

I) Definitions and Terminology

Storage group	Named set of DASD volumes of same device type and also the VSAM catalog that stores the LDS datasets. SYSDEFLT is the default g Storage group.
Bufferpool	Virtual storage in which DB2 temporarily stores pages of TSs and indexes
Database	a) Set of DB2 structures that include a collection of tables, associated indexes and the TSs in which they reside. b) Logical collection of Table spaces and Index spaces c) Will contain all the data associated with one application or with a group of related application. d) DSNDB04 is the default database
Tablespace	Physical space that hold tables .TS can consist of a number of VSAM datasets.
Tables	Ordered set of Cols and rows
Attribute	In tables, a characteristic of an entity(column. For example, the phone number of an employee is one of that employee's attributes.
Index	Based on the values of data in one or more columns. For quick access and for unique values.
Constraint	A rule that limits the values that can be inserted, deleted, or updated in a table.
Atomic key	single column key
Composite key	Key with more than one column value combined
Composite Index	Index on composite key
View	Logical table of same base table data that is created for restricting viewing or operating purpose
auxiliary index	An index on an auxiliary table in which each index entry refers to an LOB.
Commit/Rollback	Commit is to make the changes permanent in the database. Rollback is to undo the uncommitted changes
Restart	Starting from the previous first uncommitted record
Attachment facility	Begins a DB2 session. Provides interface between DB2 and another environment
Entity integrity	Primary key must not be null
Referential Integrity	Foreign key must be either one of valid primary keys or null
Ambiguous Cursors	A cursor that cannot be determined to be updatable or read-only from its definition or context. If a cursor is defined with FOR FETCH ONLY clause or the FOR UPDATE OF clause
Cardinality	Number of distinct values in a column
Resultset	Cursors that are passed from Stored procedures
Bind	Process where access paths to the data are selected and some authorization checking is performed.
Timestamp	precompiled timestamp value that is created and matched between dbrmlib and loadlib during execution
UR	Uncommitted Read also called as dirty read let the application read the uncommitted data of another task

Cascade delete	The way in which DB2 UDB for OS/390 enforces referential constraints when it deletes all descendent rows of a deleted parent row.
Auxiliary table	Table that will have LOB data
Schema	Is a collection of named objects , Is like qualifier for objects like SP, Trigger, UDFs, UDTs
Static sql	SQLs that are embedded in a program. Other than the host variable that is used in those sqls, , the SQLs are static and won't change during the program execution.
Dynamic sql	SQLs stmts are unknown during program binding and will be prepared at the time of execution.
DB2 catalog	Where system tables reside like SYSIBM.*. They are stored in DSNDB06 database
DB2 directory	DSNDB01 database is the DB name. SQL can not access this directory.
Active Log	Is a LDS, single volume single extent. Used to log changes that are done in DB2
Archive log	When active is full, it will be copied into a tape dataset, called archive log.
BSDS	Boot Strap Data Set is a KSDS file. It is a inventory of RBA range specifications, log datasets, Recent chkpoint activity and Buffer pools.
IRLM	Internal Resource Lock Manager manages locks of DB2
Address spaces	Are started tasks that are running in JES.
DDF	Distributed Data Facility facilitates access to Remote databases
Parallel sysplex	Two or process processors to access same data under datasharing technique
Datasharing group	One or more subsystems to form a datasharing group under Parallel sysplex
Temporary table	Can be DECLARED and CREATED. Instance is Non-persistent after the application is done.
Authority	Can not be explicitly granted or revoked in other words these are implicit privileges
Privileges	Can be explicitly granted or revoked
Concurrency	The shared use of resources by multiple users or application processes at the same time.

II) SQLs – Sample DDLs, DCL, Bind Package, Plan

a) Storage Group

```
CREATE STOGROUP IEFSTOG1
  VOLUMES( DV1DB4
           , DV1DB2
           , DV1DB3
           , DV1DB1 )
  VCAT      IEF2;

-- VOLUMES - CHECK SEQUENCE
```

Parameters

Volumes	-	Should be of same type
VCAT	-	Catalog name where LDS of this storage group will be created

b) Bufferpool

Creation and alteration both done using **ATLER BUFFERPOOL**

c) Database

```
CREATE DATABASE LPRUDB01
  STOGROUP SYSDEFLT
  BUFFERPOOL BP1
  INDEXBP    BP0
  CCSID EBCDIC;
```

STOGROUP	-	Give the stogroup name, SYSDEFLT is the default
BUFFERPOOL	-	bp name for data
INDEXBP	-	bp name for index
CCSID	-	EBCIDIC/ASCII

d) Tablespace

```
CREATE TABLESPACE LPRTS002 IN LPRUDB01
  USING STOGROUP SYSDEFLT
        PRIQTY 720
        SECQTY 720
        ERASE NO
  FREEPAGE 0
  PCTFREE 5
  TRACKMOD YES
  BUFFERPOOL BP1
  LOCKSIZE PAGE
        LOCKMAX 0
  LOCKPART NO
  CLOSE NO
  COMPRESS NO
  CCSID EBCDIC;
```

e) Table

```

CREATE TABLE LPRADU1.EMP
(EMPNO                CHARACTER(6)          FOR SBCS DATA
                          NOT NULL
, FIRSTNME            VARCHAR(12)          FOR SBCS DATA
                          NOT NULL
, MIDINIT             CHARACTER(1)         FOR SBCS DATA
                          NOT NULL
, LASTNAME            VARCHAR(15)         FOR SBCS DATA
                          NOT NULL
, WORKDEPT            CHARACTER(3)         FOR SBCS DATA
                          NOT NULL
, PHONENO             CHARACTER(4)         FOR SBCS DATA
                          NOT NULL
, HIREDATE            DATE                 NOT NULL
, JOB                 CHARACTER(8)         FOR SBCS DATA
                          NOT NULL
, EDLEVEL             SMALLINT            NOT NULL
, SEX                 CHARACTER(1)         FOR SBCS DATA
                          NOT NULL
, BIRTHDATE           DATE                 NOT NULL
, SALARY              DECIMAL(9 , 2)      NOT NULL
, BONUS               DECIMAL(9 , 2)      NOT NULL
, COMM                DECIMAL(9 , 2)      NOT NULL
, SPL_INSTRUCTIONS    VARCHAR(50)         FOR SBCS DATA
WITH DEFAULT NULL
, PRIMARY KEY (EMPNO)
)
CCSID EBCDIC
IN LPRUDB01.LPRTS001
AUDIT ALL;

ALTER TABLE LPRADU1.EMP
FOREIGN KEY WORKDEPT (WORKDEPT)
REFERENCES LPRADU1.DEPT
ON DELETE RESTRICT ;

```

e) Index

```

.CONNECT DBDD

SET CURRENT SQLID = 'LPRADU1';

CREATE TYPE 2 UNIQUE INDEX LPRADU1.DEPT
ON LPRADU1.DEPT
(DEPTNO                ASC )
USING STOGROUP SYSDEFLT
      PRIORITY 12
      SECURITY 12
      ERASE NO
      FREEPAGE 0
      PCTFREE 10
      BUFFERPOOL BP2
      CLOSE YES
      PIECESIZE 2097152 K;

```

f) View

```

CREATE VIEW DBMZ001."ADD_MUTL_EXCL" ( MODEL_ID , TBNAME , ID , ORG_ID
, PARENT_ACTIV_ID ) AS SELECT MODEL_ID , TBNAME , ID , ORG_ID ,
FGN_KEY_1 FROM DBMZ001.PIDATA WHERE TBNAME = 'PDD_MUTL_EXCL' ;

COMMENT ON TABLE DBMZ001.ADD_MUTL_EXCL
IS 'ADD Mutual exclusive construct          ';

COMMENT ON DBMZ001.ADD_MUTL_EXCL
(MODEL_ID          IS 'Id of the containing model  '
, TBNAME          IS 'ADD_MUTL_EXCL          '
, ID              IS 'Unique identifier
.
, ORG_ID          IS 'Original identifier
.
, PARENT_ACTIV_ID IS 'Parent activity id. Join with FUNC
TION_DEF          or PROCESS_DEF.
);

```

g) Alias

```

CREATE ALIAS DSSADS1.ZIP_CODE
FOR DSSADU1.ZIP_CODE;

```

h) Synonym

```

SET CURRENT SQLID = 'LPRADU1';

CREATE SYNONYM LPRSYN01
FOR LPRADU1.EMP;

```

i) Stored Procedure

```

CREATE PROCEDURE LPRADU1.LPRSP002
( OUT IN010 INTEGER
)
RESULT SETS 1
EXTERNAL NAME 'LPRSP002'
LANGUAGE COBOL
PARAMETER STYLE DB2SQL
NOT DETERMINISTIC
FENCED
MODIFIES SQL DATA
NO DBINFO
COLLID LPROBJU1
WLM ENVIRONMENT DBDDENVU
ASUTIME NO LIMIT
STAY RESIDENT YES
PROGRAM TYPE MAIN
SECURITY DB2
COMMIT ON RETURN NO
;

```

j) Trigger

```

CREATE TRIGGER LPRTRG01
NO CASCADE BEFORE
BEFORE INSERT
ON LPRADU1.EMP
REFERENCING NEW AS NEW_VAR
FOR EACH ROW
MODE DB2SQL
WHEN (NEW_VAR . WORKDEPT NOT IN ( SELECT DEPTNO FROM
LPRADU1.DEPT) )
BEGIN ATOMIC SIGNAL SQLSTATE 'LPR00' (
'DEPERATMENT NOT VALID');
END

```

k) UDF

```

CREATE FUNCTION LPRADU1.LPRTSTUF
( CHARACTER(30)
, CHARACTER(30)
)
RETURNS CHARACTER(6)
SPECIFIC LPRTSTUF
EXTERNAL NAME 'LPRUF001'
LANGUAGE COBOL
PARAMETER STYLE DB2SQL
NOT DETERMINISTIC
NOT NULL CALL
FENCED
READS SQL DATA
NO EXTERNAL ACTION
SCRATCHPAD 100
FINAL CALL
ALLOW PARALLEL
NO DBINFO
WLM ENVIRONMENT DBDDENVU
ASUTIME NO LIMIT
STAY RESIDENT NO
PROGRAM TYPE MAIN
SECURITY DB2
;

```

l) UDT

```

CREATE DISTINCT TYPE MMDBSYS.DB2AUDIO
AS "SYSIBM".VARCHAR(250) FOR SBCS DATA CCSID EBCDIC
WITH COMPARISONS;

```

m) GRANT, REVOKE

n) BIND PACKAGE

```

DATA
  BIND PACKAGE(LPRBATU1) MEMBER(LPR150A)
  LIBRARY('UNT.CHAINA.BAT.DBRMLIB')
  OWNER(LPRADU1 ) QUALIFIER(LPRADU1 ) CURRENTDATA(YES)
  VALIDATE(BIND) EXPLAIN(NO ) ACTION(REPLACE)
  SQLERROR(NOPACKAGE) ISOLATION(CS)
  DEGREE(1 )
  NOREOPT(VARS)
  KEEPYNAMIC(NO )
  DBPROTOCOL(PRIVATE)
  FLAG(I);
ENDDATA
  
```

o) BIND PLAN

```

BIND PLAN(LPRBATU1) OWNER(DBMADM ) QUALIFIER(LPRADU1 )
PKLIST(LPRBATU1.*,
        DBMBATU0.*)
VALIDATE(BIND) EXPLAIN(NO )
ISOLATION(CS) ACQUIRE(USE ) RELEASE(COMMIT )
CURRENTDATA(NO ) CACHESIZE(0000)
DEGREE(1 ) SQLRULES(DB2) DISCONNECT(EXPLICIT)
NOREOPT(VARS)
KEEPYNAMIC(NO )
ACTION(REPLACE) RETAIN
DBPROTOCOL(PRIVATE)
FLAG(I);
  
```

p) SELECT – FETCH FIRST n ROWS ONLY

For example if you had specified SELECT * FROM MRKTHENI.EMP and it retrieves 1000 rows

EMP NO	EMP NAME	EMP ADDRESS1	EMP ADDRESS2	PHONE RES
00001	FIRST NAME	1ST STREET	#1	101010101
00002	SECOND NAME	2ND STREET	#2	201010101
00003	THIRD NAME	3RD STREET	#3	301010101
00004	FOURTH I	4TH STREET	#4	401010101
00005	FOURTH II	4TH STREET	#4	401010101
00006	FIFTH NAME	5TH STREET	#5	501010101
00007	SIXTH NAME	6TH STREET	#6	601010101
00008	SEVENTH I	7TH STREET	#7	701010101
00009	SEVENTH II	7TH STREET	#7	701010101
..				
..				
..				
00017	NINTH V	9TH STREET	#9	901010101
00018	NINTH V	9TH STREET	#9	901010101
00019	NINTH V	9TH STREET	#9	901010101
00020	NNNNN V	9TH STREET	#9	901010101
00021	NINTH V	9TH STREET	#9	901010101
00022	NINTH V	9TH STREET	#9	901010101
00023	NINTH V	9TH STREET	#9	901010101
00024	NINTH V	9TH STREET	#9	901010101
00025	NINTH V	9TH STREET	#9	901010101
..				
..				
..				
01000	NINTH V	9TH STREET	#9	901010101
01001	NINTH V	9TH STREET	#9	901010101

and you want only first 20 rows,Coding FETCH FIRST 20 ROWS ONLY would have retrieved only 20 rows but OPTIMIZE FOR 20 ROWS will fetch all of them, but will give priority to optimize the first 10 rows for efficient retrieval.

I) Auxiliary table

```
CREATE AUXILIARY TABLE LPRADU1.EMP_LOB_AUX_AUDIO
IN LPRUDB01.LPRTS130
STORES LPRADU1.EMP_LOB
COLUMN EMP_AUDIO ;
```

Though you don't give columns during creation it will create columns like below

EMP_LOB_AUX_AUDIO	LPRADU1	AUXID	VARCHAR	17
		AUXVER	SMALLINT	2
		AUXVALUE	BLOB	4

J) auxiliary index

```
CREATE TYPE 2 UNIQUE INDEX LPRADU1.EMPAUDIX
ON LPRADU1.EMP_LOB_AUX_AUDIO;
```

Specifying column names are not allowed while creating AUXILIARY INDEX

III) Interactive SQL

i) SPUFI (Sql Processing Using File Input)

- Reads the SQL statement from a text file, process those statement and put the results in an ISPF browse session.
- Use % to represent set of characters and _ for single characters.
- The input dataset must be preallocated with LRECL – 80.
- Can contain more than 1 sql statement in input dataset with ; delimiter
- -- Comments a sql stmt

```
--SELECT * FROM LPRADU1.EMP_SAMPLE;
SELECT * FROM LPRADU1.EMP;
SELECT * FROM LPRADU1.DEPT;
```

- Change the **DB2I** defaults like subsystem by selecting D in the below screen

```

                                DB2I PRIMARY OPTION MENU
COMMAND ==>> _                               SSID: DBDD

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)

```

This will put you in the below screen to change the defaults

```

                                DB2I DEFAULTS PANEL 1
COMMAND ==>> _

Change defaults as desired:

1  DB2 NAME ..... ==>> DBDD      (Subsystem identifier)
2  DB2 CONNECTION RETRIES ==>> 0   (How many retries for DB2 conne
3  APPLICATION LANGUAGE ==>> COBOL (ASM, C, CPP, COBOL, COB2, IBMC
   FORTTRAN, PLI)
4  LINES/PAGE OF LISTING ==>> 60   (A number from 5 to 999)
5  MESSAGE LEVEL ..... ==>> I     (Information, Warning, Error, S
6  SQL STRING DELIMITER ==>> DEFAULT (DEFAULT, ' or ")
7  DECIMAL POINT ..... ==>> .     (. or ,)
8  STOP IF RETURN CODE >= ==>> 8   (Lowest terminating return code
9  NUMBER OF ROWS ..... ==>> 20   (For ISPF Tables)
10 CHANGE HELP BOOK NAMES? ==>> NO (YES to change HELP data set na

```

- Changing **SPUFI** defaults regarding query related can be done inside SPUFI

```

=====
SPUFI
=====
==>

Enter the input data set name:          (Can be sequential or partitioned)
 1 DATA SET NAME ... ==> 'CSS8.V7F.DV.CNTL(LPRSEL)'
 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 ... ==> SQLOUT.OUT

Specify processing options:
 5 CHANGE DEFAULTS ... ==> Yes_    (Y/N - Display SPUFI defaults panel?)
 6 EDIT INPUT ..... ==> *        (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 informa
=====

```

in the below screen

```

=====
CURRENT SPUFI DEFAULTS
=====
==> _
SSID: DBDD

Enter the following to control your SPUFI session:
 1 SQL TERMINATOR .. ==> ;        (SQL Statement Terminator)
 2 ISOLATION LEVEL  ==> RR        (RR=Repeatable Read, CS=Cursor Stabil
 3 MAX SELECT LINES ==> 250      (Maximum number of lines to be
                                returned from a SELECT)

Output data set characteristics:
 4 RECORD LENGTH ... ==> 4092    (LRECL=Logical record length)
 5 BLOCK SIZE ..... ==> 4096    (Size of one block)
 6 RECORD FORMAT ... ==> VB      (RECFM=F, FB, FBA, V, VB, or VBA)
 7 DEVICE TYPE ..... ==> SYSDA   (Must be DASD unit name)

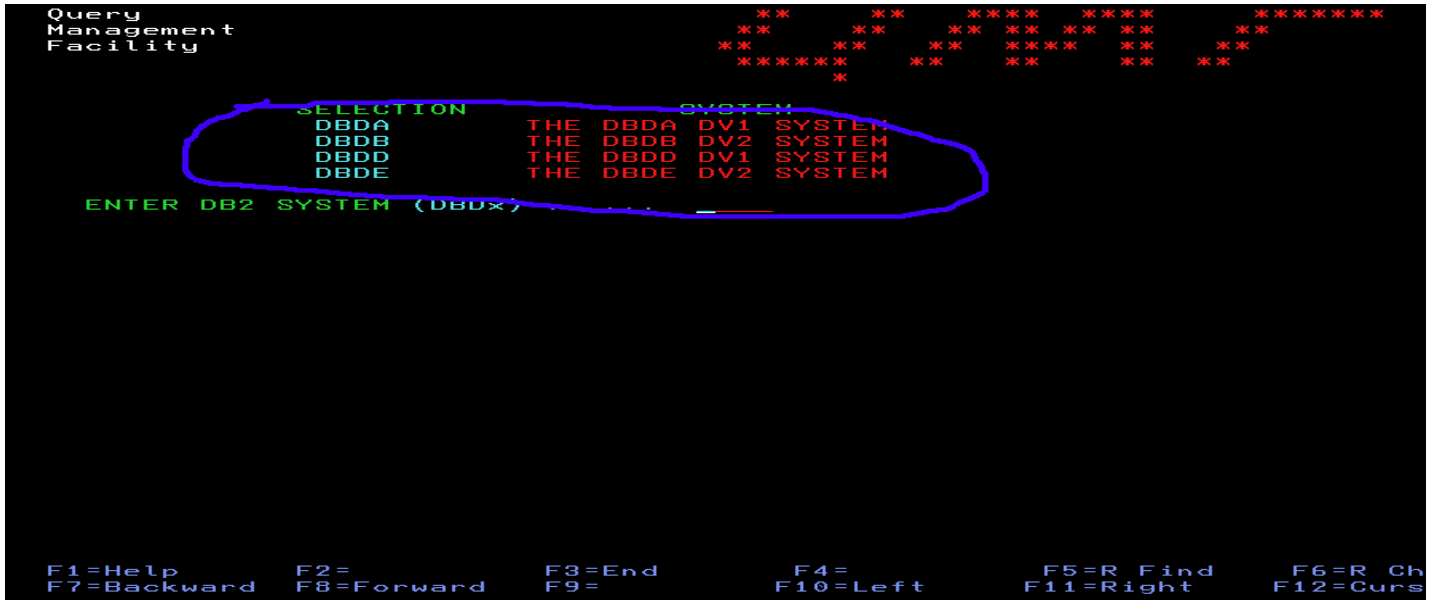
Output format characteristics:
 8 MAX NUMERIC FIELD ==> 33      (Maximum width for numeric fields)
 9 MAX CHAR FIELD .. ==> 80      (Maximum width for character fields)
10 COLUMN HEADING .. ==> NAMES   (NAMES, LABELS, ANY or BOTH)
=====

```

Change the MAX SELECT LINES in the above screen to a high value or * , if you expect to see more records than the limit in the SELECT query. Otherwise you will be deceived to think only a limited records are on the selected table

ii) QMF Query Management Facility

- This product from IBM uses its own set of DB2 tables
- This produces formatted query output. You can format the results of the SQL.
- You can place header and footer to the output.
- Only 1 SQL statement can be used in query panel
- If more than one sql statement need to be executed use QMF PROC (PF10)
- Initial screen will look like below, except the list of subsystems will vary across shops



- Type the subsystem name on the field and enter , will put in the below screen



Change the profile by pressing PF11

PROFILE

```

General Operands:
CASE      ==> UPPER      Enter UPPER, STRING, or MIXED.
DECIMAL   ==> PERIOD     Enter PERIOD, COMMA, or FRENCH.
CONFIRM   ==> YES        Enter YES or NO.
LANGUAGE  ==> SQL        Enter SQL, QBE, or PROMPTED.
MODEL     ==> REL        Enter REL.

Defaults for printing:
WIDTH     ==> 132        Number of characters per line.
LENGTH    ==> 60         Number of lines per page.
PRINTER   ==>           Printer to be used for output.

QMF Administration Operands: (Not usually changed)
SPACE     ==> "LPRUDB01"."LPRTS001"
           Enter the name of DB2 DATABASE or TABLESPACE in which
           tables will be saved by the SAVE DATA command.

TRACE     ==> NONE
           Enter ALL, NONE or a character string of function-id,
           trace-level pairs.

```

In the above screen you can change the DATABASE AND TABLESPACE NAME. You need to do this only once from next time onwards , it will use the same values

- Tables can be edited usign PF8 option

```

-----
EDIT TABLE Command Prompt                               1_ to 16 of 1
EDIT type TABLE
Name      ( _____ )
           Enter the name of the table in the database you
           want to edit.
Mode      ( CHANGE )
           Enter ADD to add new rows, or CHANGE to update
           or delete rows.
Save      ( END )
           Enter IMMEDIATE to save database alterations as
           they are made, or END to hold database alterations
           until the session is completed.
Confirm   ( YES )
           Enter NO to turn off confirmation prompting.
           Enter YES to accept default confirmation prompting.
-----
| F1=Help  F3=End  F7=Backward  F8=Forward
-----

```

Enter table name and ADD or Change on Mode like below and enter

```

+-----
EDIT TABLE Command Prompt                               1_ to 16
EDIT type TABLE
Name      ( LPRADU1.EMP SAMPLE )
           Enter the name of the table in the database you
           want to edit.
Mode      ( CHANGE )
           Enter ADD to add new rows, or CHANGE to update
           or delete rows.
Save      ( END )
           Enter IMMEDIATE to save database alterations as
           they are made, or END to hold database alterations
           until the session is completed.
Confirm   ( YES )
           Enter NO to turn off confirmation prompting.
           Enter YES to accept default confirmation prompting.
+-----
| F1=Help  F3=End  F7=Backward  F8=Forward
-----

```

Will take you to the below screen

```

SEARCH                                LPRADU1.EMP_SAMPLE                                1 to 10 of
EMP_NO. . . . . ( _____ )
EMP_NAME . . . . . ( _____ )
EMP_ADDRESS1 . . . . . ( _____ )
EMP_ADDRESS2 . . . . . ( _____ )
EMP_PHONE_RES . . . . . ( _____ )
EMP_SALARY . . . . . ( _____ )
EMP_DOB . . . . . ( _____ )
EMP DOJ . . . . . ( _____ )
EMP DOU . . . . . ( _____ )
SPL_INSTRUCTIONS . . . ( _____ )

1=Help      2=Search      3=End      4=Show Change      5=Show Field      6=Previous
    
```

Enter values on the fields and PF2 to search the table and display the record, change and

- SQL query can be entered on PF6

```

SQL QUERY                                LINE 1
-
*** END ***

1=Help      2=Run          3=End          4=Print        5=Chart        6=Draw
7=Backward  8=Forward       9=Form        10=Insert      11=Delete      12=Report
OK, QUERY is displayed.                SCROLL ==>
    
```

```

SQL QUERY                                MODIFIED LINE
SELECT * FROM LPRADU1.EMP_SAMPLE
    
```

Press PF2 to execute the sql

EMP NO	EMP NAME	EMP ADDRESS1	EMP ADDRESS2	EMP PHONE RES
00001	FIRST NAME	1ST STREET	#1	10101
00002	SECOND NAME	2ND STREET	#2	20101
00003	THIRD NAME	3RD STREET	#3	30101
00004	FOURTH I	4TH STREET	#4	40101
00005	FOURTH II	4TH STREET	#4	40101
00006	FIFTH NAME	5TH STREET	#5	50101
00007	SIXTH NAME	6TH STREET	#6	60101
00008	SEVENTH I	7TH STREET	#7	70101
00009	SEVENTH II	7TH STREET	#7	70101
00010	SEVENTH III	7TH STREET	#7	70101
00011	EIGHTH NAME	8TH STREET	#8	80101
00012	NINTH I	9TH STREET	#9	90101

- Following commands can be used in COMMAND line

SAVE QUERY AS SAMPLE	-	to save the query entered
RESET QUERY	-	to clear the QMF screen
DISPLAY SAMPLE	-	to display the saved query
SAVE DATA AS <i>table_name</i>	-	to save the displayed data in a new table
LIST QUERIES	-	to list the saved queries
LIST TABLES	-	to list tables created on your PRIMARY AUTHID
LIST ?	-	prompts to give the object type you want to list

like below

```

1=Help      2=List      3=End
7=Retrieve  8=Edit Table 9=Form
OK, you may enter a command.
COMMAND ==> list tables_
    
```

which will put you in

```

-----
Table List
-----
Action      Name                Owner
-----
LOCATION     WELZ492
ON PROJECT WELZ492
PLAN TABLE WELZ492
PROJECT    WELZ492
SAMPLE     WELZ492
XCHANGE TEST WELZ492
-----
F1=Help  F4=Command  F5=Describe  F6=Refresh  F7=Backward
F9=Clear F10=Comments F11=Sort     F12=Cancel
    
```

LIST ? on the command line will put

```

-----
LIST Command Prompt
-----
Type      (    ) 1_ to 17
Enter the name of the object type you want to list. It
can be QUERIES, PROCS, FORMS, QMF, TABLES, or ALL.

Owner     ( WELZ492 )
Enter the owner id of the object named. The owner id can
contain selection symbols "%" and "_" to specify like
owners. To list all owners, enter ALL.

Name      ( ALL )
Enter the specific object name. The name can contain
selection symbols "%" and "_" to specify like names.
To list all objects for the specified owner, enter ALL.

Location  (    )
To get a list of tables from a remote database, enter the
specific location name.
-----
F1=Help  F3=End  F7=Backward  F8=Forward
    
```

where you can enter the type of objects you want to list.

IV) Embedded SQL

1) Program Method

- programs will have SQLs with Delimiters like

```
EXEC SQL
      sqlquery
END-EXEC
```

- **Host variables**

- are used during INSERT/UPDATE/DELETE/SELECT into/from a table column
- Data type of host-variables must match with the column datatype

- **Indicator variables**

- are required if selected/inserted column allow null
- If the column is null indicator variable is set to negative value
- SQLCODE -305 will be returned when you failed to use indicator variable and the result is NULL.
- Define indicator variables as **S9(4) COMP**

```
EXEC SQL
      SELECT EMP_NAME, EMP_DOJ
      FROM EMP_SAMPLE
      INTO :EMP-NAME, :EMP-DOJ :INDEMP-DOJ
      WHERE EMP_NO= :EMP-NO
END-EXEC
```

- **CURSORS**

- When the sql return more than one row
- Cursor names should be unique through out the program
- Cursors will be closed with COMMIT, use WITH HOLD to hold a cursor on COMMIT
- Steps to followed in using a cursor
 - DECLARE - Can be before procedure division
 - OPEN - Executes the query during OPEN
 - FETCH - Retrieves a row and populates the respective host variables
 - CLOSE - Closes the cursor

2) SQLCA

- SQL programs must use SQLCA, sql communication area.
- Structure of SQLCA follows the language type that is used to write the program like COBOL version, PL/1 version.
- Used like,


```
EXEC SQL      INCLUDE SQLCA
END-EXEC
```
- Provides information describing the status of last sql statement executed
- Communications between DB2 DBMS and application PGMs
- Generally checked by program after each SQL request
- SQLCODE which is a variable in SQLCA is checked after the SQL stmt to know about the outcome of the SQL.

3) DCLGEN

- Cobol copy structure of the tables

```
//STEP01 EXEC PGM=IKJEFT01,DYNAMNBR=20
//SYSTSPRT DD SYSOUT=*
//SYSPRINT DD SYSOUT=*
//SYSUDUMP DD SYSOUT=*
//SYSTSIN DD *
DSN SYSTEM(DSN)
DCLGEN TABLE (qualifier.table_name)
LIBRARY( 'Z8MMK0V.DB2.DCLGEN(EMPDCL)
LANGUAGE(COBOL)
NAMES(MUR)
STRUCTURE(EMPLOYEE_STRUCTURE)
ACTION(REPLACE)
QUOTE
/*
```

- subsystem name where the table resides
- Table name to what DCLGEN is created
- PDS(E) where DCLGEN should be stored
- language on which the DCLGEN is created
- table column variables name's prefix
- 01 group name

4) Program Preparation – DB2 and Non CICS pgm

No	Process	Executed Pgr	Input	DDNAME	Output	DDNAME	Description
1	Precompile	DSNHPC	Cobol pgm DCLGEN libs DSNHPC lib	SYSIN SYSLIB STEPLIB	DBRMLIB Modified source	DBRMLIB SYSCIN	Include members will be expanded Checks the SQL syntax SQL statements are converted to COBOL CALL stmts to give modified source The SQL statements will be extracted and written in a DBRM member Places a time stamp token in the DBRM member and load module (consistency token)
2	Compile	IGYRCTL	Modified source Copylib libs Cobol compile lib	SYSIN SYSLIB STEPLIB	Obj module	SYSLIN	Expands COPYBOOKs COBOL syntax will be done Places the time stamp written in the precompile step in the OBJ module
3	Link	IEWL	Obj module Subpgm load lib Link module lib Attaches like DSNHLI/DBAHLI	SYSLIN SYSLIB SYSLIB SYSIN	Loadmodule	SYSLMOD	Executable load module is produced Sub programs if exists will be merged along with the language modules Timestamp will be written in the load module
4	Bind	IKJEFT01 TMP	DBRMLIB		Bind package		Reads the SQL statements from DBRMs and produces a mechanism to access data from tables Package versioning can be used to Bind two versions of DBRMLIBs of same Package into one collection

5) Sample compile JCL for DB2 pgm-

```
//PC          EXEC PGM=DSNHPC, REGION=2M, PARM='HOST(COB2),APOST,SOURCE,XREF'
//STEPLIB    DD DSN=DSND01.SDSNLOAD,DISP=SHR           → DSNHPC is picked from here
//DBRMLIB    DD DSN=Z8MMK0V.CLASS.DBRMLIB(EMPLOYEE),DISP=SHR
//SYSCIN     DD DSN=&&DSNHOUT,DISP=(NEW,PASS),UNIT=SYSDA,SPACE=(CYL,(1,1))
//SYSIN      DD DSN=Z8MMK0V.CLASS.COBOL(EMPLOYEE),DISP=SHR
//SYSLIB     DD DSN=Z8MMK0V.CLASS.COPYLIB,DISP=SHR
//SYSPRINT   DD SYSOUT=*
//SYSTEM     DD SYSOUT=*
//SYSUDUMP   DD SYSOUT=*
//COB EXEC PGM=IGYCRCTL,COND=(4,LT,PC),REGION=6M,PARM=(LIST,NOOFF,X,S,NOTERM,MAP,NOWD,
//          NOSEQ,LIB,APOST,FLAG(I,I),FASTSRT,TEST(NONE,SYM),DYNAM,NOOPT)
//STEPLIB    DD DSN=IGY.SIGYCOMP,DISP=SHR             → compile language library
//SYSLIB     DD DSN=IRBNSOEN.PROD.COPYCOB,DISP=SHR
//SYSIN      DD DSN=&&DSNHOUT,DISP=(,DELETE)
//SYSLIN     DD DSN=&&OBJMOD,DISP=(NEW,PASS)
//SYSPRINT   DD SYSOUT=*
//SYSUDUMP   DD SYSOUT=*
//LKED EXEC PGM=IEWL,PARM="",COND=(4,LT,COB)
//SYSLIB     DD DSN=IRBNSOPN.OFFSHORE.CPE.LOCL.LOADLIB,DISP=SHR
//          DD DSN=DSND001.SDSNLOAD,DISP=SHR
//          DD DSN=CEE.SCEELKED,DISP=SHR
//          DD DSN=IGY.SIGYCOMP,DISP=SHR
//SYSLIN     DD DSN=&&OBJMOD,DISP=(,DELETE)
//SYSLMOD    DD DSN=Z8MMK0V.CLASS.LOADLIB(EMPLOYEE),DISP=SHR
//SYSPRINT   DD SYSOUT=*
//SYSUDUMP   DD SYSOUT=*
//SYSUT1     DD UNIT=SYSDA,SPACE=(1024,(50,50))
//SYSDBOUT   DD SYSOUT=*
//SYSIN      DD *
          ENTRY EMPLOYEE
          NAME EMPLOYEE(R)
/*
//BIND EXEC PGM=IKJEFT01,DYNAMNBR=20,COND=(4,LT)
//STEPLIB    DD DSN=DSND001.SDSNLOAD,DISP=SHR
//          DD DSN=DSND001.RUNLIB.LOAD,DISP=SHR
//SYSUDUMP   DD DUMMY
//SYSOUT     DD SYSOUT=*
//SYSPRINT   DD SYSOUT=*
//SYSTSIN    DD *
          DSN SYSTEM(DSN)           → subsystem where the bind occurs
          BIND PACKAGE(MRKBATU1) MEMBER(MRK001) → collection name in package, and program name in member
          QUALIFIER(Z8MMK0V)       → qualifier for the tables used in the program
          LIBRARY('HLQ.DB2.DBRMLIB') → library where the DBRMLIB is stored
          OWNER(plan_creator)      → Plan owner
          ACT(REPLACE) -           → add or replace the package in the collection
          ISOLATION(RR) -         → whether the page lock is released on commit or usage end
          ACQUIRE(USE/ALLOCATE) - → TS locks while allocating the plan or the first usage
          RELEASE(COMMIT/DEALLOCATE) → TS locks to be released at COMMIT or DEALLOCATE of the plan
          VALIDATE(BIND/RUN) -     → Validate authorization at run time or bind time
          EXPLAIN(NO)             → whether PLAN_TABLE is used to explain or not
          FLAG                     → for messages
          END
/*
```

6) Program Preparation – DB2 and CICS pgm

Will have additional step between PRECOMPILER and COMPILER called TRANSLATOR

No	Process	Executed Pgm	Input	Output	Description
2a	Translator	DFHECP1\$	Modified source -Sysin Map library - SYSLIB CICS library - steplib	Translated Modified source	CICS stmts EXEC CICS are converted to COBOL calls to give translated souce

v) Executing a DB2 program in BATCH

- -818 if the time stamp differs between load and plan, if all the DBRMLIBs are directly bound into the PLAN(old way, not recommended)
- -805 if the time stamp differs between load module and package, if the DBRMLIBs are bound into packages in a collection and then into a plan(efficient)
- **Sample run JCL using TSO terminal program, IKJEFT01.**

```
//STEP1      EXEC PGM=IKJEFT01
//.....
//.....
//CEEDUMP DD  SYSOUT=*
//SYSABOUT DD  SYSOUT=*
//SYSPRINT DD  SYSOUT=*
//SYSTSPRT DD  SYSOUT=*
//SYSDBOUT DD  SYSOUT=*
//SYSTSIN   DD  *
    DSN SYSTEM(subsystem name)           - DB2 subsystem to which the pgm communciate
    PROGRAM(program name)                 - program name
    PLAN(plan name)                       - plan name
    LIB('load library name')              - load library name
    PARS('parameter list to be passed to the program) - parms to the pgm
    END
/*
```

7) DB2 FLOW

