- **Evaluators**: Analyze the state of the system and identify symptoms they register in the SYMPTOMS table and make recommendations by registering them in the ACTIONS table.
- **Enactors**: Perform a specified action out of the ACTIONS table as instructed by the Autonomics Director. Enactors are also responsible for providing projections of time and CPU for actions.

The Autonomics Director Module is in charge of projecting and executing a maintenance window. Running in a stored procedure called by whatever means (Admin Scheduler, On Demand, Batch Scheduler, and so on), its job is to examine the Actions list, prioritize the required actions, and drive registered enactors to complete the highest priority actions that fit within the maintenance window.

The Autonomics Director is called through a stored procedure and is passed in the maintenance window it is supposed to run. Autonomics Director could be invoked at any time throughout a maintenance window (it need not be at the start). Autonomics Director determines how much time is remaining in the current window (if any at all) and begins to execute Actions from the Actions list.

There is the notion of a flexible window versus a hard stop window. This can be specified as the allowed overrun value in the window definition. In either case, the Autonomics Director attempts to only schedule jobs that should complete before the end of the window. However, it allows the value of overrun in seconds for enactors to complete before terminating them. A zero overrun means a hard stop at the end of the window.

Enactors must register in the Enactor Registry so they can be found by the Autonomics Director and must provide a Stored Procedure interface.

List of tables and their descriptions

All Table Definitions are shipped in the Tools Base and must be created within the DB2 catalog for Autonomics to work.

SYSAUTO.SOURCE (Table B-1): Registry of all contributors (evaluators and enactors). This is a simple registry of all the SOURCE values that appear in the other tables. This can serve as a registry of all tools (enactors and evaluators) installed and participating in Autonomics on this system.

Table B-1 SYSAUTO.SOURCE

Column name	Data type	Description
SOURCE	VARCHAR(32) NOT NULL	Registered Evaluator or Enactor
DESCRIPTION	VARCHAR(256)	Description of the registered source
VERSION	SMALLINT	Version of the registered source

SYSAUTO.SYMPTOM_REGISTRY (Table B-2): Registry of all possible symptoms each evaluator could potentially flag. This table provides a list of all symptoms that may be raised. Also provides English short and long descriptions that are used in the ISPF panels. GUI panels use the SYMPTOM_KEY to do a translation lookup for the description.

Table B-2 SYSAUTO.SYMPTOM_REGISTRY

Column name	Data type	Description
SYMPTOM_KEY	VARCHAR(64) NOT NULL	Identifier assigned by the originator of the Symptom Primary Key: should start with some Tool-related prefix
SOURCE	VARCHAR(32) NOT NULL	Owner of this symptom. Foreign key to SOURCE in SYSAUTO.SOURCE table
SHORT_DESCRIPTION	VARCHAR(64) NOT NULL	Short Description to be displayed in the Exception Profile Panel
LONG_DESCRIPTION	VARCHAR(256) NOT NULL	Long Description including replaceable variables
EXTENDED_DESCRIPTION	VARCHAR(256)	Extended description on the symptom
ZVARIABLES	SMALLINT NOT NULL	Number of &Z variables in LONG_DESCRIPTION
DEFAULT_SEVERITY	INTEGER	Default SEVERITY if one is not provided by Evaluator
SEVERITY_MULTIPLIER	FLOAT	Multiplier for tweaking how symptoms of this type are handled
SEVERITY_MAX	INTEGER	Max severity for Symptoms of this type

SYSAUTO.SYMPTOM (Table B-3): List of current symptoms.

Table B-3 SYSAUTO.SYMPTOM

Column name	Data type	Description
SYMPTOM_ID	BIGINT NOT NULL	Unique ID for Symptom (Generated)
SYMPTOM_KEY	VARCHAR(64) NOT NULL	Foreign Key to SYMPTOM_KEY in the SYMPTOM_REGISTRY table
EVAL_HISTORY_ID	BIGINT	The HISTORY_ENTRY_ID inserted into SYSAUTO.AUTORUNS_HIST by the evaluator stored procedure
TARGET_TYPE	CHAR(2) NOT NULL	Indicates the type of object this action is for: TS = Tablespace IX = Indexspace DB = Database SS = Subsystem
TARGET_QUALIFIER	VARCHAR(128) NOT NULL	The qualifier name of the DB2 object this alert applies to, that is, the database name
TARGET_OBJECT	VARCHAR(128) NOT NULL	The name of the DB2 object this alert applies to, that is, the table space name
TARGET_PARTITION	SMALLINT NOT NULL	The partition number of the DB2 object this alert applies to. Zero if alert applies to all partitions, or if the object is not partitioned

Column name	Data type	Description
STATUS	VARCHAR(32)	Active Inactive Dismissed
SEVERITY	INTEGER	Severity of symptom (entirely determined by product that flagged the symptom)
STARTTS	TIMESTAMP(12) NOT NULL GENERATED ALWAYS AS ROW BEGIN	Timestamp symptom was discovered, maintained by DB2 with temporal system time when row is inserted
ENDTS	TIMESTAMP(12) NOT NULL GENERATED ALWAYS AS ROW END	Timestamp symptom was no longer pertinent, maintained by DB2 with temporal system time when row is deleted
TRANS_ID	TIMESTAMP(12) GENERATED ALWAYS AS TRANSACTION START ID IMPLICITLY HIDDEN	Transaction start time recorded by DB2 to track when the transaction first executed causing the change. This is hidden to not appear on a typical SELECT *
PERIOD	SYSTEM_TIME (STARTTS, ENDTS)	Required by Temporal tables to add versioning
ZVALUES	VARCHAR(2048)	Source-specific data associated with symptom. For example, comma-delimited list of values used to format description of symptom reason

SYSAUTO.SYMPTOM_HISTORY: DB2 for z/OS V10 Temporal table for Symptoms.

SYSAUTO.ACTION_REGISTRY (Table B-4): Registry of all possible action recommendations that each evaluator could potentially flag. It also provides some short English descriptions of what the recommendation is for use in ISPF panels. Web UI would key off the ACTION to create translated text.

Table B-4 SYSAUTO.ACTION_REGISTRY

Column name	Data type	Description
SOURCE	VARCHAR(32) NOT NULL	Originator of this potential action recommendation
ACTION	VARCHAR(64) NOT NULL	Identifier assigned by the originator of the action. Primary Key: should start with some Tool-related prefix
SHORT DESCRIPTION	VARCHAR(64) NOT NULL	Short Description to be potentially displayed in ISPF panels
LONG_DESCRIPTION	VARCHAR(256)	Long Description including replaceable variables
ZVARIABLES	SMALLINT	Number of &Z variables in LONG_DESCRIPTION
DEFAULT_PRIORITY	SMALLINT	Optional User Defined default priority for actions of this type
ENACTOR_SCHEMA	VARCHAR(128)	The schema of the enactor stored procedure that resolves this action
ENACTOR_NAME	VARCHAR(128)	The name of the enactor stored procedure that resolves this action

SYSAUTO.ACTION (Table B-5): List of current recommended actions. This is an extension table of the SYSAUTOALERTS tables. It stores additional information about the ACTIONS we need.

Table B-5 SYSAUTO.ACTION

Column name	Data type	Description
ACTION_ID	BIGINT NOT NULL GENERATED ALWAYS AS IDENTITY	Primary Key
ACTION	VARCHAR(64) NOT NULL	The type of action requested by this alert. Foreign key to SYSAUTO.ACTION_REGISTRY
WINDOW_ID	BIGINT	Maintenance window ID where the action will be executed from
EVAL_HISTORY_ID	BIGINT	The HISTORY_ENTRY_ID inserted into SYSAUTO.AUTORUNS_HIST by the evaluator stored procedure
AUTODIRECTOR_HISTORY_ID	BIGINT	The HISTORY_ENTRY_ID inserted into SYSAUTO.AUTORUNS_HIST by the Autonomic Director stored procedure execution that resolved the action
TARGET_TYPE	CHAR(2)	Indicates the type of object this action is for (default is table space as that is what the SYSAUTOALERTS table only supports if this extension is not present): TS = Tablespace IX = Indexspace DB = Database SS = Subsystem
TARGET_QUALIFIER	VARCHAR(128) NOT NULL	The qualifier name of the DB2 object this alert applies to, that is, the database name
TARGET_OBJECT	VARCHAR(128) NOT NULL	The name of the DB2 object this alert applies to, that is, the table space name
TARGET_PARTITION	SMALLINT NOT NULL	The partition number of the DB2 object this alert applies to. Zero if alert applies to all partitions, or if the object is not partitioned
OPTIONS	VARCHAR(4000)	The options to be used when executing the corresponding action. Usually this value is 'USE PROFILE', but could it be 'TABLE'- for this table only, or 'COLUMNS', for these columns only, or 'SAMPLE', if the table space is too large and sampling is allowed
STARTTS	TIMESTAMP(12) NOT NULL GENERATED ALWAYS AS ROW BEGIN	Timestamp action was inserted, maintained by DB2 with temporal system time when row is inserted
ENDTS	TIMESTAMP(12) NOT NULL GENERATED ALWAYS AS ROW END	Timestamp action was no longer pertinent, maintained by DB2 with temporal system time when row is deleted

Column name	Data type	Description
TRANS_ID	TIMESTAMP(12) GENERATED ALWAYS AS TRANSACTION START ID IMPLICITLY HIDDEN	Transaction start time recorded by DB2 to track when the transaction first executed causing the change. This is hidden to not appear on a typical SELECT *
PERIOD	SYSTEM_TIME (STARTS, ENDTS)	Required by Temporal tables to add versioning
DURATION	INTEGER	An estimation of the time in seconds that would be needed to execute the corresponding action. Caution: If this column contains NULL, the execution plan may largely overwrite the time windows
STATUS	VARCHAR(32)	The status of the actual planned task. Valid values are: OPEN: alert is not resolved yet INPROGRESS: alert execution started FAILED: alert execution failed COMPLETED: alert execution is completed RETRACTED: alert is no longer needed
UTILITYRUN_ID	BIGINT	Foreign key to SYSAUTO.UTILITYRUNS_HISTORY. NULL if action hasn't been run yet or isn't a utility action
SOURCE	VARCHAR(32) NOT NULL	Tool that added the Action. A value of USER indicates it was placed here by the user (DBA)
DURATION_CONFIDENCE	FLOAT	Confidence in the prediction of DURATION. NULL implies no measure of confidence has been assigned; this is different than confidence of zero
CPU_PREDICTION	BIGINT	Estimation of the amount of CPU needed to execute the corresponding action
CPU_CONFIDENCE	FLOAT	Confidence in the prediction of CPU. NULL implies no measure of confidence has been assigned; this is different than confidence of zero
PRIORITY	INTEGER	Priority given to this action. These values are editable by the DBA USER or the planning module. NULL means no priority has been assigned; -1 means DO NOT RUN.
PRIORITY_OVERRIDE	CHAR	YIN Indicates the PRIORITY value has been overridden by the USER and should not be programmatically changed. The intent here is to allow user override without losing the ability to simply programmatically sort by priority

SYSAUTO.ACTION_HISTORY – Temporal table for maintaining historical list of recommendations (taken or not).

SYSAUTO.RESOLVES (Table B-6): The RESOLVES table connects SYMPTOMS to ACTIONS. This is its own independent table because a single SYMPTOM might have several ACTIONS that can resolve it (for example, a free space issue may be resolved by repartitioning a table space, but may also be resolved by running a REORG). Similarly, a single ACTION may resolve several SYMPTOMS (for example, a REORG may resolve a free space issue, as well as a fragmentation issue, and potentially a performance problem on a specific query).

It's the job of the Evaluator that places a new Action into the table that is also responsible for managing the addition of the resolves table. As such, this has a direct RI relationship with ACTIONS with cascade on delete.

It's the job of the Evaluator that places a new Action into the table that is also responsible for managing the addition of the resolves table. As such, this has a direct RI relationship with ACTIONS with cascade on delete.

Table B-6 SYSAUTO.RESOLVES

Column name	Data type	Description
ACTION_ID	BIGINT NOT NULL	Primary Key in SYSAUTOALERTS.ALERT_ID
SYMPTOM_ID	BIGINT NOT NULL	Foreign Key in SYMPTOM
CONFIDENCE	FLOAT	Confidence that specified action will resolve the specified symptom. NULL implies no measure of confidence has been assigned; this is different than confidence of zero
STARTTS	TIMESTAMP(12) NOT NULL GENERATED ALWAYS AS ROW BEGIN	Timestamp resolves correlation was added, maintained by DB2 with temporal system time when row is inserted
ENDTS	TIMESTAMP(12) NOT NULL GENERATED ALWAYS AS ROW END	Timestamp resolves correlation was no longer pertinent, maintained by DB2 with temporal system time when row is deleted
TRANS_ID	TIMESTAMP(12) GENERATED ALWAYS AS TRANSACTION START ID IMPLICITLY HIDDEN	Transaction start time recorded by DB2 to track when the transaction first executed causing the change. This is hidden to not appear on a typical SELECT *
PERIOD	SYSTEM_TIME	Required by Temporal tables to add versioning

SYSAUTO.RESOLVES_HISTORY: Temporal table for maintaining history of Resolves.

SYSAUTO.AUTORUNS_HIST (Table B-7): History of all Active Autonomics Runs (Autonomic Director Runs). This is a Tools Base version of the SYSAUTORUN_HIST table. It is used to record all activity for Automation Tool and Autonomics Director. Without an ACTION_ID field, there is no way to track this information back to the specific utilities that ran.

Table B-7 SYSAUTO.AUTORUNS_HIST

Column name	Data type	Description
HISTORY_ENTRY_ID	BIGINT NOT NULL GENERATED ALWAYS AS IDENTITY	
PROC_NAME	VARCHAR(128)	Name of Autonomic Stored Procedure
STARTS	TIMESTAMP	Time Autonomic SP started
ENDTS	TIMESTAMP	Time Autonomic SP ended
OUTPUT	CLOB(2M)	Output of Autonomic SP
RETURN_CODE	INTEGER	Return Code of Autonomic SP
ERROR_MESSAGE	VARCHAR(1331)	Error message from Autonomic SP
ROWID	ROWID	ROWID for Clob column

SYSAUTO.TIMEWINDOWS (Table B-8): An independent version of SYSAUTOTIMEWINDOWS used to define autonomic windows rather than windows just for AUTOSTATS. Defines a window in which specific tools are allowed to run.

Table B-8 SYSAUTO.TIMEWINDOWS

Column name	Data type	Description
WINDOW_ID	BIGINT NOT NULL GENERATED ALWAYS AS IDENTITY	
DB2_SSID	CHAR(4)	The DB2 member name on which the planned tasks have to be executed. If this column contains NULL, the tasks in this time window can be executed on any DB2 member
WINDOW_NAME	VARCHAR(30)	Window Name
DESCRIPTION	VARCHAR(256)	Optional Description of Window
CREATE_TIMESTAMP	TIMESTAMP	Time this window was created
CREATE_USER	CHAR(8)	User who created this window
UDATE_TIMESTAMP	TIMESTAMP	Last time this window was updated
UPDATE_USER	CHAR(8)	Last user to update window
UPDATEABLE	CHAR(1)	YIN Indicates if other users can update window
SNAPSHOT_RTS	CHAR(1)	YIN Indicates if RTS should be snapshotted before and after the execution of the maintenance window

SYSAUTO.TIMEPERIODS (Table B-9 on page 127): Represents individual time periods that make up a time window. For example, a window might be Tuesdays 9 - 10 and Thursdays 11 - 12. This would be represented by a single time window and two time periods.

Table B-9 SYSAUTO.TIMEPERIODS

Column name	Data type	Description
PERIOD_ID	BIGINT NOT NULL GENERATED ALWAYS AS IDENTITY	Unique ID for individual periods
WINDOW_ID	BIGINT NOT NULL	Foreign Key to SYSAUTOTIMEWINDOWS
START_CRON	VARCHAR(64) NOT NULL	CRON Expression for recurring start of window (Limited to simple 5 char cron expressions)
DURATION	INTEGER NOT NULL	Number of seconds in period (allows for windows to wrap over the day boundary or even over several days)
ALLOWED_OVERRUN	INTEGER	Number of seconds jobs already scheduled within window are allowed to run past the end of the window (zero indicates all incompleted jobs should be immediately terminated at the close of the window)
MAX_TASKS	INTEGER NOT NULL	The number of concurrent utilities allowed to be run during this time period. Overlapping windows should respect the lowest of all overlapping windows
SCHEDULED	CHAR(1)	Y/N: Indicates scheduled to admin scheduler

SYSAUTO.ENACTORREGISTRY (Table B-10): Look up to correlate Actions with registered Enactors capable of performing the specified action. Serves as a registry to correlate recommended actions with an enactor able to perform (and possibly predict CPU and elapsed time of) recommended action. This is not a list of all the possible recommended actions you could get, rather this is a list of all the recommended actions you can act on autonomically.

Table B-10 SYSAUTO.ENACTORREGISTRY

Column name	Data type	Description
ACTION	VARCHAR(32) NOT NULL	Action Tool is able to act on
ENACTOR	VARCHAR(128) NOT NULL	Stored Procedure to call in order to run the corresponding action

SYSAUTO.JOBS (Table B-11): Migration of DB2 Automation Tool Job Profile table into Tools Base. A Job Profile is the primary input to an Evaluator and optionally consists of the following:

- Object Profile
- Utility Profile
- Exceptions Profile
- Maintenance Window

Table B-11 SYSAUTO.JOBS

Column name	Data type	Description
NAME	VARCHAR(30) NOT NULL	Name of the job profile
TYPE	VARCHAR(8)	Type of job profile: {HAA, and so on}
CREATOR	CHAR(08) NOT NULL	Creator of the job profile; typically a TSO ID

Column name	Data type	Description
VERSION	SMALLINT NOT NULL	HAA/DLC VRM (version/release/modification): For example, V4R1M0 -> h'410' = x'019a'. Note, this could be the GGC/DLC VRM, if Change Accum generated the profile
PROF_SEL_TYPE	CHAR(01) NOT NULL	E/O/U/M (exception/object/utility/maintenance window) profile; the type of profile represented by this row in the JOBS table
PROF_SEL_ORDER	SMALLINT NOT NULL	1, 2, 3… Each profile type has an ordering. This is the same order that the profiles are processed during job generation
PROF_SEL_NAME	VARCHAR(30) NOT NULL	This is the name of the associated execution/object/utility/maintenance window profile. Similar to NAME, but NAME is for the job profile
PROF_SEL_CREATOR	CHAR(08) NOT NULL	Similar to CREATOR, but for the associated profile represented by this row in the JOBS table
UPDATED_USERID	CHAR(08) NOT NULL	The user ID that last updated the associated execution/object/utility/mw profile
UPDATED_TIMESTAMP	TIMESTAMP NOT NULL	The time stamp when the associated execution/object/utility/mw profile was last updated

SYSAUTO.OBJECT_PROFILE (Table B-12): Migration of DB2 Automation Tool Object Profile into the Tools Base. The primary table defines all profiles and has the profile metadata (name, creator, version, and so on). The related child table holding the profile rules depends on the Object Profile type.

Table B-12 SYSAUTO.OBJECT_PROFILE

Column name	Data type	Description
NAME	VARCHAR(30) NOT NULL	Name of the object profile
CREATOR	CHAR(08) NOT NULL	Creator of the object profile; typically a TSO ID
VERSION	SMALLINT NOT NULL	VRM (version/release/modification): For example, V4R1M0 -> h'410' = x'019a'
PROFILE_TYPE	VARCHAR(8) NOT NULL	Type indicates the syntax used and the related table used to define the profile. Currently, we only support LISTDEF, which indicates the rules are defined in SYSAUTO.OBJECT_LISTDEF and uses LISTDEF syntax
DESCRIPTION	VARCHAR(256)	Description of the Profile entered by the user
CREATE_TIMESTAMP	TIMESTAMP NOT NULL WITH DEFAULT	Creation Timestamp
UPDATE_USERID	CHAR(08) NOT NULL	The user ID that last updated the profile
UPDATE_TIMESTAMP	TIMESTAMP NOT NULL GENERATED BY DEFAULT FOR EACH ROW ON UPDATE	The time stamp when the profile was last updated

SYSAUTO.OBJECT_LISTDEF (Table B-13): If the Profile Type is LISTDEF, the rules for the object profile are defined in the related OBJECT_LISTDEF table. LISTDEF Object Profiles are basically a 1-1 table representation of the LISTDEF syntax familiar to any DBA that uses the IBM DB2 Utilities. For any profile, the rules consist of all rows with the PROFILE_NAME listed in OBJECT_SEQ order.

Table B-13 SYSAUTO.OBJECT_LISTDEF

Column name	Data type	Description
PROFILE_NAME	VARCHAR(30) NOT NULL	Name of the object profile. This is a foreign key to the metadata defined in the OBJECT_PROFILE table
OBJECT_SEQ	INTEGER NOT NULL	Specifies the sequence of INCLUDE and EXCLUDE clauses
RESULT_OBJECT_TYPE	CHAR(02) NOT NULL	Specifies whether to create a list of table spaces or index spaces: "TS": Specifies that the INCLUDE or EXCLUDE object expression is to create a list of related table spaces. Specify 'TABLESPACES' in the LISTDEF type-spec. "IS": Specifies that the INCLUDE or EXCLUDE object expression is to create a list of related index spaces. Specify 'INDEXSPACES' in the LISTDEF type-spec
OBJECT_TYPE	CHAR(02) NOT NULL	Specifies the object type in INCLUDE or EXCLUDE clauses: "DB": Specify DATABASE in the LISTDEF initial-object-spec. "TS": Specify TABLESPACE in the LISTDEF initial-object-spec. "TB": Specify TABLE in the LISTDEF initial-object-spec. "IS": Specify INDEXSPACE in the LISTDEF initial-object-spec. "IX": Specify INDEX in the LISTDEF initial-object-spec.
OBJECT_QUALIFIER	VARCHAR(128)	Specifies the qualifier of the object that is to be used for the initial search (initial-object-spec). Both OBJECT_QUALIFIER and OBJECT_NAME cannot be *.* or %.%. See Qualifier/Name table below for more information
OBJECT_NAME	VARCHAR(128)	Specifies the name of the object that is to be used for the initial search (initial-object-spec). Both OBJECT_QUALIFIER and OBJECT_NAME cannot be *.* or %.%. See Qualifier/Name table below for more information
OBJECT_PARTITION	INTEGER NOT NULL	Specifies a partition number. NULL: Specify LISTDEF option 'PARTLEVEL' 0 and up: Specify 'PARTLEVEL n', where n is the partition specified in OBJECT_PARTITION Note: If PROCESS_RI is Y in any of the sub-profiles, PARTLEVEL clause is not specified
EXCLUDE	CHAR(01) NOT NULL	Specifies whether to create an INCLUDE or EXCLUDE object expression: "I": Specify a LISTDEF INCLUDE clause "E": Specify a LISTDEF EXCLUDE clause

Column name	Data type	Description
PROCESS_CLONES	CHAR(01)	Specifies whether only table spaces and index spaces that contain cloned objects are to be returned in the INCLUDE or EXCLUDE clause: "N": Specify LISTDEF option 'CLONED NO' "Y": Specify LISTDEF option 'CLONED YES'
PROCESS_DEFINED	VARCHAR(03)	Specifies whether table spaces or index spaces with defined or undefined data sets are to be returned in the INCLUDED or EXCLUDE clause: "N" or "NO": Specify LISTDEF option 'DEFINED NO' "Y" or "YES": Specify LISTDEF option 'DEFINED YES' "ALL": Specify LISTDEF option 'DEFINED ALL'
PROCESS_RI	CHAR(01)	Specifies whether all objects that are referentially related to the object expression (PRIMARY KEY <--> FOREIGN KEY) are to be included in the list. "Y": Specify LISTDEF option 'RI'
PROCESS_AUX	VARCHAR(8)	Specifies an auxiliary indicator keyword to direct LISTDEF processing to follow auxiliary relationships to include related LOB or XML objects in the list: "ALL": Specify LISTDEF option 'ALL' "BASE": Specify LISTDEF option 'BASE' "LOB": Specify LISTDEF option 'LOB' "XML": Specify LISTDEF option 'XML'
PROCESS_HISTORY	CHAR(01)	Specifies that only history (versioning) objects are to be included in the resulting list clause. "Y": Specify LISTDEF option 'HISTORY'
UPDATE_USERID	CHAR(08) NOT NULL	The user ID that last updated the profile
UPDATE_TIMESTAMP	TIMESTAMP NOT NULL GENERATED BY DEFAULT FOR EACH ROW ON UPDATE	The time stamp when the profile was last updated

SYSAUTO.UTILITY (Table B-14): Migration of DB2 Automation Tool Utility Profile into the Tools Base. A Utility Profile represents all the options required for a specific utility to perform its evaluation. The specific information is stored in a proprietary format in the DATA column.

Table B-14 SYSAUTO.UTILITY

Column name	Data type	Description
NAME	VARCHAR(30) NOT NULL	Name of the job profile
CREATOR	CHAR(08) NOT NULL	Creator of the evaluation profile; typically a TSO ID
LAST_UPDATED	TIMESTAMP NOT NULL	Timestamp of the last time it was created or updated
TYPE	VARCHAR(30) NOT NULL	Identifies the Utility (long list of values)
VERSION	SMALLINT NOT NULL	Version of the Evaluation Tool this profile relates to
SEQUENCE	INTEGER	SEQUENCE is only nonzero for continuation rows.
DATA	VARCHAR(4000)	Internal representations specific to the utility.

SYSAUTO.EVALUATION (Table B-15): Migration of DB2 Automation Tool Exception Profile into the Tools Base with added Type fields and generalized for any evaluation. An Evaluation Profile (an exception profile to DB2 Automation Tool) represents all the options required for a specific evaluator to perform its evaluation. The specific information is stored in a proprietary format in the DATA column.

Table B-15 SYSAUTO.EVALUATION

Column name	Data type	Description
NAME	VARCHAR(30) NOT NULL	Name of the job profile
CREATOR	CHAR(08) NOT NULL	Creator of the evaluation profile; typically a TSO ID
LAST_UPDATED	TIMESTAMP NOT NULL	Timestamp of the last time it was created or updated
TYPE	VARCHAR(30) NOT NULL	Name of the Tool this profile represents
VERSION	SMALLINT NOT NULL	Version of the Evaluation Tool this profile relates to
SEQUENCE	INTEGER	SEQUENCE is only nonzero for continuation rows
DATA	VARCHAR(4000)	Internal representations specific to the tool. For Automation Tool, this is the Exception Profile

SYSAUTO.UTILITYRUNS_HISTORY (Table B-16): Maintains a record of all utilities that have been run. Populated by DB2 Utilities Enhancement Tool and eventually the utilities themselves.

Table B-16 SYSAUTO.UTILITYRUNS_HISTORY

Column name	Data type	Description
UTILITYRUN_ID	BIGINT NOT NULL GENERATED ALWAYS AS IDENTITY	Primary Key generated for each utility run
UTILITY_ID	CHAR(16)	Utility ID used when submitting the utility
AUTH_ID	CHAR(8)	
JOBNAME	CHAR(8)	Jobname used when running the utility
STEPNAME	CHAR(8)	Stepname under which this utility ran
UTILITY_NAME	CHAR(8)	Name of Utility
TARGET_TYPE	CHAR(2)	Indicates the type of object this action is for: TS = Tablespace IX = Indexspace DB = Database for pageset level utilities
TARGET_QUALIFIER	VARCHAR(128) NOT NULL	The qualifier name of the DB2 object this applies to, that is, the database name for pageset level utilities or schema name for table level
TARGET_OBJECT	VARCHAR(128) NOT NULL	The name of the DB2 object this applies to, that is, the table space name
TARGET_PARTITION	SMALLINT NOT NULL	The partition number of the DB2 object this applies to. Zero if alert applies to all partitions, or if the object is not partitioned
START_TS	TIMESTAMP(12)	Timestamp for the start of the utility in system time

Column name	Data type	Description
END_TS	TIMESTAMP(12)	Timestamp for the end of the utility in system time
RECORDS	BIGINT	Number of pages/records/rows/entries processed depending on the utility type
CPU	INTEGER	Total CPU time of utility run
ZIIP	INTEGER	Total IBM z Systems Integrated Information Processor (zIIP) time of utility run (subset of CPU time)
SORT_CPU	INTEGER	Total CPU time used during sort operation
SORT_ZIIP	INTEGER	Total zIIP time used during sort operation (subset of SORT_CPU time)
RETURN_CODE	SMALLINT	Utility Return Code
OUTPUT	CLOB(2M)	Sysprint output. Nullable
ROWID	ROWID	ROWID for CLOB column

SYSAUTO.SYSTABLESPACESTATS_HIST: Snapshot data of SYSIBM.SYSTABLESPACESTATS. A copy of the row data with a time stamp populated by the RTS_SNAPSHOT Stored Procedure.

SYSAUTO.SYSINDEXSPACESTATS_HIST: Snapshot data of SYSIBM.SYSINDEXSPACESTATS. A copy of the row data with a time stamp populated by the RTS_SNAPSHOT Stored Procedure.

Redbooks

Modernize Your IBM DB2 for IBM z/OS Maintenance with Utility Autonomics

(0.2”spine)
0.17”<->0.473”
90<->249 pages

SG24-8304-00

ISBN 0738440892

Printed in U.S.A.

Get connected