Module 5

DDL

Data Definition Language – create, remove, modify database structures (such as tables, indexes, views)

- Create
- Alter
- Drop

DML

Data Manipulation Language – create, remove and modify data values (the values stores in tables, indexes and views)

- Insert
- Update
- Delete
- Truncate

1- The Create Statement:

Used to create a new table or view

- Defines the table NAME and its COULMNS
- Defines the DATATYPE, LENGTH for each column
- Defines the CONSTRAINTS for each column.

Usually, we Drop a table if it exist (before creating)

DROP Statement DROP TABLE IF EXISTS <table-name>

CREATE STATEMENT SYNTAX:

```
CREATE TABLE <table-name>
(
   Column, Datatype (n)   --n is the length of the field
   Column, Datatype (n)   NOT NULL,
   Column, Datatype (n)   NOT NULL Default 0,
    Column, Datatype (n)   CONSTRAINT <constraint name> TYPE,
)
```

Example

```
CREATE TABLE Shippers
(
  ShipperiD          int             NOT NULL,
  CompanyName    char(40)        NOT NULL,
  PhoneNo           varchar(20)   NOT NULL
)
```

2- SQL Constraints:

Constraints are the rules that are applied to rows and columns that allows the database software to maintain the data integrity within the database.

The database software will not allow the condition of the constraint to be violated.

Constraints can exist at the Table level/ or the Column level.

Tabel Level constraints (Primary Key, Foreign Key).

Column Level constraints (Not Null, Check, Default, Unique, Primary Key).

Column Level constraints

```
Create Table items
(
    itemID          INT             Not Null Primary Key,
    itemcode        varchar(4)      Unique,
    itemname        varchar(40)     Not Null Default  ‘ ’,

   quantity         INT             Not Null Default  0,
   price            Real            Not Null Default 0,

   (price           Decimal(7,2)   check(price<1000))

   (check is available in Postgres not in MySQL)

)
```

Table Level Constraints

```
Create Table items
(
    itemID          INT             Not Null ,
    supplierID      INT             Not Null ,
    itemcode        varchar(4)      Unique,
    itemname        varchar(40)     Not Null Default  ‘ ’,
    quantity        INT             Not Null Default  0,
    price           Real            Not Null Default 0,
                OR
    price           Decimal(7,2)   check(price<1000)

   PRIMARY KEY (itemID),
   CONSTRAINT  fk_supplier   FOREIGN KEY (supplierID) References
   suppliers (supplierID)

)
```

About Foreign Key:

A table level foreign key can indicate how to handle situation when a foreign key in parent table is deleted or updated.

```
CONSTRAINT fk_supplier   FOREIGN KEY (supplierID),
                         References  supplier (supplierID)
        ON UPDATE    <action>
        ON DELETE    <action>
```

Actions may be:

CASCADE	The change to the parent is cascaded to all the child rows
NO ACTION	The changes to the parent is prohibited.
SET NULL	The foreign key column in the child is set to null
SET DEFAULT	The foreign key column in the child is set to default
RESTRICT	The changes to the parent is prohibited

3- The ALTER Command:

The Alter command allows you to change many different characteristics of database structures (such as tables, columns, views).

- Rename a Table
- Rename a Column

- Rename a Constraint
- ADD or DROP a column
- ADD or DROP a Constraint
- Set or DROP a default (in column)
- Set a data_type (in column)

Renaming a Table using alter:

```
Alter Table items,
Rename to “alters”;
```

Renaming a column using alter:

```
Alter Table items,
Rename Column “itemcode” to “itemunit”
```

Adding a column using alter:

```
Alter Table items,
ADD Column “itemName” varchar(20) NULL;
```