**Join statements, set operations, nested queries and grouping**

**Aim:** To get introduced to

1. Union

2. Intersection

3. Minus

4. Join

5. Nested Queries

6. Group by and having

**Q**. Amazon is one of the largest online stores operating in united states of America, they’re maintaining four tables in their database – the items table, customers table, orders table, and delivery table. Each of these tables contains the following attributes

**item**

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Item\_id (pimary key) | Item\_name(type =varchar(50) | category | price | Instock (type = int, >=0) |  |  |  |  |  |

**customer**

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Cust\_id | Cust\_name | Address | State |  |  |  |  |  |  |

**Order**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Order\_id (primary key) | Item\_id (foreign key) | Quantity (type=int) | Order\_date (type=date) | Delivery |  |  |

**Delivery**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Delivery\_id | Cust\_id (foreign key) | Order\_id (foreign key) |  |  |  |  |

Create the above tables and populate them with appropriate data.

1 .List the details of all customers who have placed an order

2. List the details of all the customers whose orders have been delivered

3. Find the order date for all customers whose name starts in the letter ‘j’

4. Display the name and price of all items bought by the customer ‘Mickey’.

5. List the details of all customers who have placed an order after jan, 2013 and not received delivery of items

6. Find the item\_id of items which has either been ordered or not delivered (use set union)

7. Find the name of all customers who have placed an order and have their orders delivered. (Use SET INTERSECTION)

8. Find the name of all customers who have placed an order but not having their orders delivered. (Use SET MINUS)

9. Find the name of the customer who has placed the most number of orders.

10. Find the details of all customers who have purchased items exceeding a price of 5000$

11. Find the name and address of customers who has not ordered a “Samsung Galaxy S4”.

12. Perform left outer join and right outer join on customers and orders table

13.Find the details of all customers grouped by ‘state’

14. Display the details of all items grouped by category and having a price greater than average price of all items.

CREATE TABLE ITEMS(ITEM\_ID INT PRIMARY KEY, ITEM\_NAME VARCHAR(50), CATEGORY VARCHAR(50), PRICE INT, INSTOCK INT);

INSERT INTO ITEMS VALUES(101,'Oneplus 6','PHONE',750,7);

INSERT INTO ITEMS VALUES(102,'GALAXY S4','PHONE',250,2);

INSERT INTO ITEMS VALUES(103,'JBL T160BT','HEADSET',50,4);

INSERT INTO ITEMS VALUES(104,'IPHONE CHARGER','CHARGER',10,6);

INSERT INTO ITEMS VALUES(105,'JBL LIVE 500BT','HEADSET',200,1);

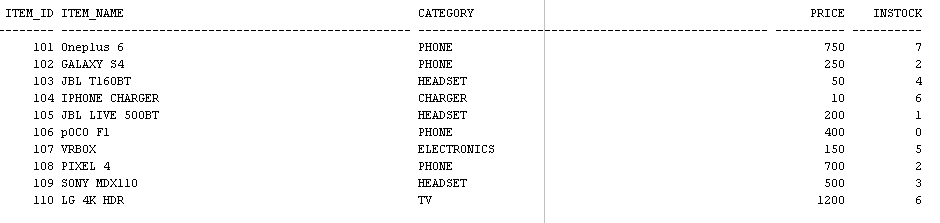
INSERT INTO ITEMS VALUES(106,'pOCO F1','PHONE',400,0);

INSERT INTO ITEMS VALUES(107,'VRBOX','ELECTRONICS',150,5);

INSERT INTO ITEMS VALUES(108,'PIXEL 4','PHONE',700,2);

INSERT INTO ITEMS VALUES(109,'SONY MDX110','HEADSET',500,3);

INSERT INTO ITEMS VALUES(110,'LG 4K HDR','TV',1200,6);



CREATE TABLE CUSTOMERS(CUST\_ID INT PRIMARY KEY,CUST\_NAME VARCHAR(20),ADDRESS VARCHAR(25),STATE VARCHAR(20));

INSERT INTO CUSTOMERS VALUES(201,'ANANDU R','ERNAKULAM','KERALA');

INSERT INTO CUSTOMERS VALUES(202,'CLARA','LONDON','BRITAN');

INSERT INTO CUSTOMERS VALUES(203,'SUCHITA','KOLAR','KARNATAKA');

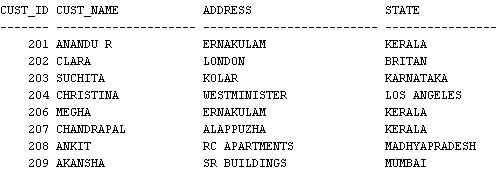
INSERT INTO CUSTOMERS VALUES(204,'CHRISTINA','WESTMINISTER','LOS ANGELES');

INSERT INTO CUSTOMERS VALUES(206,'MEGHA','ERNAKULAM','KERALA');

INSERT INTO CUSTOMERS VALUES(207,'CHANDRAPAL','ALAPPUZHA','KERALA');

INSERT INTO CUSTOMERS VALUES(208,'ANKIT','RC APARTMENTS','MADHYAPRADESH');

INSERT INTO CUSTOMERS VALUES(209,'AKANSHA','SR BUILDINGS','MUMBAI');



CREATE TABLE ORDERS(ORDER\_ID INT PRIMARY KEY,ITEM\_ID INT,CUST\_ID INT,QUANDITY INT, ORDER\_DATE DATE,FOREIGN KEY (ITEM\_ID) REFERENCES ITEMS(ITEM\_ID),FOREIGN KEY (CUST\_ID) REFERENCES CUSTOMERS(CUST\_ID));

INSERT INTO ORDERS VALUES(10,101,205,1,'11/05/2010');

INSERT INTO ORDERS VALUES(11,103,201,2,'12/11/2009');

INSERT INTO ORDERS VALUES(12,110,210,1,'15/06/2015');

INSERT INTO ORDERS VALUES(13,102,204,1,'04/04/2014');

INSERT INTO ORDERS VALUES(14,105,209,1,'08/06/2016');

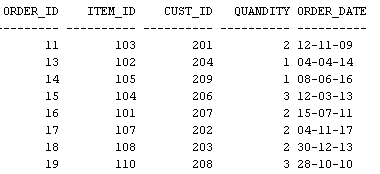
INSERT INTO ORDERS VALUES(15,104,206,3,'12/03/2013');

INSERT INTO ORDERS VALUES(16,101,207,2,'15/07/2011');

INSERT INTO ORDERS VALUES(17,107,202,2,'04/11/2017');

INSERT INTO ORDERS VALUES(18,108,203,2,'30/12/2013');

INSERT INTO ORDERS VALUES(19,110,208,3,'28/10/2010');



CREATE TABLE DELIVERY(DELIVERY\_ID INT PRIMARY KEY,CUST\_ID INT,ORDER\_ID INT,FOREIGN KEY (CUST\_ID) REFERENCES CUSTOMERS(CUST\_ID),FOREIGN KEY (ORDER\_ID) REFERENCES ORDERS(ORDER\_ID));

INSERT INTO DELIVERY VALUES(1001,201,11);

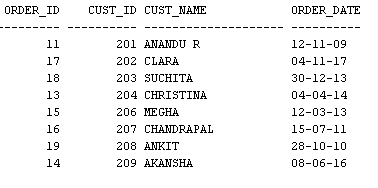
INSERT INTO DELIVERY VALUES(1002,207,16);

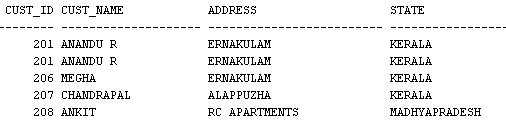
INSERT INTO DELIVERY VALUES(1003,208,19);

INSERT INTO DELIVERY VALUES(1005,201,11);

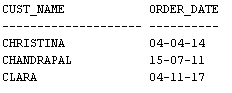
INSERT INTO DELIVERY VALUES(1006,206,15);

1. SELECT ORDERS.ORDER\_ID, CUSTOMERS.CUST\_ID , CUSTOMERS.CUST\_NAME, ORDERS.ORDER\_DATE FROM ORDERS INNER JOIN Customers ON ORDERS.CUST\_ID=CUSTOMERS.CUST\_ID;

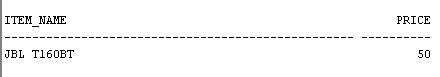


2. SELECT customers.cust\_id, customers.cust\_name,customers.address,customers.state from customers inner JOIN delivery ON DELIVERY.CUST\_ID=CUSTOMERS.CUST\_ID;

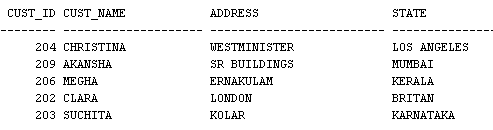
3. SELECT CUSTOMERS.CUST\_NAME, ORDERS.ORDER\_DATE FROM ORDERS INNER JOIN Customers ON ORDERS.CUST\_ID=CUSTOMERS.CUST\_ID where customers.cust\_name like 'C%';



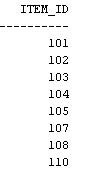
4. SELECT ITEM\_NAME, PRICE FROM ITEMS WHERE ITEM\_ID IN(SELECT ITEM\_ID FROM ORDERS WHERE CUST\_ID IN(SELECT CUST\_ID FROM CUSTOMERS WHERE CUST\_NAME='ANANDU R'));

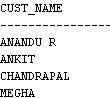


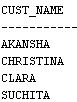
5. SELECT customers.cust\_id, customers.cust\_name,customers.address,customers.state FROM CUSTOMERS WHERE cust\_id in(select cust\_id from orders where extract(year from order\_date)>2012);

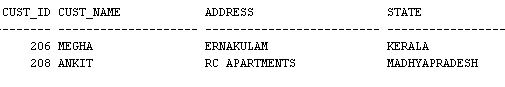


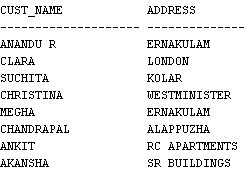
6. select item\_id from orders union select item\_id from orders where cust\_id in (select cust\_id from delivery);

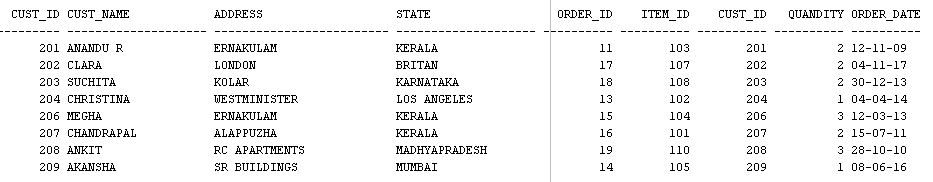


7. select cust\_name from customers where cust\_id in (select cust\_id from orders) intersect select cust\_name from customers where cust\_id in (select cust\_id from delivery);  


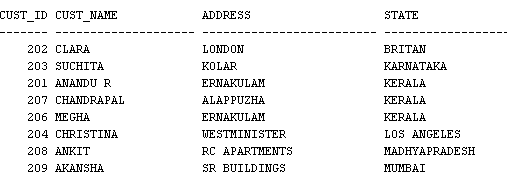
8. select cust\_name from customers where cust\_id in (select cust\_id from orders) minus select cust\_name from customers where cust\_id in (select cust\_id from delivery);  


9. select \* from customers where cust\_id in (select cust\_id from orders where quandity=(select max(quandity) from orders));   


10. select \* from customers where cust\_id in (select cust\_id from orders where item\_id in(select item\_id from items where price>5000 ));   
  
  
11. select cust\_name, address from customers where cust\_id not in (select cust\_id from orders where item\_id in (select item\_id from items where item\_name='galaxy s4'));  


12. select \* from customers left join orders on customers.cust\_id=orders.cust\_id;  


13. select \* from customers order by state;



14. select \* from items where price>(select avg(price) from items ) order by category;

