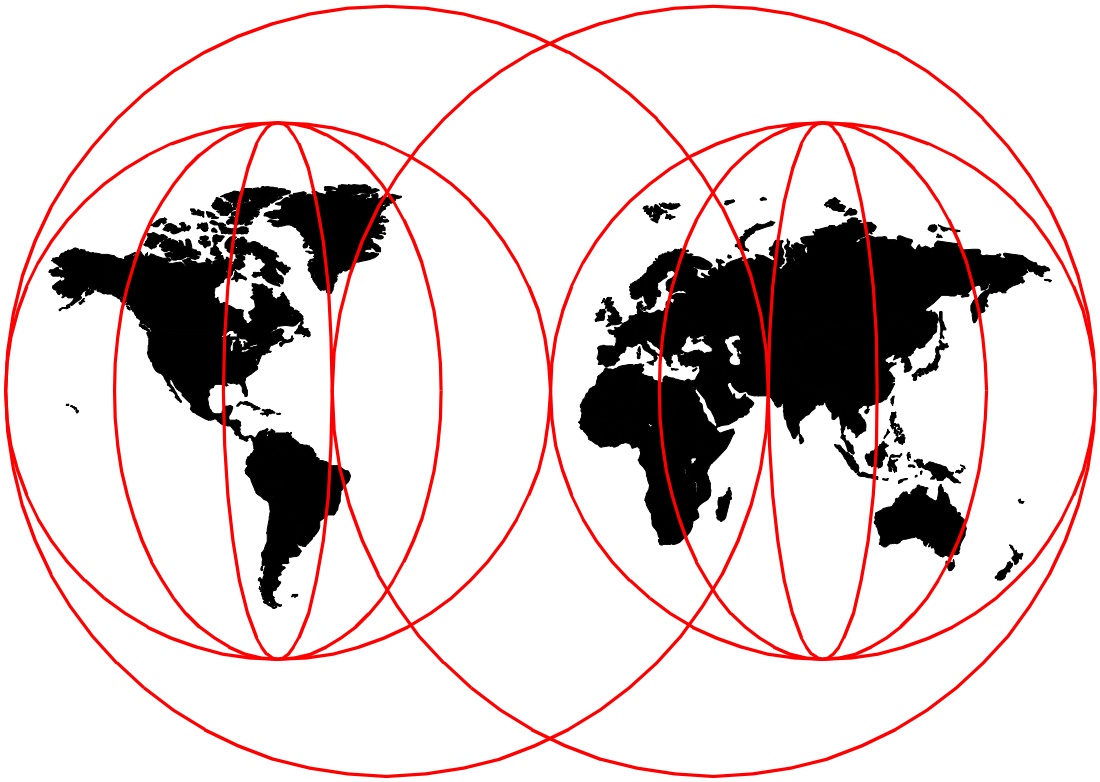
Planning for a Migration of PeopleSoft 7.5 from Oracle/UNIX to DB2 for OS/390

Viviane Anavi-Chaput, Kathryn Arrell, Jan Baisden, Richard Corrihons,

Dawn Fallon, Lee Siegmund, Nora Sokolof



International Technical Support Organization

www.redbooks.ibm.com

SG24-5648-00

International Technical Support Organization

SG24-5648-00

Planning for a Migration of PeopleSoft 7.5 from Oracle/UNIX to DB2 for OS/390

May 2000

Take Note!

Before using this information and the product it supports, be sure to read the general information in Appendix G, “Special Notices” on page 121.

First Edition (May 2000)

This edition applies to PeopleSoft Applications Release 7.5 with DB2 V6.1 and OS/390 V2R6.

Comments may be addressed to:
IBM Corporation, International Technical Support Organization
Dept. HYJ Mail Station P099
2455 South Road
Poughkeepsie, NY 12601-5400

When you send information to IBM, you grant IBM a non-exclusive right to use or distribute the information in any way it believes appropriate without incurring any obligation to you.

Contents

Figures

Tables

Preface

This redbook discusses the process to migrate PeopleSoft Applications 7.5 on Oracle on UNIX to PeopleSoft Applications 7.5 on OS/390 with DB2. We migrated the PeopleSoft Tools and the Human Resources (HR) Application.

The redbook is intended to help you plan the migration process. The information is based on installation experiences gained at the ITSO in Poughkeepsie and with customers who have completed this process.

This book will be especially useful for those who are migrating to PeopleSoft Applications on OS/390 for the first time.

The team that wrote this redbook

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

Viviane Anavi-Chaput is a DB2 specialist in the International Technical Support Organization, Poughkeepsie Center. Before joining the ITSO, Viviane worked in BI DB2 support in France.

Kathryn Arrell is an ERP Specialist with the International Technical Support Organization, Poughkeepsie Center. Before joining ITSO, Kathryn worked in RS/6000 Marketing in IBM Canada.

Jan Baisden is a Senior Market Support Representative at the International Technical Support Organization, Poughkeepsie Center. He concentrates on ERP Solutions, particularly on AIX support of S/390 uses in those solutions.

Richard Corrihons is a S/390 DB2 Specialist with the IBM S/390 Montpellier Center.

Dawn Fallon is an S/390 PeopleSoft Solutions Sales Specialist with the ERP Solutions Sales Organization, North America.

Nora Sokolof is an Information Technology Specialist in the Relational Database Management Group of the Software Migration Project Office.

Lee Siegmund is a Consulting Marketing Support Representative from the DB2 Relational Support Unit of the Dallas Systems Center. He has worked at IBM for 29 years. For the past 15 years he has supported DB2, starting with Version 1 Release 1 to the current Version 6.

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

Bob Haimowitz
Vasilis Karras
International Technical Support Organization, Poughkeepsie Center

J Kevin Lowe
Casey Townsend
IBM Global Services

Chris Chung
Mike Curtis
IBM PeopleSoft Competency Center

Ron Lack
Stephen J. Madigan
PeopleSoft Consulting

Som Sundararajan
PeopleSoft Global Support Center

Ed Presz
PeopleTools Development

Comments welcome

Your comments are important to us!

We want our redbooks to be as helpful as possible. Please send us your comments about this or other redbooks in one of the following ways:

- Fax the evaluation form found in “IBM Redbooks review” on page 133 to the fax number shown on the form.
- Use the online evaluation form found at `http://www.redbooks.ibm.com/`
- Send your comments in an internet note to `redbook@us.ibm.com`

Chapter 1. Overview of the migration process

This redbook describes experiences of migrating PeopleSoft Applications 7.5 from a Oracle database on a UNIX platform to a DB2 Database on OS/390. Some of the reasons customers consider migrating to OS/390 are:

- Scalability
- Reliability
- Systems management tools on S/390
- Existing in-house skills

The steps to install DB2 and PeopleSoft 7.5 on OS/390 are not covered here in detail as they are described in *PeopleTools 7.5: Installation and Administration of DB2 for OS/390* and *Planning to Install PeopleSoft with DB2 for OS/390.* These books are needed to do the setup of DB2 and PeopleSoft on OS/390. This book mainly covers the migration of the database.

This chapter provides a checklist of points to consider in the planning process.

1.1 Migrating a PeopleSoft Oracle database from UNIX to DB2 for OS/390

Most PeopleSoft installations have done some degree of customization of their applications. The customization can be of the following types:

- Modifications to DDL
- Use of stored procedures and triggers
- Use of non-ANSI SQL
- Use of non-ANSI data types

The two methods of migration we investigated depend on the amount of DDL customization that has been done. The two methods are:

- Minimal DDL customizations have been done.

  In this case, if the changes are few and are known, you can install a new vanilla PeopleSoft system on the S/390, make the same customizations (change the DDL), and then export and import only the data.

- Major DDL customizations have been done.

  In this case, you can extract the DDL that describe the source database and use this to set up the DB2 system on the S/390. Then when you export and import the data you will have a mirror image of the source system in the target system on DB2 for OS/390. This is the method that we used in our migration.

  There are still other customizations that must be handled, such as use of stored procedures, use on non-ANSI SQL, and so on.

The process to migrate the applications includes the following steps:

1. Planning

   - Analyze customization and platform differences.
   - Develop an infrastructure plan.
   - Design the layout of the databases, tablespaces, and tables.

- Size the DASD needed for the tablespaces and indexes for the new subsystem.
- Do a capacity plan for the CPU size.
- Analyze and choose migration tools.
- Analyze the fixes for present and future environments.
- Determine which fixes are required.
- Determine naming conventions for the DB2 environment.
- If possible, freeze development during migration.
- Use existing test scenarios (if possible) for functional tests.
- Determine how to do exception cleanup on the source PeopleSoft database (using DDDAUDIT and SYSAUDIT to ensure the database is in sync with the PeopleTools tables).
- Plan for the installation to be certified by PeopleSoft.
- Develop plans for:
  - Print migration
  - Batch migration
  - Backup and recovery
  - Security on OS/390 and DB2
  - Testing
  - Migration of data
  - Other operational activities:
    - Change management
    - Database monitoring
    - Performance monitoring

2. Planning and executing the test migration plan

- Set up OS/390 and DB2 for Peoplesoft.
- Apply all software fixes.
- Set up the DB2 subsystem, parameters, logs, VSAM files, and so on.
- Install and set up DB2 Connect and DDF.
- Create user IDs and set up PeopleSoft security on DB2.
- Establish connectivity to the database with DB2 Connect.
- Install PeopleSoft for OS/390

  Note: This step does a PeopleSoft installation up to the point where the demo data is loaded through DataMover.
- Export the source of PeopleSoft data using DataMover for the customized tables.
- Extract the customized DDL from the exported file.
- Use the `ftp` command to move the DDL to OS/390.

- Edit the DDL to make it match what you want to create on DB2 and to optimize for size and performance (consider partitioning, locks, and PCTFREE parameters, determine the lock size and so on).
- Export all the source of PeopleSoft data without the DDL.
- Import the data to DB2 using DataMover.
- Run DB2 utilities such as Reorg, Runstats, and backups.
- Run PeopleSoft Audit reports (DDDAUDIT and SYSAUDIT).
- Run the integrity tests for all customized programs and SQRs (especially any non-ANSI SQL for DB2). Does it run the same way with DB2 as it did with Oracle.

  Note: ASCII-to-EBCDIC conversions can cause problems with special characters and with sort routines. Be sure the test scenarios cover these areas. In particular, in one case, the SQR conversion required some special character conversion.

  DataMover handles the ASCII-to-EBCDIC conversion. It is important to check the locale parameter of DataMover to ensure the same locale is used for the source and the target database.
- Test the functionality of the application.
- Develop an incident tracking process.
- Test printing processes.
- Stress-test the new platform.

3. Migrating the production databases and other instances
   - Obtain functional acceptance that the functions run correctly on the new platform.
   - Ensure that the two platforms are in sync for fixes, customization, and development.
   - Migrate the database structure.
   - Migrate the data to DB2.

     If the database is large you may want to freeze changes to the source database, migrate the database structure (DDL) one weekend, complete the DB2 setup, and then migrate the data the following weekend.
4. Post-migration tasks
   - Database performance monitoring and tuning
   - Database administration procedures
   - Database backup and recovery procedures

1.2 Project planning

You will need a project manager to develop a migration plan that should include activities such as:

- Determine the migration scenario.
- List all the migration tasks.
- Develop infrastructure sustaining processes.

- Determine the training requirements.
- Determine the resources that will be required (HW,SW and people).
- Prepare a target environment.
- Test the migration scenario.
- Execute the migration plan.

You should have access to a PeopleSoft Consultant who will be able to obtain the latest information from migration experiences.

Do not attempt to make too many changes in one step. Migrate a stable release without upgrades.

1.2.1 Planning process

The planning process should include:

- The environments--test, development, production
- Security strategy for RACF, DB2 and PeopleSoft user IDs
- Customization of source applications
- Backup
- Preparation
- Compare/report/copy
- Convert data
- Batch conversion
- SQL and SQR testing

1.2.2 Migration considerations

You will have to ensure that your applications will migrate without problems by checking such things as:

- Network connectivity to application server
- Accurate sizing for S/390 and DB2 for CPU processor, storage (central and expanded memory) and DASD

  A sizing should be done with the IBM PeopleSoft Competency Center.
- Amount of data to migrate
- Amount of customization for online and batch
- Archiving of historical data to reduce the amount of data to migrate
- Parallel testing
- Use of stored procedures, triggers, non-ANSI SQL
- Conversion of UNIX scripts to JCL or CLISTs
- Modifications that have been made to PS_SQLSTMT_TBL
- Names for database objects

  Oracle allows longer names for database objects than DB2 for OS/390 does. Oracle's limit is 30 characters, while DB2'slimit is 18 characters. So, for any customized Oracle objects, the names will need to be shortened for DB2.

- Integration with other systems--either that provide data to PeopleSoft or that PeopleSoft provides data to
- Use of development, test, and training systems

1.3 Comparing Oracle terminology to DB2

The terminology used by DB2 and Oracle can be confusing when migrating from one database to the other. See Appendix A, “Oracle8 (UNIX) vs DB2 UDB for OS/390 V6.1 terminology” on page 81 for a table that lists some common terms and their meanings for each database.

1.4 Reviewing the checklist

Appendix B, “PeopleSoft migration questionnaire” on page 87, contains a questionnaire that should be completed early in the planning process to determine what extra steps need to be included in the migration plan to handle any customizations that have been made to the PeopleSoft application and database.

Chapter 2. Overview of the system we used for our tests

This chapter provides details about the systems we used and the steps we followed.

Our source system was an RS/6000 with AIX running Oracle8. The target system was an OS/390 system with DB2V6.1 as shown in Figure 1.

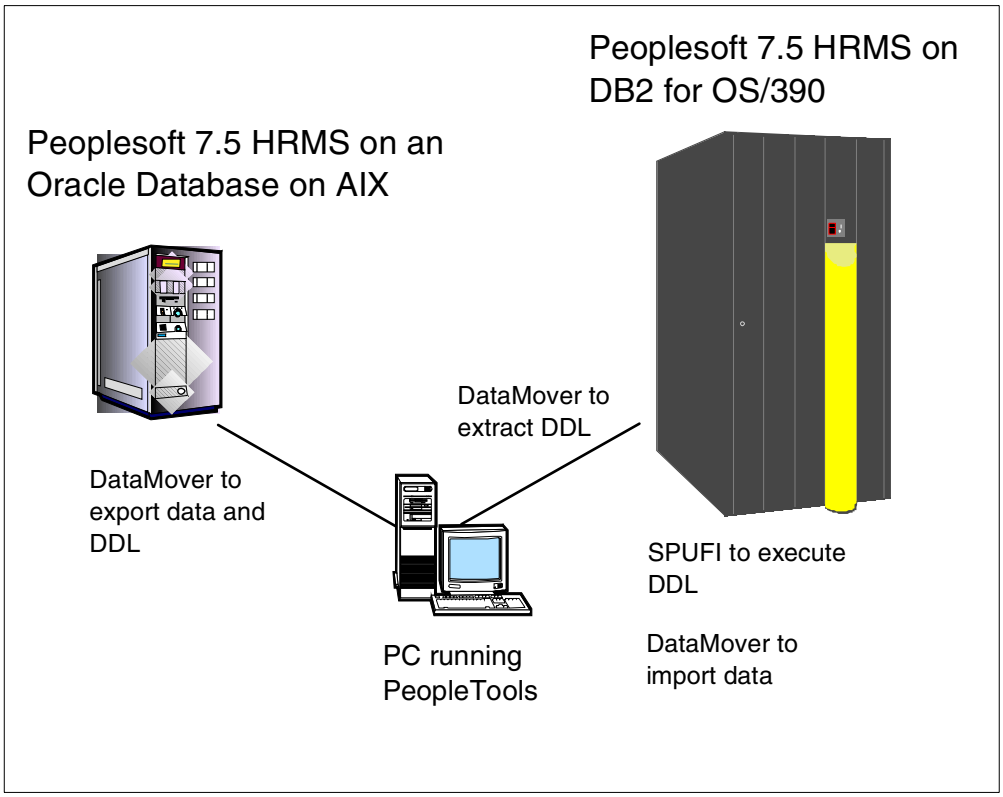


Figure 1. Systems used in the migration process

The source system contained a PeopleSoft 7.5 instance that we migrated to DB2 for OS/390.

2.1 Source system

The system we used as the source database consisted of the following:

- AIX 4.3.2
- Oracle 8.0.5.1
- PeopleSoft 7.5 HR application
- A 12 GB database we used for the migration tests

2.2 Target system

The system we used to migrate to consisted of the following:

- OS/390 V2R6 with TCP/IP (OS/390 eNetwork Communication Server TCP/IP V2R6
- DB2 UDB for OS/390 V6.1
- DB2 Connect V6 with fixpack 6.1.3
- 12 GB of DASD for the migrated database

Refer to the PeopleSoft Customer Connection Web page for the latest levels of software that are required. Chose **Customer Connection** at:
`http://www/peoplesoft.com`

2.3 Steps we followed for our test migration

Once we had the source database set up on the AIX system and a working DB2 database on the OS/390, we completed the following steps, which are described in more detail in the next several chapters:

- Set up a PC with PeopleTools establishing connectivity with Net8 to the Oracle database and DB2 Connect to the DB2 for OS/390 database.
- Set up DB2 and DDF on S/390.
- Installed PeopleSoft 7.5 code up to the point where the database is loaded.
- Extracted sizing information from the source database to determine how much DASD we would need for the target database.
- Analyzed the customization on the source database.

  This was not really necessary in our testing, as we used the extracted DDL to create the DB2 tables so the target DDL was equal to the source DDL.
- Ran the export DataMover script.
- Ran a special script to extract the DB2 DDL to create the tables on DB2 for OS/390.
- Created and executed the DDL to create the databases and tablespaces on DB2.
- Executed the extracted DDL to create the tables and indexes on DB2.
- Ran the import DataMover script.
- Validated that the target database was equal to the source database.

2.4 Other considerations

This setup was to test the migration process. It did not cover other aspects, such as handling of stored procedures or triggers. This is described in Chapter 7, “Handling the exceptions in the migration process” on page 35.

We only migrated one instance. Normally you would have to migrate or set up several instances such as production, test, and development.

Not all the steps are covered in detail because they are described in *PeopleTools 7.5: Installation and Administration of DB2 for OS/390* and *Planning to Install PeopleSoft with DB2 for OS/390.* These books will be needed to do the setup of DB2 and PeopleSoft on OS/390. This book mainly covers the migration of the database.

Chapter 3. Create a PeopleTools environment

This chapter describes the steps we took to connect the workstation using PeopleTools to the Oracle source database and the DB2 target database.

This book does not describe the installation of Oracle on AIX or DB2 for OS/390.

3.1 Set up access to the source database

In our case, the source database was an Oracle HRMS 7.5 database running on AIX. We had to set up the workstation that we were using for DataMover to access the Oracle database.

The steps to do this were:

- Install the Oracle8 Client.
- Customize the tnsnames.ora file to point to the data bases we had on AIX. There were two HR75 databases: a customized database with data used in test runs, and HR75DMO, which was a vanilla installation of PeopleSoft on AIX. Following are the lines we added to the tnsnames file in the directory D:\oracle\network\:

```
HR75.world =
  (DESCRIPTION =
    (ADDRESS_LIST =
        (ADDRESS =
          (PROTOCOL = TCP)
          (Host = 9.12.0.93)
          (Port = 1521)
                  )
    )
    (CONNECT_DATA = (SID = HR75)
    )
  )
HR75DMO.world =
  (DESCRIPTION =
    (ADDRESS_LIST =
        (ADDRESS =
          (PROTOCOL = TCP)
          (Host = 9.12.0.93)
          (Port = 1521)
                  )
    )
    (CONNECT_DATA = (SID = HR75)
    )
  )
```

- To ensure that the access to the database is functional, test the connection with the `svrmgrl` command.

  Go to a DOS screen and the directory D:\oracle and enter:

  ```
  sqlplus system/manager@HR75
  ```

  You should get a message that says you are connected to the database. This is the path that PeopleTools will use, so this must be functioning before you can run the export.

- Set up the PeopleTools configuration manager to access the Oracle database:
  - Select **Start -> Programs -> PeopleSoft 7.5 -> Configuration Manager.**
  - For database type choose **Oracle**.
  - For database name choose HR75 (this must correspond to the entry in the tnsnames.ora file).
  - For the Operator ID choose **PS**.

  Figure 2 shows the entries we made.

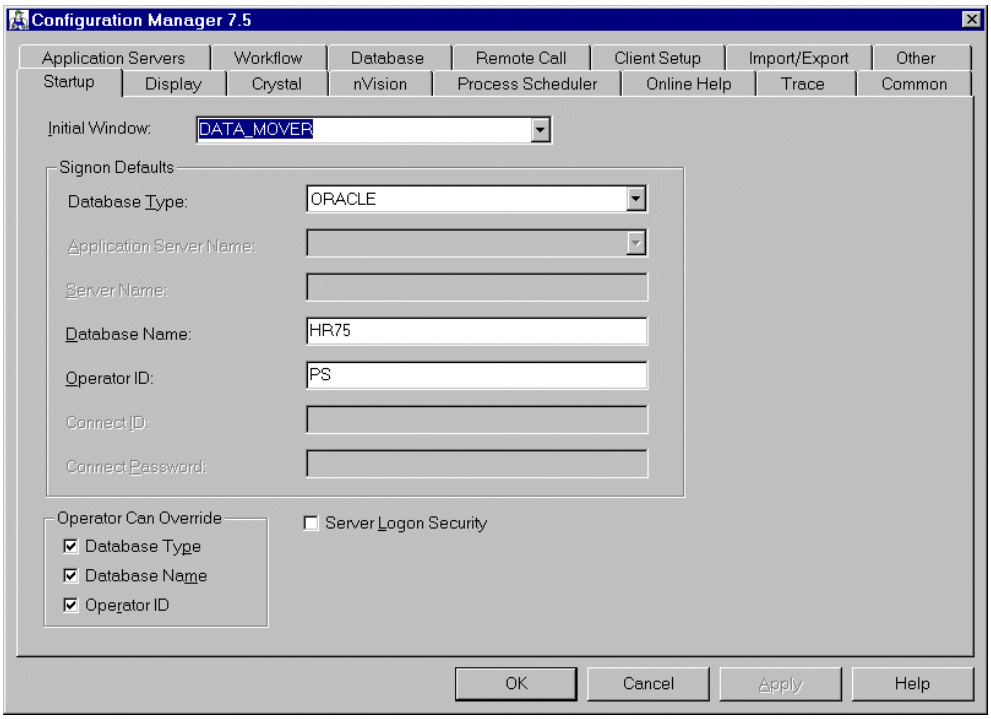


Figure 2. Startup parameters for Oracle

To set up the directories for the input, output and log files, go to the Common tab and make the entries as shown in Figure 3 on page 11.

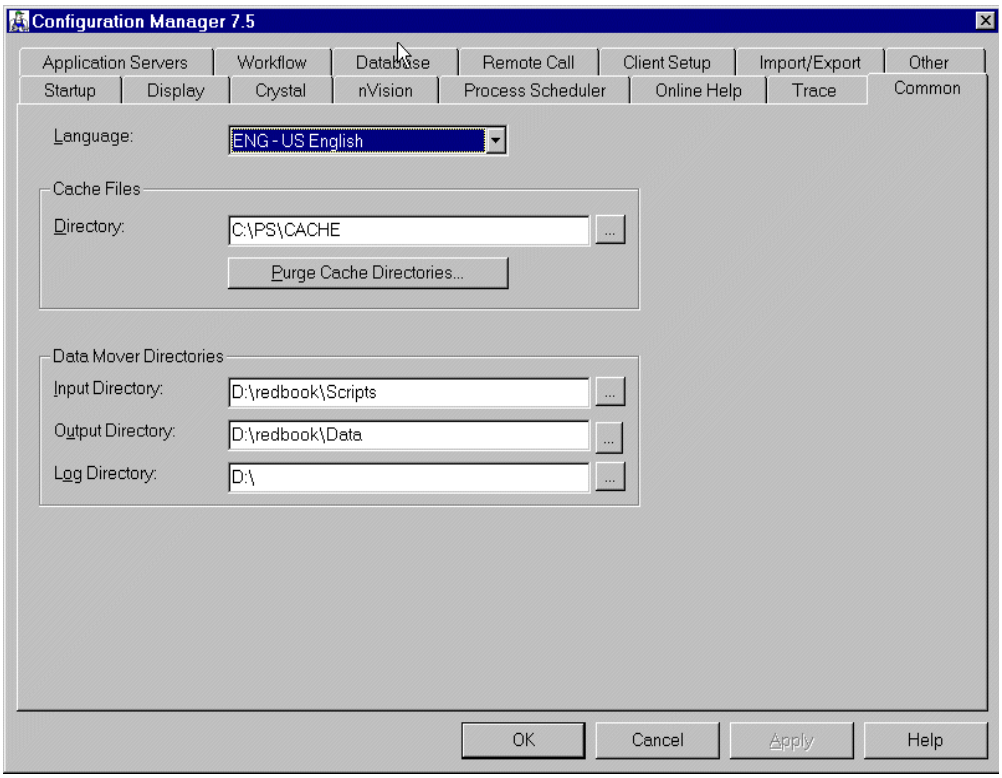


Figure 3. DataMover directories for Oracle

Then, to sign on to DataMover, select **Start -> Programs -> PeopleSoft 7.5 -> DataMover**, and choose **Oracle** for connection type, **HR75** for the database name (this must match the entry in tnsnames.ora) and **PS** for the Operator ID, as shown in Figure 4 on page 12.

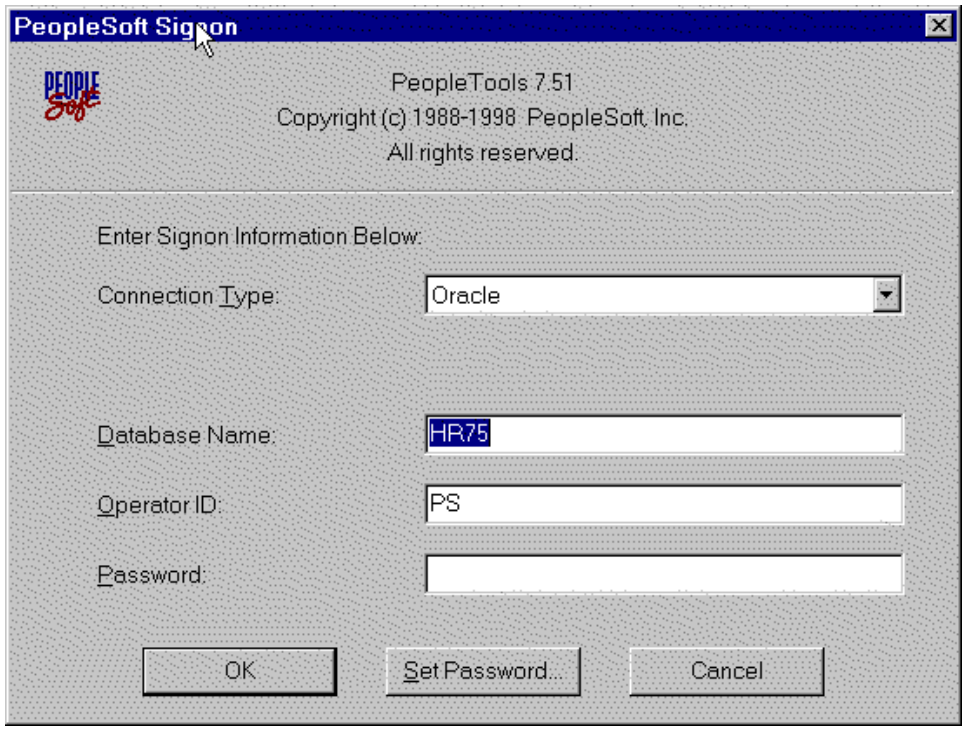


Figure 4. PeopleTools Signon screen for Oracle

Once these steps are completed, the export can be run as described in Chapter 3, “Create a PeopleTools environment” on page 9.

When you select **OK,** you will be presented with the following window:

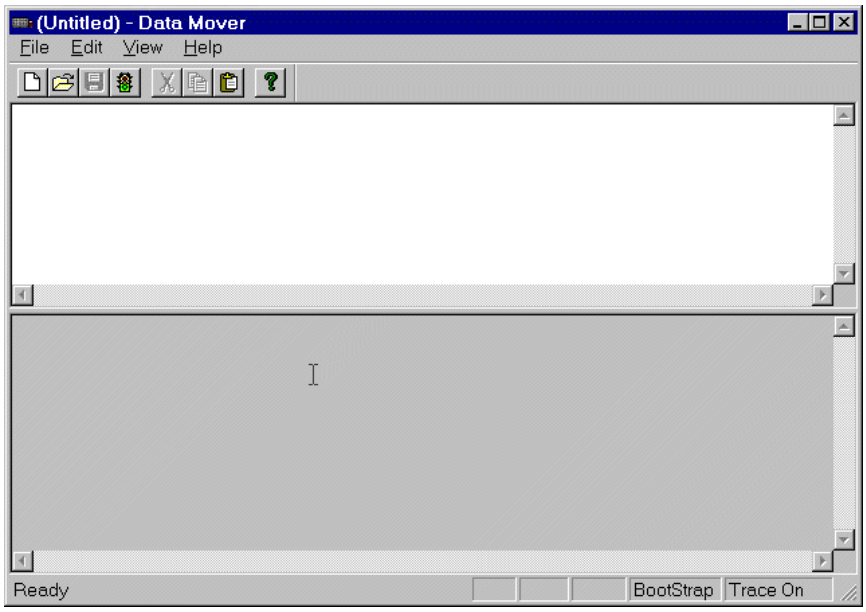


Figure 5. DataMover window

Select **File -> Open**, choose the directory, and you will be presented with the scripts you can execute, as shown in Figure 6 on page 13.

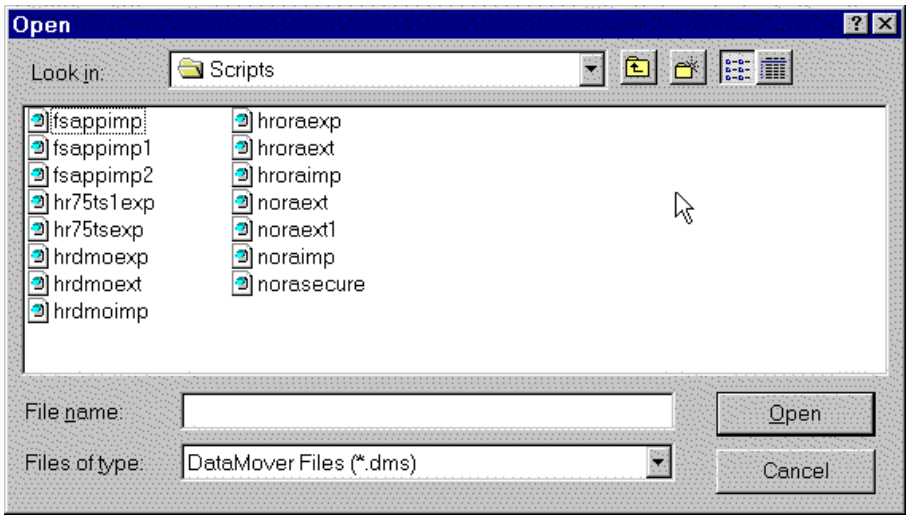


Figure 6. List of DataMover scripts we used

Now you are ready to run the scripts as described in Chapter 6, “Executing the DataMover scripts” on page 27.

3.2 Setting up the connection to DB2 UDB for OS/390 V6.1

Note: We only used one NT Workstation for DataMover for Oracle and DB2. You could use two systems and set up both at the same time--one with Oracle, and the other one with DB2.

In our case, this could only be done after we completed running the export script to complete the work with the Oracle database. When you are ready to run the extract of the DDL from the exported file, you first need to configure PeopleTools to connect to DB2 for OS/390.

The steps to do this are:

- Install DB2 Connect.
- Customize to set up the connection to DB2 on OS/390. This is described in Appendix C, “Setting up DB2 Connect” on page 91.
- Set up the PeopleSoft Configuration Manager to connect to DB2.
  - Select **Start -> Programs -> PeopleSoft 7.5 -> Configuration Manager.**
  - For database type, choose **DB2ODBC.**
  - For database name, we choose **DB2DMO** (this must correspond to the entry in the DB2 Connect customization).
  - For the Operator ID, choose **PS**.
  - For Connect ID, we chose **PSOFT1** and entered the password.

This is shown in Figure 7 on page 14.

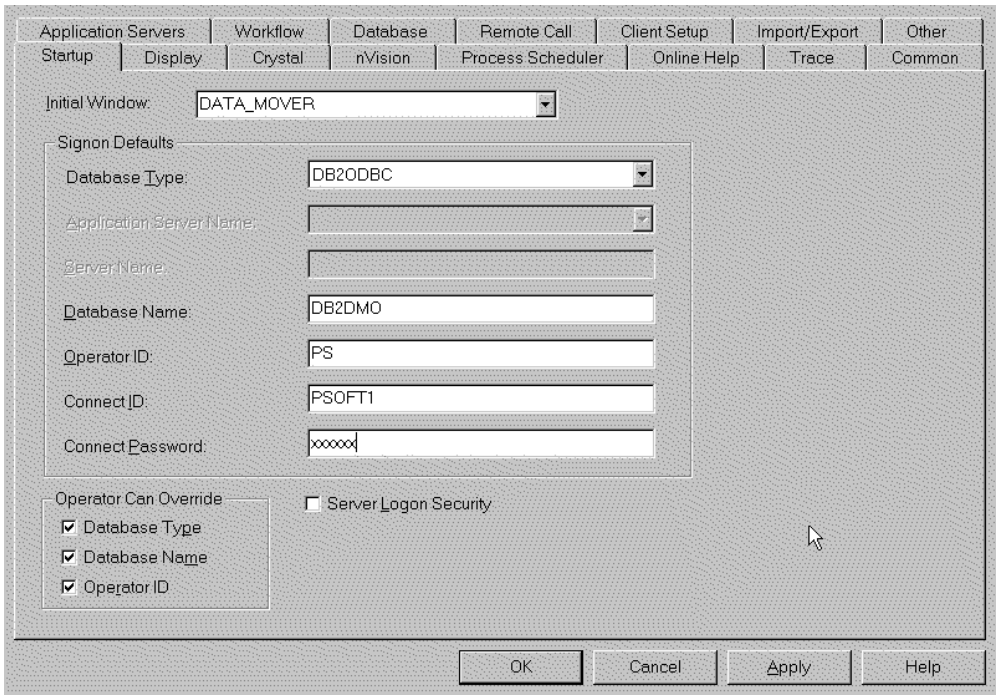


Figure 7. Setup parameters for DataMover for DB2

3.3 Set up the PeopleTools common options

To set up the directories that DataMover uses, go to the Common tab and enter the directories you will use. See Figure 8 on page 15.

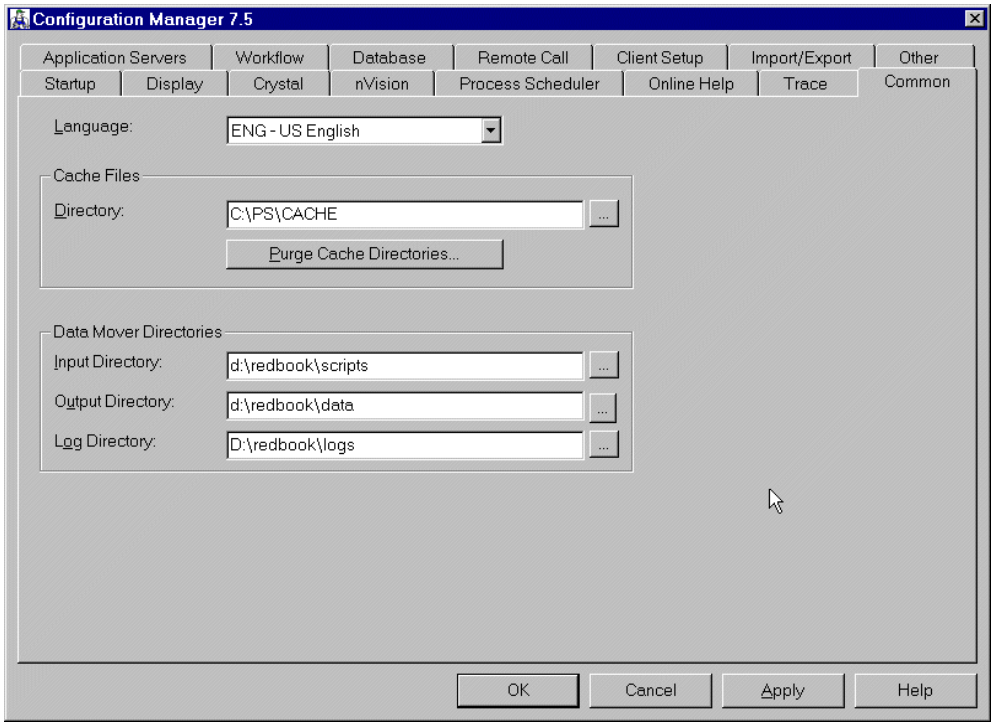


Figure 8. Directories for DataMover for DB2

Now when you start DataMover, you will have the Connection type as DB2ODBC, DB2DMO, the Operator ID and Connect ID filled in as shown in Figure 9 on page 16.

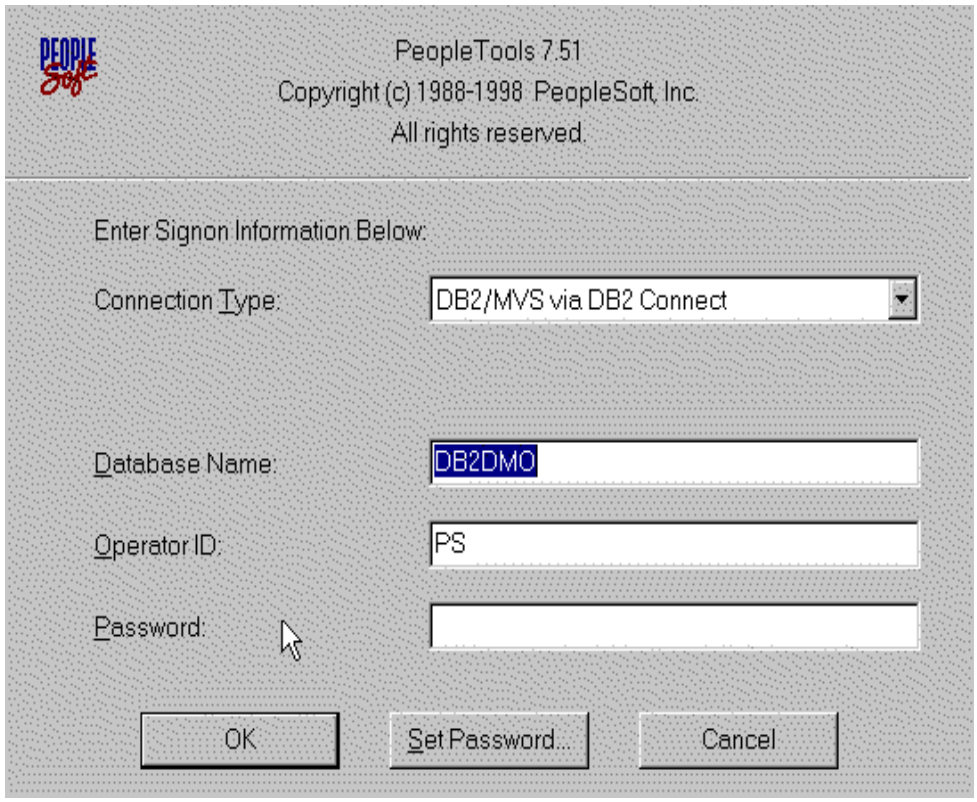


Figure 9. DataMover for DB2

3.4 DB2 security for PeopleSoft

Four types of IDs must be planned for:

1. Operator ID

   This is a client ID that may or may not exist on TSO depending on whether the Connect ID option is being used. This value must be entered into the PSOPRDEFN table.

   If the Connect ID option of PS security is not used, then each Operator ID must also exist as a TSO ID and each of these TSO IDs must have SELECT privileges on the PS.PSDBOWNER, PSLOCK and PSOPRDEFN tables.

   These IDs should not be granted DBADM (and never SYSADM) authority and are not used to access PS tables. This ID is used to connect to the subsystem and verifies the operator's authority to connect to the PS system by checking the PSOPRDEFN table. The Operator ID is also used to look up the Access ID in the PSOPRDEFN table and the Owner ID that is stored in the PS.PSDBOWNER table.

   Once this has been done, the Operator ID disconnects from the subsystem and the Access ID connects to the system and is used to perform all data access.

   If the Connect ID option is in use, then the Operator ID is not a TSO ID. Instead, the Operator ID maps to the Connect ID as defined in the PS Configuration Panel on the client workstation. However, even if the Connect ID

option is used, each Operator ID must still exist as a value in the PSOPRDEFN table.

2. Connect ID (optional)

   This is a TSO ID that has SELECT privileges on the PS.PSDBOWNER, PSLOCK and PSOPRDEFN tables. This ID should not have DBADM authority. If this option is used, all Operator IDs use the Connect ID to connect to the subsystem.

3. Owner ID

   This is a TSO ID or a RACF group (primary authid or secondary authid). This value is stored in the PS.PSDBOWNER table and is the owner of the PS tables, indexes and views. This ID should have DBADM authority for each database that is part of a single production PeopleSoft system.

   If the Owner ID is a primary ID, it is the same as the Access ID. If the Owner ID is a secondary ID (a RACF group), then the Access ID must belong to the RACF group.

4. Access ID

   This is a TSO ID that maps to the Operator ID in the PSOPRDEFN table. This ID is used to access data. If the Owner ID is a generic ID (secondary authid/RACF group) then a different Access ID may be used for each Operator ID in the PSOPRDEFN table, but they all must belong to the Owner ID (RACF group).

   The Access ID does not need to be granted the DBADM privilege because it will inherently have that privilege since it is part of the RACF group that has been granted DBADM. The advantage of this is to monitor threads of activity for each operator that is performing work.

   If all Operator IDs map to the same Access ID in the PSOPRDEFN table, then all work will appear as though the same operator were performing it.

   If the Access ID is different from the Owner ID, then before accessing data the SET CURRENT SQLID = Ownerid statement is issued first.

   If the Access ID is the same as the Owner ID, then the Owner ID is not a RACF group and a SET CURRENT SQLID = Ownerid statement is not issued. In this case, the Access ID must have DBADM authority granted on each database that makes up a single PS HR production system.

Chapter 4. Analyze the customizaton on the source system

This chapter describes the points to analyze before you attempt the migration. The objective is to understand the source database and customization of data and programs so that a complete migration plan can be established.

4.1 Database considerations

The points to consider are:

- Has database customization been done by:
  - Adding columns
  - Changing column size
  - Adding tables
  - Using special characters in column names
  - Adding indexes
  - Adding Views
- Have database changes been made outside of Application Designer?

  This would mean the changes are not reflected in the data dictionary.
- Have any of the following been used:
  - Stored procedures
  - Triggers
  - Non-ANSI SQL

These are the types of things that will require manual intervention after the import of the data has been completed.

As described previously, most PeopleSoft installations have done some degree of customization of the applications. The two methods of migration we investigated depend on the amount of customization that has been done. The two methods are:

- Minimal DDL customization has been done.

  In this case, if the changes are few and known, you can install a new vanilla PeopleSoft system on the S/390, make the same customizations (change the DDL) and then export and import only the data.
- Major DDL customizations have been done.

  In this case, you can extract the DDL that describe the source database and use this to set up the DB2 system on the S/390. Then when you export and import the data you will have a mirror image of the source system in the target system on DB2 on OS/390.

  This is the method we used in our migration, so a details analysis was not required.

4.2 Application considerations

The points to understand are:

- What level of PeopleSoft Applications are being migrated.

  We recommend that you do not try to upgrade the level of PeopleSoft Applications at the same time that you are migrating the database to DB2 on OS/390.
- What customizations have been made to the application code?
- What vendor tools are being used to manage the environment?
- Have any platform-specific changes been made to SQL in SQR or COBOL programs?
- Have other languages been used, such as PL/1 or C?
- Have triggers, stored procedures or packages been used?

4.3 Environments to be migrated

There may exist several environments for PeopleSoft Applications, such as production, test, development, training, and so on. The following facts must be ascertained:

- How many environments are in use today?
- How many databases and tablespaces in each environment?
- Are some environments clones of others?

A migration strategy and plan must be developed for each environment.

4.4 Interfaces to other applications

An analysis must be done of the systems that feed the PeopleSoft Applications and those that the PeopleSoft system feeds data to. For more information, see Appendix B, “PeopleSoft migration questionnaire” on page 87.

Chapter 5. Size and create the PeopleSoft database on OS/390

In order to properly size and allocate OS/390 data sets for the PeopleSoft Applications, we first determined the size of the existing AIX files for each tablespace. The following SQL query shows the size of the 27 files in the AIX database.

```
>select tablespace_name, file_name, bytes from dba_data_files order by 1 asc ;
```

TABLESPACE_NAME	FILE_NAME	BYTES
BNAPP	/erp/hrdemo/HR75/bnapp.dbf	9437184
BNLARGE	/erp/hrdemo/HR75/bnlarge.dbf	7340032
FSAPP	/erp/hrdemo/HR75/fsapp.dbf	3145728
GIAPP	/erp/hrdemo/HR75/giapp.dbf	3145728
GPAPP	/erp/hrdemo/HR75/gpapp.dbf	8388608
HRAPP	/erp/hrdemo/HR75/hrapp.dbf	23068672
HRLARGE	/erp/hrdemo/HR75/hrlarge.dbf	5242880
HTAPP	/erp/hrdemo/HR75/htapp.dbf	7340032
PAAPP	/erp/hrdemo/HR75/paapp.dbf	8388608
PALARGE	/erp/hrdemo/HR75/palarge.dbf	3145728
PIAPP	/erp/hrdemo/HR75/piapp.dbf	3145728
PILARGE	/erp/hrdemo/HR75/pilarge.dbf	3145728
PSIMAGE	/erp/hrdemo/HR75/psimage.dbf	68157440
PSINDEX	/dev/rhr_psindex_lv	4194304000
PSRBS	/rhr_psrbs_lv	4194304000
PSTABLE	/dev/rhr_pstable_lv	3145728000
PSTEMP	/dev/rhr_pstemp_lv	1048576000
PTAPP	/erp/hrdemo/HR75/ptapp.dbf	3145728
PTPRC	/erp/hrdemo/HR75/ptprc.dbf	3145728
PTTBL	/erp/hrdemo/HR75/pttbl.dbf	83886080
PYAPP	/erp/hrdemo/HR75/pyapp.dbf	15728640
PYLARGE	/erp/hrdemo/HR75/pylarge.dbf	38797312
STAPP	/erp/hrdemo/HR75/stapp.dbf	3145728
STLARGE	/erp/hrdemo/HR75/stlarge.dbf	3145728
SYSTEM	/dev/rhr_system_lv	419430400
TLAPP	/erp/hrdemo/HR75/tlapp.dbf	4194304
TLLARGE	/erp/hrdemo/HR75/tllarge.dbf	3145728

27 rows selected.

We used this data to determine the size of the tablespaces that would be created on DB2. Three of these tablespaces (PSRBS, PSTEMP, and PSINDEX) were for the Oracle system. We needed to create, at a minimum, 24 tablespaces on DB2 to prepare for the migration. For better performance, more databases and tablespaces are recommended. See “DB2 configuration considerations” on page 25 for some recommendations.

We used the sample JCL that is used to install the demo database on S/390 and then adjusted the sizes to accommodate the additional data. For optimum use of 3390 DASD, the number you choose should be divisible by 48. We did not do this as ours was only a test installation.

We created six databases and 24 tablespaces to accommodate the tables to be imported from the export file. Part of the setup of PeopleSoft on OS/390 requires that you first create a table PSDBOWNER. We created the database

PSOWNRDB with the tablespace PTOWNER. This is where we put our PSDBOWNER table. This is used for security and must be present for the Data Mover scripts to work.

The following table lists the layout.

Table 1. List of databases and tablespaces used for the HR demo database

Database	Tablespace
PSHRDMO	PSIMAGE, PTAPP, PTTBL, PSTABLE
PSOWNRDB	PTOWNER
PSHRDMOT	PTPRC
PSHRDMOB	BNAPP, BNLARGE,
PSHRDMOH	HRAPP, HRLARGE, HTAPP, PAAPP, PALARGE, STAPP, STLARGE
PSHRDMOP	FSAPP, GIAPP, GPAPP, PIAPP, PILARGE, PYAPP, PYLARGE, TLAPP, TLLARGE
ONLY USED FOR ORACLE	SYSTEM. PSRBS, PSINDEX

5.1 Creating the database

Using the TSO user ID PSOFT1, we had to create a STOGROUP name SGPSOFT. To do this we used the command:

```
CREATE STOGROUP SGPSOFT
       VOLUMES (PSOFT5, PSOFT6)
       VCAT DB2BMO;
COMMIT;

GRANT USE OF STOGROUP SGPSOFT TO PSOFT1;
GRANT USE OF ALL BUFFERPOOLS TO PUBLIC;
```

Then we created the databases using the following commands:

```
CREATE DATABASE PSHRDMO   STOGROUP SGPSOFT ;
CREATE DATABASE PSHRDMOT  STOGROUP SGPSOFT ;
CREATE DATABASE PSHRDMOB  STOGROUP SGPSOFT ;
CREATE DATABASE PSHRDMOH  STOGROUP SGPSOFT ;
CREATE DATABASE PSHRDMOP  STOGROUP SGPSOFT ;

GRANT DBADM ON DATABASE PSHRDMO  TO DM2DMO;
GRANT DBADM ON DATABASE PSHRDMOT TO DB2DMO;
GRANT DBADM ON DATABASE PSHRDMOB TO DB2DMO;
GRANT DBADM ON DATABASE PSHRDMOH TO DB2DMO;
GRANT DBADM ON DATABASE PSHRDMOP TO DB2DMO;
COMMIT;

SET CURRENT SQLID='DB2DMO';
```

5.2 Creating the tablespaces

We analyzed the size of the tablespaces in the source database and then made changes to the default DDL that is provided with the PeopleSoft installation code.

We had to increase the size of many of the tablespaces in order to have sufficient space to store the data.

DB2 storage is allocated in kilobytes using the primary (PRIQTY) and secondary (SECQTY) parameters.

For most efficient use of S/390 DASD (to match the size of tracks), the quantity size should be divisible by 48.

As previously mentioned, we did not follow this rule because this was just a test installation. This is a good opportunty to set up a buffer pool strategy that would benefit performance. See “DB2/PeopleSoft 7.5: HR buffer pool recommendations” on page 50, for a suggested buffer pool layout. In our example, we used only BP1, which is not what you would want to do for a production system.

Then we were ready to create the PeopleTools tablespaces using the following DDL commands using SPUFI:

```
CREATE TABLESPACE PSIMAGE  IN  PSHRDMO
        USING STOGROUP SGPSOFT PRIQTY 68158 SECQTY 20160
        FREEPAGE  0 PCTFREE  10
        SEGSIZE    8  BUFFERPOOL BP32K LOCKSIZE ANY  CLOSE NO ;
CREATE TABLESPACE PTAPP    IN  PSHRDMO
         USING STOGROUP SGPSOFT PRIQTY 3600    SECQTY 720
         FREEPAGE  0  PCTFREE 20
         SEGSIZE    4  BUFFERPOOL BP1 LOCKSIZE ANY CLOSE NO    ;

CREATE TABLESPACE PTTBL    IN  PSHRDMO
         USING STOGROUP SGPSOFT PRIQTY 95760     SECQTY  9360
         FREEPAGE  0 PCTFREE 20
         SEGSIZE    4 BUFFERPOOL BP1   LOCKSIZE ANY  CLOSE NO  ;
```

Then we created the tablespaces for the demo tables, using the following commands:

```
--  BEGIN APPLICATION TABLESPACES - PARSED

--  TABLESPACE PSHRDMOT

CREATE TABLESPACE PTPRC    IN  PSHRDMOT
          USING STOGROUP SGPSOFT    PRIQTY 3200  SECQTY 720
          FREEPAGE  0 PCTFREE 20
          SEGSIZE    4  BUFFERPOOL BP1  LOCKSIZE ANY  CLOSE NO  ;
COMMIT;

--  TABLESPACE PSHRDMOB

CREATE TABLESPACE BNAPP    IN  PSHRDMOB
          USING STOGROUP SGPSOFT   PRIQTY 9438  SECQTY 720
          FREEPAGE 0 PCTFREE  20   SEGSIZE 4
          BUFFERPOOL BP1  LOCKSIZE ANY  CLOSE NO    ;
CREATE TABLESPACE BNLARGE  IN  PSHRDMOB
          USING STOGROUP SGPSOFT PRIQTY 7341  SECQTY 720
          FREEPAGE  0 PCTFREE  20   SEGSIZE 4
          BUFFERPOOL BP1 LOCKSIZE ANY  CLOSE NO    ;
COMMIT;

--  TABLESPACE PSHRDMOH
```

```
CREATE TABLESPACE HRAPP    IN  PSHRDMOH
         USING STOGROUP SGPSOFT PRIQTY 23100 SECQTY 1440
         FREEPAGE  0 PCTFREE  20  SEGSIZE 4
         BUFFERPOOL BP1 LOCKSIZE ANY  CLOSE NO    ;
CREATE TABLESPACE HRLARGE  IN  PSHRDMOH
         USING STOGROUP SGPSOFT PRIQTY 5400 SECQTY 720
         FREEPAGE  0 PCTFREE  20   SEGSIZE 4
         BUFFERPOOL BP1 LOCKSIZE ANY  CLOSE NO    ;
CREATE TABLESPACE HTAPP    IN  PSHRDMOH
         USING STOGROUP SGPSOFT   PRIQTY 10800  SECQTY 720
         FREEPAGE  0  PCTFREE  20  SEGSIZE 4
         BUFFERPOOL BP1 LOCKSIZE ANY  CLOSE NO    ;
CREATE TABLESPACE PAAPP    IN  PSHRDMOH
         USING STOGROUP SGPSOFT PRIQTY 8400 SECQTY 720
         FREEPAGE  0 PCTFREE  20  SEGSIZE 4
         BUFFERPOOL BP1 LOCKSIZE ANY  CLOSE NO    ;
CREATE TABLESPACE PALARGE  IN  PSHRDMOH
         USING STOGROUP SGPSOFT PRIQTY 3200 SECQTY 720
         FREEPAGE  0 PCTFREE  20   SEGSIZE 4
         BUFFERPOOL BP1 LOCKSIZE ANY  CLOSE NO    ;
CREATE TABLESPACE STAPP    IN  PSHRDMOH
         USING STOGROUP SGPSOFT PRIQTY 3200 SECQTY 720
         FREEPAGE  0 PCTFREE  20  SEGSIZE 4
         BUFFERPOOL BP1 LOCKSIZE ANY  CLOSE NO    ;
CREATE TABLESPACE STLARGE  IN  PSHRDMOH
         USING STOGROUP SGPSOFT PRIQTY  3200 SECQTY 0720
         FREEPAGE  0 PCTFREE  20   SEGSIZE 4
         BUFFERPOOL BP1 LOCKSIZE ANY  CLOSE NO    ;
COMMIT:

--  TABLESPACE PSHRDMOP

CREATE TABLESPACE FSAPP    IN  PSHRDMOP
         USING STOGROUP SGPSOFT PRIQTY 3200 SECQTY 720
         FREEPAGE  0  PCTFREE 20  SEGSIZE 4
         BUFFERPOOL BP1  LOCKSIZE ANY     CLOSE NO    ;
CREATE TABLESPACE GIAPP    IN  PSHRDMOP
         USING STOGROUP SGPSOFT PRIQTY 3200 SECQTY 720
         FREEPAGE  0  PCTFREE 20  SEGSIZE 4
         BUFFERPOOL BP1  LOCKSIZE ANY     CLOSE NO    ;
CREATE TABLESPACE GPAPP    IN  PSHRDMOP
         USING STOGROUP SGPSOFT   PRIQTY 8400 SECQTY 720
         FREEPAGE 0 PCTFREE  20   SEGSIZE 4
         BUFFERPOOL BP1  LOCKSIZE ANY  CLOSE NO    ;
CREATE TABLESPACE PIAPP    IN  PSHRDMOP
         USING STOGROUP SGPSOFT PRIQTY 3200 SECQTY 720
         FREEPAGE  0 PCTFREE  20  SEGSIZE 4
         BUFFERPOOL BP1 LOCKSIZE ANY  CLOSE NO    ;
CREATE TABLESPACE PILARGE  IN  PSHRDMOP
         USING STOGROUP SGPSOFT PRIQTY 3200 SECQTY 720
         FREEPAGE  0 PCTFREE  20   SEGSIZE 4
         BUFFERPOOL BP1 LOCKSIZE ANY  CLOSE NO    ;
CREATE TABLESPACE PYAPP    IN  PSHRDMOP
         USING STOGROUP SGPSOFT PRIQTY 15800 SECQTY 1440
         FREEPAGE  0 PCTFREE  20  SEGSIZE 4
         BUFFERPOOL BP1 LOCKSIZE ANY  CLOSE NO    ;
CREATE TABLESPACE PYLARGE  IN  PSHRDMOP
```

```
          USING STOGROUP SGPSOFT PRIQTY 40000 SECQTY 2880
          FREEPAGE  0 PCTFREE  20   SEGSIZE 4
          BUFFERPOOL BP1 LOCKSIZE ANY  CLOSE NO    ;
CREATE TABLESPACE TLAPP    IN  PSHRDMOP
          USING STOGROUP SGPSOFT PRIQTY 4200   SECQTY 720
          FREEPAGE  0 PCTFREE  20  SEGSIZE 4
          BUFFERPOOL BP1 LOCKSIZE ANY  CLOSE NO    ;
CREATE TABLESPACE TLLARGE  IN  PSHRDMOP
          USING STOGROUP SGPSOFT PRIQTY 3200 SECQTY 720
          FREEPAGE  0 PCTFREE  20   SEGSIZE 4
          BUFFERPOOL BP1 LOCKSIZE ANY  CLOSE NO    ;
```

We had one extra tablespace in our source data that was not in the OS/390 installation, so we had to create one more tablespace:

```
CREATE TABLESPACE PSTABLE  IN  PSHRDMO
          USING STOGROUP SGPSOFT PRIQTY 4200   SECQTY 720
          FREEPAGE  0 PCTFREE  20  SEGSIZE 4
          BUFFERPOOL BP1 LOCKSIZE ANY  CLOSE NO    ;
```

At this point DB2 was ready for the execution of the create table DDL that were extracted from the source database.

Note: We used BP1 for all the tablespaces. You should determine your buffer pool strategy and spread out the use of buffer pools.

5.3 Running the extracted DDL

Now that we had manually created the database and tablespaces required for the target data set, we were ready to run the DataMover export and then extract the DDL to create the tables. After the tables are created the data can be loaded.

This is described in Chapter 6, “Executing the DataMover scripts” on page 27.

5.4 DB2 configuration considerations

PeopleSoft 7.5 is installed with 6 databases and 24 tablespaces. As you run your installation, you should consider increasing these numbers or, in effect, reducing the number of tables per tablespace.

The following lists recommendations for optimizing the DB2 layout for performance:

- Separate all temporary tables into their own tablespaces.
- The next candidate would be on those PeopleTools tables, especially on process scheduler-related tables
- Large tables should be moved into their own tablespaces.
- Separate those tables needed for batch from the on-line processing tables.
- If you can identify the read-only type of tables (look-up tables, static tables, reference tables, and so on), separate these.
- As a guideline, put around 50 tables in one tablespace. (The challenge is to determine which ones are read-only types of tables.)

- Since a database object is a merely a logical control block that represents DB2 objects, you can break out as many databases as needed to make a smaller DBD size.

This topic is discussed further in “DB2/PeopleSoft 7.5: HR tablespace recommendations” on page 47.

Chapter 6. Executing the DataMover scripts

This chapter documents the DataMover scenarios we ran at the ITSO in Poughkeepsie to move an entire PeopleSoft HR database from Oracle on AIX to DB2 on OS/390.

To first test the process, we moved two tables from the HR demo database. This is described in Appendix E, “Testing the migration process with two tables” on page 107.

At this point we had completed the following steps:

- We had set up PeopleTools to connect to the Oracle database on AIX and DB2 on S/390.
- We had sized the DB2 files before proceeding to allocate the database and tablespaces (we increased the allocations for the tablespaces).
- We had allocated the VSAM files by creating the database and tablespaces.

The migration method we used included the following steps:

- Run a DataMover script to export the data.
- Run a special DataMover script to separate the DDL from the data in the export file.
- Execute the DDL to create the tables.
- Run the DataMover script to import the data to DB2.
- Verify that the target database is equivalent to the source database.

This chapter does not cover the handling of:

- Very large tables outside of DataMover to reduce the migration time
- Triggers, stored procedures
- SQRs or other programs with Oracle-specific SQL

See Chapter 7, “Handling the exceptions in the migration process” on page 35 for more information on these topics.

6.1 Preparing the systems

To be able to run the DataMover utility, you must have connectivity to the database. To run the export, you must have established connectivity to the Oracle database on AIX. To extract the DDL and run the import ,you must have connectivity to the DB2 database on OS/390.

For these to run correctly, you must also correctly configure the PeopleTools configuration manager, as described in Chapter 3, “Create a PeopleTools environment” on page 9.

6.2 Running the export

This section shows:

- The export script
- A portion of the output log
- A portion of the exported file

Export script

```
REM
REM Export Oracle HR75DMO Database
REM
set output d:\redbook\data\hrdmoexp.dat;
set log d:\redbook\logs\hrdmoexp.log;
export *;
```

Log use for export

```
Started:  Mon Oct 04 19:30:29 1999
Data Mover Release: 7.5
Outputting export in d:\redbook\data\hrdmoexp.dat;
Export  ABSENCE_CAL 432
Records remaining: 3115
 Export  ABSENCE_HIST 123
Records remaining: 3114
 Export  ABSV_ACCRUAL 106
Records remaining: 3113
 Export  ABSV_ADDL_TBL 2
Records remaining: 3112
 Export  ABSV_PERIOD 2
Records remaining: 3111

Many lines removed here
After all the records are exported, then the views are exported
...
View  ZZ_DMNDCRS_SRC
Views remaining: 5
View  ZZ_PAY_LINE
Views remaining: 4
View  ZZ_PAY_LINE_CAN
Views remaining: 3
View  ZZ_PCHK_S_VCAN
Views remaining: 2
View  ZZ_PCHK_S_VW
Views remaining: 1
View  ZZ_PCHK_VCAN
Ended:  Mon Oct 04 19:51:36 1999
Successful completion
```

Export output file

The export output file is a large file (xxxx) that contains mainly unreadable characters. The first lines are shown here.

```
SET VERSION_DAM 7.5:1:0
REM Database: HR75DMO
REM Started: Wed Oct 06 10:45:21 1999
 Export  RECORD/SPACE.x
BNAPP
BNLARGE
FSAPP
GIAPP
GPAPP
HRAPP
HRLARGE
HTAPP
```

```
PAAPP
PALARGE
PIAPP
PILARGE
PSIMAGE
PTAPP
PTPRC
PTTBL
PYAPP
PYLARGE
STAPP
STLARGE
TLALL
TLAPP
TLLARGE
/
A($)B(AAAAAA)A(P)B(AA)A(BNDx)B(BLAAABAAAAAAAAAAAAAAAAAAAAAA)A(P)B(AA)A(G)
A(P)B(AAAAAAAA)A(HRAPP)B(AA)A(HRLARGE)B(AA)A(HTAPP)B(AA)A(PAAPP)B(AA)A(P)
A(ALARGE)B(AA)A(PIAPP)B(AA)A(PILARGE)B(AA)A(PSIMAGE)B(AA)A(PTAPP)B(AA
```

Note: This file should not be edited.

Large tables can be exported individually rather than using the * to export everything. In fact, it is good practice to run parallel export jobs.

6.3 Setting up the security tables

To set up our security we completed the following steps:

- We set up DB2DMO as a RACF group. We granted dbadm authority to this group on DB2.
- We set up the RACF userid PS that is part of the DB2DMO group. This is our Access ID.
- We set up the RACF userid PSOFT1 with TSO access. This is our Connect ID.
- We created three tables and granted select authority to PSOFT1 to these tables:
  - PS.PSDBOWNER
  - DB2DMO.PSLOCK
  - DB2DMO.PSOPRDEFN
- For end users we would have set up the RACF user ID PSUSER with no TSO access, part of the group DB2DMO, associated with the PeopleSoft user ID PSUSER and entered in the PSOPRDFN table, through the PeopleSoft security administration process. This is an Operator ID that can be tracked by DB2PM for performance management.

6.4 Script to extract the DDL

At this point we ran the special script provided by the IBM PeopleSoft Competency center that was designed to produce the DDL that would be used to create the tables for DB2.

This section shows:

- The extract script
- A portion of the output log file
- A portion of the extracted DDL

Extract script

```
REM

REM Extract DDL from Oracle DataMover Export into DB2 OS/390 Format
REM
set log d:\redbook\logs\hrdmoext.log;
set input d:\redbook\data\hrdmoexp.dat;
set execute_sql set current sqlid = 'DB2DMO';
set ddl table space  * input stogroup  as SGPSOFT;
set ddl index        * input stogroup  as SGPSOFT;
set ddl unique index * input stogroup  as SGPSOFT;
set ddl record       * input dbname    as HRDMO;
set ddl record       * input owner     as DB2DMO.;
set ddl index        * input owner     as DB2DMO.;
set ddl index        * input owner2    as DB2DMO.;
set ddl unique index * input owner2    as DB2DMO.;
set ddl unique index * input owner     as DB2DMO.;
set no view;
set no space;
set no trace;
set extract output d:\redbook\ddl\hrdmoddl.sql;
import * ;
```

Note: We used the same stogroup for all tablespaces for our test. In a production environment, you should use different stogroups to spread the files across more DASD units.

See 6.5.1, “Creating the tables” on page 31 for a partial listing of the extracted file.

6.4.1 DataMover EXTRACT command

The syntax of the EXTRACT command is:

```
SET EXTRACT {COMMAND | DDL | INPUT | SPACE | OUTPUT file_name};
```

It is then followed by a valid DataMover command such as IMPORT or REPLACE_ALL.

This command extracts various types of information from an export file (the .DAT file specified in the corresponding SET INPUT command that precedes the IMPORT or REPLACE ALL command) and writes this information to the user-defined output file specified in the SET EXTRACT OUTPUT file_name statement.

Note: You must use SET EXTRACT OUPUT before issuing any other SET EXTRACT statements.

EXTRACT INPUT writes out any statements from the .DAT file that are associated with the table(s) being imported. The EXTRACT DDL command writes out any CREATE TABLE, CREATE INDEX, or CREATE UNIQUE INDEX statements from the .DAT file. The EXTRACT command writes out the EXPORT statements from the .DAT file.

When EXTRACT statements are issued, no SQL CREATE or INSERT statements will be executed. The associated IMPORT or REPLACE_ALL command is not actually executed, so no import is performed.

6.5 Executing the DDL to create tables

Previously, we manually created the DDL for the database and tablespaces. Then we used the DDL that was extracted from the export file. The following are the first two of many create statements.

6.5.1 Creating the tables

We modified the extracted DDL to make the tablespace names match the ones we created. We used the `ftp` command to send the extracted file to OS/390. On OS/390, using the ISPF editor, we used the following mass change commands for 24 tablespaces to match the create tablespace commands we had executed:

```
c   HR75.HRAPP     PSHRDMOH.HRAPP all
c   HR75.HRLARGE   PSHRDMOH.HRLARGE all
c   HR75.HTAPP     PSHRDMOH.HTRAPP all
c   HR75.PAAPP     PSHRDMOH.PAAPP all
c   HR75.PALARGE   PSHRDMOH.PALARGE all
c   HR75.STAPP     PSHRDMOH.STAPP all
c   HR75.STLARGE   PSHRDMOH.STLARGE all

c   HR75.FSAPP     PSHRDMOP.FSAPP all
c   HR75.GIAPP     PSHRDMOP.GIAPP all
c   HR75.GPAPP     PSHRDMOP.GPAPP all
c   HR75.PIAPP     PSHRDMOP.PIAPP all
c   HR75.PYAPP     PSHRDMOP.PYAPP all
c   HR75.TLAPP     PSHRDMOP.TLAPP all
c   HR75.PILARGE   PSHRDMOP.PILARGE all
c   HR75.PYLARGE   PSHRDMOP.PYLARGE all
c   HR75.TLLARGE   PSHRDMOP.TLLARGE all
c   HR75.STLARGE   PSHRDMOH.STLARGE all

c   HR75.BNAPP     PSHRDMOB.BNAPP all
c   HR75.BNLARGE   PSHRDMOB.BNLARGE all

c   HR75.PTAPP     PSHRDMO.PTAPP all
c   HR75.PSIMAGE   PSHRDMO.PSIMAGE all
c   HR75.PTTBL     PSHRDMO.PTTBL all
c   HR75.PSTABLE   PSHRDMO.PSTABLE all

c   HR75.PTPRC   PSHRDMOT.PTPRC all
```

We had to change the allocation of the secondary quantity for the indexes because several of the indexes were very large and ran out of extents when we ran the import. We issued the following ISPF mass change command:

```
c 'SECQTY 40' 'SECQTY 960' all
```

We added a line to the extracted DDL to set the current SQLID to DB2DMO as this was our owner ID. Then we executed the DDL that was extracted from the source database. The beginning of the file contained the following lines:

```
SET CURRENT SQLID = 'DB2DMO';
    CREATE TABLE DB2DMO.PS_ABSENCE_CAL (CALYEAR SMALLINT NOT NULL,
       MONTHCD CHAR(2) NOT NULL,
       ABS_MONTH_END_DT DATE,
       ABS_MONTH_START_DT DATE) IN PSHRDMOH.HRAPP
    ;
    COMMIT
    ;
    CREATE UNIQUE INDEX DB2DMO.PS_ABSENCE_CAL ON DB2DMO.PS_ABSENCE_CAL
     (CALYEAR DESC,
      MONTHCD DESC) USING STOGROUP PSSINDEX PRIQTY 40 SECQTY 40 CLUSTER
     CLOSE NO
   ;
    COMMIT
    ;
    CREATE TABLE DB2DMO.PS_ABSENCE_HIST (EMPLID CHAR(11) NOT NULL,
       EMPL_RCD# SMALLINT NOT NULL,
       BEGIN_DT DATE NOT NULL,
       ABSENCE_TYPE CHAR(3) NOT NULL,
       BEGIN_TM TIME,
       RETURN_DT DATE,
       RETURN_TM TIME,
       DURATION_DAYS SMALLINT NOT NULL,
       DURATION_HOURS DECIMAL(5,1) NOT NULL,
       ABSENCE_CODE CHAR(3) NOT NULL,
       REASON CHAR(30) NOT NULL,
        PAID_UNPAID CHAR(1) NOT NULL,
   EMPLOYER_APPROVED CHAR(1) NOT NULL,
      ABS_APPROVED_DT DATE,
      NOTIFIED_DT DATE,
      NOTIFIED_TM TIME,
      NOTIFIED_BY CHAR(30) NOT NULL,
      INDUSTRIAL_INJURY CHAR(1) NOT NULL,
      INCIDENT_NBR CHAR(8) NOT NULL,
      REFER_TO_OHA CHAR(1) NOT NULL,
      COUNSELLING CHAR(1) NOT NULL,
      DISCIPLINARY CHAR(1) NOT NULL,
      ABS_RECURRENCE CHAR(1) NOT NULL,
      DOC_CONSULTED CHAR(1) NOT NULL,
      DOC_CONSULT_DT DATE,
      DUE_DT DATE,
      BIRTH_DT DATE,
   EWC_DT DATE,
    QW_DT DATE,
    MPP_START_DT DATE,
    MPP_EXPECTED_END DATE,
    MPP_EARLIEST_DT DATE,
    MATB1_RECEIVED_DT DATE,
    RTN_WRK_REMIND_DT DATE,
    RTN_CONF_RECVD_DT DATE,
    ELIGIBLE CHAR(1) NOT NULL,
    SMP_ELIGIBLE CHAR(1) NOT NULL,
    NOTIF_MLSTART_DT DATE,
    ELIG_EXT_LEAVE CHAR(1) NOT NULL,
    EXP_ML_END_DT DATE,
    AWC_DT DATE,
    NOTIFIED_RETURN_DT DATE,
   PROT_START_DT_GER DATE,
      PROT_END_DT_GER DATE,
      WAO_DATE_NLD DATE,
      WAO_PERCENT_NLD SMALLINT NOT NULL,
      ORIG_EMPL_RCD_BEL SMALLINT NOT NULL,
      ORIG_BEGIN_DT_BEL DATE,
      ORIG_ABS_TYPE_BEL CHAR(3) NOT NULL,
      ABS_ENTRY_PANEL CHAR(4) NOT NULL) IN PSHRDMOH.HRAPP
   ;
```

```
COMMIT
;
CREATE UNIQUE INDEX DB2DMO.PS_ABSENCE_HIST ON DB2DMO.PS_ABSENCE_HIST
 (EMPLID,
   EMPL_RCD#,
   BEGIN_DT DESC,
ABSENCE_TYPE) USING STOGROUP PSSINDEX PRIQTY 40 SECQTY 40 CLUSTER
  CLOSE NO
 ;
 COMMIT
 ;
Many lines not shown
```

6.5.2 Running the extracted DDL

Then we executed this DDL using SPUFI. This takes a while to run because many indexes and tables are created.

6.6 Running the import

To run the import, you logon to PeopleTools and execute the following DataMover script to import the data:

This is the Import script we used:

```
REM
REM Import data from Oracle DataMover Export into DB2 OS/390
REM
set log d:\redbook\logs\hrdmoimp.log;
set input d:\redbook\data\hrdmoexp.dat;
set execute_sql set current sqlid = 'DB2DMO';
set no view;
set no space;
set no trace;
set no record;
import *;
```

This takes a while to run. After completion, check the logs for any errors that may have occurred. This was our log file:

Log for import

```
Started:  Tue Oct 05 16:52:30 1999
Data Mover Release: 7.5
Commit done at end of record
 Import  ABSENCE_CAL  432
Records remaining: 3115
 Import  ABSENCE_HIST  123
Records remaining: 3114
 Import  ABSV_ACCRUAL  106
Records remaining: 3113
 Import  ABSV_ADDL_TBL  2
Records remaining: 3112
 Import  ABSV_PERIOD  2
Records remaining: 3111
 Import  ABSV_PLAN_TBL  7
Records remaining: 3110
 Import  ABSV_RATE_TBL  33
Records remaining: 3109
 Import  ABSV_REQUEST  2
Records remaining: 3108
```

If something happens that causes the import to not complete successfully, you can by restart by using the `set start after tablename` command to start right after

the table that was successfully imported; you can find this information in the log. If you do this, make sure to change the log file name so that it won't overlay the previous log information.

6.7 Validate the imported data

Once the import has finished, the log must be reviewed to see that there were no errors. At this point RUNSTATS should be run on the DB2 database.

The remaining tasks are to handle the following:

- Views
- Stored procedures
- Customized or non-ANSI standard SQL
- Triggers
- Other customized programs

6.7.1 Handling views

The command to recreate the views is `replace_view *`. The statement will look through the PeopleTools tables and try to create every SQL view in the database. The view text needs to be specific to the database platform. So to go from Oracle to DB2, if there is any SQL syntax that is allowed by Oracle but not by DB2, you will need to log in to PeopleTools, open the view, and format the view text for DB2.

For example, when using financials, you may need to run the script IMVWD3.DMS to delete certain views and re-add them in the correct format.

Note: Some Oracle views may use NLVL (Null Values). NLVL is not accepted by DB2. The solution is to use Coalesce. The COALESCE function is a synonym for the VALUE function. It returns the first argument that is not null.

6.7.2 Handling the Process Scheduler and cleaning up old files

These topics are covered in Chapter 9, "Migrating a database from Oracle to DB2 example" on page 39

Chapter 7. Handling the exceptions in the migration process

One of the most important aspects of a migration project is the test plan to test the migration process well before the date of the migration of the production database. The test plan includes:

- Deciding on the process
- Running a test migration
- Planning for the handling of exceptions

This chapter introduces some of the exception areas to review.

7.1 Types of exceptions

These customizations may be performed manually or be automated:

1. Customized DDL
   This is covered in Appendix 6, “Executing the DataMover scripts” on page 27.
2. Stored procedures
   Oracle stored procedures must be converted to DB2 stored procedures. DB2 stored procedures can be written in C, C++, COBOL, Assembler or PL/I, or in the DB2 SQL Stored Procedure language.
3. Triggers
   Oracle triggers may need to be converted before being DB2 compliant.

These customizations may be performed manually:

1. Packages
2. Customized applications
   Non-ANSI SQL statements and functions
3. PS_SQLSTMT_TBL
   Non-ANSI SQL or Hints
4. Shortening the name of Oracle objects for DB2

   Oracle allows longer names for database objects than DB2 for OS/390 does. Oracle's limit is 30 characters while DB2's is 18 characters. So, for any customized Oracle objects, the names will need to be shortened for DB2.

7.2 Planning the migration process

We found that by using a script to extract the DDL from the export file, we did not have to be concerned about what changes had been made to the database. If we used these extracted DDLs, then the target database on DB2 on S/390 would exactly reflect the source database on Oracle on AIX. The DataMover script takes care of creating an exact replica of the source data base.

The DataMover script ensures that all data is transferred to the DB2 for OS/390 database in a consistent manner. It will handle the ASCII-to-EBCDIC conversion. Although you can store data in ASCII on DB2 on S/390, this is *not an option* for PeopleSoft.

The bulk of the processing in the batch environment needs an EBCDIC format, which incurs the overhead of translating from ASCII to EBCDIC if the data is stored in ASCII.

7.3 Handling very large tables

If you have tables of more than 50,000 rows, you may find that using the DataMover script is too time-consuming to allow you to accomplish your migration in the available window. For large tables, you can use one of the following alternatives:

- Use third-party tools.
- Use an IGS service offering.
- Move the data manually yourself, ensuring consistency.

For more information, see Appendix F, “Handling large tables” on page 113.

7.4 Testing

Thorough testing of all aspects of the migration process must be done. Some of the things to test are:

- The customized programs
- Batch processes
- Stored procedures, triggers, and so on
- Printing
- Interfaces with other systems that depend on PeopleSoft data

Chapter 8. Migrating the production database

Once you have completed setting up the process for migration and testing all aspects of the process, the migration of the production database must take place.

How this will be done will depend on:

- How many environments must be migrated
- Whether you will run production in parallel for a certain period
- How long the migration of the data will take (based on the testing that was done)

There are several ways that could speed up the process of migrating:

1. You could freeze all changes to the database schema, extract the DDL from the source database, and use whatever time is necessary (perhaps a week) to set up the target database. Then, on the following weekend, only export the data and import the data. The DDL will have already been executed on DB2, so you will reduce the amount of time you need for the production migration.
2. If you have several very large tables, these may be the prime inhibitor to reducing the migration time. It is possible, with lots of testing, to handle large tables manually, rather than using the DataMover utility.
3. If you have set up the target DB2 subsystem to use several databases, you could run several exports and imports in parallel. In this case, you would export the data by tablespace.
4. You could consider using the DATA JOINER product.
5. You could test to see if rebuilding the indexes later improved the time to import.

DB2 may be a new environment if the migration was done from an Oracle database. If DB2 is new to the S/390 platform, there are several areas that must be addressed early in the planning process. These are covered here briefly, but an education plan and DB2 consultant skills should be included in the project plan.

8.1 Another approach

If you plan on migrating the data with something other than Data Mover, or if you have large volumes and want to create their DB2 environment prior to doing a full Data Mover export from their Oracle environment, you might consider this approach:

- Create a clone of the Oracle production database with data in only the tools tables.
- Run DataMover against this clone to export the whole database (this will be a much smaller file).
- Logon to DataMover with the DB2 connectivity.
- Run the `extract` command against this smaller file.
- Run the extracted DDL on MVS after making any necessary changes to DB2 Administration of PeopleSoft on OS/390.

Chapter 9. Migrating a database from Oracle to DB2 example

This chapter contains hints and tips that were developed based on an actual migration of a customer database. We include it in order to give you more points to consider in your migration project. It is based on manually creating the new DDL without using DataMover to extract the DDL, as described in “Script to extract the DDL” on page 29.

Prior to undertaking any of the changes outlined here, you should have a new CD that will work within the DB2 OS/390 environment. The license code should end in an “A”. This CD should be unloaded to the file server as soon as possible, because there are definite differences between the Oracle executables and the DB2 executables (for instance, the SQRBINW directory is different in Oracle and DB2). Therefore, the CD-ROM must be unloaded before you start the migration process.

Changing the values from Oracle to DB2 is really determined by the amount of customizations that have been done to the system. All the customizations should have been documented and the changes duly noted. For example, if the system has not been customized, one should be able to use the HRDDL or FSDDL that was delivered on the PeopleSoft CDs.

However, if the system been heavily modified, then the existing FSDDL and HRDDL will have to be altered to take into account the changes and additions to the existing tables. The corresponding documentation outlining the changes to the existing structures will be beneficial at this time.

New tables, indexes, and views will have to be created and the DDL executed in DB2. Lack of documentation will delay executing the DDL within DB2, as it will take additional time to identify what has changed and what has remained the same.

9.1 Examples of changes to the DDL

These examples outline the different changes that will have to occur to run the Oracle-generated DDL on OS/390.

- Modifying the DDL. The following changes will be required to use the DDL generated on the Oracle

9.1.1 Creating a tablespace

Tablespace:

ORACLE	DB2
INITIAL	PRIQTY
NEXT	SECQTY
MAXEXTENTS	NA
PCTINCREASE	PCTFREE

Following is an example of the Oracle create tablespace statement:

```
CREATE TABLESPACE PTAPP DATAFILE '/u01/oradata/<SID>/ptapp.dbf'SIZE 8M
DEFAULT STORAGE (INITIAL 64K NEXT 128K MAXEXTENTS 110 PCTINCREASE0)
```

This is an example of the DB2 create tablespace statement:

```
SET CURRENT SQLID = 'FSDV750M';
CREATE TABLESPACE PTT0070T
IN CMCPTAVD USING STOGROUP CMCNSV7S PRIQTY 720
SECQTY 720 ERASE NO FREEPAGE 0 PCTFREE 20 BUFFERPOOL BP0
LOCKSIZE ANY CLOSE NO SEGSIZE 4 LOCKMAX SYSTEM    ;
```

9.1.2 Creating a table

When modifying the create table statement, the following should be noted:

ORACLE	DB2
PRIMARY KEY	N\A
NUMBER	Smallint\Decimal
Create table tablename	Create table owner.tablename in ownerid.tablespace

This is an example of an Oracle create table statement:

```
CREATE TABLE LocState (
   LocGeoState    NUMBER NOT NULL,
LocNameState   CHAR(25) NOT NULL,
    LocAbbrevState CHAR(2) NOT NULL,
    PRIMARY KEY (LocGeoState)
);
```

This is an example of a DB2 create table statement:

```
CREATE TABLE FSDV750M.LOCCOUNTY
(LOCGEOSTATE DECIMAL NOT NULL,
LOCGEOCOUNTY DECIMAL NOT NULL,
LOCNAMECOUNTY CHAR(20) NOT NULL,
LOCABBREVCOUNTY CHAR(5) NOT NULL) IN FSDV750M.PTT0070T;
```

9.1.3 Creating an index

When modifying the create index statement, the following should be noted:

ORACLE	DB2
INITIAL	PRIQTY
NEXT	SECQTY
UNIQUE	UNIQUE
N/A	CLUSTER
INITRANS	Pending
MAXTRANS	Pending
PRIMARY KEY	N/A for Peoplesoft Databases
PCTINCREASE	PCTFREE
FREELIST	FREEPAGE
MINEXTENTS	N/A
MAXEXTENTS	N/A

This is an example of a DB2 create index statement:

```
Create index
       <creator>.<index_name>   <creator>.<index_name>
      on <tablename>   on <creator>.<table_name>
      tablespace          N/A
```

9.1.4 Verifying the values of the new tables

Once the database, tablespaces, tables, indexes, and corresponding views have been created, the next step is to verify that the values within the tables are indeed pointing to the new values. For example, to logon to the FSDV750M database, go to **Start-->Programs**. Then open your PeopleSoft Program group and click **PeopleTools.**

9.1.5 Security setup

The signon scenario within DB2 is that the PS.PSDBOWNER table gets read to verify the owner ID associated with the current database name. Then the owner ID is used to qualify the PSLOCK table, and finally the PSOPRDEFN table gets read to verify that the operator who signed onto PeopleSoft is indeed a valid operator.

For instance, if the database did not exist in DB2,then an entry will have to be added for it in the PS.PSDBOWNER table. This table is comprised of three columns; DBNAME, Owner ID, and DEDICATED_THREADS.

Next, the value of the Owner ID within the PSLOCK table will have to be changed. In the current example, the database name is FSDV750M and the Owner ID = FSDV750M. Therefore, the value for Owner ID should equal FSDV750M.

The Access ID and ACCESSPSWD are within the PSOPRDEFN table as well. The number of changes depends on the security option you have used.

The Connect ID option allows you to use the existing PeopleSoft signons that probably existed within the Oracle environment. You will have to change the Access ID and ACCESSPSWD, and then re-encrypt the password.

However, if you have decided not to use the Connect ID option, then each ID within the PSOPRDEFN table will have to be a legitimate TSO ID that is able to sign onto TSO. The Access ID and ACCESSPSWD will have to be changed in conjunction with each OPRID and OPERPSWD entry within the table.

The next table to update is the PSACCESSPRFL table. You will want to verify that the current value for Access ID is indeed the value you are using within your DB2 database. If you change the Access ID, then you will want to change the ACCESSPSWD as well, in order to keep them in sync.

Now verify that you can signon to the system via DataMover. However, note that the commands described in the following sections should be run through SPUFI or QMF, not DataMover.

9.1.6 Process scheduler setup

There are additional changes that are required, especially if you plan to utilize the PeopleSoft-provided process scheduler. It is a good idea to bring up at least one of the process schedulers and run COBOL and SQRs against the database. Some of these steps might be to verify that there are no values in these fields, as opposed to replacing values.

The first table to update is the PS_PRCSSYSTEM table and since you are now running on the OS/390, you will want to reinitialize the LASTPROCESSINSTANCE number to zero, as follows:

```
UPDATE PS_PRCSSYSTEM SET LASTPRCSINSTANCE = 0 WHERE OPSYS = '2';
```

The next table to update is the PSPRCSPRFL table. The first change is to verify the jobname of the job being submitted:

```
UPDATE PSPRCSPRFL SET MVSJOBNAME = 'FSDV75M' where MVSJOBNAME =
'XXXXXXX';
```

Then ensure that the SRVRDESTFILE equals a directory on the mainframe, as well as the client that exists, and that you have access to:

```
UPDATE PSPRCSPRFL SET SRVRDESTFILE = 'CMCNS0FD.FSDV750M'
 WHERE SRVRDESTFILE = 'SYSDA.TEMP' ;
UPDATE PSPRCSPRFL SET CLIENTDESTFILE = 'C:\TEMP\'
WHERE CLIENTDESTFILE > ' ';
```

Next, make sure that the printer output field is correct:

```
UPDATE PSPRCSPRFL SET CLIENTDESTPRNT = 'LPT1'
WHERE CLIENTDESTPRNT > ' ' OR SUBSTR(CLIENTDESTPRNT,1,1) = 'L';
```

9.1.7 Cleaning up old data

Lastly there is some table cleanup that should be done to wipe out all the old data and allow you to start with the new data:

```
DELETE FROM PSPRCSRQST;
DELETE FROM PSPRCSRQSTXFER;
DELETE FROM PS_MESSAGE_LOG;
DELETE FROM PS_MESSAGE_LOGPARM;
```

Once the old data has been cleared out of the database, run the following script either through DataMover or through SPUFI. When the DB2 database is initially built within the DB2 environment, a DMS script is created to initially load the database. The bottom of the DMS script contains some DB2 Connect-specific information, as follows:

```
Update ps_prcstypedefn
set parmlist = 'DB2ODBC/'||substr(parmlist,5,246)
where substr(parmlist,1,4) = 'DB2/'
and dbtype='1';
Update ps_prcstypedefn
set parmlist = '-CTDB2ODBC '||substr(parmlist,8,243)
where substr(parmlist,1,7) = '-CTDB2 '
and dbtype='1';
Update ps_prcstypedefn
set parmlist = '-CX%%APPSERVER%% -CTDB2ODBC '||substr(parmlist,24,226)
where substr(parmlist,1,24) = '-CX%%APPSERVER%% -CTDB2 '
and dbtype='1';
```

```
Update ps_prcstypedefn
set parmlist = '-CX%%APPSERVER%% -CT DB2ODBC'||substr(parmlist,25,226)
where substr(parmlist,1,25) = '-CX%%APPSERVER%% -CT DB2 '
and dbtype='1';

INSERT INTO PSTREESELECT20
(SELECTOR_NUM,TREE_NODE_NUM,RANGE_FROM_20,RANGE_TO_20)
SELECT 0,0,RECNAME,' '   FROM PSRECDEFN
WHERE RECTYPE IN (0, 1, 6);
```

9.1.8 Adding views

Part of the process for building the database within DB2 is that, prior to running the CREATEVW.DMS script, another script (Imvwd31.DMS) is run through DataMover against the database. The purpose of this script is to delete some old views and re-add the new views. (This is now done through DataMover, whereas before it was a manual process.) Then, when the CREATEVW.DMS script is run, these new views should be created.

You might experience some problems when running the CREATEVW.DMS script since some of the views might contain Oracle-specific syntax. It is good practice to make a note of such views and then move forward, and later come back to those views that encountered errors.

Also, if you are attempting to logon, you should enable some tracing so that if you are unsuccessful, you can at least look at the trace and try to determine why the logon was unsuccessful.

Once everything has been completed, you should encrypt the ACCESSPSWD, as follows:

```
set execute_sql set current sqlid = 'FSDV750M';
encrypt_password *;
```

Once you can signon to the new database, you should go to the DDL Model Defaults and verify that the values in there are correct. This is accomplished by selecting **Go-->PeopleTools-->Utilities-->DDL Model Defaults**. The platform ID = 1, so either press Enter or enter `1`. Verify that the values displayed are correct.

9.1.9 Verifying the SQR programs

When you are satisfied that the values are okay, run the following three SQR reports:

SETSPACE.SQR updates the PSRECDEFN table with the correct table space name. It reads through the PSRECEFN table and compares the tablespace name in DDLSPACENAME to the TSNAME column in SYSIBM.SYSTABLES. If the space names are different then it updates the PSRECDEFN table. This will result in the correct tablespace names when altering records via application designer.

S*ETDBNAM.SQR* updates the PSRECDDLPARM table with an override value for each record in PSRECDEFN if the database name in SYSIBM.SYSTABLES does not equal the database name in PSDDLDEFPARMS.

SETINDEX.SQR updates the PSIDXDDLPARM table with an override value for each record in PSINDEXDEFN for STOGROUP, PRIQTY and SECQTY. It

retrieves these values from SYSIBM.SYSINDEXPART.

The ALTER process in Application Designer uses the DDL Model defaults when recreating the indexes. If no override value exists in PSIDXDDLPARM for STOGROUP, PRIQTY and SECQTY, then the values from PSDDLDEFPARMS will be used.

If the customer has not run SETINDEX.SQR, any work that the customer has done to size indexes on the mainframe will be lost when the record is altered. SQR is targeted for release in PT7.52 (se incident T-SXK-4LI88 for details).

Chapter 10. Administration of DB2 for PeopleSoft

This chapter reviews a collection of DB2 installation and data administration topics that come from the following sources:

- Our PeopleSoft 7.5: HR installation experience
- Interviews with IBM consultants who have assisted in implementing or tuning PeopleSoft
- Interviews with PeopleSoft personnel
- Our DB2 experience

PeopleSoft provides implementation and tuning information in a variety of documents. Our suggestions for obtaining information are the following:

- Review *PeopleTools7.5: Installation and Administration of DB2 for OS/390*, including the appendix, because it contains some valuable information.
- The PeopleBooks for the application you are installing may have a section on performance optimizations.
- The PeopleSoft Web site (`www.peoplesoft.com`) Customer Connection Information library may be helpful.

Become acquainted with the PeopleSoft trace facilities. These are discussed in Chapters 6 and 7 of *PeopleTools7.5: Installation and Administration of DB2 for OS/390*. These tools provide an easy way to pinpoint SQL that may need tuning. Or, use your current DB2 monitor to obtain similar information. Document all changes that you make and the reason for making them; these changes *may* need to be reapplied when you install the next release.

10.1 DB2 installation recommendations

We strongly recommend that you create a unique DB2 subsystem for PeopleSoft.

A list of PTFs recommended for OS/390 PeopleSoft installations can be found on the PeopleSoft Web site. Make sure you have these PTFs installed.

Special consideration should be given certain DSNZPARMS. Suggested values are the following:

- CACHEDYN=Y

  Enable dynamic caching.

- CONDBAT= (number greater than MAXDBAT)

  Maximum number of active and inactive DBAT threads that can be connected to DB2 concurrently.

- DEFIXTP=2

  Use type 2 indexes. This is applicable to DB2 prior to version 6.

- DSMAX = Value greater than concurrently used datasets

  Depending on the table-to-tablespace mapping and the number of PeopleSoft applications installed, the potential number of open datasets can become quite large. Open datasets require SWA control blocks to be allocated above the 16 mb line. You can do this in one of the following ways:

- Set `JESPARM STCCLASS SWA=ABOVE`
- Code a JES exit
- Update SMF exit IEFUJV.

However, having a large number of open datasets is not without cost; for example:

- Approximately 1.8 KB of storage is required for each data set control block.
- The more open datasets there are, the longer shutdown and restart will take.

Note that not all tablespaces and indexes that are defined are used; therefore, you should set DSMAX to a number larger than the number of data sets that are *used concurrently*. Monitor the dataset open/close activity. If datasets are being opened and closed every few seconds, DSMAX should be increased. In general, for tablespaces you should specify CLOSE YES.

- EDMPOOL

  Monitor after installation and tune to maximize dynamic caching.

- IDTHTOIN=60

  This is the timeout value for active threads. Setting it to zero means no timeout value; we set it to 60 seconds so resources will not be held.

- INBUFF=60

  This is the maximum size for the input buffer used for reading active and archive logs. The recommended value speeds recovery.

- MAXDBAT= (20% greater than the number of concurrent users you expect)

- MAXKEEPD=5000

  Use the default. PeopleSoft support for KEEPDYNAMIC(YES) is emerging. See “DB2/PeopleSoft 7.5: HR bind parameter recommendations” on page 49 for a further discussion of this subject.

- NUMLKTS

  Start with the default; however, this parameter may need to be adjusted.

- NUMLKUS

  Start with the default; however, this parameter may need to be adjusted.

- CMTSTAT=INACTIVE

  This leaves the thread inactive after commit.

- RELCURHL=YES

  This releases the data page or row lock after a commit for cursors specified with hold.

- TCPALVER=NO

  This value means connection requests require user ID and password.

10.2 DB2/PeopleSoft 7.5: HR database recommendations

When following the PeopleSoft 7.5: HR installation process, the application installs with five DB2 databases. Some of the databases contain a large number of objects.

Be aware of the impact of a large database:

- The DB2 Database Descriptor (DBD) will be large.

  The DBD must be loaded contiguously into the DB2 Environmental Descriptor Manager (EDM) pool. DB2 will not load a DBD into an EDM pool if it is larger than 25% of the pool size.

- Log activity may be significant.

  DB2 logs the necessary undo/redo information for the unit of recovery in which the DBD was changed. Updating large DBDs can cause significant logging.

- Contention should be considered.

  Create, Alter, or Drop of a DB2 object locks the DBD. This may cause contention. The contention on active databases may cause DDL to fail, or users will be locked out while DDL is being executed.

For these reasons, you may want to consider splitting the larger databases.

10.3 DB2/PeopleSoft 7.5: HR tablespace recommendations

When following the PeopleSoft 7.5: HR installation process, 20 tablespaces contain 2,626 tables. One tablespace (HRAPP) contains over 900 tables. Consider creating additional tablespaces and moving tables from the existing tablespaces to these new tablespaces.

- As a rule of thumb, we recommend that a tablespace have no more than 100 tables.

  Note that having more than one table per tablespace is somewhat arbitrary. However, given the large number of tables and the challenge of managing the data sets containing those tables (a DB2 tablespace maps to a VSAM data set), having multiple tables per tablespace represents a practical strategy.

- Tablespaces containing multiple tables should be segmented. Put tables of similar size in a tablespace, then follow the recommendations for SEGSIZE listed in the *DB2 Version 6 for OS/390 Administration Guide*.

Table 2. Segment size recommendations

Number of pages in table	Segsize
< = 28 pages	4 to 28
.> 28 < 128 pages	32
> = 128 pages	64

- Some tables should be placed in a separate tablespace.
  - We recommend that you assign to a separate tablespace any table with a name of the form:
    - Some-name_TEMP
    - Some-name_TMP
    - Some-name_WRK

    This gives you the ability to move the data set containing the tablespace to minimize contention during normal operation (and it will also facilitate a recommendation to follow concerning the use of the Reorg utility).
  - PeopleSoft has grouped tables that will grow or experience high update activity in tablespaces that are named xxLARGE. Consider placing these tables in partitioned tablespaces.
  - Tables that experience high activity rates should be moved to a separate tablespace.
  - For HR, additional tables that should be placed in a separate tablespace are listed in the appendix of *PeopleTools7.5: Installation and Administration of DB2 for OS/390*.
  - TABLESPACE Parameter - Locksize

    Specify LOCKSIZE(PAGE) unless the PeopleSoft documentation recommends LOCKSIZE(ROW). (See the appendix of *PeopleTools7.5: Installation and Administration of DB2 for OS/390*.)

    Use row-level locking only when needed for concurrency. Row-level locking will increase CPU overhead.

    Some PeopleSoft applications have long Units of Work. Thus, specifying row-level locking may require adjustments to the ZPARM parameters NUMLKTS and NUMLKUS, or to the tablespace parameter LOCKMAX. The maximum value for NUMLKUS (locks per user) is 100000. NUMLKUS=0 disables this limit.

    Each lock requires 250 bytes of storage. Increasing the number of locks allowed may require that the MAXCSA parameter in the IRLM startup procedure be increased if IRLM parameter PC=NO is used. However, before increasing MAXCSA, ensure that the OS/390 specifications for ECSA can accommodate the increase.

    If you find it necessary to set NUMLKUS=0, you may want to consider setting the IRLM startup procedure PC parameter to PC=YES. There is some additional CPU overhead in doing this, but it removes the potential of bringing down OS/390 if the ECSA fills up.

10.4 DB2/PeopleSoft 7.5: HR table recommendations

If you are running DB2 V5, make sure you install PTF UQ22406. Without this PTF, if your SQL predicate values and index columns are unequal in length, then a tablespace scan is likely. With the PTF applied, an index will probably be used.

10.4.1 Modifying DDL defaults

DDL Model defaults are stored in tables PSDDLMODEL and PSDDLDEFPARMS.

If you choose to use native DB2 DDL to define objects different from the way they are shipped with PeopleSoft, then the DB2 catalog will differ from the values in the PSDDLMODEL and PSDDLDEFPARMS table. If you then use PeopleTools to recreate or alter a table, the values from the PSDDLMODEL and PSDDLDEFPARMS table will be used.

Therefore, you will need to devise a means of updating the PSDDLMODEL and PSDDLDEFPARMS to reflect the values in the DB2 catalog. SETDBNAM.SQR, SETINDEX.SQR, and SETSPACE.SQR have been developed to assist you with this task.

10.5 DB2/PeopleSoft 7.5: HR bind parameter recommendations

At the present time the following bind parameters are recommended:

- Acquire(Use)
- Currentdata(No)
- Degree(1)
- Isolation(CS)
- KeepDynamic(No)

  Currently PeopleSoft does not support the use of KeepDynamic(Yes). However, in the Application Engine - Use panel, there is a REUSE option. The purpose of this option is to prevent re-preparing a statement for each execution. Since DB2 will release a prepared statement at commit time in order to make the REUSE option effective with DB2, there is a code change available from PeopleSoft for PTPSQLRT. After making the change, PTPSQLRT must be rebound with KEEPDYNAMIC(YES). In this case, the use of dynamic caching can reduce the impact of the need to re-prepare a statement.
- Nodefer(Prepare)
- Noreopt(Vars)
- Release(Commit)
- Validate(Bind)

Changing a “common-cursor” to a “dedicated-cursor” (PeopleSoft terms) can reduce the number of SELECT statement re-prepares in COBOL programs. See Appendix G of *PeopleTools7.5: Installation and Administration of DB2 for OS/39*0 for examples.

10.6 DB2/PeopleSoft 7.5: HR EDM pool recommendations

For varying PeopleSoft HR and Financial implementations, the EDM pool size ranged from 20 MB to 150 MB. The major variants are the number of objects included in the database and the amount of storage devoted to caching of dynamic plans.

10.7 DB2/PeopleSoft 7.5: HR buffer pool recommendations

Following the PeopleSoft 7.5: HR installation procedure will cause all 4 K page tablespaces and indexes to be assigned to buffer pool 0 (BP0). The following are general recommendations for initially configuring the DB2 buffer pools in support of PeopleSoft 7.5: HR. The recommendations assume that the DB2 subsystem is dedicated to PeopleSoft 7.5: HR, and are presented as percentages of the number of buffers you assign to the DB2 buffer pools. For example if you have 100,000 DB2 buffers, you would assign 10% or 10,000 buffers to BP0.

- BP0 - 10%. Restrict BP0 to the DB2 catalog and directory.

  This facilitates the dynamic SQL that PeopleSoft 7.5: HR executes.

- BP1 - 15%. BP1 supports large tables.

  The large tables will generally have low buffer reuse. Consequently, 15% is moderate relative to the other PeopleSoft buffer pool assignments.

- BP2 - 25%. BP2 is dedicated to large-table indexes.

  Indexes tend to have high buffer reuse. Consequently, we recommend a high buffer allocation.

- BP3 - 10%. BP3 is dedicated to the smaller PeopleSoft 7.5: HR tables.

  Code tables are usually smaller and tend to experience high access. This large buffer pool allocation facilitates data-in-memory processing of the frequently accessed smaller tables.

- BP4 - 15%. BP4 is dedicated to small-table indexes.

  Indexes tend to have high buffer reuse. Consequently, we recommend a high buffer allocation.

- BP5 - 10%. BP5 is dedicated to PeopleSoft tool tables.

  PeopleSoft tool data is primarily read-only.

- BP6 - 10%. BP6 is dedicated to PeopleSoft tool indexes.

  Indexes tend to have high buffer reuse. Consequently, we recommend a high buffer allocation.

- BP7 - 5%. BP7 supports DB2 temporary storage (DSNDB07).

  The SQL Order By or Group By, for example, have the potential to require a DB2 sort which DB2 supports through the use of DSNDB07.

- BP32K- minimum. This is support for any 32 K buffer pool requirement.

Except for BP0, all these are arbitrary buffer pool assignments. Note, however, that DB2 requires BP0 for the catalog and directory tablespaces. Obviously you may, for example, use BP2 to support DSNDB07. The key is the concept of differentiating large tables, small tables, PeopleSoft tool tables, and the associated indexes.

10.8 DB2/PeopleSoft 7.5: HR reorganization recommendations

The following are considerations for the use of the DB2 Reorg utility in support of PeopleSoft 7.5: HR data:

1. Do not Reorg PeopleSoft 7.5: HR temporary or work tables.

These tables are of the form:

- Some-name_TEMP
- Some-name_TMP
- Some-name_WRK

These tables tend to start empty within a PeopleSoft 7.5: HR process. They are populated during the process, and then emptied at the end of the process. We recommended in 10.3, “DB2/PeopleSoft 7.5: HR tablespace recommendations” on page 47, that these tables be assigned to an individual tablespace. That recommendation is a prerequisite to this recommendation since the Reorg utility is invoked at the tablespace level.

2. Do not Reorg PeopleSoft 7.5: HR tables that do not have a clustering index explicitly defined.

   To enable this recommendation, the tables that are not defined with clustering indexes within PeopleSoft 7.5: HR must be defined to one or more separate tablespaces, because Reorg is invoked at the tablespace level. The Reorg utility will not resequence a table that does not have a clustering index explicitly defined (its function in this case will be limited to reestablishing free space).

   Even though the tables do not need to be reorganized, indexes may need to be reorganized.

3. Evaluate the use of the online Reorg feature introduced in DB2 V5.

   Online Reorg significantly reduces the impact of this planned outage. The online Reorg utility copies the tablespace to one or more shadow data sets where the actual reorganization is done. The DB2 log is applied in an iterative fashion to the shadow copy of the data in order to synchronize it with the actual online tablespace.

   After the log is applied, the shadow data replaces the unorganized data when the Reorg utility changes data set names. The outage is now limited to read-only during the final application of the DB2 log to the shadow copy of the tablespace, and no readers or writers are allowed during the changing of the shadow data set names to the active tablespace data set names.

4. Evaluate the usefulness of infrequently reorganizing PeopleSoft 7.5: HR tablespaces that are processed largely by single-row SQL selects and updates. The single-row processing must be through a unique index.

   Single-row processing through a unique index is not sensitive to the physical sequence of the data. Large tables frequently manifest the requirement for single-row processing, and they may benefit little from reorganization. You will have to evaluate your use of the PeopleSoft 7.5: HR tables to determine if you can Reorg infrequently.

10.9 DB2/PeopleSoft 7.5: HR RUNSTATS recommendations

- We recommend that you consider running RUNSTATS TABLE ALL frequently.

  Although RUNSTATS TABLE ALL requires more processing cycles (as opposed to RUNSTATS COLUMN), users who run decision support queries against PeopleSoft 7.5: HR tables report that the more extensive statistics enable more optimal DB2 access paths.

- We recommend that you evaluate updating the DB2 catalog with RUNSTATS statistics for the TEMP, TMP, and WORK tables.

  At the time you are likely to run RUNSTATS, the temporary tables are likely to be empty. It would be better to leave the statistics with the default (which is -1) than to run RUNSTATS on an empty table, thus updating the statistics with zeroes. With a -1 in the statistics columns, DB2 will base its access path on defaults.

  Users have reported that updating catalog statistics will influence DB2 to use indexed access, providing improved access paths.

  Some methods used to populate catalog statistics for these tables are:

  – Populate the catalog statistics with estimated values.

  – Run RUNSTATS with SHRLEVEL CHANGE while the PeopleSoft 7.5: HR is executing.

10.10 DB2/PeopleSoft 7.5: HR index usage

If you are using DB2 V5, we recommend that you define all indexes on PeopleSoft 7.5: HR tables to be type 2 indexes. This will reduce locking and position you for DB2 V6.1, which does not support type 1 indexes.

In tuning PeopleSoft 7.5: HR for your environment, you will probably add additional indexes and drop some of the initial indexes. Some of the indexes that install with the product are redundant and are not used. However, the PeopleSoft documentation states that indexes containing an underscore in the name, such as PS_ , should not be touched. Also, indexes with a letter such as PSA or PSB should not be touched.

10.11 DB2/PeopleSoft 7.5: HR point-in-time recovery recommendations

The usual reason for a point-in-time recovery is an application programming error or a flawed operational procedure. Unfortunately, this exposure is always present, regardless of your hardware/software configuration. Additionally, a point-in-time recovery has the potential to be the most disruptive outage you are likely to encounter.

The reason is that in a PeopleSoft 7.5: HR environment, you may need to recover all objects to a prior point in time. Depending on how you have mapped your tables to tablespaces, this could be from twenty to over a thousand tablespaces to recover, along with a few thousand indexes. Your usual point-in-time recovery techniques, which you probably regard as conventional at this time, may be insufficient in this environment.

The point-in-time recovery environment will be addressed as follows:

- Point-in-time recovery preventive measures
- Point-in-time recovery techniques
- Point-in-time recovery considerations

10.11.1 Point-in-time recovery preventive measures

A failure in application development (thus introducing a programming defect), or in operational procedures (perhaps by running a job twice), introduces the

requirement for point-in-time recovery. The available preventive measures are increased attention to:

- Change management
- Problem management
- Testing

Each of these disciplines is procedure-oriented and management-driven. As attention to these disciplines is increased, the need for point-in-time recovery is usually decreased. Unfortunately, the need for point-in-time recovery is never entirely eliminated. Consequently, you will want to make every effort to avoid having to do a point-in-time recovery, but you should be prepared to do one if required.

10.11.2 Point-in-time (PIT) recovery techniques

The concept behind point-in-time recovery is well understood. It usually involves resetting a table or group of tables to a prior point in time when data was determined to be consistent. The challenge in the PeopleSoft 7.5: HR environment is determining the set of tables that are logically related. It is possible that you will not be able to determine a subset of the PeopleSoft 7.5: HR tables to be reset. You will likely be required to reset *all* PeopleSoft 7.5: HR tables to a prior point of consistency. There are several techniques to effect a point-in-time recovery including:

- Point-in-time recovery using user-written application programs
- Point-In-time recovery using DB2 utilities

  The DB2 Quiesce and Copy utilities are the primary tools.
- Point-in-time recovery using a dump/restore scenario

  This scenario typically employs non-DB2 utilities.
- Point-in-time recovery using a DB2 conditional restart

  Any conditional restart scenario is potentially risky. The benefit of this scenario is the “effectively free” establishment of the point of consistency.
- Point-in-time recovery using suspension of DB2 update activity

  DB2 update activity is suspended using the SET LOG command to SUSPEND/RESUME logging. This function was introduced into DB2 Version 6 with APAR/PTF - PQ31492/UQ36695.

10.11.2.1 PIT recovery using user-written application programs

This is a strategic direction and not a scenario. It acknowledges that data can be corrupted due to program error. If this happens, you may attempt to correct the contaminated data with application programs. If you fail to correct the data with application programming, a scenario such as one of those following could be used as a last resort. This approach is gaining favor among users striving for high availability.

In implementing an approach like this, it is important to determine which transactions will make the data more corrupt or will propagate the errors. This information then can either be communicated to the end users, or the DBA can use it to disable the dangerous transaction.

10.11.2.2 Point-in-time recovery using DB2 utilities

The scenario for a point-in-time recovery using DB2 utilities is as follows:

- Determine the set of tables that are logically related.

  Typically this is a subset of the tables that make up the application database. However, it may be all PeopleSoft 7.5: HR tablespaces and indexes.

- Optionally, execute the QUIESCE utility on all of the tables that are candidates to be reset if a point-in-time recovery is required.

  This establishes a point of consistency and causes the DB2 buffers for the quiesced tables to be externalized to DASD.

- Execute the COPY utility on all of the tablespaces that are candidates to be reset if a point-in-time recovery is required.

  The obvious reason for this step is to back up the data. However, COPY will fail if it cannot externalize the DB2 buffers to DASD. That is the reason we invoked the QUIESCE utility first (to remove one reason why COPY may fail).

  You may want to place your Image Copy output on DASD. With DB2 V6.1, DASD-resident image copies enable parallelism is both the COPY and RECOVER utilities.

- For the second time, execute the QUIESCE utility on all of the tables that are candidates to be reset if a point-in-time recovery is required.

  This invocation of QUIESCE will establish a log RBA which will be the point of consistency.

- When it is necessary to recover to this point of consistency, RECOVER to the RBA established by the second invocation of the QUIESCE utility.

  With the combination of the COPY and the second QUIESCE, the RECOVER TORBA will perform as efficiently as a Recover TOCOPY would perform, assuming no logging during the execution of this scenario.

- When it is necessary to recover to this point of consistency, RECOVER all indexes on all of the tables that have been reset to the prior point of consistency. The indexes must be synchronized with the data in the recovered tablespaces. DB2 V6.1. added the functional capability to recover the index from an image copy. The recovery of the index from an image copy in V6 is a significant performance benefit over prior versions of DB2 that rebuild the index (this includes reading the table to unload and construct all keys, sorting the keys, and then rebuilding the index).

The benefits of this scenario are:

- There is minimal disruption to the user in preparing for the scenario.

  The user may see some slowness in response during the execution of the QUIESCE utility, but this can likely be scheduled during off-hours to minimize the disruption.

- There is no disruption to the user when COPY is run with Share-Level Change.

  COPY - Share-Level Change allows updates concurrent with utility execution.

- The recovery of the tablespaces will be efficient.

RECOVER TORBA will perform as well as RECOVER TOCOPY, assuming no logging between the execution of the COPY utility and the second execution of QUIESCE.

However, there is a significant disadvantage to this scenario: the requirement to recover thousands of objects may take too long, thus making this scenario impractical. If you are evaluating the use of this scenario, time it to determine if it meets your availability requirement.

10.11.2.3 Point-in-time recovery using dump/restore utilities

This scenario uses a DB2 command and usually a non-DB2 dump/restore utility program.

- Determine the set of tables that are logically related.

  Typically this is a subset of the tables that make up the application database. However, in the PeopleSoft 7.5: HR environment, it may be all of the tables.

- Use the DB2 STOP DATABASE command to stop all of the tablespaces that are logically related.

  The STOP DATABASE command will cause the DB2 buffers to be externalized to DASD, and will cause the VSAM data sets that hold the DB2 data to be closed. While the tablespaces are stopped, the data will be unavailable to users. You may evaluate bringing down DB2 as an alternative to the STOP DATABASE command.

- Dump the tablespace and index data sets using your installation high-speed dump utility.

  You may consider using FlashCopy or SnapShot to dump the data sets. The dumped data sets represent your point of consistency.

- When it is necessary to recover to this point of consistency: stop DB2, restore the data sets that were dumped in the previous step, and then restart DB2.

  Because the tablespaces were stopped when backed up, there were be no outstanding units of recovery, and the data was consistent. The restored data consists of both the tablespaces and the associated indexes.

The recovery portion of this scenario is faster than the previous one, but preparing for it is more disruptive to the user. The data is unavailable to the user while the tablespaces are stopped and while the data is being dumped. The length of time that data is unavailable can be lessened by using FlashCopy or SnapShot.

10.11.2.4 Point-in-time recovery using DB2 conditional restart

The following scenario will likely appeal to the more experienced DB2 user. The scenario requires a DB2 conditional restart, which is a part of DB2 not frequently exercised by users. Its key advantage is an “almost free” establishment of a point of consistency.

At a high level, the scenario may be defined as follows:

1. Identify a set of candidate points of consistency.
2. Select that candidate point of consistency which best meets your requirements.
3. Make that best candidate point of consistency into a true point of consistency.

This is the point at which you will do the conditional restart. The conditional restart will make your candidate point of consistency the true point of consistency on the DB2 Log.

4. Recover all tablespaces to the true point of consistency.

   The conditional restart will position you to recover all of your tablespaces. Because of the conditional restart, you will use RECOVER to currency (not RECOVER TORBA).

5. Now RECOVER all indexes on all of the tables that have been reset to the point of consistency.

   The indexes must be synchronized with the data in the recovered tablespaces.

The first three steps listed are new and will receive the major part of our attention here. Once those steps are complete, the remainder of this scenario is conceptually similar to the later steps of the scenario 10.11.2.2, “Point-in-time recovery using DB2 utilities” on page 54 with the difference that you will recover to currency and not to an RBA.

Identify a set of candidate points of consistency

Consider a list that contains many items (or rows). Each list item has two entries in columns: the first column is a timestamp, and the second column is the DB2 Log RBA associated with that time. The list can be quite long (that is, showing many timestamps). This list of timestamps is our set of candidate points of consistency.

The list of candidate points of consistency might have an entry for each hour in the day or for each minute in the day. For each entry in the list, you have a timestamp and the corresponding DB2 Log RBA. This allows you to map a specific time to a DB2 Log RBA.

How do you build a list of timestamps and the associated Log RBAs? You start by defining a dummy database and tablespace. This will be a real DB2 database and tablespace, but there will be no activity against the dummy tablespace. PeopleSoft will not know about this tablespace.

Once the dummy tablespace is defined, you will initiate a user-developed procedure that will periodically QUIESCE that dummy tablespace. Since you will allow no activity against the dummy tablespace, the QUIESCE will be very fast. The QUIESCE will cause the Log RBA and the timestamp to be entered into SYSIBM.SYSCOPY. The entries within SYSIBM.SYSCOPY for the dummy tablespace make up your list of candidate points of consistency. If you do the QUIESCE each hour, there will be an entry for the dummy tablespace in SYSIBM.SYSCOPY each hour.

Note that the defined dummy table is *only* used as an aid in determining an RBA in the log that corresponds to a given time. It is possible to calculate the RBA without the dummy table but the calculation is complex; use of the table is simpler and less prone to error.

Select the candidate point of consistency that best meets your requirements

Suppose you determine that because of an application programming error, data in your PeopleSoft system is inconsistent. This part cannot be automated. You must determine when the inconsistencies began to enter your system. Suppose

you determine that at 5:00 PM on January 14, 2000, erroneous data began to enter your system. You make the determination that you want to take your system back before that date and time.

You have one more task. Query SYSIBM.SYSCOPY for the dummy tablespace entry before 5:00 PM on January 14, 2000. Once you determine that entry from the list, note the DB2 Log RBA.

Where do you stand now? You have the Log RBA of the nearest time before the inconsistencies entered your system. You are now ready to make that Log RBA, which relates to a candidate point of consistency, into a true point of consistency.

Make the best candidate point of consistency into a true point of consistency

There is probably data inconsistency at the Log RBA you identified. You are running an active PeopleSoft system and it is likely that at the time you have identified, there was work in process (including in-flight units of recovery). However, you can make that Log RBA a true point of consistency.

By doing a DB2 conditional restart, you can make the Log RBA you identified into a point of consistency. You will use the CHANGE LOG INVENTORY DB2 utility to create a conditional restart control record using the following statement:

```
CRESTART CREATE,FORWARD=YES,BACKOUT=YES,ENDRBA=XXXX
```

where XXXX is the true point of consistency you determined from your SYSIBM.SYSCOPY query.

The conditional restart will cause DB2 to truncate the log at your true point of consistency. Log entries beyond that point will be disregarded. Additionally, DB2 will remove from SYSLGRNGX and SYSCOPY any entries that occurred after the true point of consistency.

Recover all tablespaces to the true point of consistency

After the conditional restart, this will be a recovery to currency and not a recovery to an RBA (recovery to an RBA is common in most point-in-time recovery scenarios).

Recover all indexes on the tables that have been reset to the prior point of consistency

The indexes must be made consistent with the data.

10.11.2.5 Point-in-time recovery using suspension of DB2 updating

This scenario is functionally similar to the scenario in 10.11.2.3, “Point-in-time recovery using dump/restore utilities” on page 55. The salient features of both scenarios are:

- Determine the set of objects requiring backup and recovery.
- Stop processing.
- Dump the set of objects.
- Restore the set of objects and restart DB2.

The unique characteristic of the “Point-in-time recovery using suspension of DB2 updating” scenario is a new technique to stop DB2 update processing.

The specific characteristics of this “Point-in-time recovery using suspension of DB2 updating” scenario are:

- Determine the set of objects requiring backup and recovery.

  The issues for this step are the same for all point-in-time recovery scenarios using PeopleSoft. You will likely back up your entire PeopleSoft system.

- Stop processing.

  The unique feature of this scenario is the method for stopping the processing. APAR/PTF - PQ31492/UQ36695 provides the ability to “suspend” and “resume” DB2 logging. This has the effect of “freezing” updates to your DB2 data while you make copies of the DB2 data and the DB2 log. Specifically after logging is “suspended”, the DB2 log and the data are consistent. When the SET LOG SUSPEND command is issued, the following occurs:

  – A system checkpoint is taken.

    The scope of the command is single system-only. In a data sharing environment, the command will have to be entered for each member.

  – Unwritten log buffers are externalized to DASD.

  – The BSDS is updated with the high-written log RBA.

  – A log-write latch is obtained to prevent further logging.

    Logging is suspended until the resume command is entered.

- Dump the set of objects.

  To take maximum advantage of this scenario, you will want a high-speed dump capability like FlashCopy or SnapShot. With this scenario, you will add the DB2 log to your list of objects to be dumped. The function of the APAR discussed above makes the log consistent with the DB2 data.

- Restore the set of objects and restart DB2.

  With this scenario, you will restore both the DB2 data and the log. Since the log is consistent with the data, the point-in-time recovery scenario requires only a normal DB2 restart. In-flight units of recovery will be backed out just as with any normal DB2 restart. After the normal DB2 restart, the data will represent committed units of recovery at the time that logging was initially suspended.

10.11.3 Point-in-time recovery considerations

Your strategy should be to avoid a point-in-time recovery if possible, but also to provide a process that allows it to be performed when necessary. If a point-in-time recovery must be done, then choose the most effective recovery scenario consistent with your requirements. Following are the considerations for the five options previously discussed.

1. Point-in-time recovery using user-written application programs

   This is a preferred approach among users when possible. It has the following benefits:

   - There is little or no disruption to the user.

     Data that has not been corrupted will be available to the user. (However, the user may see erroneous data prior to the recovery.)

- There is no loss of data entered between the time the data contamination was introduced into the system and the time the contaminated data was corrected.
- Fewer processor resources are likely to be required than using either of the other techniques.

Remember that this is an ideal and not a rigorous scenario that can be documented and tested. You must be prepared to reset your system by doing the point-in-time recovery when a “programming fix” is not possible or practical.

Application programming may *not* be able to repair PeopleSoft system data; if PeopleSoft defects exist, other point-in-time recovery techniques may be the only alternative.

2. Point-in-time recovery using DB2 utilities

   This scenario is preferred over the “point-in-time recovery using Dump/Restore utilities” scenario when few tablespaces and indexes are to be recovered. However, the recovery of the number of tablespaces and indexes in an PeopleSoft system may imply an outage of many hours, making this scenario potentially impractical. If you believe that this scenario may be workable in your environment, run a benchmark to assure that your availability requirements can be met.

3. Point-in-time recovery using Dump/Restore utilities

   The issue regarding this scenario is the number of tablespaces and indexes that must be recovered to an established point of consistency. As the number of tables and indexes to be recovered grows, dump-restore may become the practical alternative. Dumping packs and restoring data (using features like FlashCopy or SnapShot) is faster than a recovery-based scenario *if the regularly scheduled dumping activities and database non-availability can be tolerated*. Note that recovery is required only when errors occur, while backup occurs on a scheduled basis even if no errors ever occur.

   The major disadvantage to this scenario is that the dumping of the data is disruptive to the user and occurs on a regular basis (usually daily). The PeopleSoft system likely must be stopped in order to dump data that is consistent. Other applications must also be stopped if they are affected by PeopleSoft being unavailable.

   An additional disadvantage (affecting all scenarios to some degree, but particularly noticeable in this one) is that work is lost when data is reset to a prior point of consistency. Considering for example a once-per-week backup schedule, on average production data for a half of a week is lost when a point-in-time recovery is executed.

4. Point-in-time recovery using a DB2 conditional restart

   The main benefit to this scenario is that there is effectively no impact on the user to create the list of candidate points of consistency.

   The time required to actually recover the tablespaces and indexes will likely be somewhat longer that the time required to do the recovery described in 10.11.2.2, “Point-in-time recovery using DB2 utilities” on page 54, because this scenario will likely require more DB2 log processing during the recovery.

Since this scenario contains a conditional restart, anyone using it *must first practice it*. An improperly done conditional restart has the potential to severely damage your DB2 subsystem.

In summary, users should consider this scenario when:

- Having no disruption in defining the candidate points of consistency is of the greatest significance.
- The long outage to actually perform recovery is acceptable.
- You are willing to practice conditional restart to develop and maintain the skills necessary for its success.

5. Point-in-time recovery using suspension of DB2 updating

   This scenario has the advantages of the “Point-in-time recovery using Dump/Restore utilities” scenario with the added benefit that suspended logging is less disruptive to the user than stopping/restarting the DB2 subsystem. Data consistency is assured with this scenario through copying the log in addition to the data, and then by executing a normal DB2 restart.

   At the time of writing, this SET LOG SUSPEND/RESUME feature is being evaluated by some as the basis for their off-site disaster recovery support.

10.12 Other recovery considerations

Due to the fact that the ERP environment causes significant attention to your point-in-time recovery strategy, the previous section provides significant detail on that subject. However, you have other recovery considerations. Specifically you must also address:

- Recovery to currency

  Recovery to currency focuses on how you would handle a hardware failure like the loss of a DASD device.

- Disaster recovery

  Disaster recovery addresses how you would handle the loss of your computing center.

10.12.1 Recovery to currency

From the perspective of the ERP environment, “recovery to currency” considerations are the same as for the non-ERP environment. If you are an experienced DB2 for OS/390 user, your present “recovery to currency” strategy can be applied to your PeopleSoft ERP environment.

If you are a new user of DB2 for OS/390, you will want to do the following:

- Attend classes focused on training in the use of the DB2 recovery/restart utilities.
- Study the *DB2 Utility Guide and Reference,* focusing on the RECOVER and COPY utilities.
- Study the *DB2 Administration Guide,* focusing on the section that addresses operation and recovery.

- It is suggested that you periodically attend either local or national user group meetings. They frequently feature speakers addressing recovery-related topics.

10.12.2 Disaster recovery

From the perspective of the ERP environment, “disaster recovery” considerations are the same as for the non-ERP environment. If you have disaster recovery procedures that address your current DB2 environment, they will also likely address your PeopleSoft ERP disaster recovery requirements. If you do not presently have DB2 disaster recovery procedures, you will want to evaluate the following potential starting points for disaster recovery support:

- Disaster recover using dump/restore

  This is a frequent starting point for disaster recovery support. The scenario, 10.11.2.3, “Point-in-time recovery using dump/restore utilities” on page 55, could also provide a starting point in addressing your disaster recovery requirements.

- Documented disaster recovery scenario

  See your *DB2 Administration Guide*. The index will refer you to a section on preparation for disaster recovery, and to a disaster recovery scenario. If you ship off-site copies of both your image copies and archive logs, the documented scenario will enable you to recover to the last completed DB2 unit of recovery on the last archive you sent off-site. The scenario is popular; however, it does require practice.

- Disaster recovery with additional hardware

  This is a broad topic that is widely varied in both user requirement and scenario implementation. See “Point-in-time recovery using suspension of DB2 updating” on page 57 for a scenario that could also provide a starting point in addressing your disaster recovery requirements. This scenario will require high-speed hardware to get copies of your DB2 log and data off-site.

Chapter 11. DB2 features used by PeopleSoft applications

This chapter describes the benefits that DB2 provides for PeopleSoft applications. It covers DB2 V5 and V6 functions including the late additions (APARs), which address more specifically PeopleSoft data access and maintenance performance requirements.

IBM and PeopleSoft have a long history of cooperation in supporting the 300 enterprises that use PeopleSoft and DB2 for OS/390 today. This solid partnership made possible the teamwork development, integration, performance analysis, and customer service needed to achieve excellent PeopleSoft results on the S/390 platform.

The S/390 platform offers unmatched capacity and scalability, availability, performance, and manageability to reliably handle your information needs in an ERP solution such as PeopleSoft with DB2.

11.1 SQL and system performance enhancements in DB2 V5

This section lists DB2 V5 functions that are particularly interesting for PeopleSoft on OS/390.

You can find a detailed description of these functions in the redbook *DB2 Server for OS/390 Version 5 Recent Enhancements - Reference Guide*.

DB2 V5 function pertinent to PeopleSoft is:

- Dynamic statement caching. It has been improved through the following APARs: PQ4391, PQ09750, PQ11392, PQ11569, PQ09392, PQ12701, PQ12727, PQ13987, PQ14531, PQ14505, PQ14132, PQ17905, PQ14893, PQ19667, PQ14941, PQ14870, PQ07701.

DB2 V5 late additions that benefit PeopleSoft are the following. Most of them are DB2 V6 functions which have been retrofitted into DB2 V5 via APARs/PTFs. These enhancements are described in the DB2 V6 section:

- ORDER BY clause (APAR PQ23778)
- Unmatched column JOIN (APAR PQ22046, PQ24933)
- Outer join (partially retrofitted) (APAR PQ18710, PQ21263, PQ22087)
- Index screening in RID list processing (APAR PQ15670)
- Uncorrelated subquery - indexable IN predicates (APAR PQ23243)
- Set current precision for decimal (APAR PQ21949)
- DSMAX from 10000 to 32000 datasets (APAR PQ18543)

11.1.1 Dynamic statement caching

ERP applications use primarily dynamic SQL statements. Processing of a dynamic SQL statement requires two steps: prepare, then execute. Prepare is expensive because it consists of parsing and syntax checking, catalog searches for tables and columns, authorization checking, access path optimization, and creation of the statement executable.

For cases in which the same dynamic SQL statement is issued repeatedly, the ability to cache the previously prepared statement can significantly reduce the

cost of running those statements as they will not have to be prepared over and over again.

Dynamic statement caching is activated at two levels:

- The first level is global and permits the caching in the EDM pool of all the SQL dynamic statements submitted to DB2. This level is enabled by setting the CACHE DYNAMIC SQL field to `yes` in the installation panel DSNTIP4. The corresponding ZPARM parameter is CACHEDYN. No SQL changes are needed to benefit from this function.
- The second level applies to a package/plan and is set during the bind operation by using the KEEPDYNAMIC(YES) option. The storage used by the kept statements is controlled by setting the MAX KEPT DYN STMTS field of installation panel DSNTIPE. This function implies programming modifications.

Prepared dynamic SQL statements are cached in the EDM pool, so you should consider verifying the size of your EDM pool. Refer to the EDM pool size calculation in Chapter 2 of *DB2 for OS/390 V5 Installation Guide* for an estimate of the size of each prepared statement. This is also covered in “DB2/PeopleSoft 7.5: HR EDM pool recommendations” on page 49 of this redbook.

PeopleSoft continues to modify its own code to further increase the benefits it can get from this function. An SQL statement must have a perfect match to be able to reuse the prepared statement in the global cache; because PeopleSoft 7.0 uses literals instead of parameter markers (or host variables), the cache hit ratio is not optimal. PeopleSoft now uses a mixture of both, and is moving further towards parameter markers in PeopleSoft 7.5 and subsequent releases. Using parameter markers considerably improves the cache hit ratio.

Dynamic statement caching achieves very good savings for PeopleSoft batch processes where a limited number of SQL statements get executed many times. In contrast, ad-hoc queries, because of their random nature, benefit less from this improvement. KEEPDYNAMIC(YES) also allows you to take advantage of dedicated cursors, which causes less recompile of SQL statements repeatedly used throughout the batch COBOL programs.

11.1.2 ORDER BY clause

DB2 required that all columns referenced in an ORDER BY clause must also be named in the SELECT list of a query.

DB2 V5 now allows you to specify columns in the ORDER BY clause that are not in the SELECT list, such as the following statement:

```
SELECT name FROM q.staff.systables ORDER BY dept, years
```

You cannot use this enhancement of ORDER BY in conjunction with:

- UNION or UNION ALL
- GROUP BY
- DISTINCT
- Column functions such as MAX, MIN and SUM

This enhancement is important for applications, such as PeopleSoft, that generate SQL statements.

11.2 SQL and system performance enhancements in DB2 V6

In this section we describe the DB2 V6 functions that are beneficial to PeopleSoft on OS/390. Many of those functions have been retrofitted into DB2 V5 by APARs/PTFs. PeopleSoft V8.1 requires DB2 V6 and DB2 Connect V6 with Fix pack 2.

You can also find a very detailed explanation of those functions in the redbook *DB2 UDB for OS/390 Version 6 Performance Topics*.

DB2 V6 functions that address PeopleSoft requirements are:

- Index screening in RID list processing
- Uncorrelated subquery - indexable IN predicates
- New data types and functions: BIFs, LOBs, triggers
- 16 terabyte tables
- 255 tables per query/view
- Buffers and EDM pools in Data Spaces

DB2 V6 new additions via APARs/PTFs are the following:

- Update with subselect - APAR PQ30383, PQ31272
- Identity columns - APAR PQ30652, PQ30684
- Declared temporary tables - APAR PQ32670
- Defer defining data sets - APAR PQ30999
- ODBC performance - APAR PQ30082
- External savepoints - APAR PQ30439

11.2.1 Index screening in RID list processing

Index screening predicates reference columns in the index, but are not part of the matching index columns. For example:

```
SELECT * FROM T WHERE C1 = 1 AND C3 > 4 AND C4 =6;
```

With an index on T(C1,C2,C3), C3 > 4 is an index screening predicate. It can be applied when accessing the index, but it is not a matching index predicate like C1 = 1. The value of MATCHCOLS in the PLAN_TABLE is 1. C4 = 6 is not an index screening predicate. Any column in the index that is not one of the matching columns for the query will be considered for index screening.

Prior to this enhancement, index screening was not used when RID list processing was involved. RID list processing is used by:

- List prefetch for single table access and hybrid join
- Index ANDing and ORing during multiple index access

DB2 now allows index screening during RID list processing to filter out additional rows at index access time. These rows no longer need to be passed on by the data manager for data page access and evaluation of the predicate.The number of RIDs that require sorting also decreases by this enhancement because some of the rows have already been “screened out” at index access time. The optimizer filter factor calculations take this enhancement into consideration during list prefetch, index ANDing and ORing, and it is therefore possible to see the optimizer favoring those access paths more often to achieve better response time.

PeopleSoft benefits naturally from this enhancement every time an application performs a list prefetch, or an index ANDing or ORing.

11.2.2 Unmatched column join for CHAR

Suppose you want to join two tables, but the data types or the length of the join columns do not match. DB2 can perform a join in that case, but there is a performance impact. Because of the mismatch on data type or column length, the join predicate is considered stage 2. This means all qualifying rows of the inner table are passed back to the relational data system (RDS) component of DB2 for evaluation of the join predicate. If the join columns have the same data type and column length, the Data Manager (DM) component can deal with the join predicate at stage 1.

Example:

```
SELECT * FROM T1,T2
WHERE T1.CHAR10 = T2.CHAR5
```

Suppose there is an index on T1 (CHAR10). The T1.CHAR10 = T2.CHAR5 predicate is considered stage 2 (and the index on T1 cannot be used either) because the column length is not the same. The same is true when the join predicate is T1.VARCHAR5 = T2.CHAR5.

This problem is now partially fixed with this enhancement. In order for a predicate with unequal attributes to qualify as a stage 1 (and also indexable) predicate, it has to meet the following criteria:

- The predicate is dealing with a string data type (CHAR, VARCHAR, GRAPHIC, or VARGRAPHIC).
- The join predicate is an equal predicate.
- The predicate is a Boolean term.

These enhancements have been incorporated in DB2 for nested loop and merge scan joins. Hybrid join does not support this feature.

The problem has only been solved for join predicates on string data types. It has not been addressed for numeric data types, such as T1.INTEGERCOL = T2.SMALLINTCOL.

Another problem that is not being addressed by this enhancement is the mismatch of attributes on local predicates. For example, the local predicate CHARCOL5 = 'ABCDEF' is still stage 2 if the column length is smaller than the length of the literal.

There are many SQL statements specific to PeopleSoft applications, such as Financials, Distribution, and Manufacturing, which will benefit from this enhancement. In order to improve performances in DB2 prior releases, steps were required to manually alter columns to make column lengths match. These manual steps are no longer required.

11.2.3 Outer join

With the introduction of outer join support in DB2 V4, it has become much easier to write SQL outer join statements. This has increased programming productivity. However, the usability of outer join was sometimes limited by some of the

performance downsides of using outer join, since more data needs to be examined and more processing takes place.

DB2 V6 introduces a large number of outer join performance enhancements, thereby making outer join SQL statement performance very close to that of a similar inner join statement. Note that some--but not all--of these enhancements have been retrofitted in DB2 V5.

In addition, the SQL syntax of the ON clause has been extended to allow you to write SQL that you could not write before.

The Outer Join enhancements brought in DB2 V6 are extensively described in the redbook *DB2 UDB for OS/390 Version 6 Performance Topics*. The main items are:

- SQL ON clause extensions
- Outer join predicates classification
- Join simplification
- Removal of unnecessary work files
- Aggressive merging of views and table expressions
- Aggressive predicate evaluation
- Predicate transitive closure for outer join
- Join order permutation
- Parallel outer join

11.2.4 Set current precision for decimal

DB2 provides two sets of rules for decimal arithmetic--DEC15 and DEC31. For dynamic SQL, this option comes from the DECIMAL ARITHMETIC installation option on panel DSNTIP4.

This means that all programs in the system that use dynamic SQL must use the same option, which is a problem for systems that run different application packages within the same system. The packages need to use different values for this option. The DYNAMICRULES(BIND) option could be used, but it brings other changes in behavior that might be undesirable.

A new special register, CURRENT PRECISION, has been added. It provides the way to control DECIMAL precision in applications.This will give applications the ability to control the decimal arithmetic rules for dynamic SQL at the statement level.

The initial value of CURRENT PRECISION is determined by the value of field DECIMAL ARITHMETIC on installation panel DSNTIP4, as shown previously. The default for the initial value is DEC15 unless your installation has changed to DEC31 by modifying the value in that field.

You can change the value of the register by executing the SQL statement SET CURRENT PRECISION, which can be embedded in an application program or issued interactively. It is an executable statement that can be dynamically prepared. Only dynamic SQL will be affected by this special register.

The SET CURRENT PRECISION SQL statement replaces the value of the CURRENT PRECISION special register with the value of the string constant or host variable. The value must be DEC15 or DEC31.

When an SQL statement is dynamically prepared, the value of CURRENT PRECISION will be applied. For example, if you specify:

SET CURRENT PRECISION='DEC31'

then the subsequent statements that are prepared will use DEC31 rules for decimal arithmetic.

PeopleSoft requires DEC31 for PeopleSoft 7.5 and PeopleSoft 8.x. The solution is either to dedicate a DB2 subsystem to PeopleSoft with DECARTH=DEC31 in the DB2 dsnzparms or, if you don't want to dedicate a DB2 subsystem to PeopleSoft to use the SET CURRENT PRECISION special register, to set the DEC31 setting for each PeopleSoft connection to DB2.

The second solution allows PeopleSoft and other applications to live together without having possible negative impact on the other applications due to precision conflict by setting DECARTH=DEC15 in the DB2 dsnzparms. The key value here is allowing applications with different DECIMAL options to live together in the same DB2 subsystem instead of requiring different subsystems.

PeopleSoft began to use SET CURRENT PRECISION = 'DEC31' in PeopleTools 7.57.

11.2.5 Uncorrelated subquery - indexable IN predicates

Before this enhancement, DB2 does not use a matching index when evaluating the IN predicate against the result set of a non-correlated subquery. The non-correlated IN subquery predicate is considered a stage 2 predicate. Take the following example:

Example

```
UPDATE T1
SET SDATE = '01/01/1999' , STIME = '20:38:35'
      WHERE
PROG IN (SELECT MASTER FROM T2 WHERE INCLUDE = 'TODAY');
```

- A unique clustering index exists on T1(PROG).
- An index exists on T2(INCLUDE, MASTER).

DB2 resolves the non-correlated subselect using matching index only access, sorts and removes the duplicates, and puts the results in a workfile. The PROG IN (subselect) is then evaluated. DB2 can use a non-matching index scan on T1(PROG) or a tablespace scan to access T1 and look for a qualifying row in the workfile. DB2 cannot use a matching index to look up the values in the subselect because the predicate is stage 2.

The DB2 code has been enhanced to evaluate whether it is beneficial to process the non-correlated IN subquery in more or less the same way as DB2 handles IN list index access today. However, for non-correlated IN subqueries, DB2 fetches from a duplicate free sorted workfile and not a list of values coded by the user. The subquery is executed at cursor OPEN time. The matching with the outer query block is not done until FETCH time.

Going back to example 1 at the beginning of this section, the subquery on T2 is still executed first, the results are sorted in PROG order, and the duplicates are removed and stored in a workfile at OPEN cursor time. When the program starts

FETCHing, instead of accessing the outer table T1, and matching the rows with the results of the subquery, DB2 now uses the values from the workfile to access the index on T1(PROG) using a matching index scan to evaluate the IN predicate. The resulting set of the non-correlated subselect is evaluated as the outer table in a “nested loop” way, while T1 is now considered to be the inner table.

The non-correlated IN subquery predicate has become indexable and stage 1. The DB2 optimizer evaluates if this transformation helps performance based on the existence of an index on the column specified on the left hand side of the IN predicate and the selectivity on the IN subquery predicates. This enhancement can be used in SELECT, UPDATE, and DELETE statements.

Figure 10 on page 69 refers to a query used in a performance measurement study to evaluate this enhancement. The results show dramatic improvement both in elapsed and CPU time.

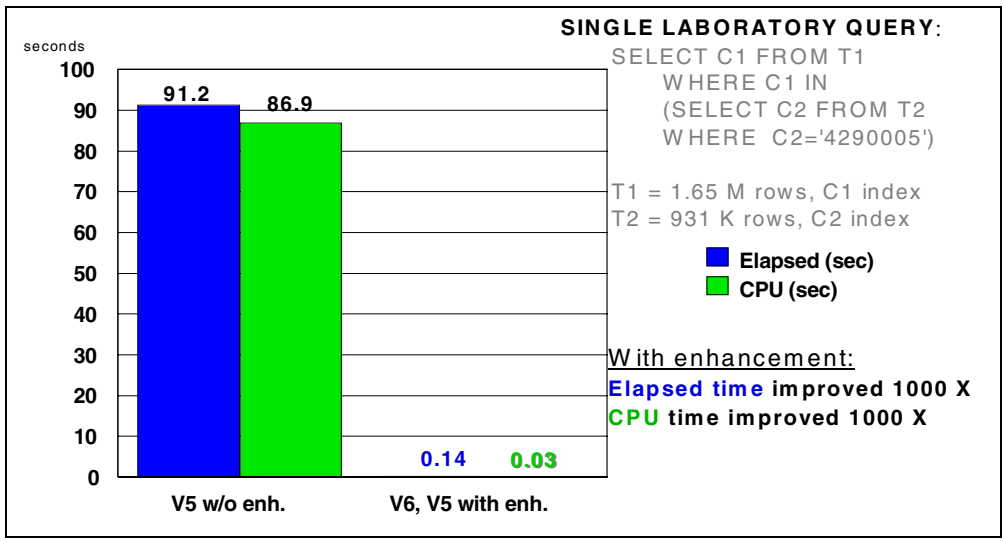


Figure 10. Uncorrelated IN subquery improvement

Roughly 10 % of PeopleSoft subqueries are uncorrelated and this enhancement will improve the performance of those queries.

Performance of correlated subqueries will be addressed in subsequent releases to PeopleSoft 7.5. PeopleSoft Global Support Group offers technical solutions which modify the code to break up the query into multiple statements or loop logic processing in COBOL programs.

11.2.6 DSMAX increased from 10000 to 32767

Support has been added to permit allocation of data sets in excess of the 10,000 data set limit. This change is made in conjunction with changes introduced in OS/390 Version 2 Release 6 and later releases. Together, they enable DB2 to dynamically allocate a much larger number of concurrent data sets.

Systems running on versions of OS/390 which are prior to V2.6 continue to be restricted to the 10,000 data set limit.

The OS/390 Version 2 Release 6 allocation has been changed to remove the architectural limit of 10,000 concurrent, dynamically allocated DB2 data sets. This

change addresses a scalability concern of large DB2 systems. This DB2 check, DSMAX, changes from the prior OS/390 limit of 10,000 data sets to the maximum of 32,767 data sets. Note the following considerations:

Consideration for allocating a high number of data sets
With OS/390 Scheduler Work Area (SWA) below the line, 10,000 open data sets is not generally possible. Therefore, you should consider moving your SWA above the 16 MB line.

Having a large number of allocated data sets has virtual storage and performance implications.

DB2 stop time can take longer as there are lots of data sets to close. If you are not able to specify STC NODETAIL for SMF, there is also a performance concern at DB2 shutdown. Abending DB2 instead of waiting for DB2 stop completion is not a recommended practice.

Checking the Storage Allocation
Calculation and checking for storage below the 16 MB line is more likely to cause a DSNT436I warning or DSNT437I error message from the DSNTINST installation clist saying storage below the line exceeds 5 MB and 8 MB with larger values of DSMAX. The parameter and maximum are shown on installations panel DSNTIPC.

This enhancement benefits PeopleSoft because PeopleSoft has many data sets; some customers have one table-to-one tablespace mapping strategy. As a result, multiple DB2 databases within the DB2 subsystem (for example, development, regression testing, systems testing) can easily exceed the limit of 10,000.

11.2.7 16 terabyte tables

Version 6 of DB2 for OS/390 greatly expands the capacity to store data in a single tablespace. As shown in Figure 10 on page 69, DB2 increases the limit for storing data in a single tablespace to 16 terabytes. This limit is up from 1 terabyte in Version 5 and 64 gigabytes in prior releases. If your data includes large objects (LOBs), you can store up to 4,000 terabytes in a LOB column. You can create tables that can be up to 16 terabytes in size, either in compressed or uncompressed format, assuming that sufficient disk space is available.

11.2.8 255 tables per query/view

In prior releases of DB2, the maximum number of base tables in a view was 15. In Version 6, the number of tables that a view can support is 255. You can also specify 255 tables in SELECT, UPDATE, INSERT, and DELETE statements.

Some PeopleSoft applications are reaching the limit of 15-tables join. When PS/Query builds a query, some additional tables/views are automatically joined to the tables specified in the user's SQL statement. These additional objects are not something a user selects. People Tools automatically joins some views to the table the user wants to access. Raising the table limit in an SQL statement from 15 to 255 tables will benefit PeopleSoft.

11.2.9 Buffers and EDM pools in data spaces

The capacity of exploiting large processor resources to run large application workloads is one of the strengths of DB2. Prior to DB2 V6, you allocate a buffer

pool in either the DBM1 address space (virtual pool) or in a hiperspace (hiperpool).

The use of hiperpools helps to relieve the 2 GB addressability limitation of MVS address spaces. DB2 hiperpools reside in expanded storage only and may not contain changed pages. The total size of all hiperpools cannot exceed 8 GB.

DB2 V6 provides an option to define buffer and EDM pools in a data space. Like hiper spaces, data spaces are data-only address spaces. That is, no program code can run in those areas.

Note

Data spaces provide a foundation for DB2 to exploit real storage larger than the 2 GB limit when the 64-bit addressability becomes available. Until the new processor is available, we recommend that customers consider using hiperpools first, before turning to data spaces, when running out of virtual storage in the DBM1 address space.

11.2.9.1 Buffer pool in data space

Unlike hiperspace, I/O can be directly done against buffers in data space (page movement occurs between the central storage and expanded storage). DB2 can also put changed pages in a virtual pool that resides in a data space, while a page in a hiperpool must be unchanged or written to DASD before it is allowed to be moved into the hiperpool.

Each data space can accommodate almost 2 GB of buffers and any single buffer pool can span multiple data spaces. However, no more than one buffer pool can be in a single data space. The sum of all data space buffers cannot exceed 8 million pages. This limit is independent of the buffer size.

A hiperspace is addressable in 4 KB blocks; in other words, it is page-addressable; a data space is byte-addressable. You cannot put a primary buffer pool into both a hiperpool and data space.

You define a buffer pool in a data space using a new VPTYPE keyword on the ALTER BUFFERPOOL command. The possible VPTYPE parameter values are PRIMARY or DATASPACE, each representing the following:

- VPTYPE(PRIMARY): Allocates a virtual buffer pool in the DBM1 address space. The sum of all DB2 virtual buffer pools cannot exceed 1.6 GB.
- VPTYPE(DATASPACE): Allocates a buffer pool in a data space. However, you need 128 bytes of buffer control storage in the DBM1 address space for each data space buffer.

The benefits of using buffer pools in data spaces are:

- To improve the buffer pool hit ratio. Since you can store more pages in a data space than in a virtual pool that resides in DBM1 address space, pages can stay in memory longer.
- To allow for more parallel processing to execute prefetch I/O streams for large queries.

The main reason to choose a data space to store your virtual buffer pools is to provide relief for virtual storage constraints in the DBM1 address space and to

provide greater opportunities for caching very large table spaces or indexes. If you are currently using hiperpools for read-intensive workloads and have not reached any DB2 virtual storage limit, there is no immediate benefit to moving to data spaces until processors are available that address more than 2 GB of real memory.

11.2.9.2 EDM pool in data space

You can choose to have the part of your EDM pool that contains cached dynamic statements in a data space. By caching prepared dynamic SQL statements in a data space, you reduce the storage you require in the DBM1 address space.

If you specify YES for CACHE DYNAMIC SQL, DB2 will calculate a default value for the EDMPOOL DATA SPACE SIZE, automatically enabling the usage of a data space for cached dynamic statements.

When using data space, take the following into consideration:

- Moving a data page back and forth between the data space and the look-aside pool may result in extra CPU usage.
- When using data spaces, make sure they are completely backed by processor storage. You do not want to see any paging activity when having to get to the buffers in the data space.

11.2.10 Defer defining data sets

This new support allows DB2 users to use the `DEFINE NO` option in the `CREATE TABLESPACE` and `CREATE INDEX` SQL statements to defer the creation of underlying VSAM datasets for the created DB2 table space or index space. The undefined table spaces or index spaces will still have a DB2 catalog entry, but are considered as empty when accessed by `SELECT` or `FETCH` operation. An existing `SQLCODE +100` (sqlcode100) is returned to any application which attempts to perform a read-only operation.

When the pageset is marked with an undefined state in the DB2 catalog (the `SPACE` column in SYSTABLEPART or SYSINDEXPART is set to -1), it is treated as an empty data set until the very first write operation occurs, either through SQL statements or certain DB2 utilities (such as LOAD). At the first write, DB2 resets the undefined status in the catalog and creates the underlying VSAM data sets to allow the write operation.

The undefined status stored in the DB2 catalog will not be modifiable by any DB2 ALTER command or any other third-party utilities. DBAs and application package providers should consider using the DEFINE NO option if the DDL performance is critical. The DEFINE NO option provides better management relief on DD limits and data usabilities by deferring the VSAM DEFINE/OPEN until the very first write.

Deferring the definition of data sets is an enhancement that can be useful for customers who use only a subset of modules from the full suite of applications provided by PeopleSoft--for example, Benefit and Payroll from HRMS. Currently, customers receive all application tables, regardless of which applications they are actually going to use. This install method allows customers to add PeopleSoft modules easily after the initial install. On the other hand, it is possible for customers to have hundreds of empty tables for applications they will not use. These tables are perfect candidates to be defined using *defer define*.

11.2.11 ODBC performance

Code path for SQLDA processing and for other common ODBC/CLI functions has been optimized for faster performance.

Database Request Module (DBRM) DSNCLIQR is updated by this APAR. Application of this APAR requires a BIND PACKAGE for DSNCLIQR. If the BIND is not performed after applying this APAR, an SQLCODE -805 can result when running CLI applications. To bind the DSNCLIQR package, refer to the sample DB2 CLI bind job DSNTIJCL in the SDSNSAMP dataset for an example on how to perform the BIND command.

11.3 Utility enhancements with DB2 V5

In DB2 V5, the ability to do online reorgs has been beneficial to users of PeopleSoft with DB2 on OS/390.

Online Reorg significantly reduces the impact of this planned outage. The online Reorg utility copies the tablespace to one or more shadow data sets where the actual reorganization is done. The DB2 log is applied in an iterative fashion to the shadow copy of the data to synchronize it with the actual online tablespace.

After the log is applied, the shadow data replaces the unorganized data when the Reorg utility changes data set names. The outage is now limited to read-only during the final application of the DB2 log to the shadow copy of the tablespace, and no readers or writers are allowed during the changing of the shadow data set names to the active tablespace data set names.

11.4 Utility enhancements with DB2 V6

DB2 on OS/390 Version 6 offers significant enhancements to the utilities which benefit the PeopleSoft/DB2 user. Many of these have be retrofitted to DB2 V5 by means of an APAR. The use of these utilities is described in detail in Chapter 9 of *DB2 Server for OS/390 Version 5 Recent Enhancements - Reference Guide*.

11.4.1 Utility parallelism

A key improvement is the ability to run the following utilities in parallel:

- COPY and RECOVER

  To reduce the elapsed time of both COPY and RECOVER, DB2 Version 6 allows these utilities to process in parallel. You may COPY a set of DB2 objects (tablespaces or indexes) in parallel. During RECOVER, the tablespaces and indexes may be restored in parallel. In previous releases of DB2, the creation (during COPY) and the restoration (during RECOVER) of image copies was serialized.

  Full parallelism can be maintained provided that the copies are made to DASD. (Image copies to tape suspend the parallelism until the tape copy is created or restored.)

  To fully exploit this feature, the DB2/PeopleSoft user will want to take image copies to DASD.

- LOAD and REORG

In DB2 Version 6, the LOAD and REORG utilities can build indexes in parallel. The DB2/PeopleSoft user will exploit the parallel index build feature of REORG (PeopleSoft does not explicitly invoke the DB2 LOAD utility).

- REBUILD Index

  With support for the recovery of indexes from image copies in DB2 Version 6, it is not likely that you will use the REBUILD index utility. However, if you find it necessary to use REBUILD to recreate indexes, the recreating or rebuilding of those indexes are done in parallel in DB2 Version 6.

- Fast Log Apply

  Prior to Version 6 of DB2, the log apply phase of RECOVER (for the RECOVER utility or recovery at restart time) was a serial process. The log record was read, the applicable tablespace page was read, and the recovery at the data page level was effected; the process was repeated for subsequent log records.

  With Fast Log Apply in DB2 Version 6, the log records are read and sorted by time of update within page within dataset. Tablespace pages are read only once, using list prefetch. Parallel tasks are dispatched at the tablespace (or dataset) level. This significantly improves performance for the DB2/PeopleSoft user during both recovery and normal DB2 restart.

  Fast Log Apply is also invoked during the Start Database command for Logical Page List (LPL) recovery and Group Buffer Pool Recovery (GRECP). This is particularly beneficial to the DB2 data sharing user.

11.4.2 Other utilities

Other utilities that have been improved are as follows:

- Consistent restart

  Prior to DB2 Version 6, it was possible for a long-running process that took infrequent (or no) commits to severely impact availability. If, during the long-running task, DB2 were to abnormally terminate, the restart process could be lengthy. Furthermore, during the restart process, the DB2 subsystem would be unavailable to users.

  Consistent restart in DB2 Version 6 addresses this issue by introducing a new state that can apply to the long-running unit of recovery. The new state is “postponed-abort”. The updates to the objects effected by the long-running unit of recovery are not backed out at restart time; instead they are put in a postponed-abort state. The DB2 subsystem is made available to the user. A new command (RECOVER POSTPONED) is used to complete the recovery of the postpone-abort objects.

  Since some PeopleSoft processes have the potential to do significant updating with few or no commits, consistent restart can significantly benefit availability for the DB2/PeopleSoft user.

- Inline RUNSTATS

  During the DB2 LOAD, REORG, and REBUILD utilities, inline statistics may be captured. This provides RUNSTATS statistics without a separate invocation of RUNSTATS and without reading the tablespace data a second time. DB2/PeopleSoft users will find this feature particularly beneficial in the REORG utility.

- REORG enhancements (APAR PQ19077)
  - Functional enhancements

    With the REORG utility in DB2 Version 6, you can select rows to be discarded during REORG. The discarded rows may optionaly be written to a discard file.

    REORG UNLOAD EXTERNAL allows records to be converted to an external format for processing by a user-written application program. A LOAD utility statement is generated to allow discarded or unloaded rows to be loaded into another tablespace.
  - Performance and availability enhancements

    REORG has made improvements that can reduce potential deadlocks which benefits both performance and availability. Additionally there have been improvements in the Display Utility command and in auto termination of online REORG (APARs PQ20032 and PQ18941)
- Reuse feature of LOAD, RECOVER, REBUILD

  Without the reuse feature, the RECOVER and REBUILD utilities delete and then redefine the VSAM datasets that contain tablespaces or indexes. With reuse, the high-used VSAM RBA is reset. Given that many tablespaces and indexes in a PeopleSoft environment can be empty, this change can have a significant impact on elapsed recovery times, particularly when it is necessary top do a point-in-time recovery, possibly recovering all DB2 objects in the PeopleSoft system.

11.5 DB2 Connect

PeopleSoft 7.5 offers several ways to establish connectivity from the client to the database. DB2 Connect is available in the Personal Edition or the Enterprise Edition. DB2 Connect can be set up in an SNA environment, but the preferred way is to use TCP/IP as the network protocol.

In the three-tier implementation, we have the presence of an intermediate application server. The thin PeopleSoft client sends PeopleSoft transaction requests (not SQL) to a PeopleSoft application server, which in turn executes a transaction that is comprised of multiple SQL requests.

The current release of DB2 Connect is V6 which requires fixpack 2 that is available on:

`http://www-4.ibm.com/software/data/db2/db2tech/indexsvc.html`

The DB2CLI.INI file requirements are as follows:

DEFERREDPREPARE chains together OPEN and PREPARE statements. This reduces network traffic, which can significantly reduce the response time. DB2 Connect by default activates the defer prepare when creating new entries with the client configuration assistant. For PeopleSoft, use the default setting DEFERPREPARE=1.

CURSORHOLD determines at what point of the transaction to release an SQL cursor. Deactivating CURSORHOLD releases cursors after a transaction has been committed. Programs within PeopleSoft control at what point a cursor needs

to be released. The DB2 Connect default is cursor hold-enabled. For PeopleSoft, use the default setting CURSORHOLD=1.

DISABLEKEYSETCURSOR is new in DB2 Connect V6. The tests done in the PeopleSoft environment demonstrated a high overhead in DB2 Connect V6 when keyset cursors are enabled. To avoid the overhead incurred by keyset cursors, PeopleSoft preferred using forward cursors. For PeopleSoft, override the default setting and set DISABLEKEYSETCURSOR=1.

11.5.1 DB2 Connect configurations for PeopleSoft

DB2 Connect can be used in two-tier or three-tier configurations. Figure 10 gives an overview of two-tier versus three-tier configurations.

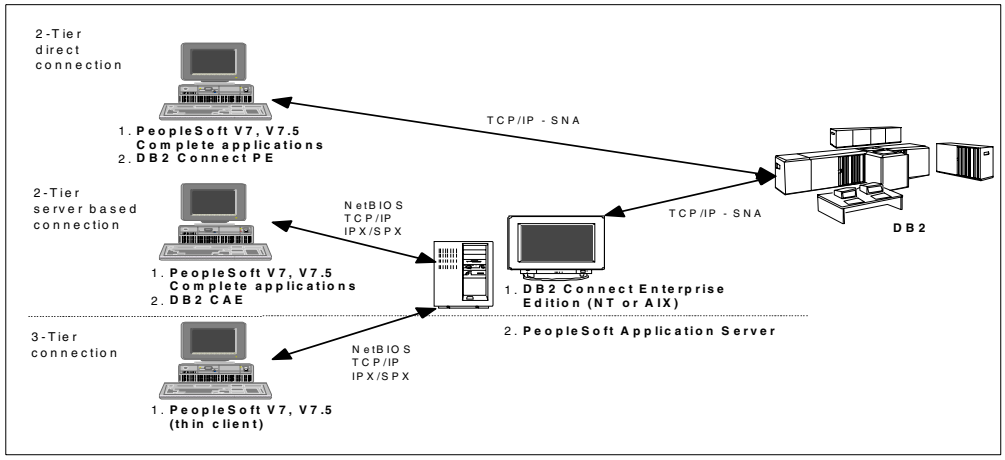


Figure 11. Two-tier and three-tier configurations

1. Two-tier configuration

   The client workstation in the two-tier implementation has the complete PeopleSoft application. There are two connection alternatives:

   - Each client workstation has DB2 Connect Personal Edition installed and uses the client configuration assistant GUI to catalog and establish direct connection between the client workstation and DB2 DDF on OS/390.

     This is how the installation client is set up to run the DataMover scripts.

   - One (or more) workstation has DB2 Connect Enterprise Edition installed and it becomes the connection gateway. Connectivity is established from it to DDF on the DB2 subsystem on OS/390.

     Then DB2 Client Application Enabler (CAE) is installed on the other client workstations. They connect to the gateway workstation and thus to the host. CAE comes with DB2 Connect V6 at no additional cost.

     This method requires less installation and maintenance on the client workstations

2. Three-tier configuration

   In a three-tier configuration the PeopleSoft application resides on an intermediary server. DB2 Enterprise Edition is installed on the middle-tier AIX or NT platform. Connectivity is established from this middle-tier to DDF on OS/390.

The thin client sends a service request via the Tuxedo API to the application server to build a panel group. This is done via a single request, and then DB2 Connect takes over and performs the send/receive ODBC transaction with the host.

In a three-tier environment no CAE is required to connect the client workstation to the middle-tier application server.

11.5.2 DB2 Connect new features

DB2 Connect is being continually enhanced with features that benefit PeopleSoft users. The following has been added recently:

- DB2 client monitoring: The monitoring tool allows for monitoring of dead clients. It can track the Operator ID.
- Password: The ability to change passwords or use encryption of passwords from the client side is now provided.

11.5.2.1 DB2 client monitoring

The PeopleSoft sign-on process performs a double sign-on: first with the Operator ID (or Connect ID), and then with the Access ID. For DBAs that have worked with PeopleSoft in the past, they know that it is a daunting task to associate PeopleSoft operators to DB2 threads. This is because all threads are displayed (via DISPLAY THREAD) with the Access ID as the ID. If your Access ID is PSOFT, then there is the potential to see hundreds or even thousands of DB2 threads all signed on with PSOFT.

One workaround to this is to give each PeopleSoft operator a Unique Access ID. This method works fine for PeopleSoft two-tier users, but it involves extra overhead to create and maintain these new Access IDs, and it will fall short for three-tier connections: all three-tier connections will be displayed with the Access ID used to boot the application server regardless of whether each Operator ID is a unique Access ID.

Adding the ability to uniquely identify a DB2 thread back to a PeopleSoft operator has been requested by customers for a long time. This problem is not specific to DB2 because the double sign-on technique is used on all platforms. The PeopleSoft API for PT7.54.10 has been enhanced to pass extra information on the DB2 thread.

With the latest enhancements made to DB2 Connect's SQLESTETI API function, you can now associate the following items with each database connection or transaction. PeopleSoft 7.54.10 and beyond populate these fields with PeopleSoft-specific information:

- User ID: A 16-character field reserved for the client's ID. The ID entered in this field is not used for connecting in DB2; it is only intended for identification purposes. PeopleSoft set this field to the PeopleSoft operator ID.
- Workstation name: An 18-character field reserved for providing the workstation name for the client. PeopleSoft populates this field with the Computer Name from the client workstation. Computer Name is set in the registry and is not dependent on the COMPUTERNAME environment variable. You can check the computer name on the client via **Control-->Panel--> Network-->Identification**, or from within REGEDIT.

- Application Name: A 32-character field reserved for the application name, such as PSQCKSRV, and so on. It is set to blanks for two-tier connections. It is set to the Domain ID for three-tier connections.Although it may appear to be the database name, it is really the Domain ID. Most customers set the Domain ID equal to the database name.

The best way to describe the new DB2 Client Monitoring feature is with some examples. The following examples all use PT7.56.

Note: You may see slightly different information for the application name in PT7.54.10 versus PT7.56. Redundant information was displayed in the application name (same as ID on the DISPLAY THREAD), and this is no longer the case in PT7.56.

Example #1: This shows a connection to a PeopleSoft/DB2 database via a two-tier connection. DataMover was signed on with Oprid = PTDMO.

```
NAME     ST A   REQ ID           AUTHID   PLAN     ASID TOKEN
SERVER   RA *  3251 PSDMT        PSOFT    DISTSERV 00D0  3009
 V437-WORKSTATION=EPRESZ050499, USERID=PTDMO,
      APPLICATION NAME=*
```

Example #2: This shows the DB2 thread when booting the application server. The workstation name is the computer name for the application server and the user ID is the Oprid used in the [startup] section of psappsrv.cfg.

Note: If using CMTSTAT=INACTIVE in the DB2 dsnzparms, then the thread will only be active for a short time, so you have to be quick to see the thread via DISPLAY THREAD.

```
NAME     ST A   REQ ID           AUTHID   PLAN     ASID TOKEN
SERVER   RA *   531 PSSAMSRV.exe PSOFT    DISTSERV 00D0  4887
 V437-WORKSTATION=EPRESZ042798, USERID=PTDMO,
      APPLICATION NAME=*
```

Example #3: This shows a three-tier connection with Client monitoring disabled for the application server domain. The database is signed with Oprid = VP1 and the application server is booted with Oprid = PTDMO. It is setting the USERID for the thread in DB2 to the Oprid set in Database Signon settings for the application server and it is setting the WORKSTATION to the workstation name of the application server machine. This is because client monitoring is not enabled for the application server in the psappsrv.cfg.

[Database Options]

EnableDBMonitoring=0

```
NAME     ST A   REQ ID           AUTHID   PLAN     ASID TOKEN
SERVER   RA *  3251 PSQCKSRV.exe PSOFT    DISTSERV 00D0  3009
 V437-WORKSTATION=EPRESZ042798, USERID=PTDMO,
      APPLICATION NAME=*
```

Example #4: This shows a three-tier connection with Client monitoring enabled for the application server domain. The database is signed with Oprid = VP1 and the application server is booted with Oprid = PTDMO. DB2 client monitoring is enabled by setting EnableDBMonitoring=1. It now displays the workstation name of the client and the user ID of the client, and the application name is set to the Domain ID from the application server.

[Database Options]

EnableDBMonitoring=1

```
NAME     ST A   REQ ID           AUTHID   PLAN     ASID TOKEN
SERVER   RA * 13237 PSQCKSRV.exe PSOFT    DISTSERV 00D0  3097
 V437-WORKSTATION=EPRESZ050499, USERID=VP1,
      APPLICATION NAME=PT750T7
```

The client monitoring feature is only available for customers using DB2 Connect. It is not supported for customers using Centura connectivity, for example.

If you're using DB2 Connect Version 5, apply fix pack 3 since it contains the new API functions that allow the extra monitoring information. Currently we recommend using PT7.5x with DB2 Connect 5.2 (including fix pack 3) or DB2 Connect 6 (including fix pack 2).

For DB2 V 5.1, you need to apply PTF UQ11841 to the DB2 subsystem so that it can process the new API functions in DB2 Connect. DB2 V6 does not require this PTF.

Since SQR does not use the PeopleSoft API, it cannot be monitored through the DB2 client monitoring feature.

11.5.2.2 Passwords

The ability to change passwords or use encryption of passwords is new to DB2 Connect 5.2.

Password change

Before DB2 Connect 5.2, customers would need to connect to the server to change their password. To avoid connecting to the server, most often customers would specify a non-expiring password, which is not recommended from a security point of view.

DB2 Connect 5.2 (fixpack 3) and DB2 Connect 6 (fixpack 2) now permit users to change their passwords without having to log on to their database server. Users can now change their passwords in any one of the following ways:

- Using the SQL CONNECT statement from the DB2 Command Line Processor
- Requesting a password change from within the ODBC login dialog
- Using the password change option of the Client Configuration Assistant
- Using the ATTACH command

In addition, application programmers can now take advantage of DB2 Connect enhanced password management to deliver more robust security mechanisms for their applications.

Password encryption on DRDA connections

Before this enhancement, DB2 Connect used to flow clear-text passwords to DB2 for OS/390, resulting in a security exposure.

Server support is provided by DB2 for OS/390. DB2 V5, acting as a server only, is adding support for a DRDA password encryption mechanism (APAR PQ21252). DRDA password encryption allows remote workstation users to be authenticated without requiring clear-text passwords to flow in the network. DB2 V5 server

supports the DRDA level 6 password encryption flows from any compliant DRDA level 6 requester.

Client support is provided by DB2 Connect. DB2 Connect V5.2 (FIXPACK 7) and higher supports DRDA level 6 password encryption.To enable DB2 Connect to flow encrypted passwords, DCS authentication must be set to DCS_ENCRYPT in the DCS directory entry (refer to the FIXPACK 7 for documentation).

When the workstation application issues an SQL CONNECT, the workstation negotiates this support with the DB2 server. If supported, a shared private key is generated by the client and server using Diffie-Hellman public key technology and the password is encrypted using 56-bit DES with the shared private key. The encrypted password is non-replayable and the shared private key is generated on every connect. If the server does not support password encryption, the application receives SQLCODE=-30073 (DRDA security manager level 6 not supported).

For more details on how to set up password encryption on DB2 Connect and DB2 for OS/390, refer to the redbook *DB2 Server for OS/390 Version 5 Recent Enhancements - Reference Guide*.

Appendix A. Oracle8 (UNIX) vs DB2 UDB for OS/390 V6.1 terminology

Table 3. Oracle 8 (UNIX) vs DB2 UDB for OS/390 V6.1 terminology

Term	Oracle	DB2 UDB for 0S/390
Database	Each instance has one database and one set of system catalog tables.	A subsystem can have more than one database. Databases are used to logically group application data. All databases share the same system catalogs, system parameters, and processes in the subsystem. DBADM authority is granted on the database level. SYSADM authority is granted at the subsystem level.
Subsystem	Equivalent to an Instance.	An installation of DB2.
Instance	An installation of Oracle.	Equivalent to a subsystem.
Tablespace	A database is logically divided into tablespaces. A tablespace can point to one or more physical database files on disk. One or more tables can reside in a tablespace.	A database is logically divided into tablespaces. There are several tablespace types: simple, segmented, partitioned and large partitioned (for 16 TB tables). A *non-partitioned* tablespace points to one physical VSAM file on DASD. A *partitioned* tablespace points to one VSAM file per partition on DASD. A *segmented* or *simple* tablespace can contain one or more tables.
Segments	A generic name given to any object that occupies storage in the database files.	Some tablespaces can be segmented, but the term “segmented tablespace” does not have the same exact meaning.
Blocks	The smallest unit of database storage. Database files are formatted into blocks, which can be from 2 K to 16 K.	Equivalent to pages; 4 K, 8 K, 16 K, 32 K.
Extents	The unit by which storage is allocated in a database file. The size of the primary and secondary extents are specified in the Storage clause of the CREATE TABLE or CREATE INDEX statements or default to the sizes specified in the CREATE TABLESPACE statement. Extents are allocated until there is no more free space in the files that make up the tablespace, or the maximum number of extents has been reached. The size of the file is specified in the CREATE TABLESPACE statement. Extents are made up of contiguous blocks of storage.	The unit by which storage is allocated for a VSAM file. The size of the primary and secondary extents is specified in the CREATE TABLESPACE statement. A VSAM file can grow up to a maximum of 119 secondary extents. Extents are made up of contiguous pages.

Table	A table can use storage from one or more database files that in are in a tablespace. A partitioned table may have each partition allocated to a different tablespace.	One or more tables can reside in a segmented tablespace. A segmented tablespace is allocated to only one file. Only one table can reside in a partitioned tablespace.
Database files	Database files are named and sized in the CREATE TABLESPACE statement. The Oracle CCF utility is run against files to prepare them for use by Oracle. This is done automatically by Oracle.	When using the STOGROUP option, database files (VSAM files) are sized, allocated, and named dynamically by the CREATE TABLESPACE statement. Database files are referred to as “data sets.”
ROWID	A datatype used for direct row accessing. Values consist of hex physical addresses for data rows.	RID list processing (optimizer controlled) and ROWID (direct row accessing) are both available. ROWID is a varying character datatype and is used for LOB data objects as a locator.
Index	An index may be created in the same tablespace as the table it accesses, or may be created in a separate tablespace. An index may use storage in one or more database files. A partitioned index may be allocated to separate tablespaces.	The CREATE INDEX statement automatically allocates the index to an indexspace created by DB2. An index cannot be stored in a tablespace. Indexes on partitioned tables may use storage from multiple datasets.
Stogroups	No equivalent.	A series of DASD volumes assigned a unique name and used to allocate VSAM datasets for DB2 tablespaces and indexes.
Triggers	Objects that are created on a table using PL/SQL logic blocks and embedded SQL. Triggers are fired when the operation (INSERT, UPDATE, DELETE) is performed on the table it was created for.	Objects that are created on a table using a set of SQL statements that are fired when a specified action (INSERT, UPDATE, DELETE) occurs on the table it was created for.
Stored Procedures	Written in PL*SQL or JAVA. Stored procedures are stored in an Oracle table and executed from within the database.	Stored procedures are written in C, C++, COBOL, Assembler, PL/1 or the new DB2 SQL Stored Procedure language. The compiled host language is stored on the DB2 server and the compiled SQL is stored on the database.
Locking	Row locking is supported. Locks are taken for a statement or transaction. No implicit locks are taken on SELECTs. Rollback segments are used to provide read consistency while data is being changed.	Row, page, table, tablespace locking is supported. Shared locks are taken on SELECTs unless the uncommitted read isolation level is used. When uncommitted read is used, no locks are taken on SELECTS. Data can be read before changes are committed.

Lock Escalation	No equivalent.	DB2 automatically trades in many lower level locks for a higher level lock when the number of lower level locks have exceeded the maximum limit. Lock escalation can be turned off.
Plan	No equivalent.	A *plan* is an executable module of SQL that is composed of one or more packages and was created from a DBRM. A *DBRM* is a module of uncompiled SQL statements that were extracted from the source program by precompilation. A DBRM is bound into a plan or a package.
Prep/Bind	Source programs with embedded SQL are precompiled or prepped. The SQL statements are replaced with calls to the Oracle SQL library. As SQL is parsed at execution time, it is saved in the SQL cache area. If the exact SQL is called again, the SQL is executed without having to be reparsed.	Source programs with embedded static SQL are prepped and the SQL is extracted into a DBRM. The DBRM is bound (BIND) and the resulting executable is called a *plan* or a *package*. An application may also have embedded dynamic SQL, which is prepared at runtime. Prepared dynamic SQL is stored in the EDM pool for future executions if the Dynamic cache zparm parameter is on.
Clusters	*Clusters* are an optional method of storing data. This approach creates an indexed cluster for groups of tables frequently joined. Each value for the cluster index is stored only once. The rows of a table that contain the clustered key value are physically stored together on disk.	No equivalent.
Clustering Index	No equivalent.	An index created on a column of a table where the data values are stored in the same physical sequence as the index. Allows for fast sequential access.
Schema	All the database objects owned by a user ID.	All the database objects owned by a user ID(creator).
Primary Authid	Same as user ID. User IDs are created using a CREATE statement.	Primary authid or individual user ID. Use IDs are not created using a CREATE statement.
Secondary Authid	No direct equivalent in Oracle. Groups of privileges known as *roles* can be granted to a user ID.	Secondary Authid or RACF Group. Privileges can be granted to a secondary authid. Primary authids are assigned to the secondary authid Group. Primary authids inherit all privileges granted to the secondary authid (group) they are in.

Partitioning	A method to break up large tables and indexes so that parts can be stored on different disks to enable parallelism.	A method to break up large tables and indexes so that parts can be stored on different DASD to enable parallelism .
Package	Written in PL*SQL and allows you to group all related programming such as stored procedures, functions, and variables in one database object that can be shared by applications.	No equivalent as known in Oracle. A “package” in DB2 has another meaning. In DB2, a package consists of a single program of executable SQL and the access paths to that SQL. The package is stored on the database and invoked by the host language executable. A package is created by doing a BIND. A package may be part of a PLAN.
Analyze	Updates statistics on tables and indexes and stores information in the system catalog tables. Statistics are used for access path selection by the optimizer.	Similar to Runstats
Runstats	Equivalent to Analyze.	Updates statistics on tables and indexes and stores information in the system catalog tables. Statistics are used by the optimizer for access path selection.
PCTFREE	Free space left in a block to allow for the increased space for a column’s value when UPDATE is performed.	Free space left in a data or index page to allow for future INSERTs. Used for data pages with a clustering index to maintain clustering of key values for sequential access. Used for index pages to reduce page splits for new index entries.
PCTUSED	A percent used level below which the amount of used space in a block must fall before new rows are inserted into the block.	No equivalent.
Hints	A way to influence the cost-based optimizer to use a different access path.	The optimizer can be influenced by identifying the desired access path in the DB2 Plan_table by a query_no. The hint will be used if the query_no is added to the end of the SQL statement and the use of hints are enabled for the subsystem. Access paths can be influenced by changing the SQL statement, sorting, and indexing.
Sequences	An object that is created for a column that automatically generates a unique integer value.	Similar to IDENTITY attribute. Available in DB2 V7.1.

Roles	Groups of privileges that can be assigned or revoked to users through normal DDL commands such as GRANT or REVOKE and used to facilitate administrative tasks.	Similar in functionality to group level security or secondary authids.
IDENTITY	Similar to Sequences.	An attribute of a column that allows for the automatic generation of unique integer values. Available in DB2/390 V7.1.
Reorg	The Export and Import utility can be used to defragment space in a tablespace.	A utility to defragment data/indexes in a tablespace/indexspace and to reestablish clustering and free space of data/indexes on pages.
Rollback Segments	Holds the previous version of the data before it was changed so that it can be used to provide consistent read access to data before changes are committed.	DB2 logs are used to roll back uncommitted changes to data. DB2 uses the Uncommitted Read isolation level to allow read access to data that is being changed but not yet committed.
Savepoints	A savepoint is used to rollback only a portion of a transaction.	Savepoints are planned to be available in DB2 V7.1.
Defragmenting	To compress pockets of free space in a tablespace by exporting and importing the data. The result is contiguous blocks of data.	A Reorg produces the same effects in addition to other benefits.
Dynamic SQL	Embedded SQL that is not known until runtime and is in string format. SQL that is parsed and optimized at runtime, like Oracle Call Interface.	Embedded SQL in string format or CLI. Is parsed and optimized at runtime.
Static SQL	Embedded SQL that is fully known before runtime but is still parsed and optimized at runtime.	Embedded SQL that is parsed and optimized (BIND) and stored before runtime. At runtime it is simply invoked.
SQL Area	Where SQL is parsed and optimized before execution. May be reused.	Similar to EDM Pool for dynamic SQL.
Dynamic Caching	SQL in SQL Area that has been prepared for execution.	Dynamic SQL that has been parsed and optimized is saved in the EDM pool for repetitive executions.

Appendix B. PeopleSoft migration questionnaire

Complete the following questionnaire during the planning phase of the project to assist you in determining the scope of the project.

Please complete this form and email or fax to: nsokolof@us.ibm.com
Fax no.: 914-238-2028

Customer Data

Customer Name:_____ Phone no.:_____

Contact Person: _____ Fax no.: _____

Technical Data

Source System

PeopleSoft Product/Release: _____ ( ) 2-tier ( ) 3-tier

PeopleSoft Modules: _____

Operating System: ( ) AIX ( ) HP-UX ( ) Solaris ( ) Windows NT ( ) Other_____

Database System: ( ) Informix ( ) MS SQL Server ( ) ORACLE ( ) Sybase

Database Version: _____

Size of Production Database _____ No. of Concurrent Users in Production _____

Avg. No. of On-line Transactions in Production per hour _____

No. of Batch Processes _____

Target System

PeopleSoft Product/Release: _____ ( ) 2-tier ( ) 3-tier

Operating System: ( ) AIX ( ) HP-UX ( ) Solaris ( ) Windows NT ( ) OS/390

OS Version _____

Database System: ( ) UDB EE ( ) DB2/OS390 Database Version _____

For s/390 only: Does DB2 Connect exists ( ) Yes ( ) No TCP/IP Version: _____

New Install of DB2? ( ) Yes ( ) No QMF Installed ? ( ) Yes ( ) No

Do Database Management Tools Exist ? ( ) Yes ( ) No Vendor: _____

Do Performance Monitoring Tools Exist ? ( ) Yes ( ) No Vendor: _____

Source PeopleSoft System

How many database instances are involved: _____

Are there more than one PeopleSoft database in one instance? ( ) Yes ( ) No

Have the table spaces of the database been modified from the delivered PeopleSoft definitions?

( ) Yes ( ) No Explain: _____

Are the PeopleSoft naming conventions used?: ( ) Yes ( ) No

Environment(s) to be migrated :

( ) Production	No. of Databases _____	No. of Tablespaces _____	Size _____
( ) Test	No. of Databases _____	No. of Tablespaces _____	Size _____
( ) Development	No. of Databases _____	No. of Tablespaces _____	Size _____
( ) Other _____	No. of Databases _____	No. of Tablespaces _____	Size _____

Have PeopleSoft DDL or applications been customized? ( ) Yes ( ) No
If NO, Please proceed to the end of the customization questions.

For customizations Only:

For each DDL customization indicate the NUMBER of modifications.

( ) New Columns _____ How many PS tables affected _____

( ) New Tables _____ # Columns _____

( ) New Indexes _____ Changes to existing indexes _____

( ) New Views _____ Changes to existing views _____

For each SQL application type below provide:

1) the total NUMBER of programs customized (both new and existing)
2) the number of new programs and the number of changes to existing programs
3) Approximate the TOTAL number of SQL calls that are new or modified

( ) New PeopleCode _____ #new _____ #existing _____
Are SQLEXEC calls used? How many? _____

( ) SQR _____ # new ____ # existing _____ tot. # SQL calls _____

( ) COBOL _____ # new ____ # existing _____ tot. # SQL calls _____

( ) PSQUERY _____ Need to be moved? ( ) Yes ( ) No

Can you provide a list of all applications that are new and modified? ( ) Yes ( ) No
Do new programs use embedded static SQL? CLI? Dynamic SQL? _____
Has the SQL in PS_SQLSTMT_TBL been modified? ( ) Yes ( ) No

Other Customer Written Programs/Reports using SQL and No. and # SQL statements:

On-line	No.	#SQL	Batch	No.	#SQL
_____	_____	_____	_____	_____	_____
_____	_____	_____	_____	_____	_____
_____	_____	_____	_____	_____	_____

End of Customized Environment Section

List Names and Number of rows for the 10 largest PeopleSoft tables in Production:

_____	_____
_____	_____
_____	_____
_____	_____
_____	_____
_____	_____
_____	_____
_____	_____
_____	_____
_____	_____

Have any stored procedures been written to access the database? ( ) Yes ( ) No how many _____
What is the average number of SQL calls per stored procedure? _____
How long are the stored procedures (total number of statements) _____
Do the stored procedures call other stored procedures? ( ) Yes ( ) No

Have any triggers been written to access the database? ( ) Yes ( ) No how many _____
What is the average number of SQL statements per trigger _____
How long are the triggers (total number of statements)? _____
Do Triggers call stored procedures? ( ) Yes ( ) No

Do applications outside of the PeopleSoft system feed or access the PeopleSoft database? Does the PeopleSoft system feed other systems? If yes, please describe:

What is the retention period for data on the PeopleSoft Database _____
Is an archival process in place ( ) Yes ( ) No

Last Run Date for DDDAUDIT and SYSAUDIT for Each Environment:

( ) Development _____ No. of Exceptions _____

( ) Test _____ No. of Exceptions _____

( ) Production _____ No. of Exceptions _____

( ) Other _____ No. of Exceptions _____

Have any database changes been made outside of Application Designer ( ) Yes ( ) No
If Yes, which environments and what type:

( ) Development _____

( ) Test _____

( ) Production _____

( ) Other _____

Appendix C. Setting up DB2 Connect

The architecture for using DB2 in a distributed environment is the Distributed Relational Database Architecture (DRDA), which has a connectivity component, Distributed Data Facility (DDF). DDF uses the TCP/IP connectivity defined previously.

After establishing connectivity using TCP/IP, the next steps are as follows:

1. Install and customize DDF on OS/390.
2. Install and customize DB2 Connect on a Windows workstation.
3. Test that you can log into the DB2 database.

C.1 Installing and customizing DRDA

The following four steps customize DDF:

1. BSDS changes.

   Most DDF parameters are stored in the DSNZPARM module. We used the DSNTIJUZ member in DSN510.NEW.SDSNSAMP to update the DSNZPARM module. One of the steps in the DSNTIJUZ JCL is to update the BSDS information, using the change log inventory utility. To execute the change log inventory, you must stop the DB2 subsystem. Following is the DSNTIJUZ job step to update the BSDS:

```
//DSNTLOG EXEC PGM=DSNJU003,COND=(4,LT)
//STEPLIB  DD  DISP=SHR,DSN=DSN510.SDSNLOAD
//SYSUT1   DD  DISP=OLD,DSN=DSN510.BSDS01
//SYSUT2   DD  DISP=OLD,DSN=DSN510.BSDS02
//SYSPRINT DD  SYSOUT=*
//SYSUDUMP DD  SYSOUT=*
//SYSIN    DD  *
 DDF    LOCATION=DB2K,LUNAME=SCPDBA1,
        NOPASSWD,RESPORT=33325,PORT=33324
//*
```

2. Define DB2 DDF to VTAM - SYS1.LOCAL.VTAMLST.

   To enable TCP/IP support, you must also define an APPL in VTAM. Do this is in SYS1.LOCAL.VTAMLST. The LUNAME keyword must be defined in BSDS and the LU must be active, prior to starting DDF communications. The following shows the APPL definitions in SYS1.LOCAL.VTAMLST:

```
**                                                              **
**      DB2 V6 LU DEFINITION FOR DRDA                           **
**                                                              **
                VBUILD TYPE=APPL
*
SCPDBA1         APPL  ACBNAME=SCPDBA1,                            X
                APPC=YES,                                         X
                ATNLOSS=ALL,                                      X
                AUTH=(ACQ),                                       X
                AUTOSES=10,                                       X
                DMINWNL=25,                                       X
                DMINWNR=25,                                       X
                DSESLIM=50,                                       X
```

```
EAS=509,                                                    X
ENCR=NONE,                                                  X
MODETAB=ISTINCLM,                                           X
PARSESS=YES,                                                X
SECACPT=ALREADYV,                                           X
SONSCIP=NO,                                                 X
SYNCLVL=SYNCPT,                                             X
VERIFY=NONE,                                                X
VPACING=2,                                                  X
VTAMFRR=NO
```

3. Define DB2 DDF to RACF.

   DDF uses OpenEdition to perform TCP/IP services. Some of the OpenEdition functions that DDF executes require an authorized user with certain privileges. To execute the authorized functions, the user ID associated with the DDF started task must be defined for OpenEdition as a superuser. To define a user ID as a superuser, you must set the User Identifier (UID) parameter of the RACF user profile to zero.

   To set the UID parameter for your DDF user, you can issue one of the following RACF commands:

   ```
   ADDUSER userid OMVS(UID(0))
   ALTUSER userid OMVS(UID(0))
   ```

   The ADDUSER RACF command adds a new user profile and should be used when creating a new user for DDF. The ALTUSER RACF command changes the RACF profile for the existing DDF user. To check whether your DDF user ID is already correctly defined to RACF, issue the following RACF command:

   ```
   LISTUSER userid OMVS
   ```

   If you specify both a user ID and a group in the RACF Started Procedure Table ICHRIN03 for the DDF address space, the group must also have a valid OpenEdition group ID (GID) setting. To define RACF groups to be OpenEdition groups, use the RACF panels or one of the following commands:

   ```
   ADDGROUP groupid OMVS(GID(n))
   ```

   where *groupid* is the name of the RACF group associated with the DDF address space, and can be any valid unique identifier.

4. Define DB2 DDF to TCP/IP.

   Part of the DDF customization process is to select port numbers when updating the BSDS. The DDF statement of the change log inventory has been enhanced with PORT and RESPORT values. If PORT and RESPORT are defined, DDF accepts TCP/IP connections from any client that provides valid security information. DB2 also allows outbound connections to other DRDA servers using TCP/IP.

   To define the port numbers in TCP/IP you must update the TCP/IP PROFILE data set. In our case, we used SYS1.TCPPARMS member PROFILE. You must register the TCP/IP port numbers you have specified during DB2 installation or when using the change log inventory utility. We defined two port numbers required by our DB2 subsystem, DB2K.

   In the PORT statement you must use TCP as the protocol, and the name of the OpenEdition started procedure (in our case, OMVS). Because DB2 uses OpenEdition services to connect to TCP/IP, the DB2 ports are reserved for the

OpenEdition address space, and not for the DDF address space, xxxxDIST. The PORT definitions are shown here:

```
SYS1.TCPPARMS(PROFILE)
    PORT
      23 TCP INTCLIEN              ;
      33324 TCP OMVS                 ; DRDA SQL PORT for DB2K
      33325 TCP OMVS               ; DRDA SQL resync port for DB2K
```

For more detailed information on customizing DDF, refer to *WOW! DRDA Supports TCP/IP: DB2 Server for OS/390 and DB2*.

C.2 Installing and customizing DB2 Connect on the client

We installed DB2 Personal Connect using the CD-ROM for Personal Edition DB2 Connect on Windows 95, and the Enterprise Edition of DB2 Connect on AIX.

For DB2 Connect V6, fixpak 2 is required. It is available from:

```
http://www-4.ibm.com/software/data/db2/db2tech/indexsvc.html
```

When we installed the product, we selected **DB2 Connect Personal Edition**, then **Typical** on the installation menu.

Note: If you select **Compact** instead of **Typical**, you will not have all the functions.

Install DB2 Connect on *each* PeopleSoft workstation if you use it to communicate to the OS/390 DB2 database.

In our environment, we needed the following:

- A Windows NT workstation for the installation client
- The DB2 Connect Personal Edition software

Note: This configuration depends on whether you to have a two-tier or a three-tier setup. In a typical situation in three-tier mode, you have DB2 Connect Enterprise Edition on the application servers and CAE installed for those client workstations that need two-tier and three-tier connections.

C.2.1 Client Configuration Assistant

After you have installed DB2 Connect on the workstation, start the customization wizard as shown in Figure 12 on page 94.

C.2.2 Determining parameters

To avoid confusion, you should determine all the information you will need before you start this process. Using inconsistent information will prevent you from establishing connectivity, and it will be difficult to determine what is incorrect. Table 4 on page 94 lists the parameters we used for our installation.

Table 4. DB2 Connect parameters for Windows

Parameter	Source	Value used
Protocol	Database Services	TCP/IP
Target operating system		MVS/ESA to OS/390
Hostname	SYS1.TCPPARMS (PROFILE)	WTSC04 or 9.12.14.207
Port number	Port number for DRDA in SYS1.TCPPARMS (PROFILE)	33324
Target database	Location name	DB2W
Alias database name	For DB2 Connect and PeopleTools	DB2DMO
TSO user ID	Valid user ID with RACF and DB2 dbadm (use uppercase)	PSOFT1
TSO password	RACF password (use uppercase)	abcdef

C.2.3 Add a database

After starting the client configuration assistant, choose the option to add a database by clicking **Add**.

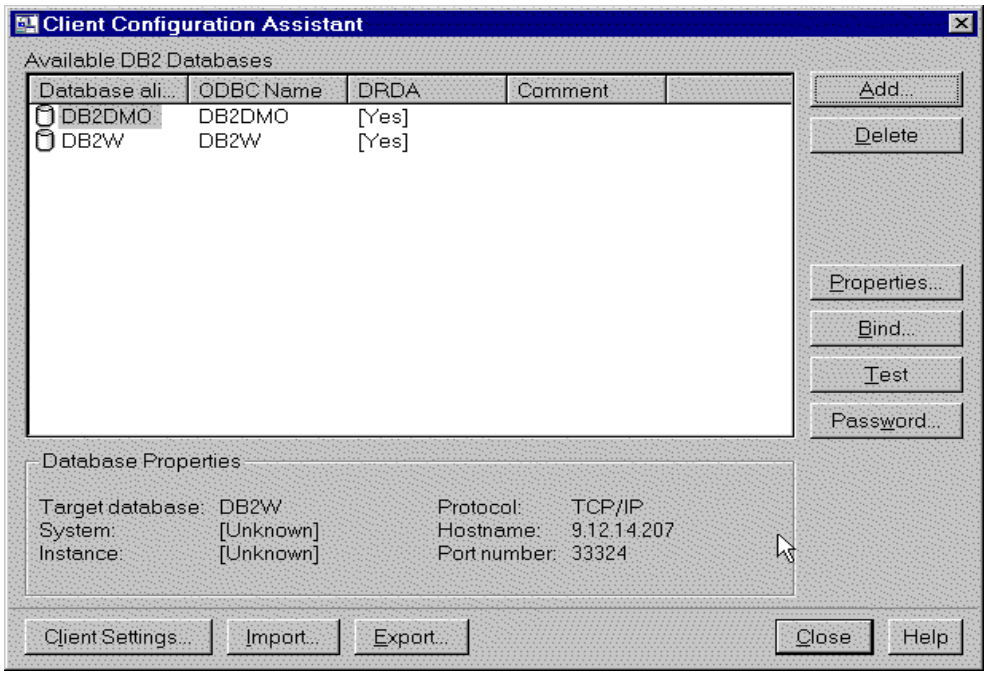


Figure 12. Client Configuration Assistant screen after the add function was executed

C.2.4 Choosing the source

Choose the option to manually configure the connection as shown in, and then click **Next**.

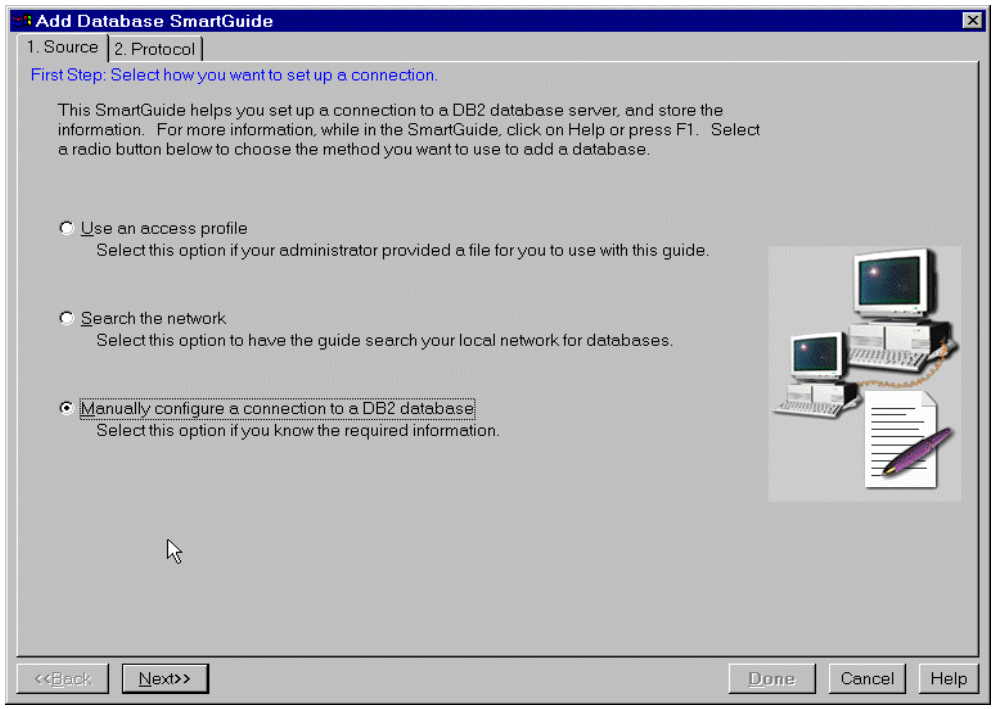


Figure 13. Choosing the manual connection

C.2.5 Choosing the protocol

Choose **TCP/IP**. This adds the option to choose the target operating system. Also choose **MVS/ESA** or **OS/390**, as shown in Figure 14.

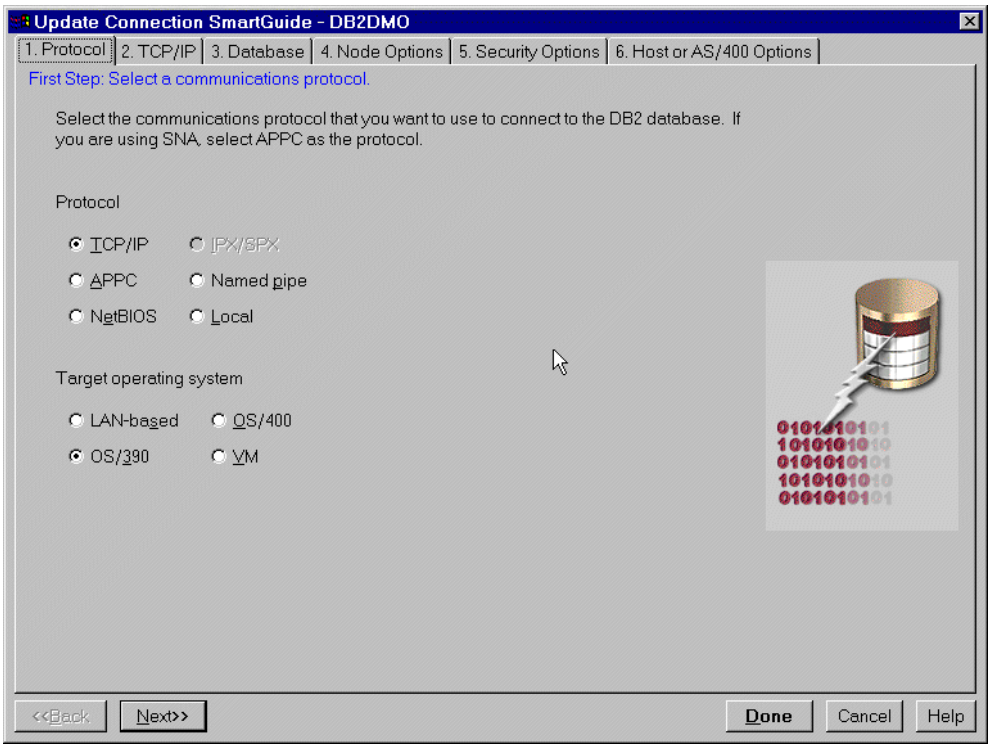


Figure 14. Protocol selection

C.2.6 Specify TCP/IP communication parameters

TCP/IP must already be configured before you do this step. We used the TCP/IP address at this point rather than the hostname, as we had not set up a hostname file. We left the Service name blank. The Port number must be the same Port number that DDF is listening to on OS/390. We used the Port number 33324, as shown in Figure 15.

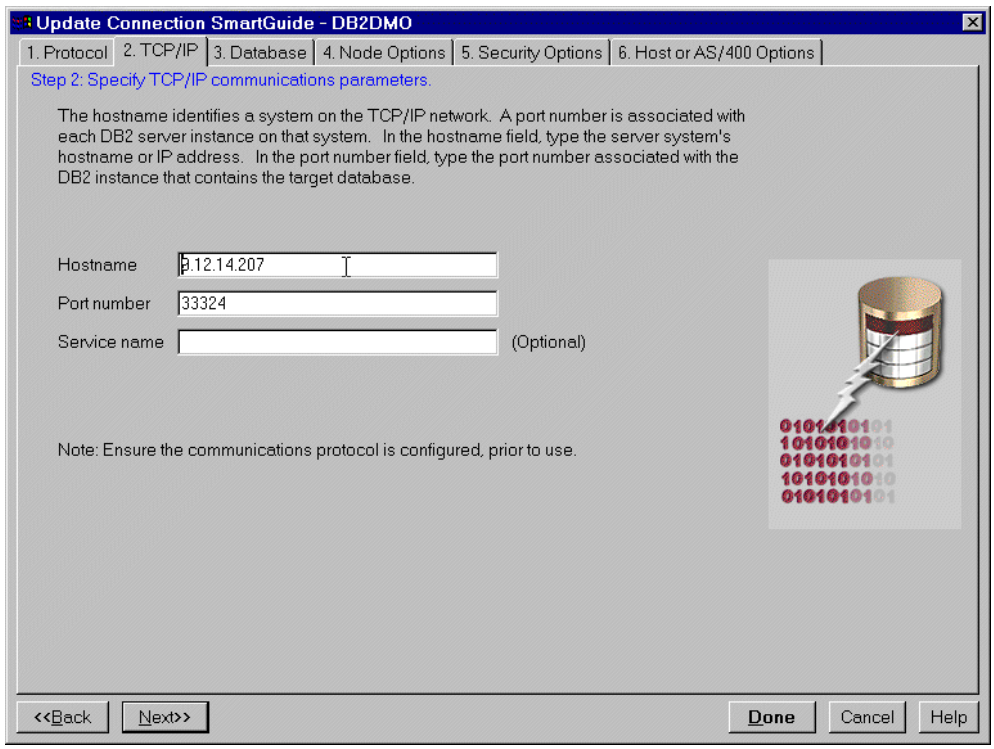


Figure 15. TCP/IP address and DDF port number

C.2.7 Choosing the target database

Enter the name of the target database. In our case this was DB2W as shown in Figure 16. DB2W is the location name defined in the DDF parameters on OS/390, as shown in C.1, “Installing and customizing DRDA” on page 91.

C.2.8 Alias name

Enter the alias name as shown in Figure 16. In our case, this was DB2DMO.

Each Peoplesoft application instance using DB2 for OS/390 needs its own alias name (such as DB2DMO, PSHRDMO or PSFSDMO). After we finished adding DB2DMO as an alias, we repeated this procedure, starting with the Client Configuration Assistant, to add the alias DB2W.

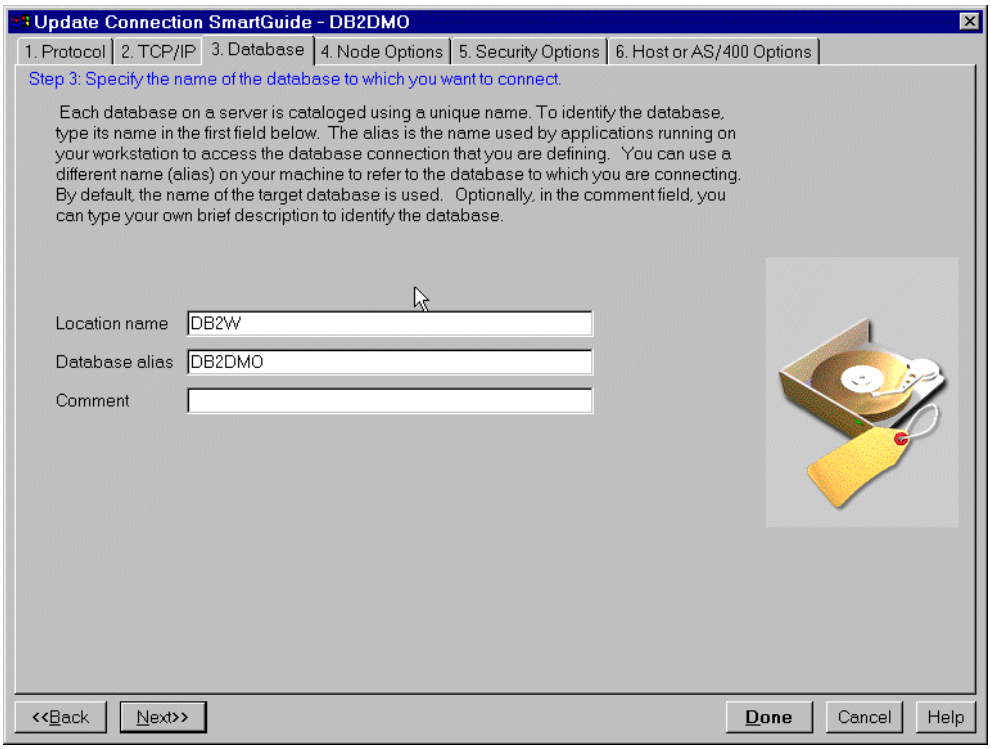


Figure 16. Database name and alias

C.2.9 OBDC Name

We used the default to register with ODBC. Click **Done** to complete the customization. See Figure 17.

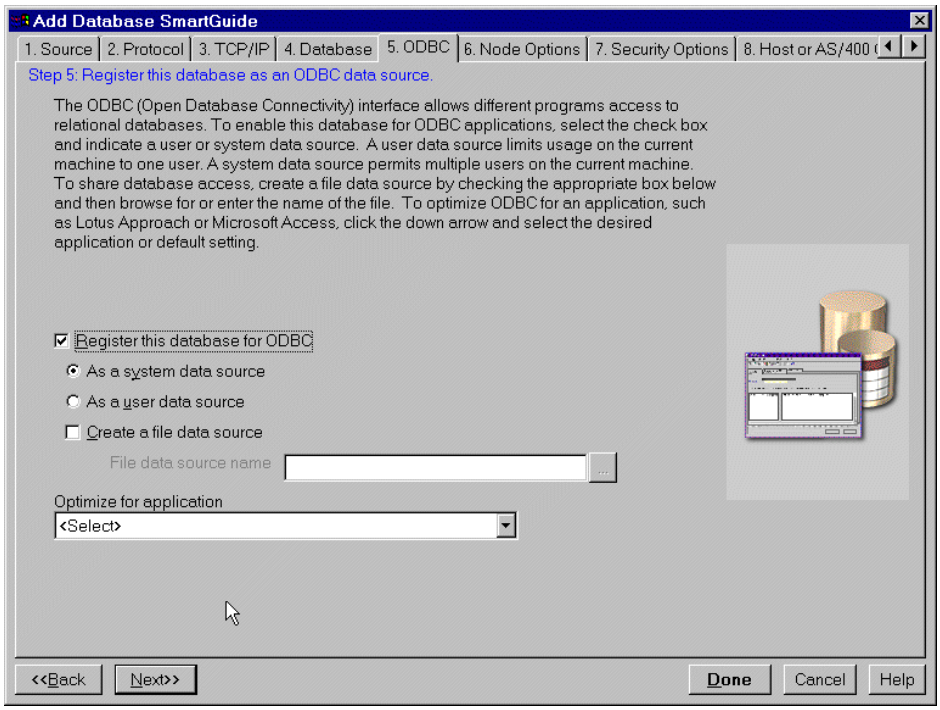


Figure 17. ODBC tab

C.2.10 Security options

This is a new screen in DB2 Connect V6. We had to choose host to allow DB2 to connect correctly as shown in Figure 18.

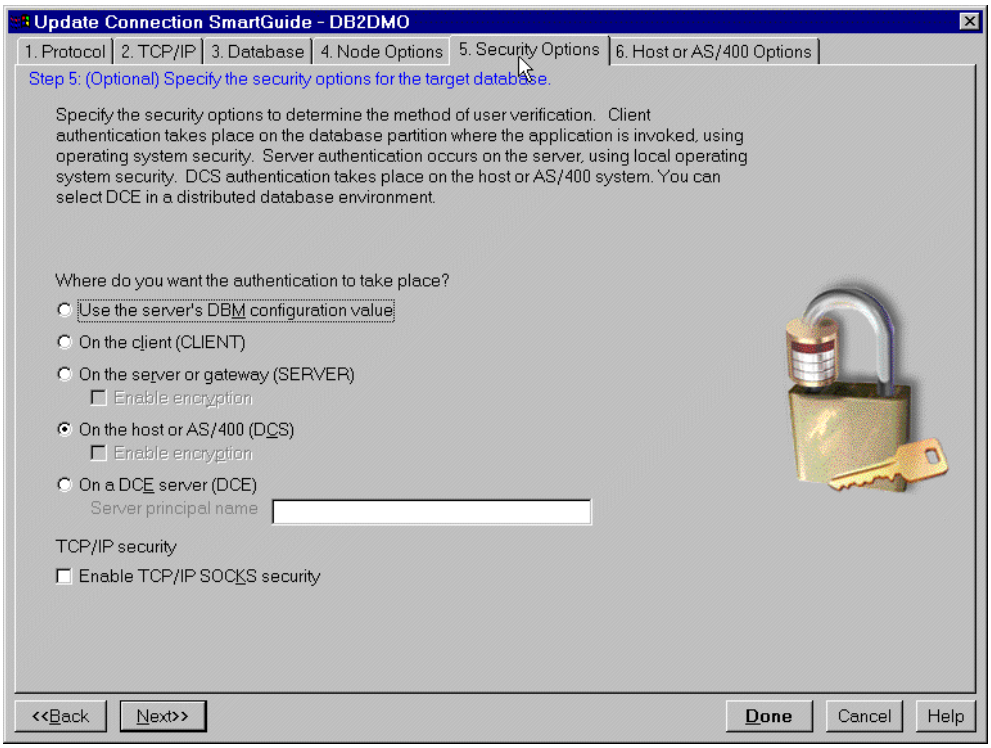


Figure 18. Security options

C.2.11 Customization complete

At this point you should receive confirmation that the connection configuration was added successfully.

You can add additional aliases for the database now, or you can do this at the main screen (Client Configuration Assistant).

C.2.12 Testing the connection

There are three ways to test the connection:

- Use the Test Connection option at the end of the customization.
- Use the test button on the primary screen.
- Connect to DB2 using the command line processor.

C.2.12.1 The Test Connection option

After clicking **Test Connection**, enter a valid TSO user ID and password in uppercase.

A successful connection will result in the message shown in Figure 19 on page 101.

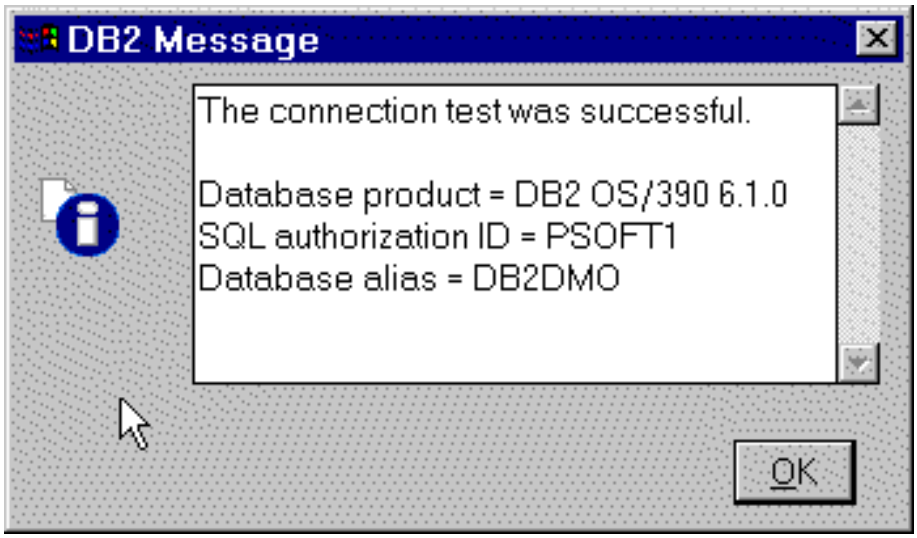


Figure 19. Connection test successful

C.2.12.2 Test button on the database screen

At any time, you can click **Test** on the primary screen. If you receive an error message, you should check that:

- DB2 and DDF are started on OS/390.
- Your parameters are consistent with source data.
- You can ping the connection.
- You used uppercase for the TSO user ID and password.

C.2.12.3 Connection to DB2

Use the command option of DB2 Connect or issue the following command at prompt of C:

```
C:./>db2cmd
```

This will set up an environment and bring up another DOS session. In this session, enter the following command:

```
C:connect to DB2W user tsouserid using tsopassword
```

The following lines appear:

```
DB2CLP C:\>db2 connect to DB2W user PSOWNER using PSOWNER
Database Connection Information
Database product          = DB2 OS/390 5.2
 SQL authorization ID      = PSOWNER
 Local database alias      = DB2W
```

At the prompt DB2CLP C:, enter:

```
\db2 select * from ps.psdbowner
```

```
DBNAME    OWNERID            DEDICATE_THREADS
------    -------            ----------------
DB2W PSOWNER            Y
DB2W PSUSER             Y
DB2W PS                 Y
DB2W OPRID              Y
```

4 record(s) selected

DSB2CLP C:\>

At this point we have established connectivity to DB2 from the client workstation that will be used to install PeopleSoft applications.

C.2.13 Setting up a second database alias

We set up the alias name DB2DMO that we would use later in the installation process, as shown in Figure 20.

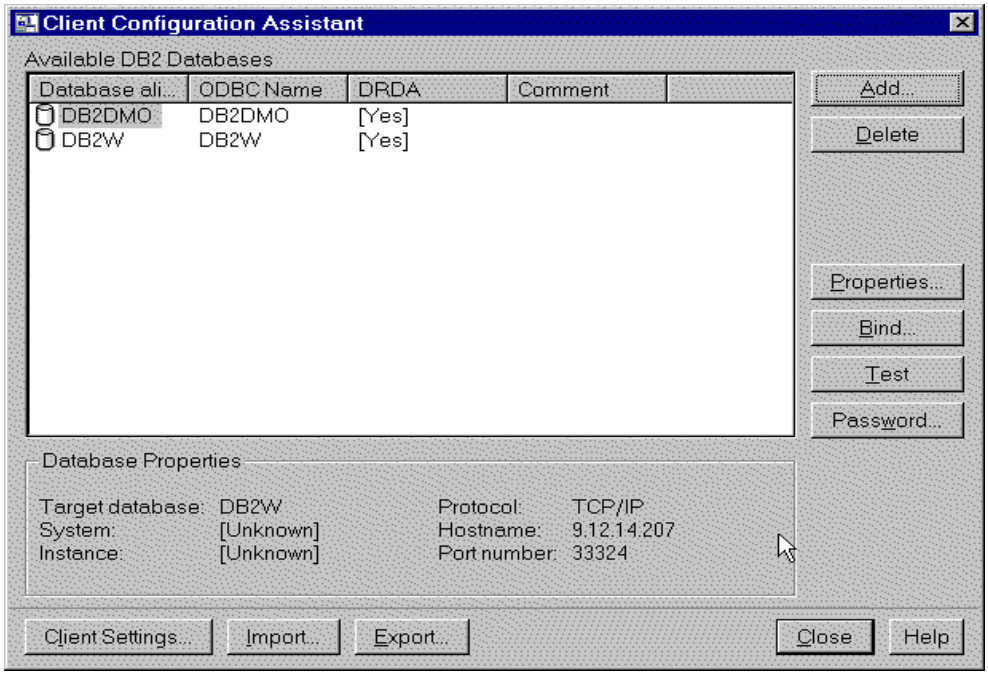


Figure 20. Client Configuration Assistant after entering another alias

C.2.13.1 Parameter change for performance

It is recommended that you deactivate the CURSORHOLD option to prevent lock holding. You can do this by clicking the sequence:

Properties-->Settings-->Advanced-->Cursor hold

CURSORHOLD-->OK

You can add these two lines in the db2cli.ini file under the [common] section:

- Cursorhold = default
- Disablekeysetcursor=1

Appendix D. Connectivity options

This appendix discusses considerations for connectivity to the DB2 Database Server. You should appreciate that communications is an area of continuous development; additional options may have become available after this writing, or additional developments in the existing options may change these considerations.

In this section we concentrate on the connectivity from application servers to a database server. While other communications is at least as important for a large PeopleSoft installation, it was felt that this particular connectivity is most likely to be new for companies performing a migration; we thought that more discussion is necessary than is found in other documents.

Whatever connectivity option is chosen, it must be expected that the connection between the application server and the database server could become a bottleneck as the applications usage grows or when more PeopleSoft applications are used.

In this discussion we mainly concentrate on hardware alternatives. Since PeopleSoft uses DB2 Connect and DB2 Connect is based on TCP/IP, software and protocol alternatives are eliminated. Also note that the discussion is theoretical; for information on how to actually implement the connectivity, consult your communications staff, TCP/IP documentation, and the hardware manufacturers' guides. For an example of how DB2 Connect was implemented in a PeopleSoft installation, see *Planning to Install PeopleSoft with DB2 for OS/390.*

The hardware selections we discuss are:

- ESCON channel adapter
- Gigabit Ethernet
- FDDI
- Fast Ethernet

You should note the possibilities *not* included in the list. Token-ring, normal EtherNet, and telephone line connections were excluded because the speed of these connections is usually inadequate for the amount of data and response requirements of PeopleSoft applications. ATM was eliminated because it is mostly being used as a means of combining small packets in higher-speed backbones of extended networks.

Token ring is not a viable option for the connection from the application server to the database server. However, it *is* a viable option for the connection of the clients to the application server.

D.0.1 ESCON channel adapter

Channel connection hardware offers very fast transmission. From the view of S/390 systems programmers, it is the easiest to implement. However, it does require a support infrastructure for fiber technology, so you should anticipate building this infrastructure if it is not already in place.

Because channel connections are implemented with a point-to-point approach (application servers only communicate to the database server, not to one another), the highest *aggregate* data rates of all the options are possible. Also

note that total network congestion plays a small role in response performance; each connection (application server to database server) may be thought of as an independent LAN.

Channel connections may have the highest cost, since the hardware in the application servers is expensive. Because of this, companies that use channel adapters sometimes use a *gateway* approach: some application servers with channel adapters serve as gateways to the database server, and other application servers connect to the gateways using one of the other LAN technologies. This adds complexity to the design, especially in the areas of availability, recovery and performance.

D.0.2 Gigabit Ethernet

A Gigabit Ethernet LAN with nodes for the application servers and the database server provides high-speed connectivity with adequate bandwidth. This technology does require an OSA-2 adapter on the database server, so systems programmer effort for defining and customizing that hardware is necessary. Gigabit Ethernet networks are implemented with fiber technology, so the fiber support infrastructure noted in D.0.1, “ESCON channel adapter” on page 103 is also necessary for a Gigabit Ethernet.

Communications specialists should also be asked to examine the traffic rates to application servers. Since Gigabit Ethernet is a LAN, it is possible that traffic rates could surpass the LAN capacity; so it might be necessary to use more than one Gigabit Ethernet LAN to support your requirements.

D.0.3 FDDI

Most of the discussion in D.0.2, “Gigabit Ethernet” on page 104 applies to FDDI, but note some differences:

- FDDI is not as fast as Gigabit Ethernet; this has two consequences:
  1. The network capacity required by an *individual* application server may not be met, thus a different option must be chosen.
  2. The network capacity required by *all* the application servers may not be met; multiple FDDI LANs may be necessary.
- FDDI may be implemented with either fiber or copper technology. Building the fiber support infrastructure might not be necessary, or you might plan a later implementation of fiber connectivity.
- FDDI LANs are usually less expensive and less complex than Gigabit Ethernet.

D.0.4 Fast Ethernet

These LANs are marginally adequate for the bandwidth requirements of PeopleSoft applications. You may want to evaluate this alternative as part of a migration strategy if applications use small bandwidths (for example, if applications are to be in "test mode" for an extended period). You may also want to consider this option as a backup for the primary LAN in availability or disaster scenarios.

Fast Ethernet does not require fiber technologies. It is the least expensive of the options to implement, but involves a high risk of inadequate bandwidth.

D.0.5 Connectivity performance

Whatever connectivity option is chosen, it must be expected that the connection could become a bottleneck as the applications grow or when more PeopleSoft applications are used. Therefore, measuring the performance of the connection and comparing its usage with capacity should be an ongoing activity. General performance and measurements are subjects outside the scope of this book. However, you should plan to address those subjects. There are many tools and products to assist you; one that you should investigate is Network Traffic Analysis (NTA). For information about NTA, refer to the following Web address:

```
http://www.ibm.com/services/tsm/nta
```

Appendix E. Testing the migration process with two tables

This appendix documents the DataMover scenarios we ran at the ITSO in Poughkeepsie to move two tables from the HR demo database. We did this to test out the processes we wanted to use. It was a faster way to check out the process than using the entire database.

The method included the following steps:

- Run a DataMover script to export the data.
- Run a special DataMover script to separate the DDL from the data in the export file.
- Allocate the VSAM file to create the database and tablespaces.
- Execute the extracted DDL to create the tables.
- Run the DataMover script to import the data to DB2.
- Verify the migration.

E.1 Preparing the systems

To be able to run the DataMover utility, you must have connectivity to the database. To run the export you must have established connectivity to the Oracle database on AIX. To extract the DDL and run the import, you must have connectivity to the DB2 database on OS/390.

For these to run correctly you must also correctly configure the PeopleTools configuration manager.

E.2 Running the export script

This is the script used to export only two tables:

```
REM Export Oracle HR75DMO tablespace
set output d:\redbook\data\fsapp.dat;
set log d:\redbook\logs\hroraexp.log;
export ps_run_fs_rt_gen;
export ps_run_fs_rt_gen_d ;
```

Figure 21 on page 108 shows the DataMover window after we completed the export.

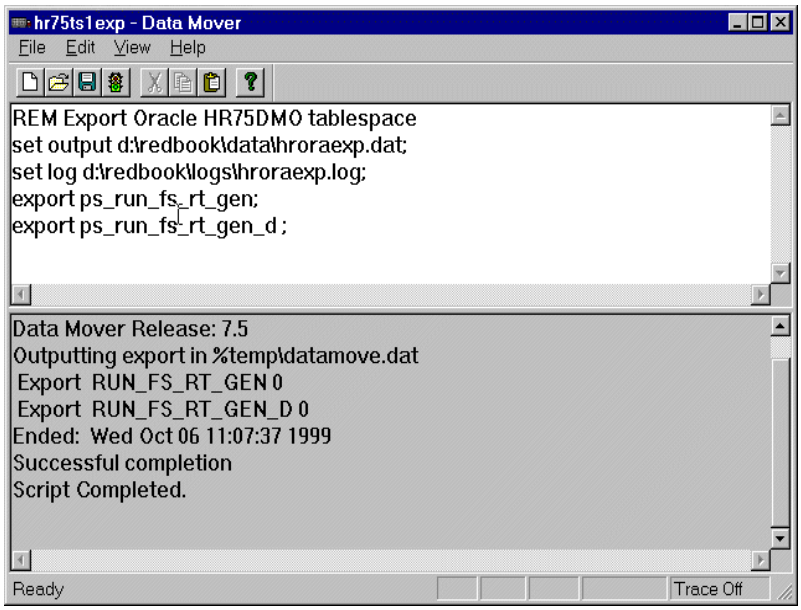


Figure 21. DataMover window after the export

Log for exporting two tables

This is the output file from the export:

```
d:\redbook\data\hroraexp.dat;

Started: Wed Oct 06 11:07:36 1999
Data Mover Release: 7.5
Outputting export in d:\redbook\data\fsapp.dat;
 Export  RUN_FS_RT_GEN 0
 Export  RUN_FS_RT_GEN_D 0
Ended:  Wed Oct 06 11:07:37 1999
Successful completion
```

Sample output in the export file

The following are the first few lines of the exported data file:

```
SET VERSION_DAM 7.5:2:0

REM Database: HR75DMO
REM Started: Wed Oct 06 11:07:36 1999
 Export  RECORD/SPACE.x
BNAPP
BNLARGE
FSAPP
GIAPP
GPAPP
HRAPP
HRLARGE
HTAPP
PAAPP
PALARGE
PIAPP
PILARGE
PSIMAGE
PTAPP
PTPRC
PTTBL
PYAPP
PYLARGE
STAPP
STLARGE
```

```
TLALL
TLAPP
TLLARGE
/
A($)B(AAAAAA)A(P)B(AA)A(BNDx)B(BLAAABAAAAAAAAAAAAAAAAAAAAAA)A(P)B(AA)A(G)
A(P)B(AAAAAAAA)A(HRAPP)B(AA)A(HRLARGE)B(AA)A(HTAPP)B(AA)A(PAAPP)B(AA)A(P)
A(ALARGE)B(AA)A(PIAPP)B(AA)A(PILARGE)B(AA)A(PSIMAGE)B(AA)A(PTAPP)B(AA)
B()A(PTPRC)B(AA)A(PTTBL)B(AA)A(PYAPP)B(AA)A(PYLARGE)B(AA)A(STAPP)B(AA)
B()A(STLARGE)B(AA)A(TLALL)B(AA)A(TLAPP)B(AA)A(TLLARGE)B(AAAAAAAAAAAAAA)
B(AAAAAAAAAAAAAAA

Many lines not shown - they are basically unreadable lines
```

E.3 Extracting the DDL from the export file

The IBM PeopleSoft Competency Center provided us with a script that could be used to extract the DDL from the export file. The DDL was then used as input for SPUFI on DB2.

To run this script, select

START-->Programs-->PeopleSoft7.5-->Data Mover

In this case we were using the database name DB2W.

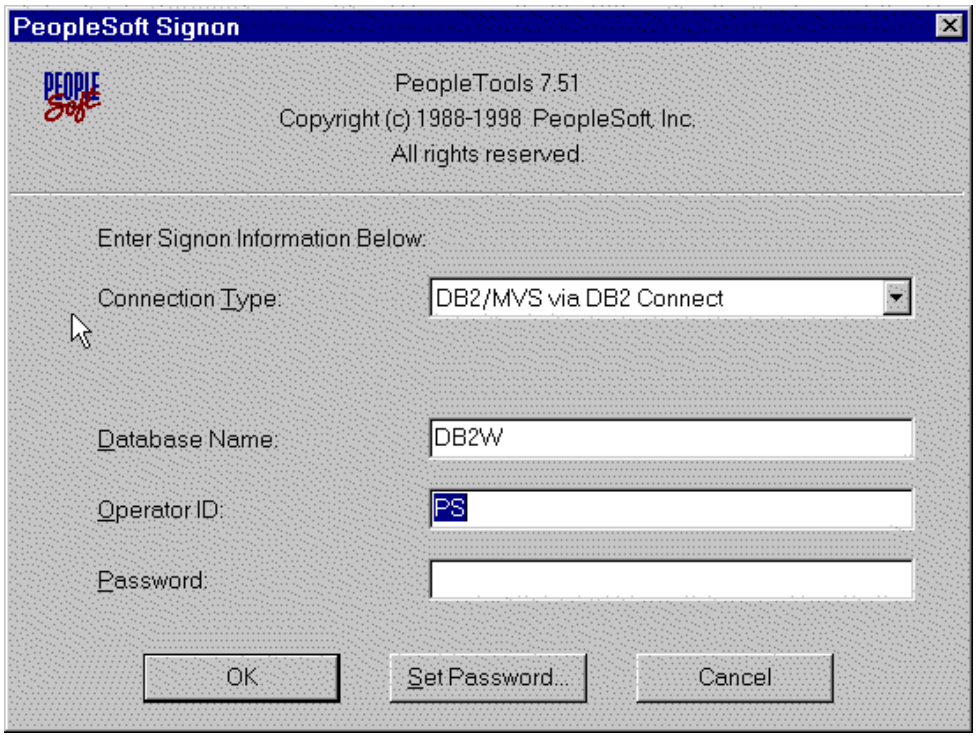


Figure 22. PeopleSoft Signon to DB2 using DB2W

If you had set up the alias name in DB2 Connect, you could set up DataMover to access DB2DMO as shown in Figure 23 on page 110.

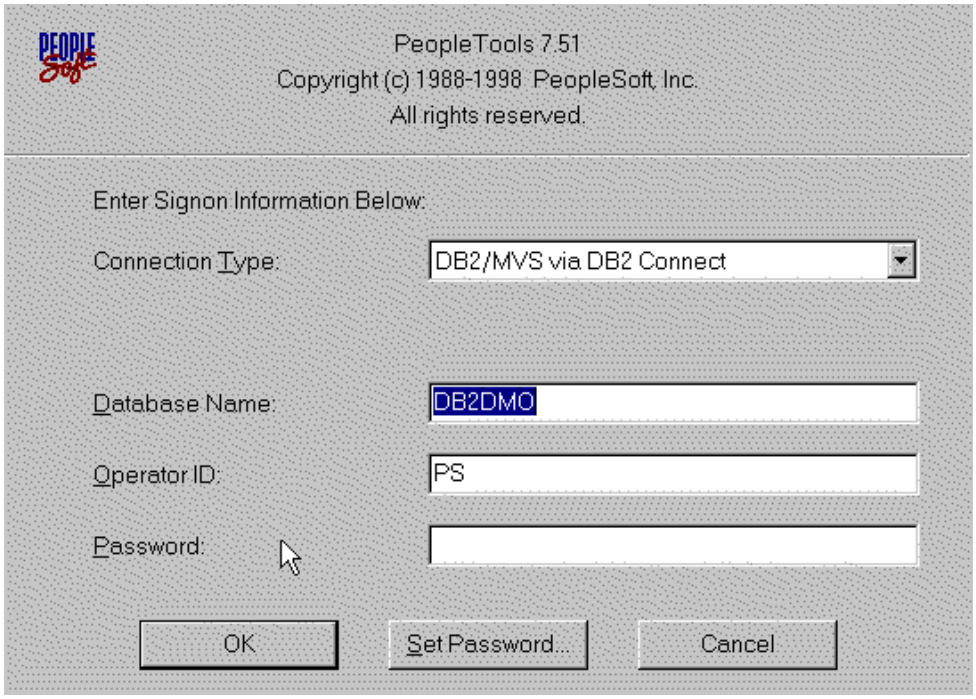


Figure 23. PeopleSoft signon using DB2DMO as the database alias name

This is the file we used to extract the DDL:

```
REM
REM Extract DDL from Oracle DataMover Export into DB2 OS/390 Format
REM
set log d:\redbook\logs\fsapp.log;
set input d:\redbook\data\fsapp.dat;
set execute_sql set current sqlid = 'DB2DMO';
set ddl table space  * input stogroup  as PSSTABLE;
set ddl index        * input stogroup  as PSSINDEX;
set ddl unique index * input stogroup  as PSSINDEX;
set ddl record       * input dbname    as HRDMO;
set ddl record       * input owner     as DB2DMO.;
set ddl index        * input owner     as DB2DMO.;
set ddl index        * input owner2    as DB2DMO.;
set ddl unique index * input owner2    as DB2DMO.;
set ddl unique index * input owner     as DB2DMO.;
set no view;
set no space;
set no trace;
set extract output d:\redbook\ddl\fsapp.sql;
import *;
```

In this example, we extracted the DDL to fsapp,txt. After extracting the DDL via DataMover, the next step was to move the DDL up to the mainframe and execute it via SPUFI or DSNTEP2. Once the DDL is created, the next step is to replace the following line:

```
Remove the line: SET EXTRACT OUTPUT d:\redbook\ddl\fsapp.sql;
Add the line SET NO RECORD;
```

Then the same DataMover script can be used to load the tables. In this way there is no problem in generating the DB2-specific DDL from the export file.

E.4 DDL used to create the database and tablespaces on S/390

The extracted DDL does not handle the database and tablespace create commands. We created these manually and then used the extracted DDL to create the tables. This is the DDL that we used with SPUFI to prepare the DB2 database for the import:

Create the stogroups

```
 CREATE STOGROUP PSSTABLE
        VOLUMES (PSOFT3, PSOFT4, PSOFT5, PSOFT6)
        VCAT DB2V610W;

 COMMIT;
 CREATE STOGROUP PSSINDEX
        VOLUMES (PSOFT3, PSOFT4, PSOFT5, PSOFT6)
        VCAT DB2V610W;

 COMMIT;
```

Create the database and tablespace

```
 SET CURRENT SQLID = 'PS';

  CREATE DATABASE PSHRDMOP  STOGROUP PSSTABLE BUFFERPOOL BP1;

  COMMIT;

  CREATE TABLESPACE  FSAPP      IN  PSHRDMOP
         USING STOGROUP PSSTABLE PRIQTY 48    SECQTY  48
         FREEPAGE  0 PCTFREE 20
         SEGSIZE    4 BUFFERPOOL BP1   LOCKSIZE ANY  CLOSE NO  ;
  COMMIT;
```

E.5 Creating the tables

Then we used the extracted DDL to create the tables:

```
CREATE TABLE PS_RUN_FS_RT_GEN (OPRID CHAR(8) NOT NULL,
    RUN_CNTL_ID CHAR(30) NOT NULL,
    RT_RATE_INDEX CHAR(10) NOT NULL,
    TERM INTEGER NOT NULL,
    FROM_CURRENCY CHAR(3) NOT NULL,
    CUR_RT_TYPE CHAR(5) NOT NULL,
    AS_OF_DATE DATE NOT NULL,
    OVERRIDE_CUR_RT CHAR(1) NOT NULL,
    GENERATE_REPORT CHAR(1) NOT NULL,
    GENERATE_RECIP CHAR(1) NOT NULL,
    GENERATE_CROSS_RT CHAR(1) NOT NULL,
    RATE_TRIANGULATE CHAR(1) NOT NULL) IN PSHRDMOP.FSAPP
;
 COMMIT
 ;
 CREATE UNIQUE INDEX PS_RUN_FS_RT_GEN ON PS_RUN_FS_RT_GEN (OPRID,
    RUN_CNTL_ID) USING STOGROUP PSSINDEX PRIQTY 40 SECQTY 40 CLUSTER
  CLOSE NO
 ;
 COMMIT
 ;
 CREATE TABLE PS_RUN_FS_RT_GEN_D (OPRID CHAR(8) NOT NULL,
    RUN_CNTL_ID CHAR(30) NOT NULL,
    RATE_TRIANGULATE CHAR(1) NOT NULL,
    RT_RATE_INDEX CHAR(10) NOT NULL,
    TERM INTEGER NOT NULL,
    FROM_CURRENCY CHAR(3) NOT NULL,
CUR_RT_TYPE CHAR(5) NOT NULL,
   AS_OF_DATE DATE,
   FROM_CUR CHAR(3) NOT NULL,
   TO_CUR CHAR(3) NOT NULL) IN PSHRDMOP.FSAPP
;
COMMIT
;
CREATE UNIQUE INDEX PS_RUN_FS_RT_GEN_D ON PS_RUN_FS_RT_GEN_D (OPRID,
```

```
   RUN_CNTL_ID,
   FROM_CUR,
   TO_CUR) USING STOGROUP PSSINDEX PRIQTY 40 SECQTY 40 CLUSTER CLOSE NO
;
CREATE INDEX PS#RUN_FS_RT_GEN_D ON PS_RUN_FS_RT_GEN_D (RUN_CNTL_ID,
   RATE_TRIANGULATE,
   RT_RATE_INDEX,
TERM,
    FROM_CURRENCY) USING STOGROUP PSSINDEX PRIQTY 40 SECQTY 40  CLOSE NO
 ;
 COMMIT ;
```

E.6 Importing the data

At this point we ran the import Data Mover script:

Script to import two tables:

This is the file that was used to import the data once we had used the DDL:

```
REM
REM Import data from Oracle DataMover Export into DB2 OS/390
REM
set log d:\redbook\logs\fsapptbl.log;
set input d:\redbook\data\fsapp.dat;
set execute_sql set current sqlid = 'PS';
set ddl table space  * input stogroup  as PSSTABLE;
set ddl index        * input stogroup  as PSSINDEX;
set ddl unique index * input stogroup  as PSSINDEX;
set ddl record       * input dbname    as PSHRDMOP;
REM set ddl record       * input owner     as DB2DMO.;
REM set ddl index        * input owner     as DB2DMO.;
REM set ddl index        * input owner2    as DB2DMO.;
REM  set ddl unique index * input owner2    as DB2DMO.;
REM  set ddl unique index * input owner     as DB2DMO.;
REM  set ddl unique index * input owner as DB2DMO.;
set no view;
set no space;
set no trace;
REM set extract output d:\redbook\ddl\fsapp.sql;
set no record;
import *;
```

Log to import two tables:

This was the log produced from the import.

```
Started:  Wed Oct 06 16:24:50 1999
Data Mover Release: 7.5
Commit done at end of record
Importing  RUN_FS_RT_GEN
 Creating Table  RUN_FS_RT_GEN
 Building required indexes for RUN_FS_RT_GEN
Records remaining: 1
Importing  RUN_FS_RT_GEN_D
 Creating Table  RUN_FS_RT_GEN_D
 Building required indexes for RUN_FS_RT_GEN_D
SQL Spaces: 0  Tables: 2  Indexes: 3  Views: 0
Ended:  Wed Oct 06 16:24:51 1999
Successful completion
```

E.7 Validate the import

Then we verified that the tables were accessible in DB2.

Appendix F. Handling large tables

In some cases we have found that DataMover is not efficient for large tables or if the platform migration also includes changes to a table defintion. An alternative to using DataMover is to develop shell scripts and awk programs to be run on the Oracle platform to extract large tables and create the DB2 LOAD utility control cards. The data and control cards can then be sent to the DB2 platform using the ftp command and LOADed into the new database. Elapsed time is typically much lower than with DataMover.

A version of these scripts was originally published in the redbook *Converting from Oracle AIX to DB2 for OS/390*. The scripts listed in this appendix have been modified by adding some processes specific to the PeopleSoft application.

There are two versions of the shell scripts:

1. A version for tables that will be run through a conversion process (*unldconv.sh*) on the DB2 platform
2. A version for tables that will be loaded directly (*unldstrt.sh*) into DB2

These invoke different awk programs and produce output in two different formats.

- For tables that will be loaded directly to DB2, *unldstrt.sh* produces a fixed-length file and will produce LOAD control cards for that file format.
- For tables that will be processed by a conversion routine, *unldconv.sh* produces a file with the individual fields delimited by a “pipe” character (|); this format was chosen to allow simple file processing with the widest range of programming tools (C, C++, awk, Perl, SQR, etc.).

The process within each of the scripts is similar:

1. Read a record from the controlling input file.

   Each record contains two fields: the name of the table to be extracted, and a unique identifier for that table. A four-digit number is recommended as the identifier. This is necessary because MVS has an 8-character limit for filenames and for PDS member names.

   A simple way to assign the number is to extract all of the tables names for a PeopleSoft installation, load the list of names to a spreadsheet, and use the spreadsheet's Fill function to assign consecutive numbers. Records in the input file will look like the following:

   ```
   PS_JOB  0248
   PS_ACCOMPLISHMENTS 0023
   PS_EARNINGS_BAL  0147
   PS_PAY_TAX  1234
   ```

2. Extract the table definition from the Oracle Dictionary and the PeopleSoft record definition tables.
3. Invoke *count.awk* or *countcnv.awk* to compute the record length of the file that you are going to create.
4. Create the SQL*Plus script that will SELECT all of the rows of the table. This script will invoke either *colfmt.awk* or *numfmt.awk* to create SQL*Plus Column directives, depending on whether this is a fixed format or delimited file extract, and will invoke either *collist.awk* or *desc.awk*, again depending on whether this

is a fixed format or delimited file extract, to create the column list for the SELECT statement.

5. Run the SQL*Plus script to create the extract file.
6. Run *loadctl.awk* to create DB2 LOAD utility control cards if this is a fixed format extract.

This migration process assumes that the target DB2 database has been defined already and that JCL for the DB2 LOAD utility is available on the mainframe. The LOAD control cards should be sent using the *ftp* command to a PDS and the extracted data files should be sent using the *ftp* command to sequential datasets. The existing JCL should only need to be altered to point to the appropriate PDS member and the appropriate sequential dataset.

The following are sample scripts that have been used in the migration of large tables:

unldstrt.sh

```
#!/bin/ksh
echo 'This is unldstrt.sh'
echo "running against $ORACLE_SID"
# # # # # # # # # # # # # # # # # # # # # # # # # # # # # # # # # # # # # # # #
# Shell script unldstrt.sh
#
#
# Starting from an flat file containing a list of all the tables and
#  a unique identifier number (4 digits) for each table
#
# extracts data from Oracle for each table and writes data into
# a file named identifier.DAT, formatted in columns
#
# Syntax: unldstrt.sh table_list_file
#
# This script uses the awk command with the following awk command files:
#
# desc.awk formats the query command files using RPAD and DECODE
#      to obtain a column-formatted output
#
# count.awk computes the total length of a record
#
# numfmt.awk creates a fixed length column definition for numbers
#
# loadctl.awk creates DB2 LOAD utility control statements
#
# # # # # # # # # # # # # # # # # # # # # # # # # # # # # # # # # # # # # # # #
#
# Define the environment variables for the oracle user and password
CPRS_USR=sysadm
CPRS_PWD=sysadm
echo "signed on as $CPRS_USR"
#
# Start of main program
#
# Loop on all the tables listed in the input file
{
while read i j; do
#for i in `cat $1`
```

```
#do
# Define some environment variables for temporary files
print " table $i identifier $j"
date
export OUTFILE=$j.DAT
DSCFILE=$j.dsc
SQLFILE=$j.sql
VARFILE=$j.var
ALLFILE=$j.all
DB2FILE=$j.CTL
POSFILE=$j.pos
rm -f $OUTFILE
rm -f $DSCFILE
rm -f $SQLFILE
rm -f $DB2FILE
# Extract the table description from Oracle catalog
#sqlplus  $CPRS_USR/$CPRS_PWD <<EOF >/dev/null 2>&1
sqlplus -s $CPRS_USR/$CPRS_PWD <<EOF >$j.out 2>$j.err
clear columns
clear breaks
set pagesize 50000
set linesize 250
set newpage 1
set feedback off
spool $DSCFILE
select a.column_name -
 , a.data_type -
 , b.length -
 , b.length-b.decimalpos -
 , b.decimalpos -
 ,decode(b.fieldtype,0,'Character' -
 ,1,'Long Character' -
 ,2,'Number' -
 ,3,'Signed Number' -
 ,4,'Date' -
 ,5,'Time' -
 ,6,'DateTime' -
 ,7,'SubRecord' -
 ,8,'Image' -
 ,9,'Ver' -
   ,'Unknown') -
from all_tab_columns a, -
   psdbfield b -
where b.fieldname = a.column_name -
 and a.table_name = '$i'  -
order by a.column_id;
EOF
# Cut head and tail from the file containing the descriptions of the tables
# Change also the NOT NULL clause in a blank string
# and cut the blanks in the first column
tail +4 $DSCFILE | sed 's/NOT NULL/ /; s/^ //' > $DSCFILE.tmp1
NL=`wc -l < $DSCFILE.tmp1`
NLM1=`expr $NL - 1`
head -$NLM1 $DSCFILE.tmp1 > $DSCFILE.tmp2
cp $DSCFILE.tmp2 $VARFILE
mv $DSCFILE.tmp2 $DSCFILE
rm -f $DSCFILE.tmp*
# Compute the record length of the table
```

```
# using the count.awk awk script
LS=`awk -f count.awk $DSCFILE`
# Prepare the heading of the query statement on the table
# by echoing the statements into the sql file
echo "clear columns" > $SQLFILE
echo "clear breaks" >> $SQLFILE
echo "set pagesize 50000" >> $SQLFILE
echo "set linesize $LS" >> $SQLFILE
echo "set feedback off" >> $SQLFILE
echo "set heading off" >> $SQLFILE
echo "set space 0" >> $SQLFILE
echo "set newpage 1" >> $SQLFILE
# Append to the query statement the numeric column formats using
# the numfmt.awk awk script
awk -f numfmt.awk $DSCFILE >> $SQLFILE
echo "spool $OUTFILE" >> $SQLFILE
echo "select '' " >> $SQLFILE
# Append to the query statement file the list of the table fields
# to obtain the column layout, using the desc.awk awk script
awk -f desc.awk $DSCFILE >> $SQLFILE
# Append to the query statement file the "from" clause
# and the closing instructions
echo "from  $i;" >> $SQLFILE
echo "spool off" >> $SQLFILE
echo "quit" >> $SQLFILE
# Execute the query statement; unless the table definition
# changes you can rerun from here
print "starting the extract"
date
#sqlplus -s $CPRS_USR/$CPRS_PWD @$SQLFILE >/dev/null 2>&1
sqlplus $CPRS_USR/$CPRS_PWD @$SQLFILE >/dev/null 2>&1
date
# Cut the first line from the output file
tail +2 $OUTFILE > $OUTFILE.tmp
print "extract finished"
date
mv $OUTFILE.tmp $OUTFILE
# Change the DATE data type into its DB2 external length, 26 bytes
sed 's/ DATE/ 26/' $DSCFILE > $DSCFILE.tmp4
mv $DSCFILE.tmp4 $DSCFILE
# Generate the DB2 LOAD control cards
awk -v tblname=$i -f loadctl.awk $DSCFILE > $DB2FILE
# End of the loop
print "Done with $i"
done
} < $1
```

unldconv.sh

```
#!/bin/ksh
print "This is unldconv.sh"
print "running against $ORACLE_SID"
# # # # # # # # # # # # # # # # # # # # # # # # # # # # # # # # # # # # # # # #
# Shell script unldconv.sh
#
# This script will unload data from Oracle into a delimited file.  The
```

```
# delimited file is intended to be used as the the load source for an SQR
program
# that will modify the data prior to loading it to the DB2 database.
#
# Starting from an flat file containing a list of all the tables
# and a unique number for each table, extracts data from Oracle for
# each table and writes data into a file named identifier.DAT.
#
# Syntax: unldconv.sh table_list_file
#
# This script uses the awk command with the following awk command files:
#
# colfmt.awk creates a SQL*Plus column format for DATE fields
#
# collist.awk creates the column list for the extract SELECT statement
#
# count.awk computes the total length of a record
#
# # # # # # # # # # # # # # # # # # # # # # # # # # # # # # # # # # # # # #
#
# Define the environment variables for the oracle user and password
CPRS_USR=sysadm
CPRS_PWD=sysadm
#SYSADM=sysadm1
print "signed on as $CPRS_USR"
#
# Start of main program
#
# Loop on all the tables listed in the input file
{
while read i j; do
#for i in `cat $1`
# Define some environment variables for temporary files
print "table $i identifier $j"
date
export OUTFILE=$j.DAT
DSCFILE=$j.dsc
SQLFILE=$j.sql
rm -f $OUTFILE
rm -f $DSCFILE
rm -f $SQLFILE
# Extract the table description from Oracle catalog
#sqlplus  $CPRS_USR/$CPRS_PWD <<EOF >/dev/null 2>&1
sqlplus -s $CPRS_USR/$CPRS_PWD <<EOF >$j.out 2>$j.err
clear columns
clear breaks
set pagesize 50000
set linesize 250
set newpage 1
set feedback off
spool $DSCFILE
select a.column_name -
 , a.data_type -
 , b.length -
 , b.length-b.decimalpos -
 , b.decimalpos -
 ,decode(b.fieldtype,0,'Character' -
 ,1,'Long Character' -
```

```
 ,2,'Number' -
 ,3,'Signed Number' -
 ,4,'Date' -
 ,5,'Time' -
 ,6,'DateTime' -
 ,7,'SubRecord' -
 ,8,'Image' -
 ,9,'Ver' -
   ,'Unknown') -
from all_tab_columns a, -
   psdbfield b -
where b.fieldname = a.column_name -
 and a.table_name = '$i'  -
order by a.column_id;
EOF
# Cut head and tail from the file containing the descriptions of the tables
# Change also the NOT NULL clause in a blank string
# and cut the blanks in the first column
tail +4 $DSCFILE | sed 's/NOT NULL/ /; s/^ //' > $DSCFILE.tmp1
NL=`wc -l < $DSCFILE.tmp1`
NLM1=`expr $NL - 1`
head -$NLM1 $DSCFILE.tmp1 > $DSCFILE.tmp2
mv $DSCFILE.tmp2 $DSCFILE
rm -f $DSCFILE.tmp*
# Compute the record length of the table
# using the countcnv.awk awk script
LS=`awk -f countcnv.awk $DSCFILE`
# Prepare the heading of the query statement on the table
# by echoing the statements into the sql file
echo "clear columns" > $SQLFILE
echo "clear breaks" >> $SQLFILE
echo "set pagesize 50000" >> $SQLFILE
echo "set linesize $LS" >> $SQLFILE
echo "set feedback off" >> $SQLFILE
echo "set heading off" >> $SQLFILE
echo "set space 0" >> $SQLFILE
echo "set newpage 1" >> $SQLFILE
echo "set colsep '|' " >> $SQLFILE
# set the column format and length for Date columns
awk -f colfmt.awk $DSCFILE >> $SQLFILE
echo "spool $OUTFILE" >> $SQLFILE
echo "select  " >> $SQLFILE
# append to the query the list of columns to extract
awk -f collist.awk $DSCFILE >> $SQLFILE
# Append to the query statement file the "from" clause
# and the closing instructions
echo "from  $i;" >> $SQLFILE
echo "spool off" >> $SQLFILE
echo "quit" >> $SQLFILE
# Execute the query statement;  unless the table definition
# changes you can rerun from here
print "starting the extract"
date
#sqlplus -s $CPRS_USR/$CPRS_PWD @$SQLFILE >/dev/null 2>&1
sqlplus $CPRS_USR/$CPRS_PWD @$SQLFILE >/dev/null 2>&1
# Cut the first line from the output file
tail +2 $OUTFILE > $OUTFILE.tmp
print "extract finished"
```

```
date
mv $OUTFILE.tmp $OUTFILE
# End of the loop
print "Done with $j table $i "
done
} < $1
```

count.awk

```
# Generates a byte count of the output of SELECT statement
BEGIN { total=0 }
{
if ($2 == "DATE")
total +=26
else
total += $3
}
END { print total }
```

countcnv.awk

```
#generates a byte count for a delimeted output from a SELECT statement
BEGIN { total=0 }
{
if ($2 == "DATE")
total +=26 +1
else
total += length($1) + 1
}
END { print total }
```

numfmt.awk

```
# Generates an Oracle column format for numeric data
# the format will provide fixed length output
BEGIN {}
function getnines(len) {
 if (len <= 0)
  return ""
 else
  return "9" getnines(len - 1)}
$6 == "Number" && $5 != "0" { print " COLUMN " $1 " FORMAT " getnines($4+0)
"V" getnines($5+0)}
$6 == "Signed" && $5 != "0" { print " COLUMN " $1 " FORMAT S" getnines($4+0)
"V" getnines($5+0)}
$6 == "Signed" && $5 =="0" { print " COLUMN " $1 " FORMAT S" getnines($4+0) }
$6 == "Number" && $5 == "0" { print " COLUMN " $1 " FORMAT " getnines($4+0) }
```

collist.awk

```
# generates the column list for an Oracle SELECT statement
# Dates are transformed into something DB2 likes
BEGIN {}
{
if ($2 == "DATE")
 print " , DECODE("$1",NULL,' ',TO_CHAR("$1",'YYYY-MM-DD-HH24.MI.SS'))"
else
 if ( NR > 1 )
  print ","$1
 else
```

```
  print $1
}
```

colfmt.awk

```
# Generates an Oracle column format for Dates
# the format will provide fixed length output
BEGIN {}
$6 == "Date" { print " COLUMN
DECODE("$1",NULL,'',TO_CHAR("$1",'YYYY-MM-DD-HH24.MI.SS')) FORMAT A21"}
```

desc.awk

```
BEGIN {}
{
if ($2 == "DATE")
print " || rpad(DECODE("$1",NULL,' ',TO_CHAR("$1",'YYYY-MM-DD-HH24.MI.SS')
|| '.000000'),26)"
else
 if (substr($2,1,3) == "NUM")
  print " || lpad(DECODE("$1",NULL,'0',"$1"),"$3",'0') "
 else
  print " || rpad(DECODE("$1",NULL,' ',"$1"),"$3",' ') "
}
```

loadctl.awk

```
# loadctl.awk
# Generates DB2 LOAD utility control cards
#    Input file format:
#      Field 1        Column Name
#      Field 2        Oracle Data type
#      Field 3        Field Length
#      Field 4        Length of the integer portion of the field
#      Field 5        Length of the decimal portion of the field
#      Field 6        PeopleSoft field type
# ######################################################################
BEGIN { print "LOAD DATA RESUME YES INDDN SYSRC00"
        print " INTO TABLE " toupper(tblname) " ("
        sep = " " }
if (NR > 1) { sep = ',' }
$6 == "Character" { print sep $1 " CHAR(" $4 ")" }
$6 == "Date" { print sep $1 " TIMESTAMP EXTERNAL(26)"}
$6 == "Number" && $5 == 0 { print sep $1 " DECIMAL EXTERNAL("$3")"}
$6 == "Number" && $5 != 0 { print sep $1 " DECIMAL EXTERNAL("$3","$5")"}
$6 == "Signed" && $5 == 0 { print sep $1 " DECIMAL EXTERNAL("$3")"}
$6 == "Signed" && $5 != 0 { print sep $1 " DECIMAL EXTERNAL("$3","$5")"}
END {print ")" }
```

Appendix G. Special Notices

This publication is intended to help migrate the ERP solution of Peoplesoft 7.5 using DB2 on OS/390. The information in this publication is not intended as the specification of any programming interfaces that are provided by OS/390, DB2, PeopleSoft 7.5 and the associated products.

See the PUBLICATIONS section of the IBM Programming Announcement for OS/390, DB2 V6 and DB2 Connect for more information about what publications are considered to be product documentation.

References in this publication to IBM products, programs or services do not imply that IBM intends to make these available in all countries in which IBM operates. Any reference to an IBM product, program, or service is not intended to state or imply that only IBM's product, program, or service may be used. Any functionally equivalent program that does not infringe any of IBM's intellectual property rights may be used instead of the IBM product, program or service.

Information in this book was developed in conjunction with use of the equipment specified, and is limited in application to those specific hardware and software products and levels.

IBM may have patents or pending patent applications covering subject matter in this document. The furnishing of this document does not give you any license to these patents. You can send license inquiries, in writing, to the IBM Director of Licensing, IBM Corporation, North Castle Drive, Armonk, NY 10504-1785.

Licensees of this program who wish to have information about it for the purpose of enabling: (i) the exchange of information between independently created programs and other programs (including this one) and (ii) the mutual use of the information which has been exchanged, should contact IBM Corporation, Dept. 600A, Mail Drop 1329, Somers, NY 10589 USA.

Such information may be available, subject to appropriate terms and conditions, including in some cases, payment of a fee.

The information contained in this document has not been submitted to any formal IBM test and is distributed AS IS. The information about non-IBM ("vendor") products in this manual has been supplied by the vendor and IBM assumes no responsibility for its accuracy or completeness. The use of this information or the implementation of any of these techniques is a customer responsibility and depends on the customer's ability to evaluate and integrate them into the customer's operational environment. While each item may have been reviewed by IBM for accuracy in a specific situation, there is no guarantee that the same or similar results will be obtained elsewhere. Customers attempting to adapt these techniques to their own environments do so at their own risk.

Any pointers in this publication to external Web sites are provided for convenience only and do not in any manner serve as an endorsement of these Web sites.

Any performance data contained in this document was determined in a controlled environment, and therefore, the results that may be obtained in other operating

environments may vary significantly. Users of this document should verify the applicable data for their specific environment.

This document contains examples of data and reports used in daily business operations. To illustrate them as completely as possible, the examples contain 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.

Reference to PTF numbers that have not been released through the normal distribution process does not imply general availability. The purpose of including these reference numbers is to alert IBM customers to specific information relative to the implementation of the PTF when it becomes available to each customer according to the normal IBM PTF distribution process.

SET, SET Secure Electronic Transaction, and the SET Logo are trademarks owned by SET Secure Electronic Transaction LLC.

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

Appendix H. Related publications

The publications listed in this section are considered particularly suitable for a more detailed discussion of the topics covered in this redbook.

H.1 IBM Redbooks publications

For information on ordering these publications see “How to Get ITSO Redbooks” on page 127.

- *Planning to Install PeopleSoft with DB2 for OS/390,* SG24-5156
- *WOW! DRDA Supports TCP/IP: DB2 Server for OS/390 and DB2,* SG24-2212
- *Using RVA and SnapShot for Business Intelligence Applications for OS/390 and DB2,* SG24-5333
- *Implementing DFSMSdss SnapShot and Virtual Concurrent Copy,* SG24-5268
- *Implementing SnapShot,* SG24-2241
- *DB2 for OS/390 and Data Compression,* SG24-5261
- *Converting from Oracle AIX to DB2 for OS/390, SG24-5478*

H.2 IBM Redbooks collections

Redbooks are also available on the following CD-ROMs. Click the CD-ROMs button at http://www.redbooks.ibm.com/ for information about all the CD-ROMs offered, updates and formats.

CD-ROM Title	Collection Kit Number
System/390 Redbooks Collection	SK2T-2177
Networking and Systems Management Redbooks Collection	SK2T-6022
Transaction Processing and Data Management Redbooks Collection	SK2T-8038
Lotus Redbooks Collection	SK2T-8039
Tivoli Redbooks Collection	SK2T-8044
AS/400 Redbooks Collection	SK2T-2849
Netfinity Hardware and Software Redbooks Collection	SK2T-8046
RS/6000 Redbooks Collection (BkMgr Format)	SK2T-8040
RS/6000 Redbooks Collection (PDF Format)	SK2T-8043
Application Development Redbooks Collection	SK2T-8037
IBM Enterprise Storage and Systems Management Solutions	SK3T-3694

H.3 Other resources

These publications are also relevant as further information sources and are available through the the PeopleSoft Web site at: http://www.peoplesoft.com/

- *PeopleTools 7.5: Installation and Administration of DB2 for OS/390*
- *The Three-Tier Answer Book*
- *Hardware and Software Requirements*

H.4 Referenced Web sites

This Web site is also relevant as a further information source:

- `http://www.peoplesoft.com/` PeopleSoft home page

How to Get ITSO Redbooks

This section explains how both customers and IBM employees can find out about ITSO redbooks, redpieces, and CD-ROMs. A form for ordering books and CD-ROMs by fax or e-mail is also provided.

- **Redbooks Web Site** `http://www.redbooks.ibm.com/`

  Search for, view, download, or order hardcopy/CD-ROM redbooks from the redbooks Web site. Also read redpieces and download additional materials (code samples or diskette/CD-ROM images) from this redbooks site.

  Redpieces are redbooks in progress; not all redbooks become redpieces and sometimes just a few chapters will be published this way. The intent is to get the information out much quicker than the formal publishing process allows.

- **E-mail Orders**

  Send orders by e-mail including information from the redbooks fax order form to:

	e-mail address
In United States	usib6fpl@ibmmail.com
Outside North America	Contact information is in the “How to Order” section at this site: `http://www.elink.ibmlink.ibm.com/pbl/pbl/`

- **Telephone Orders**

United States (toll free)	1-800-879-2755
Canada (toll free)	1-800-IBM-4YOU
Outside North America	Country coordinator phone number is in the “How to Order” section at this site: `http://www.elink.ibmlink.ibm.com/pbl/pbl/`

- **Fax Orders**

United States (toll free)	1-800-445-9269
Canada	1-403-267-4455
Outside North America	Fax phone number is in the “How to Order” section at this site: `http://www.elink.ibmlink.ibm.com/pbl/pbl/`

This information was current at the time of publication, but is continually subject to change. The latest information may be found at the redbooks Web site.

IBM Intranet for Employees

IBM employees may register for information on workshops, residencies, and redbooks by accessing the IBM Intranet Web site at `http://w3.itso.ibm.com/` and clicking the ITSO Mailing List button. Look in the Materials repository for workshops, presentations, papers, and Web pages developed and written by the ITSO technical professionals; click the Additional Materials button. Employees may access `MyNews` at `http://w3.ibm.com/` for redbook, residency, and workshop announcements.

IBM Redbook Fax Order Form

Please send me the following:

Title	Order Number	Quantity

First name	Last name	
Company		
Address		
City	Postal code	Country
Telephone number	Telefax number	VAT number

☐ Invoice to customer number

☐ Credit card number

Credit card expiration date	Card issued to	Signature

We accept American Express, Diners, Eurocard, Master Card, and Visa. Payment by credit card not available in all countries. Signature mandatory for credit card payment.

List of abbreviations

AIX	Advanced Interactive Executive (IBM's flavor of UNIX)
API	application program interface
BSDS	bootstrap dataset (DB2)
CD-ROM	(optically read) compact disk - read-only memory
COBOL	common business-oriented language
DASD	direct access storage device
DBA	database administrator
DBD	database description
DBMS	database management system
DB2	DATABASE 2 (an IBM relational database management system)
DCS	Database Connection Services (part of DDCS)
DDCS	Distributed Database Connection Services (IBM program product)
DDF	Distributed Data Facility (DB2)
DDL	database definition language
DFDSS	Data Facility Data Set Services (IBM software product)
DFHSM	Data Facility Hierarchical Storage Manager
DRDA	**D**istributed Relational Database Architecture
ECSA	extended common service area
EDM	Environmental Description Manager
ERP	enterprise resource planning
ESCON	Enterprise Systems Connection (architecture, IBM System/390)
FBA	fixed block architecture
FDDI	fiber distributed data interface
FS	financial system
GB	gigabyte (10**9 bytes or 1,000,000,000 bytes)
HLQ	high-level qualifier
HR	human resources
IBM	International Business Machines Corporation
IP	Internet Protocol
ITSO	International Technical Support Organization
JCL	job control language
MB	megabyte, 1,000,000 bytes (1,048,576 bytes memory)
NFS	**N**etwork File System
NT	Microsoft Windows NT (New Technology)
ODBC	open database connectivity
OMVS	Open MVS (Multiple Virtual Storage - OS/390)
OLAP	Online Analytical Processing
OSA	Open Systems Adapter (IBM System/390)
OS/390	Operating System (for the) IBM System/390
PC	personal computer
PROC	command procedure
PTF	program temporary fix
RACF	Resource Access Control Facility
RAMAC	Raid Architecture with Multi-Level Adaptive Cache
RS/6000	IBM RISC System 6000
S/390	IBM System/390
SMIT	System Management Interface Tool
SPUFI	SQL processor using file input
SQL	structured query language
SYSADM	system administrator
TCP/IP	Transmission Control Protocol/ Internet Protocol
TSO	Time Sharing Option
VSAM	Virtual Storage Access Method (IBM)
VTAM	Virtual Telecommunications Access Method
VTOC	volume table of contents

Index

IBM Redbooks review

Your feedback is valued by the Redbook authors. In particular we are interested in situations where a Redbook "made the difference" in a task or problem you encountered. Using one of the following methods, **please review the Redbook, addressing value, subject matter, structure, depth and quality as appropriate.**

- Use the online **Contact us** review redbook form found at http://www.redbooks.ibm.com/
- Fax this form to: USA International Access Code + 1 914 432 8264
- Send your comments in an Internet note to redbook@us.ibm.com

Document Number **Redbook Title**	SG24-5648-00 Planning for a Migration of PeopleSoft 7.5 from Oracle/UNIX to DB2 for OS/390
Review	
What other subjects would you like to see IBM Redbooks address?	
Please rate your overall satisfaction:	O Very Good O Good O Average O Poor
Please identify yourself as belonging to one of the following groups:	O Customer O Business Partner O Solution Developer O IBM, Lotus or Tivoli Employee O None of the above
Your email address: The data you provide here may be used to provide you with information from IBM or our business partners about our products, services or activities.	O Please do not use the information collected here for future marketing or promotional contacts or other communications beyond the scope of this transaction.
Questions about IBM's privacy policy?	The following link explains how we protect your personal information. http://www.ibm.com/privacy/yourprivacy/

SG24-5648-00
Printed in the U.S.A.

Planning for a Migration of PeopleSoft 7.5 from Oracle/UNIX to DB2 for OS/390

SG24-5648-00