# KIET GROUP OF INSTITUTIONS

## DEPARTMENT OF COMPUTER APPLICATIONS

### LAB ASSIGNMENT 8

### DBMS Lab (KCA – 252)

### Assignments on Join

Prassan Varshney Roll_Number - 36

**Note – Questions from 1 to 14 refer the sample tables Salesman, Customer, Order.**

***Sample table*: salesman**

| salesman_id | name | city | commission |
|---|---|---|---|
| 5001 | James Hoog | New York | 0.15 |
| 5002 | Nail Knite | Paris | 0.13 |
| 5005 | Pit Alex | London | 0.11 |
| 5006 | Mc Lyon | Paris | 0.14 |
| 5007 | Paul Adam | Rome | 0.13 |
| 5003 | Lauson Hen | San Jose | 0.12 |

***Sample table*: customer**

| customer_id | cust_name | city | grade | salesman_id |
|---|---|---|---|---|
| 3002 | Nick Rimando | New York | 100 | 5001 |
| 3007 | Brad Davis | New York | 200 | 5001 |
| 3005 | Graham Zusi | California | 200 | 5002 |
| 3008 | Julian Green | London | 300 | 5002 |
| 3004 | Fabian Johnson | Paris | 300 | 5006 |
| 3009 | Geoff Cameron | Berlin | 100 | 5003 |
| 3003 | Jozy Altidor | Moscow | 200 | 5007 |
| 3001 | Brad Guzan | London | | 5005 |

***Sample table*: orders**

| ord_no | purch_amt | ord_date | customer_id | salesman_id |
|---|---|---|---|---|
| 70001 | 150.5 | 2012-10-05 | 3005 | 5002 |
| 70009 | 270.65 | 2012-09-10 | 3001 | 5005 |
| 70002 | 65.26 | 2012-10-05 | 3002 | 5001 |
| 70004 | 110.5 | 2012-08-17 | 3009 | 5003 |
| 70007 | 948.5 | 2012-09-10 | 3005 | 5002 |
| 70005 | 2400.6 | 2012-07-27 | 3007 | 5001 |
| 70008 | 5760 | 2012-09-10 | 3002 | 5001 |
| 70010 | 1983.43 | 2012-10-10 | 3004 | 5006 |
| 70003 | 2480.4 | 2012-10-10 | 3009 | 5003 |
| 70012 | 250.45 | 2012-06-27 | 3008 | 5002 |
| 70011 | 75.29 | 2012-08-17 | 3003 | 5007 |
| 70013 | 3045.6 | 2012-04-25 | 3002 | 5001 |

1. Write a SQL statement to prepare a list with salesman name, customer name and their cities for the salesmen and customer who belongs to the same city.

   ```
   SELECT s.name AS Salesman_Name, c.cust_name AS Customer_Name, s.city
   FROM salesman s
   JOIN customer c ON s.city = c.city;
   ```
2. Write a SQL statement to make a list with order no, purchase amount, customer name and their cities for those orders which order amount between 500 and 2000.
   ```
   SELECT o.ord_no, o.purch_amt, c.cust_name, c.city
   FROM orders o
   JOIN customer c ON o.customer_id = c.customer_id
   WHERE o.purch_amt BETWEEN 500 AND 2000;
   ```
3. Write a SQL statement to know which salesman are working for which customer.
   ```
   SELECT s.name AS Salesman_Name, c.cust_name AS Customer_Name
   FROM salesman s
   JOIN customer c ON s.salesman_id = c.salesman_id;
   ```
4. Write a SQL statement to find the list of customers who appointed a salesman for their jobs who gets a commission from the company is more than 12%.
   ```
   SELECT c.cust_name, s.name AS Salesman_Name
   FROM customer c
   JOIN salesman s ON c.salesman_id = s.salesman_id
   WHERE s.commission > 0.12;
   ```
5. Write a SQL statement to find the list of customers who appointed a salesman for their jobs who does not live in the same city where their customer lives, and gets a commission is above 12%.
   ```
   SELECT c.cust_name, s.name AS Salesman_Name, c.city AS Customer_City, s.city AS Salesman_City
   FROM customer c
   JOIN salesman s ON c.salesman_id = s.salesman_id
   WHERE s.commission > 0.12 AND c.city <> s.city;
   ```
6. Write a SQL statement to find the details of a order i.e. order number, order date, amount of order, which customer gives the order and which salesman works for that customer and how much commission he gets for an order.
   ```
   SELECT o.ord_no, o.ord_date, o.purch_amt, c.cust_name, s.name AS Salesman_Name, s.commission
   FROM orders o
   JOIN customer c ON o.customer_id = c.customer_id
   JOIN salesman s ON o.salesman_id = s.salesman_id;
   ```
7. Write a SQL statement to make a list in ascending order for the customer who works either through a salesman or by own.
   ```
   SELECT c.*
   FROM customer c
   ORDER BY c.cust_name ASC;
   ```
8. Write a SQL statement to make a list in ascending order for the customer who holds a grade less than 300 and works either through a salesman or by own.
   ```
   SELECT c.cust_name, c.city, o.ord_no, o.ord_date, o.purch_amt
   FROM customer c
   LEFT JOIN orders o ON c.customer_id = o.customer_id
   ```

```
ORDER BY o.ord_date ASC;
```

9. Write a SQL statement to make a report with customer name, city, order number, order date, and order amount in ascending order according to the order date to find that either any of the existing customers have placed no order or placed one or more orders.

```
SELECT c.cust_name, c.city, o.ord_no, o.ord_date, o.purch_amt
FROM customer c
LEFT JOIN orders o ON c.customer_id = o.customer_id
ORDER BY o.ord_date ASC;
```

10. Write a SQL statement to make a report with customer name, city, order number, order date, order amount salesman name and commission to find that either any of the existing customers have placed no order or placed one or more orders by their salesman or by own.

```
SELECT c.cust_name, c.city, o.ord_no, o.ord_date, o.purch_amt, s.name AS Salesman_Name, s.commission
FROM customer c
LEFT JOIN orders o ON c.customer_id = o.customer_id
LEFT JOIN salesman s ON c.salesman_id = s.salesman_id
ORDER BY o.ord_date ASC;
```

11. Write a SQL statement to make a list in ascending order for the salesmen who works either for one or more customer or not yet join under any of the customers.

```
SELECT s.*
FROM salesman s
LEFT JOIN customer c ON s.salesman_id = c.salesman_id
GROUP BY s.salesman_id, s.name, s.city, s.commission
ORDER BY s.name ASC;
```

12. Write a SQL statement to make a list for the salesmen who works either for one or more customer or not yet join under any of the customers who placed either one or more orders or no order to their supplier.

```
SELECT s.*
FROM salesman s
LEFT JOIN customer c ON s.salesman_id = c.salesman_id
LEFT JOIN orders o ON c.customer_id = o.customer_id
GROUP BY s.salesman_id, s.name, s.city, s.commission
ORDER BY s.name ASC;
```

13. Write a SQL statement to make a list for the salesmen who either work for one or more customers or yet to join any of the customer. The customer may have placed, either one or more orders on or above order amount 2000 and must have a grade, or he may not have placed any order to the associated supplier.

```
SELECT s.*
FROM salesman s
LEFT JOIN customer c ON s.salesman_id = c.salesman_id
LEFT JOIN orders o ON c.customer_id = o.customer_id AND o.purch_amt >= 2000
WHERE c.grade IS NOT NULL OR o.ord_no IS NULL
GROUP BY s.salesman_id, s.name, s.city, s.commission
ORDER BY s.name ASC;
```

14. Write a SQL statement to make a cartesian product between salesman and customer i.e. each salesman will appear for all customer and vice versa.

```
SELECT s.*, c.*
FROM salesman s
CROSS JOIN customer c;
```

**Note – For questions 15 to 19 use sample table company_mast and item_mast**

*Sample table*: company_mast

| COM_ID | COM_NAME |
|---|---|
| 11 | Samsung |
| 12 | iBall |
| 13 | Epsion |
| 14 | Zebronics |
| 15 | Asus |
| 16 | Frontech |

*Sample table*: item_mast

| PRO_ID | PRO_NAME | PRO_PRICE | PRO_COM |
|---|---|---|---|
| 101 | Mother Board | 3200 | 15 |
| 102 | Key Board | 450 | 16 |
| 103 | ZIP drive | 250 | 14 |
| 104 | Speaker | 550 | 16 |
| 105 | Monitor | 5000 | 11 |
| 106 | DVD drive | 900 | 12 |
| 107 | CD drive | 800 | 12 |
| 108 | Printer | 2600 | 13 |
| 109 | Refill cartridge | 350 | 13 |
| 110 | Mouse | 250 | 12 |

15. Write a SQL query to display all the data from the item_mast, including all the data for each item's producer company.

```
SELECT s.*, c.*
FROM salesman s
CROSS JOIN customer c;
```

16. Write a SQL query to display the item name, price, and company name of all the products.

```
SELECT i.pro_name, i.pro_price, c.com_name
FROM item_mast i
JOIN company_mast c ON i.pro_com = c.com_id;
```

17. Write a SQL query to display the average price of items of each company, showing the name of the company.

```
SELECT c.com_name, AVG(i.pro_price) AS Avg_Price
FROM item_mast i
JOIN company_mast c ON i.pro_com = c.com_id
GROUP BY c.com_name;
```

18. Write a SQL query to display the names of the company whose products have an average price larger than or equal to Rs. 350.

```
SELECT c.com_name
FROM item_mast i
JOIN company_mast c ON i.pro_com = c.com_id
GROUP BY c.com_name
HAVING AVG(i.pro_price) >= 350;
```

19. Write a SQL query to display the name of each company along with the ID and price for their most expensive product.

```
SELECT c.com_name, i.pro_id, i.pro_price
FROM item_mast i
JOIN company_mast c ON i.pro_com = c.com_id
WHERE i.pro_price = (SELECT MAX(pro_price) FROM item_mast WHERE pro_com = c.com_id);
```

Note – For questions 20 to 23 use the sample tables emp_deptmen, emp_details.

*Sample table*: emp_departmen

| DPT_CODE | DPT_NAME | DPT_ALLOTMENT |
|---|---|---|
| 57 | IT | 65000 |
| 63 | Finance | 15000 |
| 47 | HR | 240000 |
| 27 | RD | 55000 |
| 89 | QC | 75000 |

*Sample table*: emp_details

| EMP_IDNO | EMP_FNAME | EMP_LNAME | EMP_DEPT |
|---|---|---|---|
| 127323 | Michale | Robbin | 57 |
| 526689 | Carlos | Snares | 63 |
| 843795 | Enric | Dosio | 57 |
| 328717 | Jhon | Snares | 63 |
| 444527 | Joseph | Dosni | 47 |
| 659831 | Zanifer | Emily | 47 |
| 847674 | Kuleswar | Sitaraman | 57 |
| 748681 | Henrey | Gabriel | 47 |
| 555935 | Alex | Manuel | 57 |
| 539569 | George | Mardy | 27 |
| 733843 | Mario | Saule | 63 |
| 631548 | Alan | Snappy | 27 |
| 839139 | Maria | Foster | 57 |

20. Write a query in SQL to display all the data of employees including their department.

```
SELECT e.*, d.*
FROM emp_details e
JOIN emp_departmen d ON e.emp_dept = d.dpt_code;
```

21. Write a query in SQL to display the first name and last name of each employee, along with the name and sanction amount for their department.

```
SELECT e.emp_fname, e.emp_lname, d.dpt_name, d.dpt_allotment
FROM emp_details e
JOIN emp_departmen d ON e.emp_dept = d.dpt_code;
```

22. Write a query in SQL to find the first name and last name of employees working for departments with a budget more than Rs. 50000.

```
SELECT e.emp_fname, e.emp_lname
FROM emp_details e
```

```
JOIN emp_departmen d ON e.emp_dept = d.dpt_code
WHERE d.dpt_allotment > 50000;
```

23. Write a query in SQL to find the names of departments where more than two employees are working

```
SELECT d.dpt_name
FROM emp_departmen d
JOIN emp_details e ON e.emp_dept = d.dpt_code
GROUP BY d.dpt_name
HAVING COUNT(e.emp_idno) > 2;
```