IBM Software
Information Management Skills Development

Introduction to DB2® LUW Performance

Version/Revision# **1**

Module ID **10300**

Length **1.5 hours**

Table of Contents

1 Introduction

DB2 10 is an enterprise database engine with a rich set of features and tools providing unmatched capabilities to achieve best performance. It contains innovative features for delivering information on demand and scaling databases to new levels.

2 About this Lab

In this first introductory lab we take you through setting up a new database for optimal performance. Specifically, you will learn

- How to setup tablespaces and buffer pools
- How to use the Configuration Advisor to establish an initial well-performing database and instance configuration
- How to collect information that identifies system resources
- How to collect information about the database environment

In this part of the lab, the attendee will work with simply preparing the DB2 environment for proceeding through the labs.

3 Suggested Reading

Best practices for DB2 for Linux, UNIX, and Windows

Practical guidance for the most common DB2 10 product configurations and use this knowledge to improve the value of your DB2 data servers.

http://www.ibm.com/developerworks/data/bestpractices/

4 Environment Requirements and Setup

To complete this lab you will need the following:

- DB2 Performance Clinic VMware® image
- VMware Player 2.x or VMware Workstation 6.5 or later

4.1 Unpacking the image

The image is delivered in a self-extractable set of zip files. For easy handling the files are compressed to 700MB volumes. Download all the volumes to the same directory.

Double click the executable file and select the destination folder.

4.2 Preparation Step

1. Choose the **DB2_BLU-AWSE_10.5.…vmx** file when first opening the lab image in the **/DB2_BLU_PerfMonitoring...** folder and run it.

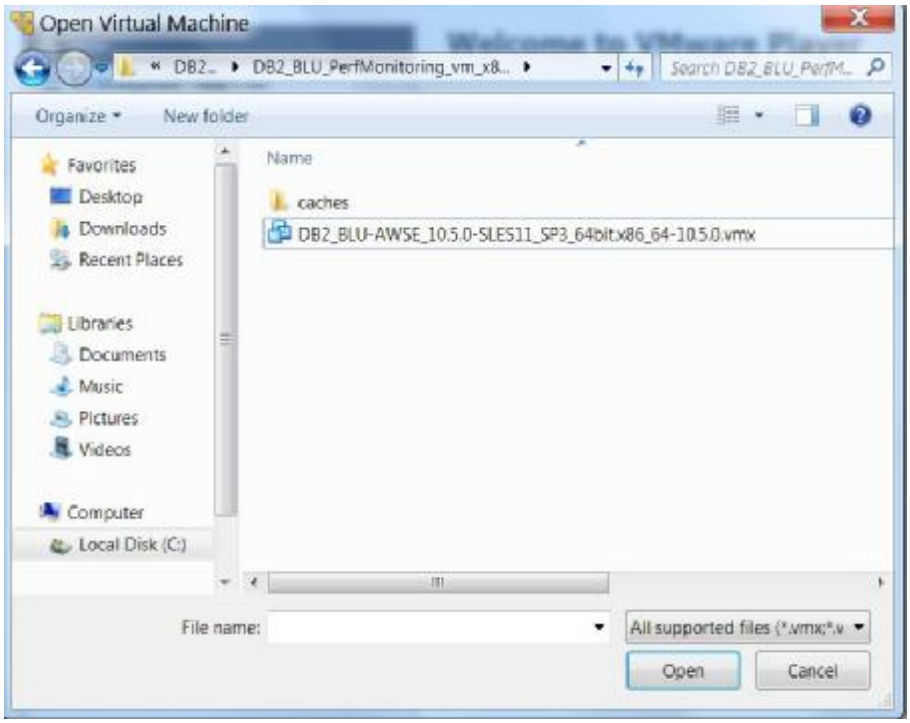

Figure 1 – Opening a VMware Image File

4.3 Start the Virtual Machine

Start the VMware image by clicking the 

button in VMware Workstation if it is not already on.

ℹ ***Note***: **Please wait for the first boot setup process to complete.**

The system will power up like any other Linux system and will come to the state as shown in the next picture below.

Figure 2 – Start the Virtual Machine

After the virtual machine has finished booting up, you can now work inside the virtual machine environment. To bring focus into the virtual machine environment, click inside the virtual machine screen with your mouse or click on the "**`Full Screen`**" button  in the toolbar on top of the VMware window.

After clicking on the screen, you may not see your mouse pointer anymore, this is normal as you are now operating in a command line mode inside the virtual machine.
You can bring focus to the host operating system at any point by pressing **"`Alt + Ctrl`"** at the same time.

4.4 Login to the Virtual Machine and Accept the License Agreement

You will see a pop-up message asking you to read and accept the License Agreement.

Figure 3 – Linux Distribution Statement

Use the **"Page Down"** key to scroll down and read the full license agreement. Once you have read the agreement, select **"Yes, I Agree to the License Agreement"** and click **"Next"** to go to the next screen.
This will take you to the **VMware Tools License Agreement**.

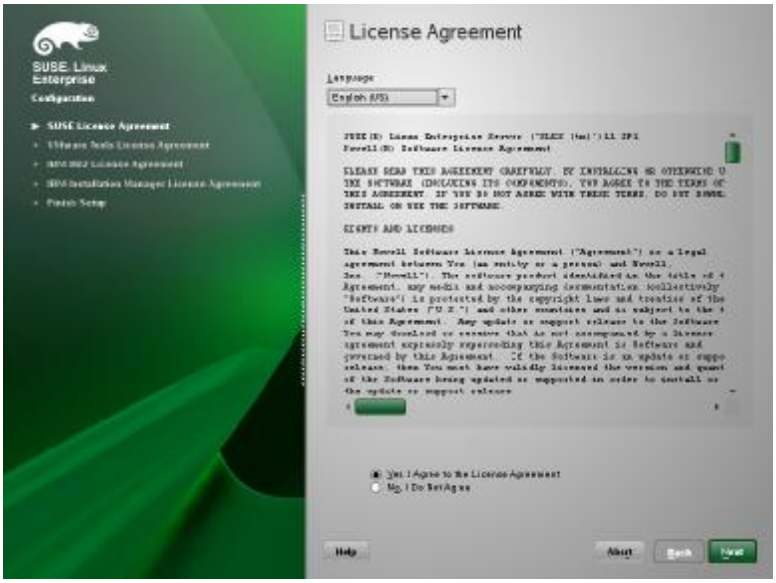

Figure 4 – SUSE Linux Enterprise Server License Agreement

Figure 5 – VMware Tools License Agreement

Use the **"`Page Down`"** key to scroll down and read the full license agreement. Once you have read the agreement, select **"`Yes, I Agree to the License Agreement`"** to go to the next screen. This will take you the **IBM DB2 10 License Agreement**:

Figure 6 – DB2 V10 License Agreement

Use the **"Page Down"** key to scroll down and read the full license agreement. Once you have read the agreement, select **"Yes, I Agree to the License Agreement"** to go to the next screen. This will take you the **IBM Installation Manager License Agreement**:

Figure 7 – DB2 V10 Trial Version Notice

This screen shows all of the Software Licenses, and Non-IBM license agreements that were displayed in all of the previous screens. In order to use this image, you must accept all of the listed agreements that were displayed in the previous screens. Select **"Next"** to finish. If you do not agree with the license agreements, select **"Abort"** and the virtual machine will be shutdown automatically.

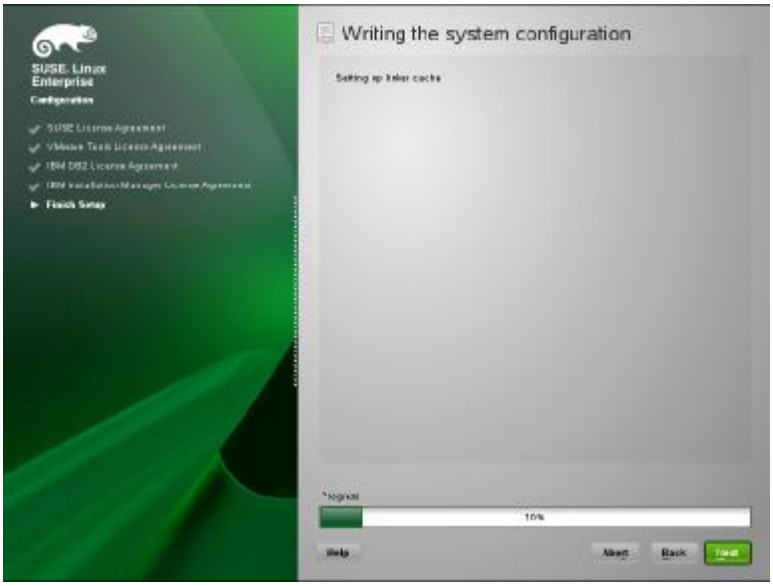

Figure 8 – License Agreements

At the login prompt, login with the following credentials:

- Username: **`db2inst1`**
- Password: **`password`**

Figure 9 – Login credentials

4.5 Open the Terminal Window

1. Open a new terminal window by right-clicking on the **Desktop** and choosing the "**Open Terminal**" item:

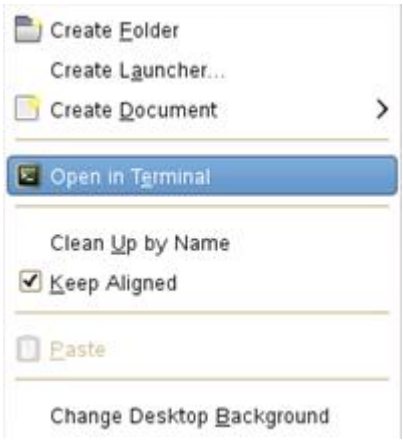

Figure 10 – Opening a Terminal

2. Start the Database Manager as follows:

    4 Enter the command "**`db2start`**" to start the database instance. Make sure the instance is started successfully. (**Note**: If you notice a message that says "**`The database manager is already active`**", that's OK. Please continue to the next section).

4 In order to enable the aliases defined for this lab and all labs that follow, copy the /home/db2inst1/labbashrc to /home/db2inst1/.bashrc and source that file.

```
cp /home/db2inst1/labbashrc .bashrc
. .bashrc
```

4 If your LabScripts folder in is compressed.

```
cd /home/db2inst1/Documents
unrar x LabScripts
```

i **Attention: You need to type a space between the two dots.**

5 Database Creation Best Practices and Exercises

It is important to be aware of the default behavior when using the simplest form of the CREATE DATABASE command, as many of the default settings has performance implications. For instance, the command “CREATE DATABASE <dbname>” results in:

- Configuration Advisor running against the database with somewhat conservative options
  - Best Practice: Run the Configuration Advisor, after database has been created and input information, such as the concurrent number of users and types of transactions. Doing this assumes that DB2 tuning experience is at a beginner level.
- Automatic statistics collection being enabled
  - Best Practice: To ensure that the optimizer has a good set of working statistics leave this setting as is. Certain tables may require special designations, like “Volatile.” Additionally, a statistics profile can be registered for tables requiring more than the default collection method. This profile will be respected by the automatic statistics gathering process. See the DB2 Information Center for more information and restrictions on this feature.
- Adaptive Self Tuning Memory being enabled
  - Best Practice: Leave this feature on when the system is a dedicated DB2 server. However, with respect to the nature and criticality of memory allocation, it is best employed after thorough system testing. Monitor system memory and the db2diag.log for actual heap allocations. For a DPF environment, see DB2 Information Center for further considerations including selecting a “tuning node”.
- Unicode database
  - This designation is required if you intend to use the XML native data type or if multi byte code set is required. If required, be mindful to the collating sequence you choose, Unicode culturally correct collating sequences like UCA500R1_xxx can cause additional CPU overhead. There are alternative collations like SYSTEM_819_BE that take advantage of Unicode databases that contain data in only one language. They use the same lookup table-based collation algorithm as single-byte collations such as SYSTEM_819, making them very efficient.

- Automatic Storage using a single location for containers (defined by the database manager configuration parameter DFTDBPATH)
    - Though additional storage paths can be added to a database with the ALTER DATABASE command, it is advisable to establish multiple storage paths on dedicated devices before the database is populated. When you create tablespaces, by default they will be designated to use automatic storage, be mindful to specify attributes like INITIALSIZE, MAXSIZE, and INCREASESIZE.
- Database control and log files being stored in same location as tablespace containers (also defined by the database manager configuration parameter DFTDBPATH)
    - For optimal performance move the transaction log files to their own dedicated file system after database creation.
- Default page size of 4K for default buffer pool and the SYSCATSPACE, TEMPSPACE1, and USERSPACE1 tablespaces
- Isolation level being set to Cursor Stability with currently committed semantics
- Detailed deadlocks event monitor (DB2DETAILDEADLOCK) created and enabled (This monitor is deprecated, but used for transitional purposes

Some of the best practices cited above cannot actually be addressed until after the database has been created, while other factors can be addressed with the CREATE DATABASE command itself. The autonomic and automatic default options should be kept.

You will definitely want to address the placement of log files and tablespace containers. By default, db2 database metadata, transaction log files and tablespace containers are all placed in the same default location. While this works, in a real production system this can result in severe performance degradation, especially for large, transaction laden databases.

5.1 Create customized database

1. Create a database with non default locations for transactional log files and tablespaces. Issue the following command from the terminal window as db2inst1:

```
db2 "create database INTRO ON /data1, /data2 DBPATH ON /home/db2inst1
RESTRICTIVE"
```

This command uses the **ON** clause which indicates that the database will use automatic storage and all automatic tablespace containers will be created under specified locations. The **"DBPATH ON"** clause indicates that database metadata along with log control and data files will be created under: **/home/db2inst1.** If DBPATH is not specified these things will be created on the first path found in the ON clause.

2. The home directory is acceptable for database metadata but leaving the log data and control files in the same location is not the best for transactional performance. Issue the following command to verify location of log files:

```
db2 get db cfg for intro |grep -i "path to log"
```

Expected output:

```
Changed path to log files (NEWLOGPATH) =
Path to log files                      = /home/db2inst1/db2inst1/NODE0000/SQL00001/LOGSTREAM0000/
```

You can see that the log file location is the same as the one for database metadata.

DB2 Object	Location
Database configuration file SQLDBCON	home/db2inst1/db2inst1/NODE0000/SQL00001
Database directory Contains files needed for: • buffer pool information • history information • log control files • storage path information • tablespace information	home/db2inst1/db2inst1/NODE0000/SQL00001
Directory for event monitor data	home/db2inst1/db2inst1/NODE0000/SQL00001/MEMBER0000/db2event
Directory for transaction log files	home/db2inst1/db2inst1/NODE0000/SQL00001/LOGSTREAM0000
Local database directory for the instance	home/db2inst1/db2inst1/NODE0000/sqldbdir

3. Change the path to a completely separate location. Issue this command from terminal:

```
db2 "update db cfg for intro using NEWLOGPATH /logs"
```

4. Verify the path has been changed. Issue this command:

```
db2 get db cfg for intro |grep -i "path to log"
```

Expected output:

```
Changed path to log files (NEWLOGPATH) = /logs/NODE0000/LOGSTREAM0000
Path to log files                      = /home/db2inst1/db2inst1/NODE0000/SQL00001/LOGSTREAM0000/
```

Attention: Note that "Path to log files" still shows the old location but the effective log path is the location indicated by NEWLOGPATH parameter. The new log path will be used after a restart.

We had also specified the RESTRICTIVE key word in the create database command. (This can be checked by displaying the database configuration file with the "get db cfg" command under Restrict Access). Without it, db2 catalog information is readable by PUBLIC, i.e. anyone connected to the database. This will revoke the authorization to PUBLIC and make the database more secure. See DB2 Information center for other restrictions.

5.2 Registry Variables

There are three types of configurations in a DB2 environment:

- Database manager configuration
- Database configuration
- Registry variables

So far, we have looked at database and database manager configurations. In this section we look at some significant performance related registry variables. Note that the Configuration Advisor does not enable any registry variables. Remember that for registry variables to take effect a complete db2 recycle using db2stop and db2start is required.

5.2.1 DB2_MEM_TUNING_RANGE

This registry variable is related to self tuning memory manager (STMM). If STMM is being used, it will take more memory, as needed, on the server as long as there is free memory available. This may cause problems on systems where you need to leave some memory for another application or for the system. The syntax is as follows:

db2set DB2_MEM_TUNING_RANGE=MINFREE, MAXFREE

MINFREE and MAXFREE are percentage numbers. So if you want to dedicate no more than 60% of system memory to db2, you would set MINFREE to 40% which means STMM should take no more than 60% of system memory so there is always at least 40% free memory available. STMM will release memory when it is not needed but you may want to control how much memory STMM gives up. For example, if the requirement is that STMM should retain al least 20% of system memory, then MAXFREE would be set to 80%.

1. In order to make these settings, execute the following from command terminal:

```
db2set DB2_MEM_TUNING_RANGE=40,80
```

5.2.2 DB2MEMDISCLAIM

DB2 agents may have some associated paging space. This paging space may remain allocated even after the associated memory has been freed depending on the platform and virtual memory tunings. `DB2MEMDISCLAIM`, when set, indicates that explicit requests should be made by DB2 to free associated paging space as well. This leads to smaller paging space requirements and possibly less disk activity from paging. On systems where memory is tight and paging space is being used, this may provide a performance improvement.

1. Enable it as follows from command terminal

```
db2set DB2MEMDISCLAIM=YES
db2stop force
db2start
```

5.2.3 DB2_SMS_TRUNC_TMPTABLE_THRESH

By default, when a temporary table is not needed it is truncated to zero pages. If there is a lot of temp table activity, this can degrade performance. Using this registry variable, one can specify the number of extents that should be preserved upon truncation so that when the table is reused there are pre allocated extents available for use.

1. Run the following command to indicate that 10 extents should be preserved when truncating any temp table:

```
db2set DB2_SMS_TRUNC_TMPTABLE_THRESH=10
```

Restart DB2 so that all changes take effect.

```
db2stop force
db2start
```

5.3 Buffer pools and Tablespaces

1. When a database is created, a default user tablespace and bufferpool are created. Issue this command to check which tablespace has been created:

```
db2 connect to INTRO
db2 list tablespaces show detail
```

There should be three tablespaces created:

- SYSCATSPACE (catalog data)
- TEMPSPACE1 (system temp)
- USERSPACE1 (user data)

USERSPACE1 is the default tablespace created and it has a page size of 4k. DB2 supports four page sizes: 4k, 8k, 16k and 32k. You may have tables with large rows that do not fit on a 4k page, so tablespaces with larger page sizes should also be created. However, larger page size tablespaces need buffer pools with matching page size.

2. Check to see what kind of buffer pools have been created by using the following query:

```
db2 "select substr(BPNAME, 1,25) as BPNAME, NPAGES, PAGESIZE FROM
SYSCAT.BUFFERPOOLS"
```

Expected output looks like:

```
BPNAME                    NPAGES      PAGESIZE
------------------------- ----------- -----------
IBMDEFAULTBP                   -2         4096
```

There is only one bufferpool for all type of tables and activities in the entire database. Also note that it is enabled for Automatic memory tuning. NPAGES value of -2 indicates that IBMDEFAULTBP buffer pool has been enabled for automatic tuning.

5.3.1 Create customized Buffer pools and Tablespaces

Create automatic buffer pools for the rest of the page sizes. Create a dedicated buffer pool for the system temp tablespace. It is always a good design to segregate temp activity to its own buffer pool. Temp space activity, when heavy, can be disruptive to regular database activity due to frequent creation and removal of temp tables.

1. Run the following statements to achieve the above stated goals:

```
db2 create bufferpool bp8k size automatic pagesize 8k
db2 create bufferpool bp16k size automatic pagesize 16k
db2 create bufferpool bp32k size automatic pagesize 32k
db2 create bufferpool tempbp size automatic pagesize 4k
```

2. Run the following command to verify the results:

```
db2 "select substr(BPNAME, 1,25) as BPNAME, NPAGES, PAGESIZE FROM
SYSCAT.BUFFERPOOLS"
```

Expected output:

```
BPNAME                    NPAGES      PAGESIZE
------------------------- ----------- -----------
IBMDEFAULTBP                       -2        4096
BP8K                               -2        8192
BP16K                              -2       16384
BP32K                              -2       32768
TEMPBP                             -2        4096
```

3. Now that buffer pools have been created, we can create the corresponding tablespaces. Execute the following from command terminal:

```
db2 create tablespace tbsp8k  pagesize 8k bufferpool bp8k
db2 create tablespace tbsp16k pagesize 16k bufferpool bp16k
db2 create tablespace tbsp32k pagesize 32k bufferpool bp32k
```

4. Verify that all tablespaces are associated with the intended buffer pools. Run the following query:

```
db2 "select substr(bpname,1,25) as BPNAME, substr(tbspace,1,25) as TBSPACE
from syscat.bufferpools B ,syscat.tablespaces T
where B.BUFFERPOOLID=T.BUFFERPOOLID"
```

Expected output:

```
BPNAME                  TBSPACE
----------------------- -----------------------
IBMDEFAULTBP            SYSCATSPACE
IBMDEFAULTBP            TEMPSPACE1
IBMDEFAULTBP            USERSPACE1
IBMDEFAULTBP            SYSTOOLSPACE
BP8K                    TBSP8K
BP16K                   TBSP16K
BP32K                   TBSP32K
```

All looks fine except that TEMPSPACE1 is still associated with IBMDEFAULTBP. Recall that we had created a dedicated bufferpool for TEMPSPACE1 but we never altered the tablespace to point to it. So TEMPBP buffer pool exists but is not being used by TEMPSPACE1.

5. Run the following statement to fix it:

```
db2 alter tablespace TEMPSPACE1 bufferpool TEMPBP
```

6. Run previous query *(step #4)* again to verify TEMPSPACE1 is pointing to BPTEMP.

5.3.2 Create dedicated tablespace for LOB/LONG data

If the database is going to contain significant amount of LOB/LONG data, it is a good idea to create dedicated tablespace(s) for LOB data. There are two design reasons for it. First, LOB data can be quite large so it can quickly max out the capacity of your regular data tablespaces. Secondly, LOB data is not buffered in DB2 bufferpools, therefore you want to ensure that it is being buffered in the file system cache. Since regular data is buffered in buffer pools there is no need for double buffering it in file system cache. However, for LOB data file system caching is critical.

1. Execute the following command to create dedicated tablespace for LOB data:

```
db2 create LARGE tablespace tbsplong managed by automatic storage FILE
SYSTEM CACHING
```

5.4 Configuration Advisor

1. When a database is created in DB2, the Configuration Advisor is run under the covers automatically using default values. This behavior can be disabled with the db2set command before creating a database**:**

```
db2set DB2_ENABLE_AUTOCONFIG_DEFAULT=NO
```

Configuration advisor was also run for INTRO database with default values but we want to run it again with user supplied input.

2. You will use the AUTCONFIGURE command to provide database usage characteristics and get recommended values to improve the initial configuration of the database with respect to the input. Here we are only retrieving recommendations (`apply none`) but you can also optionally apply recommendations as well (`apply db only`, `apply db and dbm`).

```
db2 connect to INTRO
db2 autoconfigure using mem_percent 50  tpm 200 admin_priority performance
is_populated no num_remote_apps 200 isolation cs apply none
```

Keyword	Values	Default	Explanation	Lab Setting
Mem_percent	1 – 100	25	Percentage of instance memory that is assigned to the database.	50
Workload_type	Simple, Mixed, Complex	Mixed	Simple workloads tend to be I/O intensive mostly transactions whereas complex ones tend to be CPU intensive and mostly queries	Default
Num_stmts	1 – 1000000	10	Number of statements per unit of work	Default
Tpm	1 – 2000000	60	Transactions per minute	200
Admin_priority	Performance, Recovery, Both	Both	Optimize for better performance or better recovery time.	Perf.
Is_populated	Yes, no	Yes	Is the database populated with data	No
Num_local_apps	0 – 5000	10	Number of connected local applications	Default
Num_remote_apps	0 – 5000	10	Number of connected remote applications	200
Isolation	RR, RS, CS. UR	RR	Maximum Isolation level applications connecting to this database	CS
Bp _resizable	Yes, no	Yes	Indicates if buffer pools can be re-sized.	Default

AUTOCONFIGURE OUTPUT:

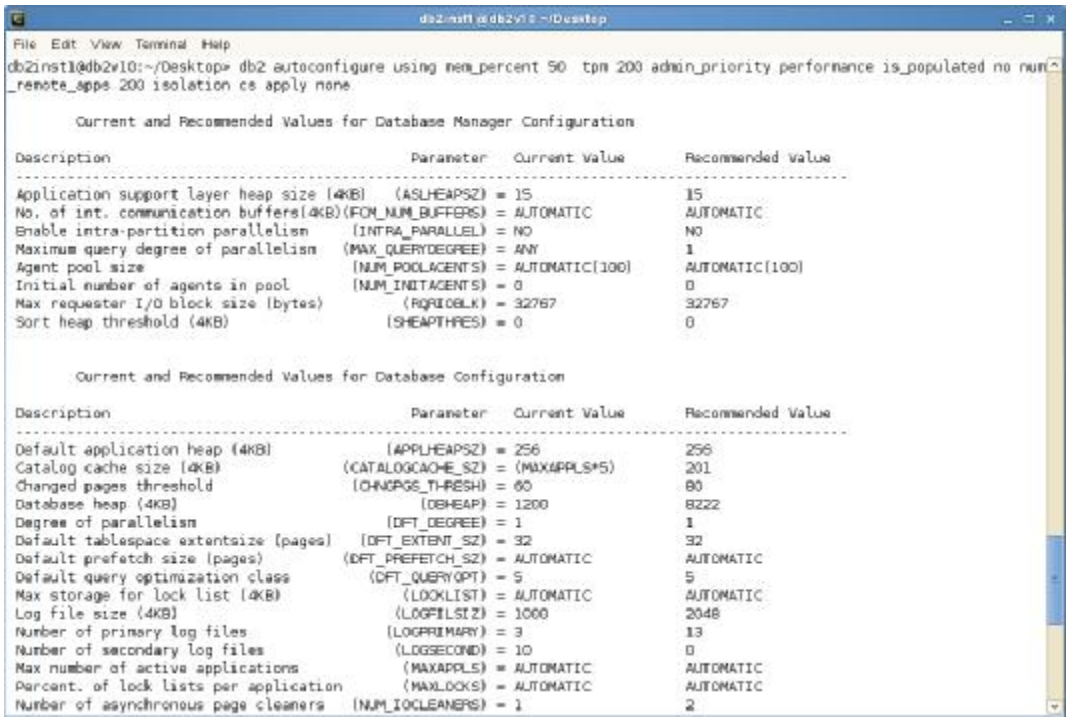

Figure 11 – Auto configure output

6 Collecting Baseline Information

When you start to think about how to configure your DB2 database and system environment, you need to be aware of what system resources are at your disposal. This includes things like processing power, memory, and disk storage. Though DB2 can detect most resources and configure them accordingly, it is always a good idea to understand them yourself.

In this section, we will look at some of the tools that are typically used to collect this information. We will also explore how to collect DB2 configuration information. Keeping track of such system and database level information is important in both understanding how the system changes over time and to help gauge what effects of any tuning has on the system.

6.1 System Level

In this part of the exercise, we are going to use the DB2 Problem Determination tool (`db2pd`), which is both useful and flexible, to better understand our system environment. The tool is much more robust and powerful than demonstrated here, and it is very efficient to use it to collect system-level information.

Additionally, it nicely transcends the different operating systems and can be used anywhere where DB2 is installed. The content can differ across operating systems, but more or less provides the same information,

For this purpose, issue the following command:

```
db2pd –osinfo disk /data1
```

Using this information, please identify the following:

What Operating System and Version is running? _____

What is the Nodename? _____

What processors are being used? _____

How many processors? _____

What is the speed of the processors in MHz? _____

How much total memory is on the machine in MBs? _____

How much virtual memory has been set? _____

How much memory is currently free? _____

What is total disk storage size? _____

How much disk storage is available? _____

As you can tell, this is all very useful information to know prior to database tuning.

Similar information can also be obtained using SQL if you have already created a database and are connected to it. This is useful if you want to programmatically collect such information.

1. To see the system information for INTRO database, issue the following commands:

```
db2 connect to INTRO
db2 "select substr(name,1,20) as name, substr(value,1,20) as value from
sysibmadm.env_sys_resources"
```

6.2 Database Level

Once you have created a database, it is usually useful to maintain some kind of historical records of its configuration. This can help you better understand how the configuration of the database evolves over time, as well as help identify configuration changes that affect performance.

Such details usually include configuration information about the database manager and database configuration files, buffer pools, tablespaces, and even the DDL of your database objects.

The `db2support` problem analysis and environment collection tool is essentially a one-stop-shop for all environment information. With one command you can collect all of the environment and database information such as database manager configuration, database configuration, registry variables, tablespace layouts, buffer pool sizes and a lot more. Though it is primarily intended for DB2 support purposes, it also works nicely to collect information about a system with which you may not be familiar.

1. Run db2support from the command window terminal:

```
cd /home/db2inst1/
mkdir envdata
cd ~/envdata
db2support . –d INTRO –c
```

Attention: For the purpose of this lab, you should take a look at the generated files, which can be found in the /home/db2inst1/envdata directory, please continue in the next instruction.

2. Once the tool finishes, it will have all the information collected and available in db2support.zip file. Unzip the file to see the db2support.html file by issuing the following command.

```
unzip db2support.zip
```

3. Start up **Firefox** from **My computer** menu in the lower left hand corner. Once Firefox browser is up, go to **File à Open File**, specify location for **db2support.html** file to bring it up in the browser.

(**Note**: If you notice a message that says “Server not found”, just ignore it)

You should now be able to see all the environment configuration information in the browser broken up into different sections. A sample of the output produced is listed below:

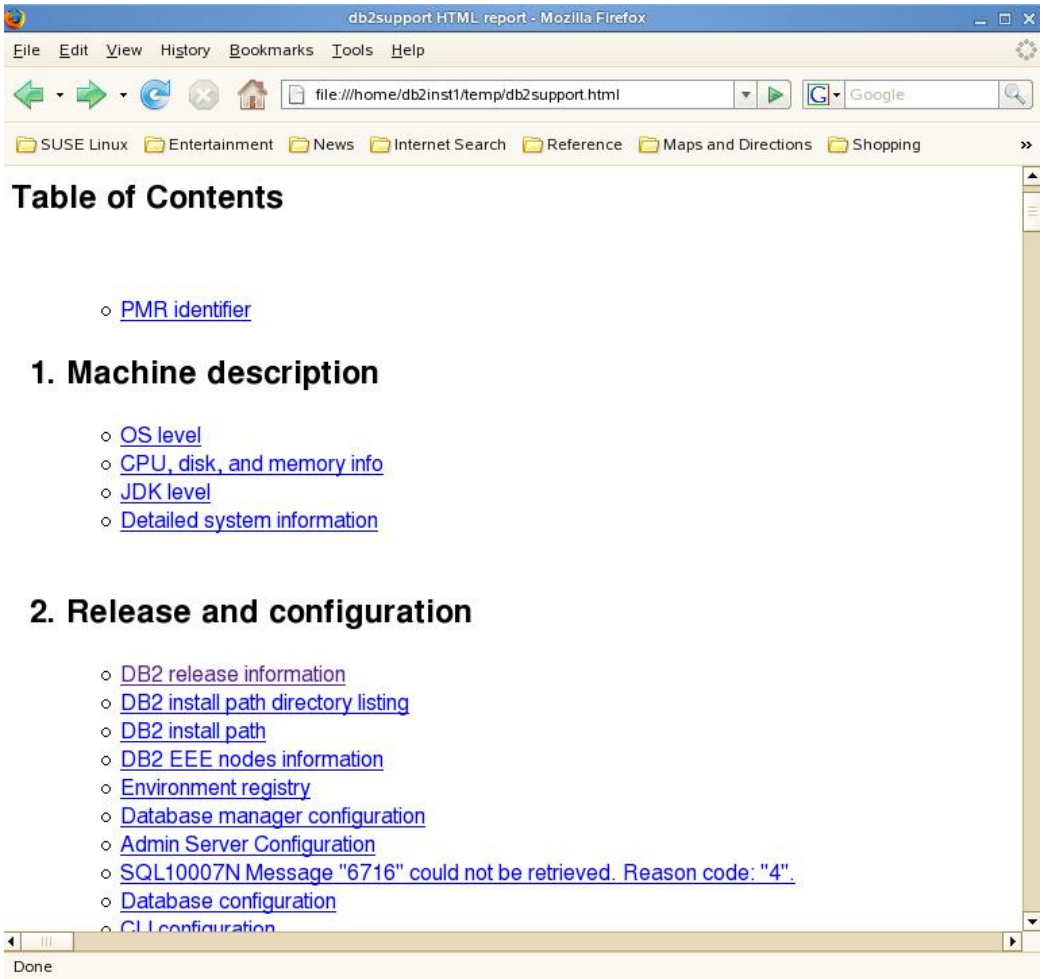

Figure 12 – Opening a db2support HTML report

Browse and review the configuration information dumped by db2support. Based on the output produced, answer the following questions:

Q1: What databases exist, other than INTRO database?

Q2: Can you identify the tablespace containers for all the tablespaces in the INTRO database?

Q3: What is db2 version and fixpack level being used?

So far we have looked at tools to check the configuration information. There are times when you need to dump the database schema containing DDL for all the tables, indexes, triggers, stored procedures and functions that were created by the user. This can be done with the `db2look` command. This command offers a host of options to tailor the results to your needs. With respect to performance related information the following parameters can be used:

- -l : bufferpools and tablespaces, created
- -f : database parameters that impact the query optimizer
- -x or -xd : authorizations specified with the RESTRICTIVE parameter of the CREATE DATABASE command
- -m ; statistics.

Issue the following command from command terminal:

```
db2look -d INTRO –f –o queryParameters.out
db2look –d INTRO –l –o BPSandTBS.out
```

View the output files specified. In the queryParameters.out file, you will see the database manager and database parameters along with any registry settings that affect performance. In the BPS_TBS.out file, you will see the bufferpools and tablespaces and their DDL.

7 Summary

You can see by now that there are three key areas that are important to understand when trying to avoid performance degradations of your system: configuration, monitoring, and performance troubleshooting. One of the best practices around performance tuning and monitoring is to collect baseline information. We explored some of the tools provided by DB2 to gather the initial data that will help you to understand system performance under both operational and troubleshooting conditions.

8 Cleanup

At this point, we will must clean up both the database and environment to prepare for the next section. Please issue the following commands:

```
db2 force applications all
db2 terminate
db2 drop db intro
db2set DB2_SMS_TRUNC_TMPTABLE_THRESH=
db2set DB2_MEM_TUNING_RANGE=
db2set DB2MEMDISCLAIM=NO
db2stop
```

IBM Canada
8200 Warden Avenue
Markham, ON
L6G 1C7
Canada