TOPICS COVERED

- DATA MANIPULATION
- ALIAS
- COMPLEX QUERIES
  - sub queries
  - joins
  - unions
- DCL
- TCL
- SYNONYMS and ALIASES

ALIAS - Column Aliases

- Heading of a column can be changed while displaying data in a select statement, in order to make the display more meaningful.
- A column Alias gives a column an alternative name for that attribute in the o/p.
- The Alias has to be specified after the column in the select list.
- The scope of the alias name is only with in that SQL statement.

ALIAS - Column Aliases

The Column Alias name can be given in two ways

1. The new ALIAS name can be written along with column name separated by a single space

Column name

ALIAS NAME

SELECT **EMPNO as "EMPLOYEE NUMBER"**,
SALARY FROM EMP;

2. The new ALIAS name can be written along with column name separated by a word “AS”.

Column name

ALIAS NAME

SELECT **EMPNO AS EMPLOYEENUMBER**,

SALARY FROM EMP;

ALIAS - Column Aliases

- The Alias headings are also converted to uppercase in the o/p list, and unless it is enclosed in double quotes cannot contain spaces.

```
SELECT EMPNO EMPLOYEENUMBER,
SALARY FROM EMP;
```

- If the aliases contain embedded spaces it should be enclosed within double quotation marks

```
SELECT EMPNO “EMPLOYEE NUMBER”,
SALARY FROM EMP;
```

ALIAS - Column Aliases

```
--*****************************************************************  00295398
--  SELECT USING SCALAR FUNCTIONS GIVING ALIAS NAME                  00295499
--*****************************************************************  00295598
  SELECT SUBSTR(FIRST_NAME,1,3) AS NICKNAME FROM EMPLOYEE;           00295699
---------+---------+---------+---------+---------+---------+---------+---------+
NICKNAME
---------+---------+---------+
ARU
VAR
TIM
SAL
PER
DSNE610I NUMBER OF ROWS DISPLAYED IS 5
DSNE616I STATEMENT EXECUTION WAS SUCCESSFUL, SQLCODE IS 100
---------+---------+---------+---------+---------+---------+---------+---------+
```

ALIAS NAME

THE CONCATENATION OPERATOR

- || allows columns to be linked to other columns,constant values or arithmetic expressions to result in a single character column.

Example:

SELECT FNAME||LNAME “EMPLOYEE NAME” FROM EMP;

O/P

EMPLOYEE NAME

LINDAGOODMAN

SIDNEYSHELDON

THE CONCATENATION OPERATOR

```
--*********************************************************************  00295398
--  SELECT USING CONCATENATION,SCALAR FUNCT AND ALIAS NAME              00295499
--*********************************************************************  00295598
  SELECT SUBSTR(FIRST_NAME,1,3)||LAST_NAME NICKNAME FROM EMPLOYEE;       00295699
---------+---------+---------+---------+---------+---------+---------+---------+
NICKNAME
---------+---------+---------+---------+---------+---------+---------+---------+
ARUKUMAR
VARCHAND
TIMTOMMY
SALSAMMY
PERPRECY
DSNE610I NUMBER OF ROWS DISPLAYED IS 5
DSNE616I STATEMENT EXECUTION WAS SUCCESSFUL, SQLCODE IS 100
---------+---------+---------+---------+---------+---------+---------+---------+
```

THE CONCATENATION OPERATOR

```
--*****************************************************************      00295398
--  SELECT USING CONCATENATION AND ALIAS NAME                            00295499
--*****************************************************************      00295598
  SELECT FIRST_NAME||LAST_NAME NICKNAME FROM EMPLOYEE;                   00295699
---------+---------+---------+---------+---------+---------+---------+---------+
NICKNAME
---------+---------+---------+---------+---------+---------+---------+---------+
ARUN      KUMAR
VARUN     CHAND
TIMY      TOMMY
SALY      SAMMY
PERC      PRECY
DSNE610I NUMBER OF ROWS DISPLAYED IS 5
DSNE616I STATEMENT EXECUTION WAS SUCCESSFUL, SQLCODE IS 100
---------+---------+---------+---------+---------+---------+---------+---------+
```

COMPLEX SQL'S

- One terms a SQL to be complex when data that is to be retrieved comes from more than one table.

Important Note:

1. A table can be referred and considered as more than one table by giving it more than one alias name.
2. The tables can be different.

Complex SQL's

- SQL provides THREE ways of coding a complex SQL

1. **Sub queries** (15)
2. **Joins** - non related tables
3. **unions**

SUBQUERIES

- Nested Select statements(16 levels max)
- A query in side a query.
- Specified using the IN (or NOT IN) predicate, equality or non-equality predicate(‘=‘ or ‘<>‘) and comparative operator(<, <=, >, >=)
- When using the equality, non-equality or comparative operators, the inner query should return only a single value
- The nested loop statements gives the user the flexibility for querying multiple tables

SUBQUERIES

SUBQUERIES

correlated

- the nested select statement refers back to the columns in previous select statements
- TOP-BOTTOM – UP approach

Non- correlated

- the nested queries in which the inner query completes first and based on that the outer query functions.
- BOTTOM - UP approach

SUB QUERIES
Non correlated

- Locate the Project information of Employee whose first name is ‘TIMY’

```
SELECT PROJNO , PROJNAME
 FROM DSN8710.PROJ
 WHERE PROJID =
                    ( SELECT PROJID
                      FROM DSN8710.EMPLOYEE
                     WHERE FIRST_NAME =
                     ‘TIMY’);
```

Content of the tables EMPLOYEE

```
 SELECT * FROM EMPLOYEE;                                                  00292165
---------+---------+---------+---------+---------+---------+---------+---------+
EMPNO    FIRST_NAME LAST_NAME  DEPTID   PHONENO  HIRTHDATE     SALARY  PROJID  PA
---------+---------+---------+---------+---------+---------+---------+---------+
1001     ARUN       KUMAR      D01      453459   2011-01-15   400000.  PJ001   12
1002     VARUN      CHAND      D01      456769   2008-07-05   200000.  PJ002   23
1004     TIMY       TOMMY      D03      109289   2005-01-29   800000.  PJ004   98
1005     SALY       SAMMY      D04      878769   2009-06-29   560000.  PJ003   23
1003     PERC       PRECY      D02      456512   2005-01-29   800000.  PJ003   98
DSNE610I NUMBER OF ROWS DISPLAYED IS 5
DSNE616I STATEMENT EXECUTION WAS SUCCESSFUL, SQLCODE IS 100
---------+---------+---------+---------+---------+---------+---------+---------+
```

Content of the tables PROJ

```
 SELECT * FROM PROJ;                                                    00291968
---------+---------+---------+---------+---------+---------+---------+---------+
PROJID  PROJNAME    PROJLEAD    PROJLOC
---------+---------+---------+---------+---------+---------+---------+---------+
PJ001   ICICI       NEWTON      CHENNAI
PJ002   HDFC        SACHIN      CHENNAI
PJ003   ISRO        SELTON      DELHI
PJ004   BHEL        FREDDI      PUNE
DSNE610I NUMBER OF ROWS DISPLAYED IS 4
DSNE616I STATEMENT EXECUTION WAS SUCCESSFUL, SQLCODE IS 100
---------+---------+---------+---------+---------+---------+---------+---------+
```

Content of the tables DEPT

```
  SELECT * FROM DEPT;                                                    00292069
---------+---------+---------+---------+---------+---------+---------+---------+
DEPTID  DEPTNAME    NO_OF_EMPLOYS  HOD
---------+---------+---------+---------+---------+---------+---------+---------+
D01     TRAINEE              20.  SAM
D02     DEVE                 30.  TOM
D03     MRKT                 10.  TIM
D04     HR                    5.  TAM
DSNE610I NUMBER OF ROWS DISPLAYED IS 4
DSNE616I STATEMENT EXECUTION WAS SUCCESSFUL, SQLCODE IS 100
---------+---------+---------+---------+---------+---------+---------+---------+
```

Ex: Non- correlated Sub query.

Selecting from proj table based on a value in employee table

```
--*****************************************************************
--  SELECT SUB QUERIES - NON-CORREALTED
--*****************************************************************
  SELECT PROJNAME,PROJLOC FROM PROJ WHERE PROJID =
      (SELECT PROJID FROM EMPLOYEE WHERE FIRST_NAME = 'TIMY');
---------+---------+---------+---------+---------+---------+---------+---------+
PROJNAME    PROJLOC
---------+---------+---------+---------+---------+---------+---------+---------+
BHEL        PUNE
DSNE610I NUMBER OF ROWS DISPLAYED IS 1
DSNE616I STATEMENT EXECUTION WAS SUCCESSFUL, SQLCODE IS 100
---------+---------+---------+---------+---------+---------+---------+---------+
```

INNER query

Non- correlated Sub query using EXISTS

```
--*****************************************************************      00297099
--  SELECT SUB QUERIES - NON-CORREALTED - EXISTS                          00298099
--*****************************************************************      00299099
   SELECT EMPNO,FIRST_NAME FROM EMPLOYEE WHERE EXISTS                     00300099
      (SELECT * FROM PROJ WHERE PROJLOC = 'CHENNAI');                     00310099
---------+---------+---------+---------+---------+---------+---------+---------+
EMPNO   FIRST_NAME
---------+---------+---------+---------+---------+---------+---------+---------+
1001    ARUN
1002    VARUN
1004    TIMY
1005    SALY
1003    PERC
DSNE610I NUMBER OF ROWS DISPLAYED IS 5
DSNE616I STATEMENT EXECUTION WAS SUCCESSFUL, SQLCODE IS 100
---------+---------+---------+---------+---------+---------+---------+---------+
```

Non- correlated Sub query using EXISTS

```
--*****************************************************************
--  SELECT SUB QUERIES - NON-CORREALTED - EXISTS
--*****************************************************************
   SELECT EMPNO,FIRST_NAME FROM EMPLOYEE WHERE EXISTS
      (SELECT * FROM PROJ WHERE PROJLOC = 'BHOPAL');
---------+---------+---------+---------+---------+---------+---------+---------+
EMPNO   FIRST_NAME
---------+---------+---------+---------+---------+---------+---------+---------+
DSNE610I NUMBER OF ROWS DISPLAYED IS 0
DSNE616I STATEMENT EXECUTION WAS SUCCESSFUL, SQLCODE IS 100
---------+---------+---------+---------+---------+---------+---------+---------+
```

Non- correlated Sub query –
Referring to the same table in both the queries

```
--*****************************************************************      00311099
--  SELECT SUB QUERIES - NON-CORREALTED 2ND MAXIMUM                      00312099
--*****************************************************************      00313099
  SELECT MAX(SALARY) FROM EMPLOYEE WHERE                                 00320099
  SALARY<(SELECT MAX(SALARY)FROM EMPLOYEE);                              00330099
---------+---------+---------+---------+---------+---------+---------+---------+


---------+---------+---------+---------+---------+---------+---------+---------+
 560000.
DSNE610I NUMBER OF ROWS DISPLAYED IS 1
DSNE616I STATEMENT EXECUTION WAS SUCCESSFUL, SQLCODE IS 100
---------+---------+---------+---------+---------+---------+---------+---------+
```

EMPLOYEE table is referred in both the queries

CORRELATED SUB QUERIES

- A specialized form is Correlated Sub query - the nested Select statement refers back to the columns in previous select statements
- It works on Top-Bottom-Top fashion
- Ex: to find the Nth maximum salary from a table.

CORRELATED SUB QUERIES

```
--*****************************************************************      00295398
--  SELECT SUB QUERIES - CORREALTED                                      00295499
--*****************************************************************      00295598
  SELECT EMPNO,FIRST_NAME FROM EMPLOYEE A WHERE 4 =                      00295699
  (SELECT COUNT (SALARY) FROM EMPLOYEE B WHERE                           00295799
     (A.SALARY <= B.SALARY));                                            00295899
---------+---------+---------+---------+---------+---------+---------+---------+
EMPNO   FIRST_NAME
---------+---------+---------+---------+---------+---------+---------+---------+
1001    ARUN
DSNE610I NUMBER OF ROWS DISPLAYED IS 1
DSNE616I STATEMENT EXECUTION WAS SUCCESSFUL, SQLCODE IS 100
---------+---------+---------+---------+---------+---------+---------+---------+
```

CORRELATED SUB QUERIES
- using DISTINCT

```
--*****************************************************************
--  SELECT SUB QUERIES - CORREALTED
--*****************************************************************
  SELECT EMPNO,FIRST_NAME FROM EMPLOYEE A WHERE 4 =
  (SELECT COUNT (DISTINCT(SALARY)) FROM EMPLOYEE B WHERE
     (A.SALARY <= B.SALARY));
---------+---------+---------+---------+---------+---------+---------+---------+
EMPNO   FIRST_NAME
---------+---------+---------+---------+---------+---------+---------+---------+
1002    VARUN
DSNE610I NUMBER OF ROWS DISPLAYED IS 1
DSNE616I STATEMENT EXECUTION WAS SUCCESSFUL, SQLCODE IS 100
---------+---------+---------+---------+---------+---------+---------+---------+
```

JOINS –
(INTERSECTION)

- Main feature which distinguished relational from non relational systems
- Used when a SQL query requires data from more than one table
- Rows in one table may be joined to rows in another table according to common values existing in corresponding columns
- Two main types of joins
  - INNER Joins
  - OUTER Joins

JOINS –
(INTERSECTION)

JOINS

INNER

Based on the common values in both the tables
Ex: A intersection B

OUTER

LEFT

Based on the common values in both the tables
Plus extras in the left table

RIGHT

Based on the common values in both the tables
Plus extras in the right table

FULL

Based on the common values in both the tables
Plus extras from the left table and then from right table

JOINS

Old method of writing a join

```
SELECT empname, deptname
FROM emp, dept
WHERE emp.deptno = dept.deptno
```

ANSI SQL 1999 standard

```
SELECT empname,deptname
FROM emp
Join dept
On emp.deptno = dept.deptno
```

Using TABLE ALIASES

- If table names need to be typed repeatedly, it can get tedious
- Temporary Labels or Aliases can be used for table names in WHERE clause
- This remain valid for current SELECT clause
- Can be up to 30 character in length
- If a table alias is used for a table name in FROM clause then that alias must be substituted for the table name throughout the SELECT statement.

INNER JOIN

```
SELECT    e. empname,  d.deptname
FROM      employee e
JOIN   dept d
ON     e. deptno = d. deptno
```

```
SELECT    e. empname, d.deptname
FROM      employee e
INNER JOIN dept d
ON     e. deptno = d. deptno
```

Note: Default is INNER joins

Employee Table - sample

EMPNO	EMPNAME	DEPTNO	SALARY
E001	ABC	D001	12000
E002	DEF	D002	15000
E003	GHI	D001	11000
E004	UVW	NULL	17000

DEPT TABLE – sample

DEPTNO	DEPTNAME
D001	Finance
D002	Accounts
D003	Sales

INNER JOIN

```
SELECT    e. empname, d.deptname
FROM      employee e
INNER JOIN dept d
ON      e. deptno = d. deptno
```

EMPNAME	DEPTNAME
ABC	Finance
DEF	Accounts
GHI	Finance

So we get only those rows which satisfy the condition e.deptno = d.deptno This join is called "INNER JOIN"

OUTER JOIN RIGHT

```
SELECT    e.empname,d.deptname
FROM      employee  e
Right outer JOIN   dept d
ON      e.deptno = d.deptno
```

EMPNAME	DEPTNAME
ABC	Finance
DEF	Accounts
GHI	Finance
	Sales

We get those rows which satisfy the condition

e.deptno = d.deptno and non-matching rows from the right side table.In this case “Dept”. This join is called “RIGHT OUTER JOIN”

OUTER JOIN
LEFT

```
SELECT    e.empname,d.deptname
FROM      employee e
LEFT outer JOIN          dept d
ON      e.deptno = d.deptno
```

EMPNAME	DEPTNAME
ABC	Finance
DEF	Accounts
GHI	Finance
UVW	

We get those rows which satisfy the condition e.deptno = d.deptno and non-matching rows from the left side table.In this case “Emp”. This join is called “LEFT OUTER JOIN”

OUTER JOIN
FULL

```
SELECT    e.empname,d.deptname
FROM      emp e
FULL outer JOIN           dept d
ON      e.deptno = d.deptno
```

EMPNAME	DEPTNAME
ABC	Finance
DEF	Accounts
GHI	Finance
UVW	
	SALES

We get those rows which satisfy the condition e.deptno = d.deptno and non-matching rows from the left side and right side table. This join is called “FULL OUTER JOIN”

UNION

- The UNION operation combines two sets of rows into a single set composed of all the rows in either or both the original sets.

UNION

RULES FOR UNION

- The two sets must contain the same number of columns
- Each column in the first set must be either of the same data type as the corresponding column of the second set or convertible to the same data type as the corresponding column of the second set

UNION

Example

▪ PERM-EMP
▪ Empno
▪ Empname
▪ EmpAddress
▪ EmpPhone
▪ Salary

TEMP-EMP
TEmpno
TEmpname
TEmpAddress
TEmpPhone
Wages

UNION

Example Contd.

- Someone wants the address and phone of all the employees of the company irrespective of whether they are temporary or permanent

```
Select empno, empname, empaddress, empphone
        from perm-emp
        Union
Select tempno, tempname, tempaddress, tempphone
        from temp-emp ;
```

UNION ALL

- Union eliminates duplicates. So if we want all the rows to be printed then we have to use “UNION ALL”

DB2 PRIVILEGES

- IMPLICIT – Automatic privileges for CREATOR of object
- EXPLICIT – Specific privilege provided by GRANT and REVOKE SQL Statement

DB2 PRIVILEGES

1. Implicit Privileges on Table

- All DML
- ALTER and DROP of table
- CREATE and DROP indexes
- References – referential constraints
- Run certain DB2 utilities

3. DCL (Data Control Language)

DB2 PRIVILEGES

2. Explicit privilege

DCL (Data Control Language)

- Explicit privileges can be given to the user by the DCL Statements
  - GRANT
  - REVOKE

DB2 PRIVILEGES
2. Explicit – DCL - GRANT

GRANT

- Grants privileges on different DB2 objects such as the Tables, Views, Plans, Packages, Databases etc. to the required set of users.
- Is used to grant Use privileges to user on requirement.
- Is also used to grant **system privileges** to select few users.
- User with a SYSADM privilege will be responsible for overall control of the system.
- Operation specific and Attribute (column) specific privilleges can be granted.

DB2 PRIVILEGES
2. Explicit – DCL - GRANT

- Syntax : GRANT <privileges> TO <users/PUBLIC>
  [WITH GRANT OPTION]

  E.g. GRANT SELECT, UPDATE(NAME, NO)
  ON Table EMPL To A, B, C (or PUBLIC);

- Note: Select operation is granted for all the attributes to users A, B and C. But, Update operation is allowed only for NAME and NO attributes.

  Eg. Grant ALL ON TABLE EMPS TO USER1

- All the possible operation on all the attributes of table EMPS are granted to USER1.

DB2 PRIVILEGES
2. Explicit – DCL - GRANT

* Some table (or View) privileges are :

  - Select, Update, Delete and Insert.

* Privileges specific to Tables are:

  - Alter & Index (create).

* There are no specific DROP privileges;

* The table can be dropped by its owner or a SYSADM.

* A user having authority to grant privilege to another, also has the authority to grant the privilege with "**with the GRANT Option**".

DB2 PRIVILEGES
2. Explicit – DCL - REVOKE

- Revoke is primarily used to revoke (call back) the privileges given to a user on specific Objects.
- The user granting the privileges has the authority to Revoke also.
- It is not possible to be column specific when revoking an **Update** privilege.
- It is possible to be user specific when revoking a privilege granted.

DB2 PRIVILEGES
2. Explicit – DCL - GRANT

Removing privileges on Table from the user

- REVOKE SELECT

  ON TABLE EMP,DEPTS FROM CMAP112;

- REVOKE ALL ON TABLE EMP FROM USER1;

- REVOKE ALL ON TABLE EMP FROM PUBLIC;

4. TCL
(TRANSACTION CONTROL LANGUAGE)

- COMMIT
- ROLLBACK

4. TCL
(TRANSACTION CONTROL LANGUAGE)

Commit Transaction

- Indicates successful end of a unit work
- What ever changes done till this command is given will be written in the actual memory.
- All page LOCKS released.
- TABLE (Space) Locks released if Release (Commit) on BIND.
- Cursor is Closed.

Syntax : COMMIT ;

4. TCL
(TRANSACTION CONTROL LANGUAGE)

Rollback Transaction

- Current unit of work abandoned
- Changes to data since last COMMIT are undone.
- ALL Page Locks released.
- TABLE (Space) Locks released if Release (Commit) on BIND.
- Cursor Closed.

Syntax :

```
ROLLBACK;
```

SYNONYMS AND ALIASES

These are alternate names for tables and views.

SYNONYM

- An alternative name for a table or view.
- Mainly to hide qualifier of a table
- It is dropped when table / table space is dropped
- Synonyms can only be used to refer to objects at the subsystem in which the synonym is defined.

SYNONYM

Syntax

CREATE SYNONYM <syn name> FOR
<user id > . <table name>;

DAVIN5.EMPS can be referred as SYEMPS
If table is deleted, then synonym also will get deleted

```
---------+---------+---------+---------+---------+---------+---------+---
--
--
     CREATE SYNONYM SYEMPS FOR
            DAVIN5.EMPS;
---------+---------+---------+---------+---------+---------+---------+---
DSNE616I STATEMENT EXECUTION WAS SUCCESSFUL, SQLCODE IS 0
```

ALIASES

- An alternative name for a table or view.
- Mainly to hide qualifier of a table
- Can be accessible creator as well as any user to whom the privilege is granted
- It is not dropped when table / table space is dropped Aliases name can only be used to refer to objects across different sub sytems.

ALIASES

Alias.

An alternate name that can be used in SQL statements to refer to a table or view in the same or a remote DB2 subsystem

```
CREATE  ALIAS  TESTTAB  FOR

locationname.userid.EMP;
```