SQL DDL DML

DDL – Data Definition Language

Modify Database Structures

`Create`

`Alter`

`Drop`

DML – Data Manipulation Language

Modify the data values within the tables/rows

`Insert`

`Update`

`Delete`

SQL Insert

INSERT statement

```
INSERT INTO <table name> (column, column,
column)
   VALUES (value, value, value)
```

(if no column/value is specified, NULL or default will be assigned)

```
INSERT INTO <table name>
   VALUES (value, value, value, value)
```

(must have a value or NULL for every column in the table)

SQL Insert

INSERT statement

```
INSERT INTO nwEmployees
    (LastName, FirstName, Title, TitleOfCourtesy,
    BirthDate, HireDate, Address, City, Region,
    PostalCode, Country, HomePhone, Extension)
    VALUES
    ('Dunn','Nat','Sales Representative', 'Mr.',
    '1970-02-19', '2014-01-15',
    '4933 Jamesville Rd.','Jamesville','NY',
    '13078','USA','315-555-5555','130');
```

SQL Insert

INSERT statement

```
INSERT INTO nwEmployees
    VALUES
    ('20','Thomas','Tammy','Database Administrator',
    'Ms.','1990-08-27', '2017-06-18',
    '5012 Arapahoe St.','Boulder','CO',
    '80304','USA');
```

```
INSERT INTO nwEmployees
    VALUES
    ('20','Thomas','Tammy','Database Administrator',
    'Ms.','1990-08-27', '2017-06-18',
    '5012 Arapahoe St.','Boulder','CO',
    '80304','USA',NULL,NULL,'','','');
```

SQL DDL DML

CREATE statement

```
CREATE TABLE <table name>
   (column  DATATYPE(L),
    column  DATATYPE(L) NOT NULL,
    column  DATATYPE(L) NOT NULL Default 0,
    column  DATATYPE(L) CONSTRAINT
         <constraint name> TYPE,
    column  DATATYPE(L))
```

DESCRIBE statement

shows you what MySQL knows about a table

SQL DDL DML

CREATE statement

```
CREATE TABLE IF NOT EXISTS items (
    itemID     INT          NOT NULL AUTO_INCREMENT,
    itemCode   CHAR(3)      ,
    itemname   VARCHAR(40)  NOT NULL DEFAULT ' ',
    quantity   INT          NOT NULL DEFAULT 0,
    price      DECIMAL(9,2)NOT NULL DEFAULT 0,
    PRIMARY KEY  (itemID)
         );

DROP TABLE IF EXISTS items;

DESC items;
```

CREATE statement

```
CREATE OR REPLACE TABLE items (
    itemID     INT         NOT NULL ,
    itemCode   CHAR(3)     ,
    itemname   VARCHAR(40)  NOT NULL DEFAULT ' ',
    quantity   INT          NOT NULL DEFAULT 0,
    price      DECIMAL(9,2) NOT NULL DEFAULT 0
    PRIMARY KEY  (itemID)
         );
```

SQL DDL DML

TRUNCATE statement – removes all rows, keeps structure

```
TRUNCATE TABLE <table name>
```

DROP statement -- removes all rows, removes structure

```
DROP TABLE <table name>
```

SQL DDL DML

ALTER statement

```
ALTER TABLE <table name>
   ADD/MODIFY/DROP
            COLUMN <column name>  DATATYPE(L),

   RENAME <new table name>

ALTER TABLE <table name>
   DROP COLUMN
```

SQL Insert

ALTER statement

```
ALTER TABLE nwemployees
   MODIFY COLUMN EmployeeID INT(11) PRIMARY KEY
   AUTO_INCREMENT;

ALTER TABLE Items
   ADD PRIMARY KEY (ItemID) ;

ALTER TABLE Items
   ADD COLUMN InventoryDate DATE AFTER  ;

ALTER TABLE Items
   DROP COLUMN InventoryDate;
```

SQL Insert

BULK INSERT statement

```
INSERT INTO items
    SELECT ProductID, CategoryID,  ProductName,
    CURDATE(), unitsInStock, UnitPrice
        FROM nwProducts
    ;
```

SQL DDL DML

UPDATE statement

```
UPDATE <table name>
  SET column = <value>
  WHERE <condition>
```

DELETE statement

```
DELETE FROM <table name>
  WHERE <condition>
```

Note: Without the WHERE clause,
the DELETE will affect ALL rows

SQL Insert

UPDATE statement

```
UPDATE items
   SET price = (price + (price * .05))
   WHERE itemcode = 1;

UPDATE items
   SET price = ROUND((price + (price * .05)),2)
   WHERE itemcode = 1;
```

Delete statement

```
DELETE FROM items
   WHERE itemcode = 2;
```

The VIEW

- A “VIEW” is an empty shell of a table definition
- The view contains no data until it is queried
- Sometimes considered a “Virtual Table”
- Each time the view is queried, the underlying query that populates the view is re-executed

CREATING a VIEW

```
CREATE VIEW <view name> AS
   SELECT <col1>, <col2>, <col3>
      FROM <table1>
      WHERE <condition>
```

Why VIEWs?

- The base table or specific columns in the base table can be hidden from certain users who are only allowed access to the view
- Very complex SQL to create the view can be hidden from end users

SQL DDL DML

First “why”:

Base Table:

Employees(EmpID, Lastname, Firstname, Salary, HireDate)

View:

Employees(EmpID, Lastname, Firstname, HireDate)

SQL DDL DML

Second “why”:

Base Query:

```
Create VIEW TopEmployeeOrders AS
   Select LastName, Firstname,
    sum(UnitPrice * Quantity) as 'OrderValue'
   from nwEmployees E, nwOrders O,
    nwOrderDetails D
   where E.EmployeeID = O.EmployeeID
     and O.OrderID = D.OrderID
   GROUP BY LastName, FirstName
   Order By 3 desc
```

View

```
Select * from TopEmployeeOrders;
```

SQL DDL DML

```
CREATE OR REPLACE VIEW TopEmployeeOrders AS
   SELECT LastName, Firstname,
    SUM(UnitPrice * Quantity) AS 'OrderValue'
   FROM nwEmployees E, nwOrders O,
    nwOrderDetails D
   WHERE E.EmployeeID = O.EmployeeID
     AND O.OrderID = D.OrderID
   GROUP BY LastName, FirstName
   ORDER BY 3 DESC;
```