Db2 for z/OS Implementation

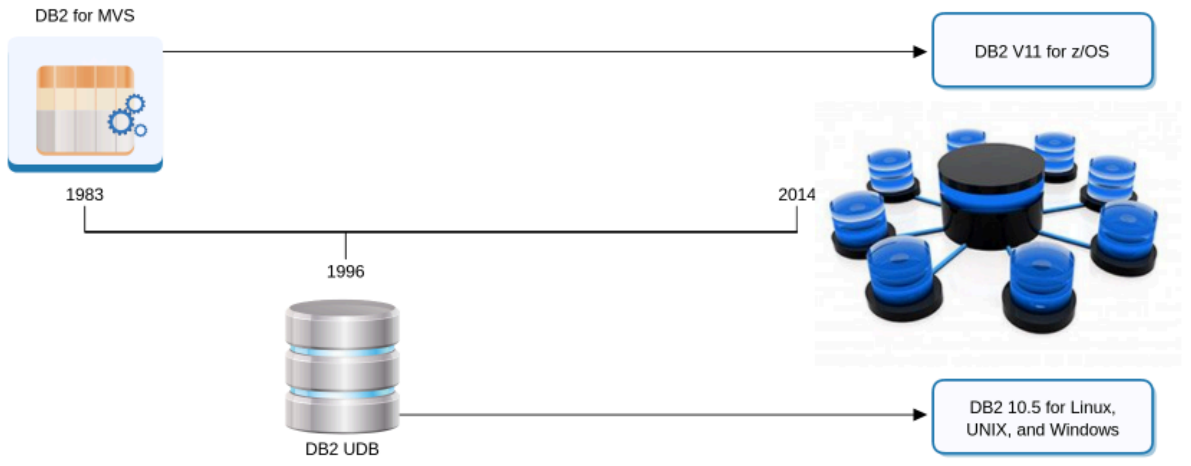
Objectives

Db2 for z/OS Implementation

In this module you will see how Db2 has evolved and look at some of the tools and utilities that are used to connect and interact with Db2 for z/OS. You will discover how storage for Db2 in a z/OS environment is configured and how SQL is prepared for application access to Db2 data.

After completing the module, you should be able to:

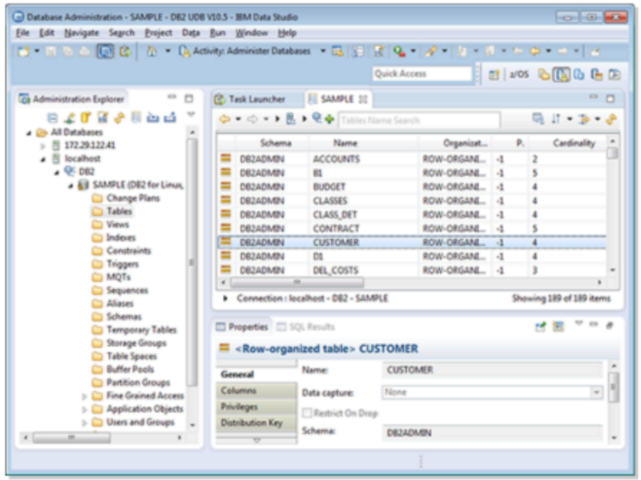
- Identify the Methods Used for Connecting to Db2 for z/OS
- Describe How z/OS Storage Can be Configured for Use by Db2 for z/OS
- Explain How SQL is Prepared for Application Use



While Db2 origins were well before the Db2 for MVS version 1 release in 1983, this version marked the birth of Db2 for the mainframe. Db2 ran exclusively on mainframes for several years, until the late 80's and 90's where versions of Db2 for distributed systems such as OS/2, AIX and UNIX appeared. These distributed versions combined in 1996 to form DB Universal Database (UDB) version 5.

These iterations of Db2 have evolved throughout the years to become: Db2 V11 for z/OS and Db2 10.5 for Linux, UNIX, and Windows.

Db2 for z/OS differs in several significant ways to Db2 for Linux, UNIX, and Windows. This module looks at these differences, in particular the interfaces, physical storage and implementation.



Schema	Name	Organizat...	P.	Cardinality
DB2ADMIN	ACCOUNTS	ROW-ORGANI...	-1	2
DB2ADMIN	B1	ROW-ORGANI...	-1	5
DB2ADMIN	BUDGET	ROW-ORGANI...	-1	4
DB2ADMIN	CLASSES	ROW-ORGANI...	-1	4
DB2ADMIN	CLASS_DET	ROW-ORGANI...	-1	4
DB2ADMIN	CONTRACT	ROW-ORGANI...	-1	5
DB2ADMIN	CUSTOMER	ROW-ORGANI...	-1	4
DB2ADMIN	D1	ROW-ORGANI...	-1	4
DB2ADMIN	DEL_COSTS	ROW-ORGANI...	-1	3

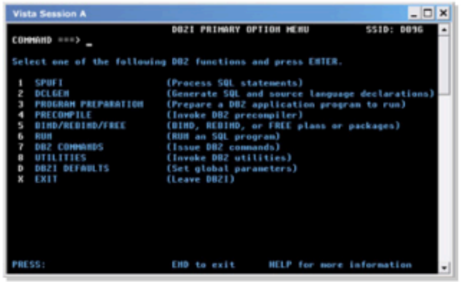
Connection : localhost - DB2 - SAMPLE　　Showing 189 of 189 items

<Row-organized table> CUSTOMER

General
Columns
Privileges
Distribution Key

Name: CUSTOMER
Data capture: None
Restrict On Drop
Schema: DB2ADMIN

IBM Data Studio offers administration and development capabilities of DB2 for Linux, UNIX and Windows. It also provides collaborative database development tools for DB2 for z/OS users.



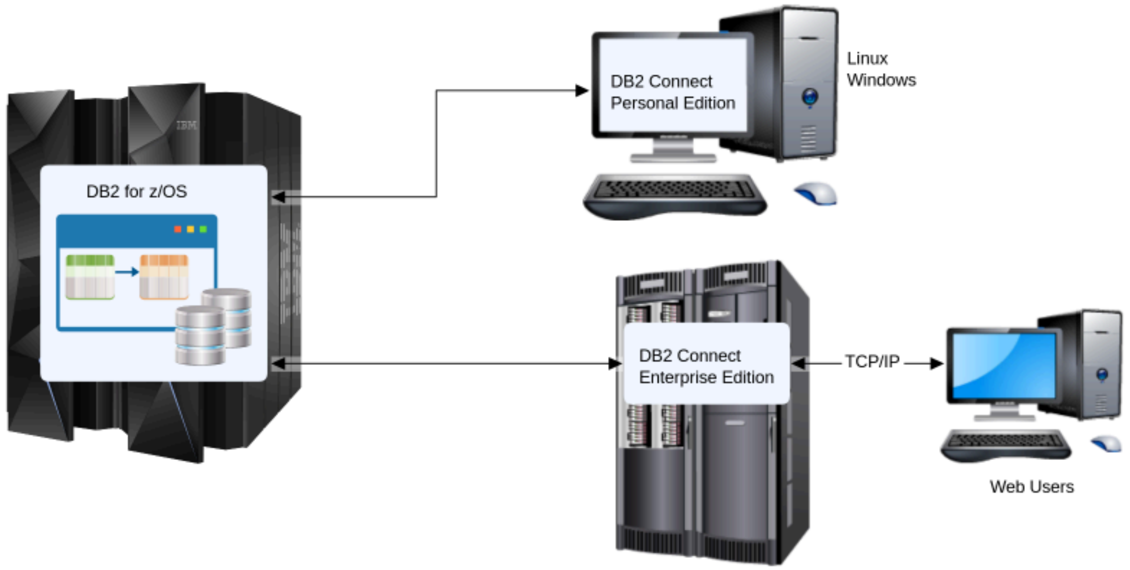
```
                        DB2I PRIMARY OPTION MENU              SSID: DB9G
COMMAND ===> _

Select one of the following DB2 functions and press ENTER.

 1  SPUFI                  (Process SQL statements)
 2  DCLGEN                 (Generate SQL and source language declarations)
 3  PROGRAM PREPARATION    (Prepare a DB2 application program to run)
 4  PRECOMPILE             (Invoke DB2 precompiler)
 5  BIND/REBIND/FREE       (BIND, REBIND, or FREE plans or packages)
 6  RUN                    (RUN an SQL program)
 7  DB2 COMMANDS           (Issue DB2 commands)
 8  UTILITIES              (Invoke DB2 utilities)
 D  DB2I DEFAULTS          (Set global parameters)
 X  EXIT                   (Leave DB2I)




PRESS:                  END to exit      HELP for more information
```

DB2 for z/OS tools and utilities are accessible through an ISPF interface.

Db2 provides a variety of tools that simplify the tasks that you need to do to manage Db2.

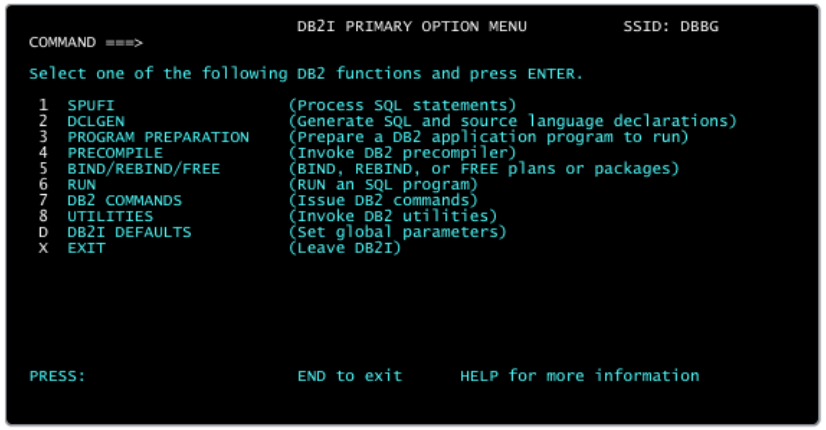
Most of the database tools that support Db2 for z/OS provide a graphical user interface (GUI) and also contain an ISPF (Interactive System Productivity Facility) interface that allows you to perform most Db2 tasks interactively.

Many GUI tools that run on other platforms can be used to manage Db2 on z/OS, including IBM Data Studio.



As discussed in an earlier module, Db2 Connect allows web, Windows, UNIX, Linux and mobile applications to connect with IBM z/OS Db2 data.

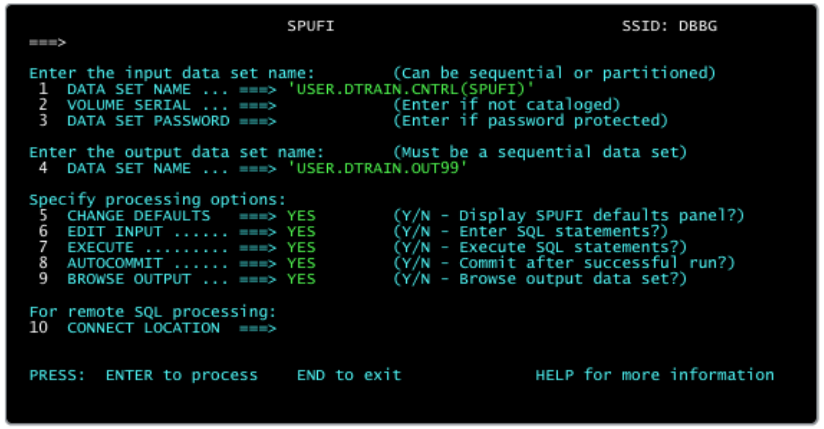
Db2 Connect can exist in two formats: as a single point of connection server supporting numerous workstations and a variety of applications, or as a Db2 Connect client that uses the DRDA protocol to connect directly to the mainframe Db2 database.



```
                            DB2I PRIMARY OPTION MENU          SSID: DBBG
COMMAND ===>

Select one of the following DB2 functions and press ENTER.

 1  SPUFI                  (Process SQL statements)
 2  DCLGEN                 (Generate SQL and source language declarations)
 3  PROGRAM PREPARATION    (Prepare a DB2 application program to run)
 4  PRECOMPILE             (Invoke DB2 precompiler)
 5  BIND/REBIND/FREE       (BIND, REBIND, or FREE plans or packages)
 6  RUN                    (RUN an SQL program)
 7  DB2 COMMANDS           (Issue DB2 commands)
 8  UTILITIES              (Invoke DB2 utilities)
 D  DB2I DEFAULTS          (Set global parameters)
 X  EXIT                   (Leave DB2I)






PRESS:                      END to exit      HELP for more information
```

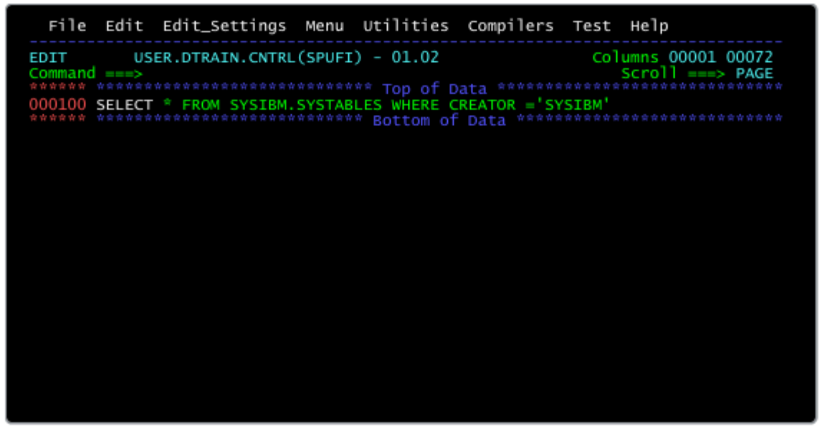
Db2 for z/OS provides a comprehensive set of tools to develop and maintain Db2 from TSO/ISPF. Many of these tools are restricted to database administrator (DBA) use.

The DB2I Primary Option Menu provides access to most of the tools required to manage Db2, but you should remember that Db2 for z/OS may be in use by thousands of clients and applications at any time, so the tools and commands available from this interface must be used carefully so as not to disrupt their access to the data.



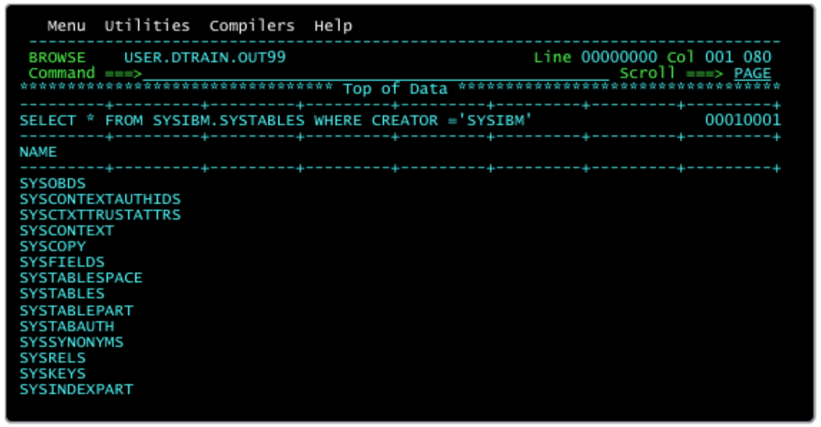
```
                           SPUFI                            SSID: DBBG
===>

Enter the input data set name:         (Can be sequential or partitioned)
 1  DATA SET NAME ... ===> 'USER.DTRAIN.CNTRL(SPUFI)'
 2  VOLUME SERIAL ... ===>             (Enter if not cataloged)
 3  DATA SET PASSWORD ===>             (Enter if password protected)

Enter the output data set name:        (Must be a sequential data set)
 4  DATA SET NAME ... ===> 'USER.DTRAIN.OUT99'

Specify processing options:
 5  CHANGE DEFAULTS   ===> YES         (Y/N - Display SPUFI defaults panel?)
 6  EDIT INPUT ...... ===> YES         (Y/N - Enter SQL statements?)
 7  EXECUTE ......... ===> YES         (Y/N - Execute SQL statements?)
 8  AUTOCOMMIT ...... ===> YES         (Y/N - Commit after successful run?)
 9  BROWSE OUTPUT ... ===> YES         (Y/N - Browse output data set?)

For remote SQL processing:
10  CONNECT LOCATION  ===>


PRESS:  ENTER to process    END to exit            HELP for more information
```

One useful tool within this ISPF option is SQL Processor Using File Input (SPUFI). This facility enables users to write and run SQL statements against a Db2 system.


```
   File  Edit  Edit_Settings  Menu  Utilities  Compilers  Test  Help
 -------------------------------------------------------------------------------
 EDIT       USER.DTRAIN.CNTRL(SPUFI) - 01.02                Columns 00001 00072
 Command ===>                                                  Scroll ===> PAGE
 ****** ***************************** Top of Data ******************************
 000100 SELECT * FROM SYSIBM.SYSTABLES WHERE CREATOR ='SYSIBM'
 ****** **************************** Bottom of Data ****************************
```

SPUFI accepts input in the form of SQL code stored in a sequential data set, or partitioned data set member. In the example shown here, the data set and member name would be entered in the SPUFI panel and the SQL code that appears within it would be invoked against the specified Db2 database.

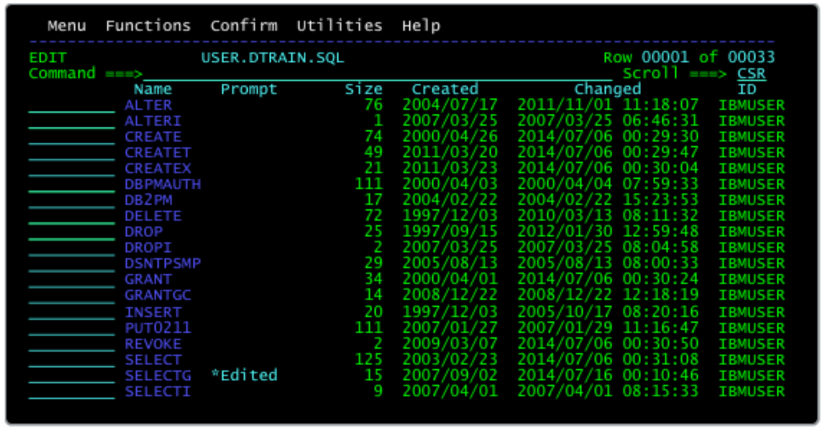
Storing code in this format allows you to create commonly used SQL statements and easily make modifications when required.



```
 BROWSE    USER.DTRAIN.OUT99                               Line 00000000 Col 001 080
 Command ===>                                                   Scroll ===> PAGE
********************************* Top of Data **********************************
---------+---------+---------+---------+---------+---------+---------+---------+
SELECT * FROM SYSIBM.SYSTABLES WHERE CREATOR ='SYSIBM'                  00010001
---------+---------+---------+---------+---------+---------+---------+---------+
NAME
---------+---------+---------+---------+---------+---------+---------+---------+
SYSOBDS
SYSCONTEXTAUTHIDS
SYSCTXTTRUSTATTRS
SYSCONTEXT
SYSCOPY
SYSFIELDS
SYSTABLESPACE
SYSTABLES
SYSTABLEPART
SYSTABAUTH
SYSSYNONYMS
SYSRELS
SYSKEYS
SYSINDEXPART
```

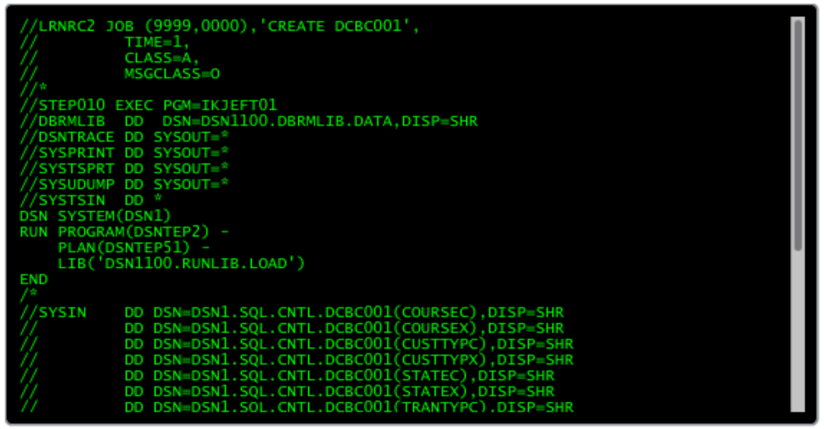
After processing the SQL statements, SPUFI then returns the results in a data set and displays it on the screen.

The SQL could have just as easily been an Update or Delete, or even a Create, Alter, or Drop of a database object.



```
  Menu  Functions  Confirm  Utilities  Help
 ------------------------------------------------------------------------------
EDIT              USER.DTRAIN.SQL                               Row 00001 of 00033
Command ===> ________________________________________________ Scroll ===> CSR
           Name     Prompt       Size  Created           Changed          ID
_________ ALTER                    76  2004/07/17  2011/11/01 11:18:07  IBMUSER
_________ ALTERI                    1  2007/03/25  2007/03/25 06:46:31  IBMUSER
_________ CREATE                   74  2000/04/26  2014/07/06 00:29:30  IBMUSER
_________ CREATET                  49  2011/03/20  2014/07/06 00:29:47  IBMUSER
_________ CREATEX                  21  2011/03/23  2014/07/06 00:30:04  IBMUSER
_________ DBPMAUTH                111  2000/04/03  2000/04/04 07:59:33  IBMUSER
_________ DB2PM                    17  2004/02/22  2004/02/22 15:23:53  IBMUSER
_________ DELETE                   72  1997/12/03  2010/03/13 08:11:32  IBMUSER
_________ DROP                     25  1997/09/15  2012/01/30 12:59:48  IBMUSER
_________ DROPI                     2  2007/03/25  2007/03/25 08:04:58  IBMUSER
_________ DSNTPSMP                 29  2005/08/13  2005/08/13 08:00:33  IBMUSER
_________ GRANT                    34  2000/04/01  2014/07/06 00:30:24  IBMUSER
_________ GRANTGC                  14  2008/12/22  2008/12/22 12:18:19  IBMUSER
_________ INSERT                   20  1997/12/03  2005/10/17 08:20:16  IBMUSER
_________ PUT0211                 111  2007/01/27  2007/01/29 11:16:47  IBMUSER
_________ REVOKE                    2  2009/03/07  2014/07/06 00:30:50  IBMUSER
_________ SELECT                  125  2003/02/23  2014/07/06 00:31:08  IBMUSER
_________ SELECTG  *Edited         15  2007/09/02  2014/07/16 00:10:46  IBMUSER
_________ SELECTI                   9  2007/04/01  2007/04/01 08:15:33  IBMUSER
```

In an enterprise environment it is good practice for developers and database administrators to maintain their own PDS members containing frequently used SQL statements. Once created, the SQL code in the member can be invoked through the interactive SPUFI screen just discussed, or can be referenced using a batch job executing the DSNTEP2 or DSNTEP4 programs.

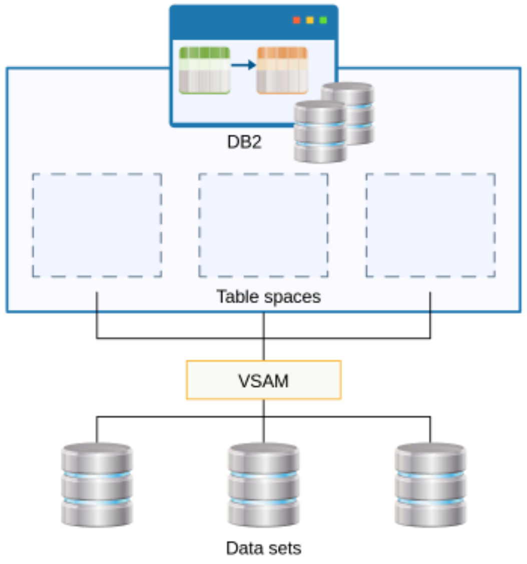


```
//LRNRC2 JOB (9999,0000),'CREATE DCBC001',
//         TIME=1,
//         CLASS=A,
//         MSGCLASS=O
//*
//STEP010 EXEC PGM=IKJEFT01
//DBRMLIB  DD  DSN=DSN1100.DBRMLIB.DATA,DISP=SHR
//DSNTRACE DD SYSOUT=*
//SYSPRINT DD SYSOUT=*
//SYSTSPRT DD SYSOUT=*
//SYSUDUMP DD SYSOUT=*
//SYSTSIN  DD *
DSN SYSTEM(DSN1)
RUN PROGRAM(DSNTEP2) -
    PLAN(DSNTEP51) -
    LIB('DSN1100.RUNLIB.LOAD')
END
/*
//SYSIN    DD DSN=DSN1.SQL.CNTL.DCBC001(COURSEC),DISP=SHR
//         DD DSN=DSN1.SQL.CNTL.DCBC001(COURSEX),DISP=SHR
//         DD DSN=DSN1.SQL.CNTL.DCBC001(CUSTTYPC),DISP=SHR
//         DD DSN=DSN1.SQL.CNTL.DCBC001(CUSTTYPX),DISP=SHR
//         DD DSN=DSN1.SQL.CNTL.DCBC001(STATEC),DISP=SHR
//         DD DSN=DSN1.SQL.CNTL.DCBC001(STATEX),DISP=SHR
//         DD DSN=DSN1.SQL.CNTL.DCBC001(TRANTYPC),DISP=SHR
```

This is an example of a Db2 batch job running the DDL to create the objects of a Db2 application system. The SYSTSIN DD statement provides the code used to invoke the DSN command processor for the DSN1 subsystem, then runs the DSNTEP2 program using the DSNTEP51 plan. The DD statements concatenated under the SYSIN DD identify the data set members, which in this case contain SQL code that will be executed in the order that they appear here. The job output can then be viewed.

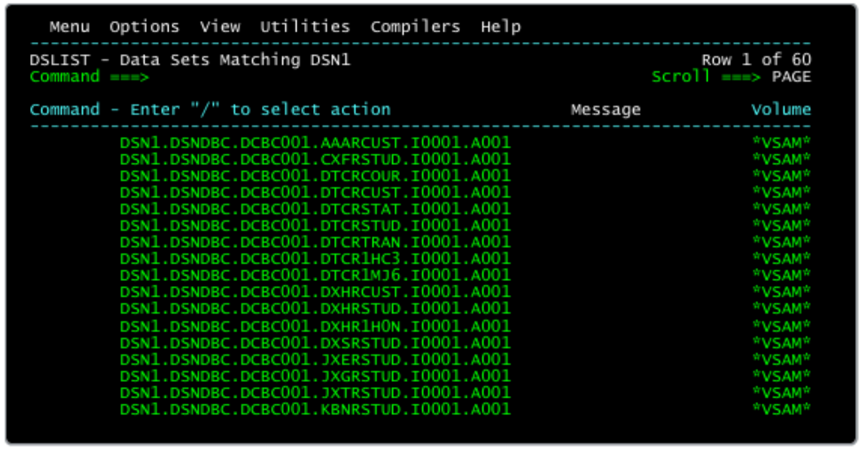
This method can be used for any SQL task or Db2 command, and is a much safer way of managing Db2.

Scroll to see more of the job.



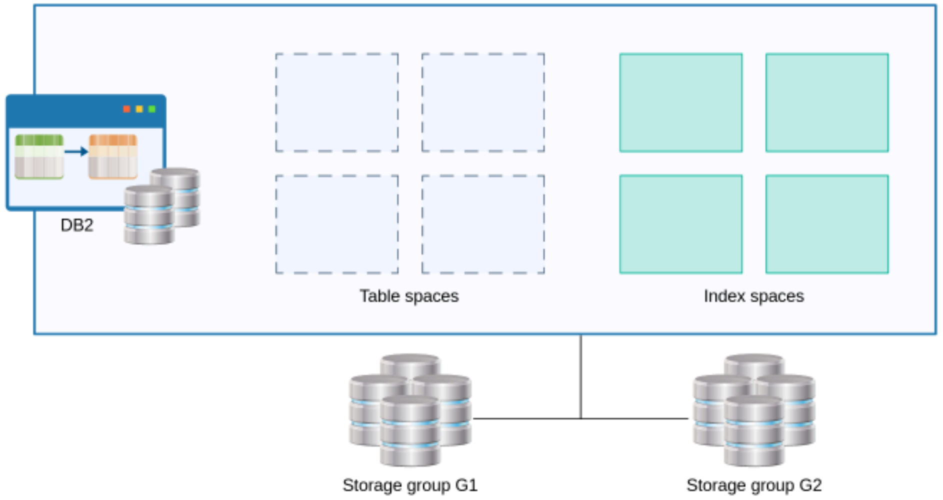
Apart from the interface, the other main difference between Db2 for z/OS and most other Db2 products is the way in which Db2 objects are physically stored.

As Db2 is running on z/OS, it uses Virtual Storage Access Method (VSAM) to store its table space, indexes, and other objects.



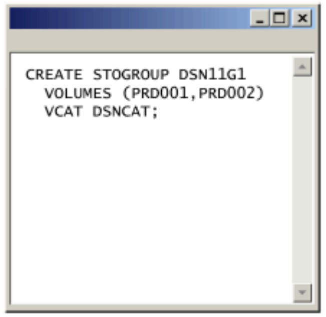
```
  Menu  Options  View  Utilities  Compilers  Help
------------------------------------------------------------------------------
DSLIST - Data Sets Matching DSN1                                  Row 1 of 60
Command ===>                                                  Scroll ===> PAGE

Command - Enter "/" to select action                  Message          Volume
------------------------------------------------------------------------------
         DSN1.DSNDBC.DCBC001.AAARCUST.I0001.A001                       *VSAM*
         DSN1.DSNDBC.DCBC001.CXFRSTUD.I0001.A001                       *VSAM*
         DSN1.DSNDBC.DCBC001.DTCRCOUR.I0001.A001                       *VSAM*
         DSN1.DSNDBC.DCBC001.DTCRCUST.I0001.A001                       *VSAM*
         DSN1.DSNDBC.DCBC001.DTCRSTAT.I0001.A001                       *VSAM*
         DSN1.DSNDBC.DCBC001.DTCRSTUD.I0001.A001                       *VSAM*
         DSN1.DSNDBC.DCBC001.DTCRTRAN.I0001.A001                       *VSAM*
         DSN1.DSNDBC.DCBC001.DTCR1HC3.I0001.A001                       *VSAM*
         DSN1.DSNDBC.DCBC001.DTCR1MJ6.I0001.A001                       *VSAM*
         DSN1.DSNDBC.DCBC001.DXHRCUST.I0001.A001                       *VSAM*
         DSN1.DSNDBC.DCBC001.DXHRSTUD.I0001.A001                       *VSAM*
         DSN1.DSNDBC.DCBC001.DXHR1H0N.I0001.A001                       *VSAM*
         DSN1.DSNDBC.DCBC001.DXSRSTUD.I0001.A001                       *VSAM*
         DSN1.DSNDBC.DCBC001.JXERSTUD.I0001.A001                       *VSAM*
         DSN1.DSNDBC.DCBC001.JXGRSTUD.I0001.A001                       *VSAM*
         DSN1.DSNDBC.DCBC001.JXTRSTUD.I0001.A001                       *VSAM*
         DSN1.DSNDBC.DCBC001.KBNRSTUD.I0001.A001                       *VSAM*
```

This is a list of the VSAM data sets that make up part of a Db2 database. They reside on volumes beside other data sets and must be managed similarly.

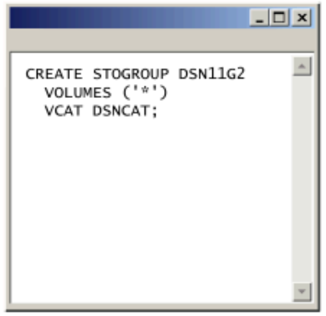


In a z/OS environment you can create Db2 storage groups, which are defined groups of volumes on disks that hold the data sets in which Db2 tables and indexes are stored. Once created you have several options in how the storage groups and Db2 data sets are managed. You can allow Db2 to manage the data sets, let the Storage Management Subsystem (SMS) manage some or all of them, or they can be managed manually using the VSAM Access Method Services.

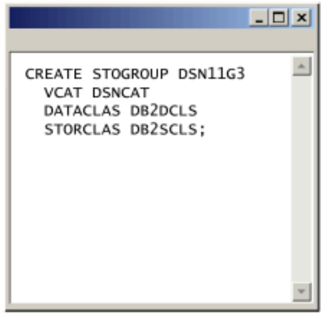
IBM recommends that Db2 storages groups are implemented and that you use SMS to manage them.



```
CREATE STOGROUP DSN11G1
  VOLUMES (PRD001,PRD002)
  VCAT DSNCAT;
```



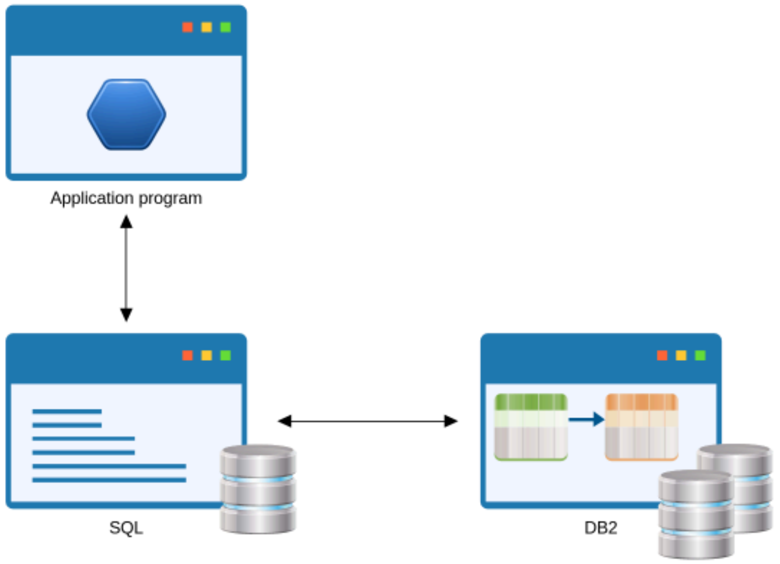
```
CREATE STOGROUP DSN11G2
  VOLUMES ('*')
  VCAT DSNCAT;
```



```
CREATE STOGROUP DSN11G3
  VCAT DSNCAT
  DATACLAS DB2DCLS
  STORCLAS DB2SCLS;
```

The storage group management type is actually defined when the administrator creates the storage group. In the example on the left, because volume serial names are specified it is informing the system that data sets placed on those volumes are to be Db2 managed. In the example in the middle, the asterisk indicates that it is to be managed by SMS. The example on the right does not contain a VOLUMES parameter, which defaults to being SMS managed. In addition, specific SMS Data class and Storage class attributes are assigned to the storage group.

These statements are discussed in more detail in later modules.



As discussed previously, SQL is used to access Db2 data and return the required data to the user, or program. There are a number of ways that you can invoke SQL code and in this section you will see how it is embedded in an application program.

Static SQL

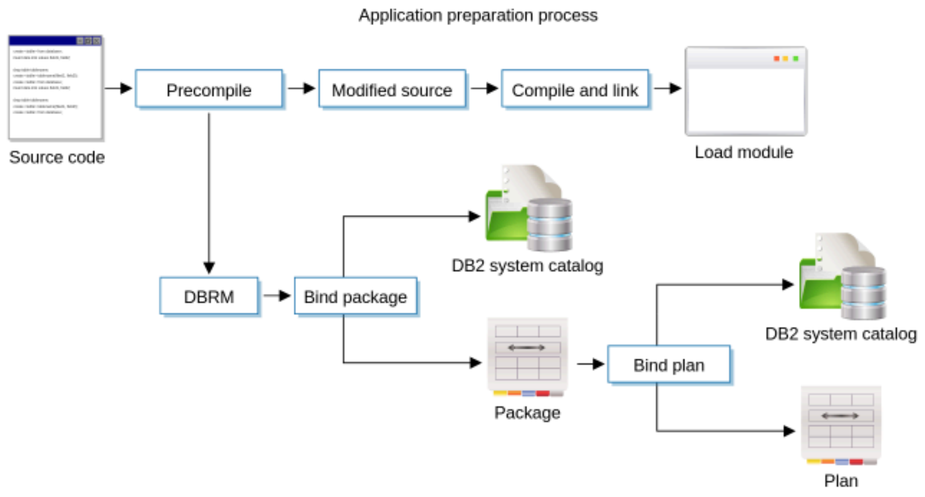
Static SQL are complete SQL statements that are written in the source code. DB2 works out access paths for the statements during the program preparation process and these are recorded in DB2. The SQL never changes from one run to another and the same predetermined access paths are used again without DB2 having to expend the overheads to work them out again.

Dynamic SQL

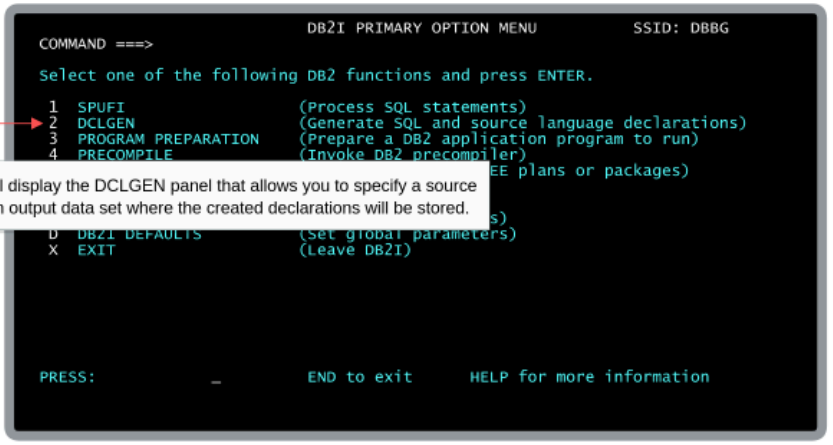
Dynamic SQL are SQL statements that are partially or totally unknown when the program is written. Only when the program runs can DB2 know what the statements are and then determine the appropriate access paths. These paths are not recorded in DB2 because they can change from one run to another, adding an overhead to execution.

Embedded SQL statements are written within application programming languages and preprocessed by a SQL preprocessor prior to the application program being compiled.

The two types of embedded SQL are static and dynamic.



The development cycle of an application program that contains SQL has a few extra steps. It is necessary to convert the SQL into something that the compiler computes, and process it so that Db2 can retrieve the data from the tables.



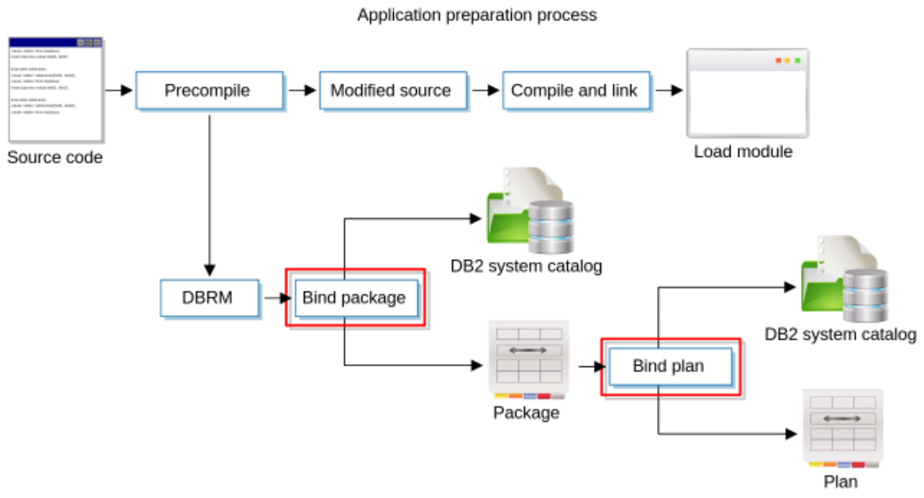
```
                              DB2I PRIMARY OPTION MENU          SSID: DBBG
 COMMAND ===>

 Select one of the following DB2 functions and press ENTER.

   1  SPUFI                    (Process SQL statements)
   2  DCLGEN                   (Generate SQL and source language declarations)
   3  PROGRAM PREPARATION      (Prepare a DB2 application program to run)
   4  PRECOMPILE               (Invoke DB2 precompiler)
                                                   EE plans or packages)

                                                   s)
   D  DB2I DEFAULTS            (Set global parameters)
   X  EXIT                     (Leave DB2I)




 PRESS:                       END to exit      HELP for more information
```

Option 2 will display the DCLGEN panel that allows you to specify a source table and an output data set where the created declarations will be stored.

Before your program's SQL statements can be processed you need to specify the tables and views that they are referencing. This can be undertaken manually by coding a DECLARE TABLE statement in your SQL code and specifying the details of each column and its data type.

Another commonly used method of performing this task is by using the declarations generator (option 2) shown here in the DB2I Primary Option Menu.

Click **Play** to see a demonstration of this concept.



For an application program, the bind is a separate step that must be performed before the program is ready for execution. Before a program can issue SQL statements, a Db2 plan must exist.

The result of a BIND PACKAGE is package that should be associated with a particular application plan. In general, plans are created using the BIND PLAN command. A package contains control structures that Db2 uses when it runs SQL statements. An application plan relates an application process to a local instance of Db2 and specifies processing options.

There is no need to bind the entire plan again when you change one SQL statement. You need to bind only the package that is associated with the changed SQL statement.

Summary

Db2 for z/OS Implementation

In this module you saw how Db2 has evolved and looked at some of the tools and utilities that are used to connect and interact with Db2 for z/OS. You discovered how storage for Db2 in a z/OS environment is configured and how SQL is prepared for application access to Db2 data.

You should now be able to:

- Identify the Methods Used for Connecting to Db2 for z/OS
- Describe How z/OS Storage Can be Configured for Use by Db2 for z/OS
- Explain How SQL is Prepared for Application Use