**SQL TABLES QUERIES AND SCREENSHOTS**

SQL Stands for Structured Query Language.It allows us to create the databases for any E-commerce company for storing the data in a relational manner here, we are creating tables and Retrieving data using our business criteria using SQL queries.

**3.Creating Table Schema**

1. Customer Table

-- create a table

CREATE TABLE Customers (

  customer\_id INTEGER PRIMARY KEY,

  org\_person TEXT NOT NULL,

  org\_name TEXT NOT NULL,

  gender TEXT NOT NULL,

  first\_name TEXT NOT NULL,

  middle\_initial TEXT  NULL,

  last\_name TEXT NULL,

  email\_address TEXT NOT NULL,

  login\_name TEXT NOT NULL,

  login\_pwd TEXT NOT NULL,

  ph\_no TEXT NOT NULL,

  address\_line\_1 TEXT NOT NULL,

  address\_line\_2 TEXT NOT NULL,

  address\_line\_3 TEXT NOT NULL,

  address\_line\_4 TEXT NOT NULL,

  town\_city TEXT NOT NULL,

  county TEXT NOT NULL,

  country TEXT NOT NULL

);

-- insert some values

INSERT INTO Customers VALUES (1, 'Haridas', 'Flipkartrt','M','Haridas','','','hari@gmail.com','hari','hari@123','+912333342345','Fliporganization','Cherukara','Malappuram','Kerala','Cherukara','Kerala','Kerala\_North');

INSERT INTO Customers VALUES (2, 'Ramu', 'Amazon','M','Ramu','','','ram@gmail.com','ram','ram@123','+912333342345','AmazonOrganization','Cherukara','Kanyakumari','Tamilnadu','Cheruka','Tamilnadu','Tamilnadu\_South');

INSERT INTO Customers VALUES (3, 'Abhijith', 'Myntra','M','Abhijith','','','abhi@gmail.com','abhi','abhi@123','+912333342345','Fliporganization','Cherukara','Trivandrum','Kerala','Cherukara','Kerala','Kerala\_South');

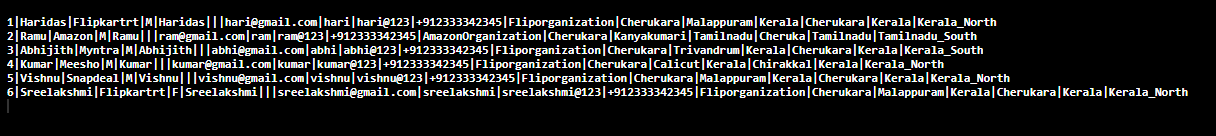
INSERT INTO Customers VALUES (4, 'Kumar', 'Meesho','M','Kumar','','','kumar@gmail.com','kumar','kumar@123','+912333342345','Fliporganization','Cherukara','Calicut','Kerala','Chirakkal','Kerala','Kerala\_North');

INSERT INTO Customers VALUES (5, 'Vishnu', 'Snapdeal','M','Vishnu','','','vishnu@gmail.com','vishnu','vishnu@123','+912333342345','Fliporganization','Cherukara','Malappuram','Kerala','Cherukara','Kerala','Kerala\_North');

INSERT INTO Customers VALUES (6, 'Sreelakshmi', 'Flipkartrt','F','Sreelakshmi','','','sreelakshmi@gmail.com','sreelakshmi','sreelakshmi@123','+912333342345','Fliporganization','Cherukara','Malappuram','Kerala','Cherukara','Kerala','Kerala\_North');

SELECT \* FROM Customers;

Output:



1. Products Table

-- create a table

CREATE TABLE Products (

  product\_id INTEGER PRIMARY KEY,

  product\_type\_code INTEGER FOREIGN KEY REFERENCES Ref\_Product\_Types(product\_type\_code),

  return\_merchandise\_authorization\_nr TEXT NOT NULL,

  product\_name TEXT NOT NULL,

  product\_price FLOAT NOT NULL,

  product\_color TEXT  NOT NULL,

  product\_size INTEGER NOT NULL,

  product\_description TEXT NOT NULL,

  other\_product\_details TEXT NOT NULL

);

INSERT INTO Products VALUES (1001,'NR001','Shoe',999,'Grey',10,'Footwear','Large Size');

INSERT INTO Products VALUES (1002,'NR002','Computer',999,'White',10,'Electronics','Large Size');

INSERT INTO Products VALUES (1003,'NR003','Watch',1599,'Black',10,'Wearable','Large Size');

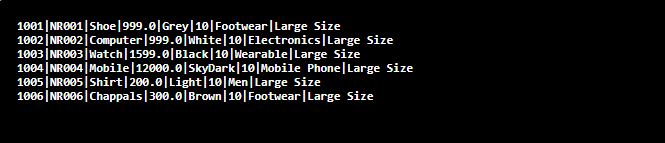
INSERT INTO Products VALUES (1004,'NR004','Mobile',12000,'SkyDark',10,'Mobile Phone','Large Size');

INSERT INTO Products VALUES (1005,'NR005','Shirt',200,'Light',10,'Men','Large Size');

INSERT INTO Products VALUES (1006,'NR006','Chappals',300.0,'Brown',10,'Footwear','Large Size');

SELECT \* FROM Products;

Output:



1. Ref Product Types Table

-- create a table

CREATE TABLE Ref\_Product\_Types (

  product\_type\_code INTEGER PRIMARY KEY,

  parent\_product\_type\_code INTEGER FOREIGN KEY REFERENCES Ref\_Product\_Types(product\_type\_code),

  product\_type\_description TEXT NOT NULL

);

INSERT INTO Ref\_Product\_Types VALUES (1001,'desc1');

INSERT INTO Ref\_Product\_Types VALUES (1002,'desc2');

INSERT INTO Ref\_Product\_Types VALUES (1003,'desc3');

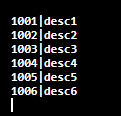
INSERT INTO Ref\_Product\_Types VALUES (1004,'desc4');

INSERT INTO Ref\_Product\_Types VALUES (1005,'desc5');

INSERT INTO Ref\_Product\_Types VALUES (1006,'desc6');

SELECT \* FROM Ref\_Product\_Types;

Output:



1. Orders Table

-- create a table

CREATE TABLE Orders (

  order\_id INTEGER PRIMARY KEY,

  customer\_id INTEGER FOREIGN KEY REFERENCES Customers(customer\_id),

  order\_status\_code INTEGER FOREIGN KEY REFERENCES Ref\_Order\_Status\_Codes(order\_status\_code),

  date\_order\_placed DATE NOT NULL,

  order\_details TEXT NOT NULL

);

INSERT INTO Orders VALUES (1001,'12-03-2021','Completed');

INSERT INTO Orders VALUES (1002,'12-04-2021','Not Completed');

INSERT INTO Orders VALUES (1003,'12-05-2021','Not done Payment');

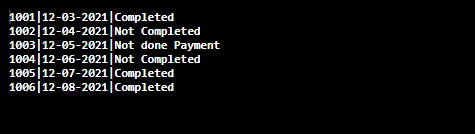
INSERT INTO Orders VALUES (1004,'12-06-2021','Not Completed');

INSERT INTO Orders VALUES (1005,'12-07-2021','Completed');

INSERT INTO Orders VALUES (1006,'12-08-2021','Completed');

SELECT \* FROM Orders;

Output:



1. Ref\_Order\_Status\_Code Table:

-- create a table

CREATE TABLE Ref\_Order\_Status\_Codes (

  order\_status\_code INTEGER PRIMARY KEY,

  order\_status\_description TEXT NOT NULL

);

INSERT INTO Ref\_Order\_Status\_Codes VALUES (1001,'Completed');

INSERT INTO Ref\_Order\_Status\_Codes VALUES (1002,'Not Completed');

INSERT INTO Ref\_Order\_Status\_Codes VALUES (1003,'Not done Payment');

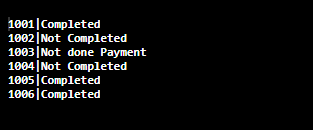
INSERT INTO Ref\_Order\_Status\_Codes VALUES (1004,'Not Completed');

INSERT INTO Ref\_Order\_Status\_Codes VALUES (1005,'Completed');

INSERT INTO Ref\_Order\_Status\_Codes VALUES (1006,'Completed');

SELECT \* FROM Ref\_Order\_Status\_Codes;

Output:



1. Customer\_Payment\_Methods Table:

-- create a table

CREATE TABLE Customer\_Payment\_Methods (

  customer\_payment\_id INTEGER PRIMARY KEY,

  customer\_id INTEGER FOREIGN KEY REFERENCES Customers(customer\_id),

  payment\_method\_code INTEGER FOREIGN KEY REFERENCES Ref\_Payment\_Methods(payment\_method\_code),

  credit\_card\_number TEXT NOT NULL,

  payment\_method\_details TEXT NOT NULL

);

INSERT INTO Customer\_Payment\_Methods VALUES (1001,'1111111111111111','Completed');

INSERT INTO Customer\_Payment\_Methods VALUES (1002,'1111111111111111','Not Completed');

INSERT INTO Customer\_Payment\_Methods VALUES (1003,'1111111111111111','Not done Payment');

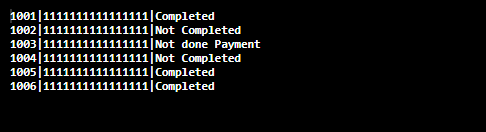
INSERT INTO Customer\_Payment\_Methods VALUES (1004,'1111111111111111','Not Completed');

INSERT INTO Customer\_Payment\_Methods VALUES (1005,'1111111111111111','Completed');

INSERT INTO Customer\_Payment\_Methods VALUES (1006,'1111111111111111','Completed');

SELECT \* FROM Customer\_Payment\_Methods;

Output:



1. Ref\_Payment\_Methods:

-- create a table

CREATE TABLE Customer\_Payment\_Methods (

  payment\_method\_code INTEGER PRIMARY KEY,

  payment\_method\_description TEXT NOT NULL

);

INSERT INTO Customer\_Payment\_Methods VALUES (1001,'Completed');

INSERT INTO Customer\_Payment\_Methods VALUES (1002,'Not Completed');

INSERT INTO Customer\_Payment\_Methods VALUES (1003,'Not done Payment');

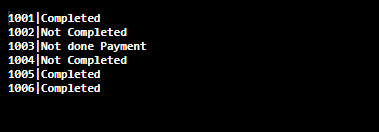
INSERT INTO Customer\_Payment\_Methods VALUES (1004,'Not Completed');

INSERT INTO Customer\_Payment\_Methods VALUES (1005,'Completed');

INSERT INTO Customer\_Payment\_Methods VALUES (1006,'Completed');

SELECT \* FROM Customer\_Payment\_Methods;

Output:



1. Invoice Table

-- create a table

CREATE TABLE Invoices (

  invoice\_no INTEGER PRIMARY KEY,

  order\_id INTEGER FOREIGN KEY REFERENCES Orders(order\_id),

  invoice\_status\_code INTEGER FOREIGN KEY REFERENCES Ref\_Invoice\_Status\_Codes(invoice\_status\_code),

  invoice\_date DATE NOT NULL,

  invoice\_details TEXT NOT NULL

);

INSERT INTO Invoices VALUES (10001,'10-11-2020','Completed');

INSERT INTO Invoices VALUES (10002,'11-11-2020','Not Completed');

INSERT INTO Invoices VALUES (10003,'12-11-2020','Not done Payment');

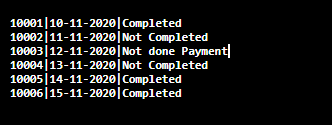
INSERT INTO Invoices VALUES (10004,'13-11-2020','Not Completed');

INSERT INTO Invoices VALUES (10005,'14-11-2020','Completed');

INSERT INTO Invoices VALUES (10006,'15-11-2020','Completed');

SELECT \* FROM Invoices;

Output:



1. Ref\_Invoice\_Status\_Codes Table:

-- create a table

CREATE TABLE Ref\_Invoice\_Status\_Codes (

  invoice\_status\_code INTEGER PRIMARY KEY,

  invoice\_status\_description TEXT NOT NULL

);

INSERT INTO Ref\_Invoice\_Status\_Codes VALUES (10001,'Completed');

INSERT INTO Ref\_Invoice\_Status\_Codes VALUES (10002,'Not Completed');

INSERT INTO Ref\_Invoice\_Status\_Codes VALUES (10003,'Not done Payment');

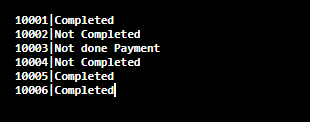
INSERT INTO Ref\_Invoice\_Status\_Codes VALUES (10004,'Not Completed');

INSERT INTO Ref\_Invoice\_Status\_Codes VALUES (10005,'Completed');

INSERT INTO Ref\_Invoice\_Status\_Codes VALUES (10006,'Completed');

SELECT \* FROM Ref\_Invoice\_Status\_Codes;

Output:



1. Payments:

-- create a table

CREATE TABLE Payments (

  invoice\_no INTEGER PRIMARY KEY,

  invoice\_no INTEGER FOREIGN KEY REFERENCES Invoices(invoice\_no),

    payment\_date DATE NOT NULL,

  payment\_amount FLOAT NOT NULL

);

INSERT INTO Payments VALUES (10001,'10-11-2020',2000);

INSERT INTO Payments VALUES (10002,'11-11-2020',660);

INSERT INTO Payments VALUES (10003,'12-11-2020',199);

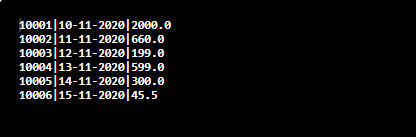
INSERT INTO Payments VALUES (10004,'13-11-2020',599);

INSERT INTO Payments VALUES (10005,'14-11-2020',300);

INSERT INTO Payments VALUES (10006,'15-11-2020',45.50);

SELECT \* FROM Payments;

Output:



1. - Shipments:

-- create a table

CREATE TABLE Shipments (

  shipment\_id INTEGER PRIMARY KEY,

  order\_id INTEGER FOREIGN KEY REFERENCES Orders(order\_id),

  invoice\_no INTEGER FOREIGN KEY REFERENCES Invoices(invoice\_no),

  shipment\_trace\_no INTEGER NOT NULL,

  shipment\_date DATE NOT NULL,

  other\_ship\_details TEXT NOT NULL

);

INSERT INTO Shipments VALUES (10001,10,'10-11-2019','Completed');

INSERT INTO Shipments VALUES (10002,20,'11-11-2019','Not Completed');

INSERT INTO Shipments VALUES (10003,30,'12-11-2019','Not done Payment');

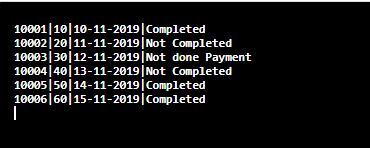
INSERT INTO Shipments VALUES (10004,40,'13-11-2019','Not Completed');

INSERT INTO Shipments VALUES (10005,50,'14-11-2019','Completed');

INSERT INTO Shipments VALUES (10006,60,'15-11-2019','Completed');

SELECT \* FROM Shipments;

Output:



1. Order\_Items Table:

-- create a table

CREATE TABLE Order\_Items (

  order\_item\_id INTEGER PRIMARY KEY,

   product\_id INTEGER FOREIGN KEY REFERENCES Products(product\_id),

  order\_id INTEGER FOREIGN KEY REFERENCES Orders(order\_id),

   order\_item\_status\_code INTEGER FOREIGN KEY REFERENCES Ref\_Order\_Item\_Status\_Codes(order\_item\_status\_code),

  order\_item\_quantity INTEGER NOT NULL,

  order\_item\_price FLOAT NOT NULL,

  RMA\_no TEXT NOT NULL,

  RMA\_Issued\_By TEXT NOT NULL,

  RMA\_Issued\_date DATE NOT NULL,

  Other\_Order\_item\_details TEXT NOT NULL

);

INSERT INTO Order\_Items VALUES (10001,10,999,'R01','Amazon','21-02-2020','NA');

INSERT INTO Order\_Items VALUES (10002,20,2999,'R02','Amazon','22-02-2020','NA');

INSERT INTO Order\_Items VALUES (10003,30,299,'R03','Amazon','23-02-2020','NA');

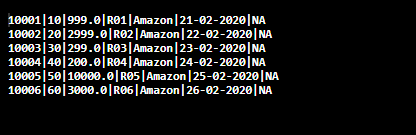
INSERT INTO Order\_Items VALUES (10004,40,200,'R04','Amazon','24-02-2020','NA');

INSERT INTO Order\_Items VALUES (10005,50,10000,'R05','Amazon','25-02-2020','NA');

INSERT INTO Order\_Items VALUES (10006,60,3000,'R06','Amazon','26-02-2020','NA');

SELECT \* FROM Order\_Items;

Output:



1. Ref\_Order\_Item\_Status\_Code:

-- create a table

CREATE TABLE Ref\_Order\_Item\_Status\_Codes (

  order\_item\_status\_code INTEGER PRIMARY KEY,

  Order\_item\_status\_description TEXT NOT NULL

);

INSERT INTO Ref\_Order\_Item\_Status\_Codes VALUES (100001,'Purchased');

INSERT INTO Ref\_Order\_Item\_Status\_Codes VALUES (100002,'Purchased');

INSERT INTO Ref\_Order\_Item\_Status\_Codes VALUES (100003,'Purchased');

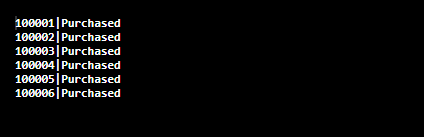
INSERT INTO Ref\_Order\_Item\_Status\_Codes VALUES (100004,'Purchased');

INSERT INTO Ref\_Order\_Item\_Status\_Codes VALUES (100005,'Purchased');

INSERT INTO Ref\_Order\_Item\_Status\_Codes VALUES (100006,'Purchased');

SELECT \* FROM Ref\_Order\_Item\_Status\_Codes;

Output:



1. Shipment\_items table:

-- create a table

CREATE TABLE Shipment\_items (

  shipment\_item\_id INTEGER PRIMARY KEY,

   shipment\_id INTEGER FOREIGN KEY REFERENCES Shipments(shipment\_id),

  order\_item\_id INTEGER FOREIGN KEY REFERENCES Orders(order\_id),

);

-- INSERT INTO Order\_Items VALUES (10001,10,999,'R01','Amazon','21-02-2020','NA');

-- INSERT INTO Order\_Items VALUES (10002,20,2999,'R02','Amazon','22-02-2020','NA');

-- INSERT INTO Order\_Items VALUES (10003,30,299,'R03','Amazon','23-02-2020','NA');

-- INSERT INTO Order\_Items VALUES (10004,40,200,'R04','Amazon','24-02-2020','NA');

-- INSERT INTO Order\_Items VALUES (10005,50,10000,'R05','Amazon','25-02-2020','NA');

-- INSERT INTO Order\_Items VALUES (10006,60,3000,'R06','Amazon','26-02-2020','NA');

SELECT \* FROM Shipment\_items;

Output:

1. **Data Retrieval Queries From The Table**

Select Statments are used to retrieve or pull the data from the data base.We can use SELECT statement for getting the data from the database.And also we can use select statement with other statement like Update, Delete etc.Here we are using some data retrieval queries for analyse the data.

1. **UPDATE**

Update statements are used to modify the existing data in a table.The important thing is we must specify the WHERE clause when using update statement.Otherwise all records will be updated.Normal case we are updating using the primary key column in the table.

And we can update multiple columns in a single update query.

UPDATE Customers

SET login\_name = 'haridas', ph\_no= '910333342345'

WHERE customer\_id = 1;

SELECT \* FROM Customers

1. **COUNT**

Count is an aggregate function in SQL like AVG(), SUM() functions.It returns number of rows that matches the condition we given in the query.If we omit the condition, it returns number of rows in the table.And also we have DISTINCT keyword for considering the unique records in the table.

For getting nomber of rows in the table we can use the query like:

SELECT COUNT(\*)FROM Table\_name;

SELECT DISTINCT(COUNT(product\_name))

FROM Products

WHERE product\_price >=100 AND product\_size=10;

1. **INNER JOIN**

A Join means joining tables according to some criteria.We have four common joins in sql.Inner Join,left join,right join,Full join.For inner join, it selects the common rows in two tables.Then , Full Join it takes all rows from both table.For left join it takes all rows from the left table and common rows in the right table.At last, In right join it takes all the rows from the right table and common rows in the left table.

We can join multiple tables with single sql query.

SELECT \*

FROM Orders

INNER JOIN Order\_Items

ON Orders.order\_id = Order\_Items.order\_id;

1. **GROUP BY WITH LEFT JOIN**

Group by statements are the statements for grouping one column according to another columns.For example, number of employees in each country.Group by statements are mostly used with aggregated functions like COUNT(),AVG(),SUM() etc. To group the result with one or more columns.

SELECT COUNT(customer\_id), country

FROM Customers

GROUP BY country;

Here we are using the group by with LEFT JOIN.

SELECT Invoices.invoice\_date, COUNT(Orders.order\_id) AS NumberOfOrders FROM Orders

LEFT JOIN Invoices ON Orders.order\_id = Invoices.order\_id

GROUP BY date\_order\_placed;

1. **DELETE**

Delete statement is used to delete a record from the table using where clause.If we are not using the where clause, we can delete the table using the query like

DELETE FROM Products

When Products is a table name.

And for deleting one record we can use WHERE clause and the query like

DELETE FROM Products WHERE product\_name='Shoe';