Exp No: 8 Date : 14-08-2019

JOIN STATEMENTS, SET OPERATIONS, NESTED QUERIES AND GROUPING

AIM:

To get introduced to

-UNION	- JOIN
-INTERSECTION	- NESTED QUERIES
-MINUS	- GROUP BY & HAVING

QUESTION:

Amazon is one of the largest online stores operating in the United States of America. They are maintaining four tables in their database. The Items table, Customers table, Orders table and Delivery table. Each of these tables contains the following attributes:

Items: itemid (primary key)
Itemname(type =varchar(50))
category
Price
Instock (type=int, greater than or equal to zero)

Customers: custid (primary key)
Custname
Address
state

Orders: orderid (primary key)
Itemid(refers to itemid of Items table)
Quantity (type=int)
Orderdate (type=date)

Delivery:- delivery id (primary key)
Custid (refers to custid in customers table)
Orderid (refers to ordered in orders table)

Create the above tables and populate them with appropriate data.

- items

```
hishamalip@savage: ~
amazon=# CREATE TABLE items(
                            itemid INT NOT NULL PRIMARY KEY,
                            itemname VARCHAR(50) NOT NULL,
                            category TEXT NOT NULL,
                            price INT NOT NULL,
                            instock INT CHECK(instock >= 0)
                           );
CREATE TABLE
amazon=#
```

```
hishamalip@savage: ~
amazon=# INSERT INTO items VALUES(100, 'realme 3', 'mobile', 8999, 75);
INSERT 0 1
amazon=# INSERT INTO items VALUES(101, 'sony hd tv', 'tv', 75999, 15);
INSERT 0 1
amazon=# INSERT INTO items VALUES(102, 'one plus 7', 'mobile', 35000, 20),
                                  (103, 'the alchemist', 'book', 250, 100),
                                  (104, 'allen solly', 'shirt', 1139, 35);
INSERT 0 3
amazon=# INSERT INTO items VALUES(105, 'samsung m10', 'mobile', 9999, 100);
INSERT 0 1
amazon=# SELECT * FROM items;
 itemid |   itemname    | category | price | instock
--------+---------------+----------+-------+---------
    100 | realme 3      | mobile   |  8999 |      75
    101 | sony hd tv    | tv       | 75999 |      15
    102 | one plus 7    | mobile   | 35000 |      20
    103 | the alchemist | book     |   250 |     100
    104 | allen solly   | shirt    |  1139 |      35
    105 | samsung m10   | mobile   |  9999 |     100
(6 rows)

amazon=#
```

- customers

```
hishamalip@savage: ~

amazon=# CREATE TABLE customers(
                                custid INT NOT NULL PRIMARY KEY,
                                custname VARCHAR(20) NOT NULL,
                                address VARCHAR(50),
                                state VARCHAR(20) NOT NULL
                               );
CREATE TABLE
amazon=#
```

```
hishamalip@savage: ~

amazon=# INSERT INTO customers VALUES(1000, 'hisham', 'calicut', 'kerala'),
                                     (1001, 'sharuk', 'new delhi', 'delhi'),
                                     (1002, 'vijay', 'chennai', 'tamilnadu'),
                                     (1003, 'alia', 'mumbai', 'maharashtra'),
                                     (1004, 'dhoni', 'ranchi', 'jharkhand'),
                                     (1005, 'sourav', 'trivandrum', 'kerala');
INSERT 0 6
amazon=# SELECT * FROM customers ;
 custid | custname |  address   |    state
--------+----------+------------+-------------
   1000 | hisham   | calicut    | kerala
   1001 | sharuk   | new delhi  | delhi
   1002 | vijay    | chennai    | tamilnadu
   1003 | alia     | mumbai     | maharashtra
   1004 | dhoni    | ranchi     | jharkhand
   1005 | sourav   | trivandrum | kerala
(6 rows)

amazon=#
```

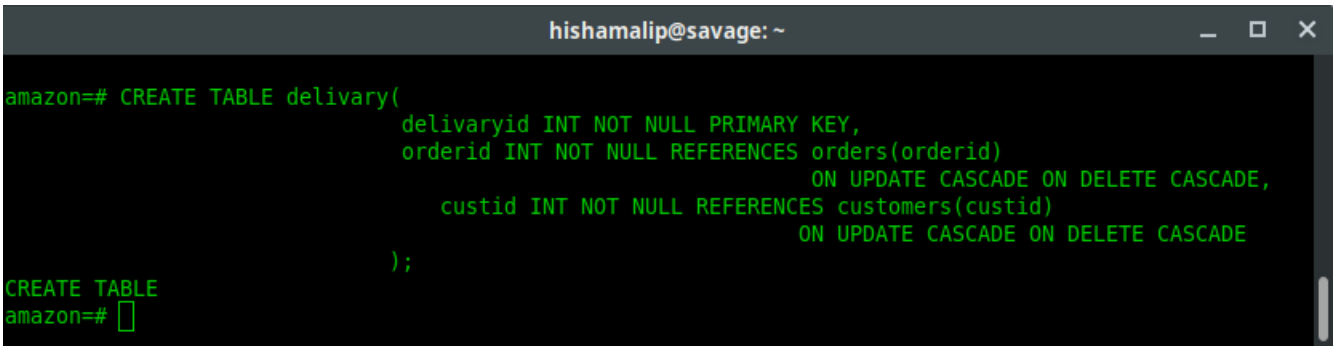
- orders

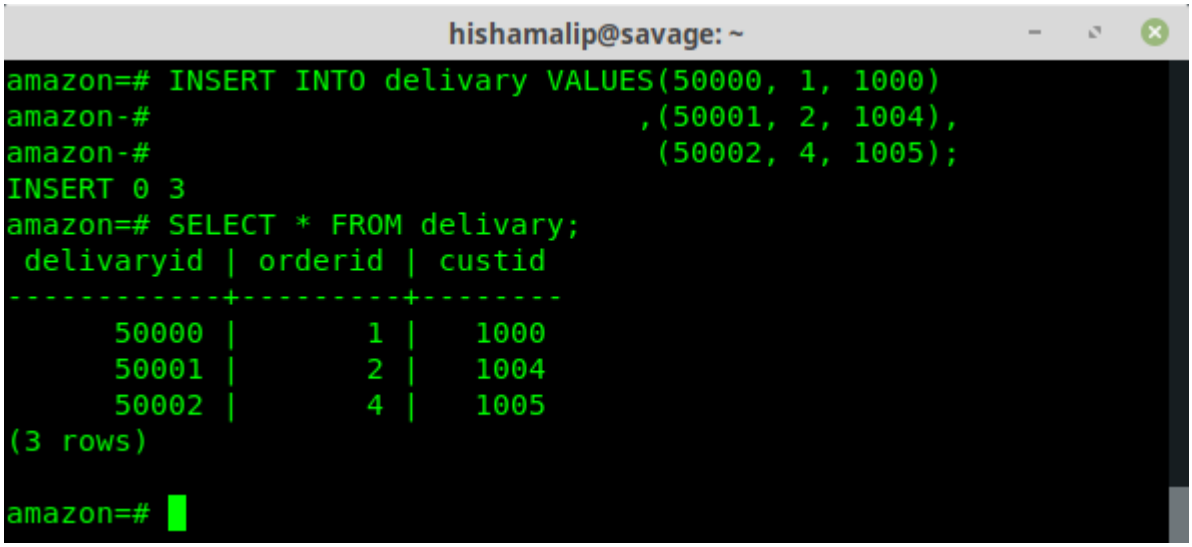
```
hishamalip@savage: ~
amazon=# CREATE TABLE orders(
                         orderid INT NOT NULL PRIMARY KEY,
                         itemid INT NOT NULL REFERENCES items(itemid)
                                                ON UPDATE CASCADE ON DELETE CASCADE,
                         quandity INT NOT NULL,
                         orderdate DATE,
                         custid INT NOT NULL REFERENCES customers(custid)
                                                ON UPDATE CASCADE ON DELETE CASCADE
                        );
CREATE TABLE
amazon=#
```

```
hishamalip@savage: ~
amazon=# INSERT INTO orders VALUES(1, 100, 3, '2019-08-13', 1000),
amazon-#                          (2, 101, 1, '2017-11-20', 1004),
amazon-#                          (3, 104, 2, '2018-10-27', 1001),
amazon-#                          (4, 103, 5, '2019-07-03', 1005);
INSERT 0 4
amazon=# SELECT * FROM orders;
 orderid | itemid | quandity | orderdate  | custid
---------+--------+----------+------------+--------
       1 |    100 |        3 | 2019-08-13 |   1000
       2 |    101 |        1 | 2017-11-20 |   1004
       3 |    104 |        2 | 2018-10-27 |   1001
       4 |    103 |        5 | 2019-07-03 |   1005
(4 rows)

amazon=#
```

- delivary


hishamalip@savage: ~
amazon=# CREATE TABLE delivary(
delivaryid INT NOT NULL PRIMARY KEY,
orderid INT NOT NULL REFERENCES orders(orderid)
ON UPDATE CASCADE ON DELETE CASCADE,
custid INT NOT NULL REFERENCES customers(custid)
ON UPDATE CASCADE ON DELETE CASCADE
);
CREATE TABLE
amazon=#


hishamalip@savage: ~
amazon=# INSERT INTO delivary VALUES(50000, 1, 1000)
amazon-# ,(50001, 2, 1004),
amazon-# (50002, 4, 1005);
INSERT 0 3
amazon=# SELECT * FROM delivary;
delivaryid | orderid | custid
50000 | 1 | 1000
50001 | 2 | 1004
50002 | 4 | 1005
(3 rows)
amazon=#

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

```
hishamalip@savage: ~
amazon=# SELECT customers.custid, custname, address, state
         FROM customers, orders
         WHERE customers.custid = orders.custid;
 custid | custname |  address   |   state
--------+----------+------------+-----------
   1000 | hisham   | calicut    | kerala
   1004 | dhoni    | ranchi     | jharkhand
   1001 | sharuk   | new delhi  | delhi
   1005 | sourav   | trivandrum | kerala
(4 rows)

amazon=#
```

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

```
hishamalip@savage: ~
amazon=# SELECT customers.custid, custname, address, state
         FROM customers, delivary
         WHERE customers.custid = delivary.custid;
 custid | custname |  address   |   state
--------+----------+------------+-----------
   1000 | hisham   | calicut    | kerala
   1004 | dhoni    | ranchi     | jharkhand
   1005 | sourav   | trivandrum | kerala
(3 rows)

amazon=#
```

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

```
hishamalip@savage: ~
amazon=# SELECT orderdate
amazon-# FROM orders, customers
amazon-# WHERE customers.custname LIKE 's%'
amazon-#   AND orders.custid = customers.custid;
 orderdate
------------
 2018-10-27
 2019-07-03
(2 rows)

amazon=#
```

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

```
hishamalip@savage: ~

amazon=# SELECT itemname, price
amazon-# FROM items, orders, customers
amazon-# WHERE items.itemid = orders.itemid
amazon-#   AND customers.custid = orders.custid
amazon-#   AND customers.custname = 'hisham';
 itemname | price
----------+-------
 realme 3 |  8999
(1 row)

amazon=#
```

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

```
hishamalip@savage: ~

amazon=# SELECT cus.*
         FROM customers AS cus, orders
         WHERE orders.custid = cus.custid
               AND orderdate >= '2013-01-01'
               AND cus.custid NOT IN(SELECT custid FROM delivary);
 custid | custname |  address  | state
--------+----------+-----------+-------
   1001 | sharuk   | new delhi | delhi
(1 row)

amazon=#
```

6.Find the itemid of items which has either been ordered or not delivered. (Use SET UNION)

```
hishamalip@savage: ~

amazon=# (SELECT i.itemid FROM items AS i, orders AS o WHERE i.itemid = o.itemid)
         UNION
         (SELECT i.itemid FROM items AS i, orders AS o WHERE i.itemid = o.itemid
                                                       AND o.orderid NOT IN(SELECT orderid FROM delivary)
amazon(#                                                                   );
 itemid
--------
    100
    101
    103
    104
(4 rows)

amazon=#
```

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

```
hishamalip@savage: ~
amazon=# (SELECT custname FROM customers AS c, orders AS o WHERE o.custid = c.custid)
amazon-# INTERSECT
amazon-# (SELECT custname FROM customers AS C, delivary AS d WHERE d.custid = c.custid);
 custname
----------
 sourav
 dhoni
 hisham
(3 rows)

amazon=#
```

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

```
hishamalip@savage: ~
amazon=# (SELECT custname FROM customers AS c, orders AS o WHERE o.custid = c.custid)
         EXCEPT
         (SELECT custname FROM customers as c, delivary AS d WHERE d.custid = c.custid);
 custname
----------
 sharuk
(1 row)

amazon=#
```

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

```
hishamalip@savage: ~
WHERE custid = (SELECT custid
                FROM orders
                GROUP BY custid
                ORDER BY count(*) DESC LIMIT 1);
 custid | custname | address |   state
--------+----------+---------+-----------
   1004 | dhoni    | ranchi  | jharkhand
(1 row)

amazon=#
```

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

```
hishamalip@savage: ~
amazon=# SELECT DISTINCT(c.*)
         FROM customers AS c, items as i, orders as o
         WHERE o.itemid = i.itemid
               AND c.custid = o.custid
               AND price > 5000;
 custid | custname | address |   state
--------+----------+---------+-----------
   1004 | dhoni    | ranchi  | jharkhand
   1000 | hisham   | calicut | kerala
(2 rows)

amazon=#
```

11. Find the name and address of customers who has not ordered a ‘sony hd tv’

```
hishamalip@savage: ~
amazon=# (SELECT custname, address FROM customers)
         EXCEPT
         (SELECT c.custname, c.address
          FROM customers AS c, orders AS o, items AS i
          WHERE o.itemid = i.itemid
                AND c.custid = o.custid
                AND itemname = 'sony hd tv');
 custname |  address
----------+-----------
 sharuk   | new delhi
 hisham   | calicut
 vijay    | chennai
 sourav   | trivandrum
 alia     | mumbai
(5 rows)

amazon=#
```

12. Perform Left Outer Join and Right Outer Join on Customers & Orders Table.

```
hishamalip@savage: ~
amazon=# SELECT * FROM customers
amazon-# LEFT OUTER JOIN orders
amazon-# ON customers.custid = orders.custid;
 custid | custname |  address   |    state    | orderid | itemid | quandity | orderdate  | custid
--------+----------+------------+-------------+---------+--------+----------+------------+--------
   1000 | hisham   | calicut    | kerala      |       1 |    100 |        3 | 2019-08-13 |   1000
   1004 | dhoni    | ranchi     | jharkhand   |       2 |    101 |        1 | 2017-11-20 |   1004
   1001 | sharuk   | new delhi  | delhi       |       3 |    104 |        2 | 2018-10-27 |   1001
   1005 | sourav   | trivandrum | kerala      |       4 |    103 |        5 | 2019-07-03 |   1005
   1003 | alia     | mumbai     | maharashtra |         |        |          |            |
   1002 | vijay    | chennai    | tamilnadu   |         |        |          |            |
(6 rows)

amazon=#
```

13. Find the details of all customers grouped by state

```
hishamalip@savage: ~
amazon=# SELECT COUNT(*), state FROM customers GROUP BY state;
 count |    state
-------+-------------
     1 | jharkhand
     1 | delhi
     1 | maharashtra
     2 | kerala
     1 | tamilnadu
(5 rows)

amazon=#
```

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

```
hishamalip@savage: ~

amazon=# SELECT * FROM items
amazon-# GROUP BY category, itemid
amazon-# HAVING price >= (SELECT AVG(price) FROM items);
 itemid |  itemname   | category | price | instock
--------+------------+----------+-------+---------
    101 | sony hd tv | tv       | 75999 |      15
    102 | one plus 7 | mobile   | 35000 |      20
(2 rows)

amazon=#
```

RESULT: The given set of queries are executed and output is obtained.