```
                        /* Non local orders */

-- Creating salesman table & inserting data into the table

CREATE TABLE salesman
(
      Id INTEGER,             -- Id of integer datatype
      Name VARCHAR(10),       -- Name of string datatype
      City VARCHAR(10),       -- City of string datatype
      Commission INTEGER,           -- Commission of integer datatype
      PRIMARY KEY (Id)        -- Represents uniue Id of salesman
);

INSERT INTO salesman VALUES (101,'Jmaes','NewYork',150);
INSERT INTO salesman VALUES (102,'Nail','Paris',130);
INSERT INTO salesman VALUES (103,'Alex','London',110);
INSERT INTO salesman VALUES (104,'Mcoy','Paris',140);
INSERT INTO salesman VALUES (105,'Paul','Rome',130);
INSERT INTO salesman VALUES (106,'Lauson','SanJose',120);

-- Checking for table salesman in DB and verifying Attributes & Records

SELECT * FROM salesman;

-- Creating customer table & inserting data into the table

CREATE TABLE customer
(
      Id INTEGER,                   -- Id of integer datatype
      Name VARCHAR(10),             -- Name of string datatype
      City VARCHAR(10),             -- City of string datatype
      Grade INTEGER,                      -- Grade of integer datatype
      salesman_id INTEGER,                -- salesman id of integer datatype
      PRIMARY KEY(Id),              -- Represents uniue Id of Customer
      FOREIGN KEY(salesman_id)            -- Child table
            REFERENCES salesman(Id)       -- parent table
);

INSERT INTO customer VALUES (1,'Nick','NewYork',100,101);
INSERT INTO customer VALUES (2,'Brad','NewYork',200,102);
INSERT INTO customer VALUES (3,'Graham','California',200,103);
INSERT INTO customer VALUES (4,'Julie','London',300,104);
INSERT INTO customer VALUES (5,'Fabian','Paris',300,105);
INSERT INTO customer VALUES (6,'Geoff','Berlin',300,106);
INSERT INTO customer VALUES (7,'Jos','Moscow',100,101);
INSERT INTO customer VALUES (8,'Guzan','London',200,102);


-- Checking for table customer in DB and verifying Attributes & Records

SELECT * FROM customer;

-- Creating orders table & inserting data into the table

CREATE TABLE orders
(
      Order_no INTEGER,                   -- Id of integer datatype
      Purchase_Amount INTEGER,                  -- Name of string datatype
      Order_Date DATE,                    -- City of string datatype
```

```
      Customer_id INTEGER,                    -- Grade of integer datatype
      salesman_id INTEGER,                    -- salesman id of integer datatype
      FOREIGN KEY(salesman_id) REFERENCES salesman(Id),
      FOREIGN KEY(Customer_id) REFERENCES customer(Id)
);

INSERT INTO orders VALUES(7001,250,'2012-10-05',5,105);
INSERT INTO orders VALUES(7009,270,'2012-12-10',1,105);
INSERT INTO orders VALUES(7002,60,'2012-10-05',5,105);
INSERT INTO orders VALUES(7004,110,'2012-08-17',2,105);
INSERT INTO orders VALUES(7007,950,'2012-09-10',3,105);
INSERT INTO orders VALUES(7005,2450,'2012-07-27',8,105);
INSERT INTO orders VALUES(7008,5700,'2012-09-10',4,105);
INSERT INTO orders VALUES(7003,1980,'2012-10-10',7,105);
INSERT INTO orders VALUES(7006,75,'2012-10-10',6,105);
INSERT INTO orders VALUES(7010,3045,'2012-06-27',1,105);

-- Checking for table ordes in DB and verifying Attributes & Records

SELECT * FROM orders;

                -- Joins concept is reuired to solve this problem

                           /* query to fetch the required result*/


SELECT
      o.order_no, c.name, o.customer_id, s.id  -- required fields from tables
FROM
      salesman s                    -- Inner Join & alliasing
      INNER JOIN customer c
      ON c.salesman_id = s.id
      INNER JOIN orders o
      ON o.customer_id = c.id             -- Appropriate relations on common fields
WHERE
      s.city != c.city              -- for non local orders
ORDER BY
      order_no                            -- sorting the orders

      ------------------------------------------------------------------------
------------------------------
                        /* Employee Incentive calculation */

-- Creating employee table & inserting data into the table

CREATE TABLE employee
(
      employee_id INTEGER,          -- employee_id of integer datatype
      Name VARCHAR(10),       -- Name of string datatype
      Pos_id VARCHAR(3),            -- Position of employee of string datattype
      Emp_sale INTEGER        -- Sales of integer datatype
);

INSERT INTO employee VALUES (201,'Jackson','P61',456);
INSERT INTO employee VALUES (202,'Nick','P62',267);
INSERT INTO employee VALUES (203,'Joffrey','P62',324);
INSERT INTO employee VALUES (204,'Misty','P63',154);
INSERT INTO employee VALUES (205,'Jack','P61',532);
```

```
-- Checking for table employee in DB and verifying Attributes & Records

SELECT * FROM employee;

-- Creating incentive_details table & inserting data into the table

CREATE TABLE incentive_details
(
      P_id VARCHAR(3),              -- uniue position of string datatype
      Position_name VARCHAR(5),           -- position name of string datatype
      Sales_milestone INTEGER,            -- sales data of integer datatype
      Incentive INTEGER,                  -- incentive of integer datatype
      Cap INTEGER                   -- Maximum limit of incentive of integer type
);

INSERT INTO incentive_details VALUES ('P61','GM',120,15000,50000);
INSERT INTO incentive_details VALUES ('P62','AM',85,8000,23500);
INSERT INTO incentive_details VALUES ('P63','Staff',45,5000,13000);

-- Checking for table incentive_details in DB and verifying Attributes & Records

SELECT * FROM incentive_details;

                              /* query for employee incentive made */


SELECT
      e.employee_id,
CASE
      WHEN ((ROUND(e.emp_sale/i.sales_milestone))*i.incentive) < i.cap
      THEN ((ROUND(e.emp_sale/i.sales_milestone))*i.incentive)
      WHEN ((ROUND(e.emp_sale/i.sales_milestone))*i.incentive) > i.cap
      THEN i.cap
END incentive_made      -- use of CASE clause to categorise the incentive made

FROM
      employee e INNER JOIN incentive_details i
      ON e.pos_id = i.p_id
ORDER BY
      e.employee_id
      ------------------------------------------------------------------------------
------------------------------

                              /* Research papers in institute */
-- Creating researchers table and inserting data

CREATE TABLE researchers
(
      r_id CHAR(4),
      r_name VARCHAR(5),
      r_gender CHAR(1)
);

INSERT INTO researchers VALUES ('R001','Marry','F');
INSERT INTO researchers VALUES ('R002','Alice','F');
INSERT INTO researchers VALUES ('R003','Bob','M');
INSERT INTO researchers VALUES ('R004','Ben','M');
INSERT INTO researchers VALUES ('R005','Mike','M');
INSERT INTO researchers VALUES ('R006','Tara','F');
```

```
INSERT INTO researchers VALUES ('R007','Anita','F');
INSERT INTO researchers VALUES ('R008','Lisa','F');
INSERT INTO researchers VALUES ('R009','Missy','F');
INSERT INTO researchers VALUES ('R010','Johm','M');

-- Checking for table researchers in DB and verifying Attributes & Records

SELECT * FROM researchers

-- Creating mentors table and inserting data

CREATE TABLE mentors
(
      m_id CHAR(4),
      m_name VARCHAR(7),
      m_gender CHAR(1)
);

INSERT INTO mentors VALUES ('M001','Rob','M');
INSERT INTO mentors VALUES ('M002','Jessica','F');
INSERT INTO mentors VALUES ('M003','Rachel','F');
INSERT INTO mentors VALUES ('M004','Joey','M');
INSERT INTO mentors VALUES ('M005','Monica','F');

-- Checking for table mentors in DB and verifying Attributes & Records

SELECT * FROM mentors;

-- Creating papers table and inserting data

CREATE TABLE papers
(
      p_id CHAR(4),
      p_subject VARCHAR(20)
);

INSERT INTO papers VALUES ('P001','Microbiology');
INSERT INTO papers VALUES ('P002','Theoretical Physics');
INSERT INTO papers VALUES ('P003','Theoretical Physics');
INSERT INTO papers VALUES ('P004','Microbiology');
INSERT INTO papers VALUES ('P005','Microbiology');

-- Checking for table papers in DB and verifying Attributes & Records

SELECT * FROM papers;

-- creating table research_paper & inserting data

CREATE TABLE research_paper
(
      r_id CHAR(4),
      p_id CHAR(4)
);

INSERT INTO research_paper VALUES ('R001','P001');
INSERT INTO research_paper VALUES ('R002','P002');
INSERT INTO research_paper VALUES ('R003','P003');
INSERT INTO research_paper VALUES ('R004','P004');
INSERT INTO research_paper VALUES ('R005','P005');
```

```
INSERT INTO research_paper VALUES ('R006','P006');
INSERT INTO research_paper VALUES ('R007','P007');
INSERT INTO research_paper VALUES ('R008','P008');
INSERT INTO research_paper VALUES ('R009','P009');
INSERT INTO research_paper VALUES ('R010','P010');

-- Checking for table research_paper in DB and verifying Attributes & Records

SELECT * FROM research_paper;

-- creating table research_mentor & inserting data

CREATE TABLE research_mentor
(
      r_id CHAR(4),
      m_id CHAR(4)
);

INSERT INTO research_mentor VALUES ('R001','M005');
INSERT INTO research_mentor VALUES ('R002','M004');
INSERT INTO research_mentor VALUES ('R003','M003');
INSERT INTO research_mentor VALUES ('R004','M002');
INSERT INTO research_mentor VALUES ('R005','M001');
INSERT INTO research_mentor VALUES ('R006','M005');
INSERT INTO research_mentor VALUES ('R007','M004');
INSERT INTO research_mentor VALUES ('R008','M003');
INSERT INTO research_mentor VALUES ('R009','M002');
INSERT INTO research_mentor VALUES ('R010','M001');

-- Checking for table research_mentor in DB and verifying Attributes & Records

SELECT * FROM research_mentor;

                  /* Query to fetch results for assigned task */

/* Query to fetch results for assigned task */

SELECT
      p_subject
FROM
      (SELECT
            m_gender, p_subject, COUNT(*)
      FROM
            mentors me INNER JOIN research_mentor rm  -- Inner join of tables
            ON me.m_id = rm.m_id
            INNER JOIN researchers r
            ON r.r_id = rm.r_id
            INNER JOIN research_paper rp
            ON rp.r_id = r.r_id
            INNER JOIN papers p
            ON p.p_id = rp.p_id
       WHERE
            m_gender = 'F'          -- filter to get female mentors
       AND
            p_subject LIKE '%b%'    -- subject with b letter
      GROUP BY m_gender, p_subject) a

      /* research paper task by another query */
-- fetching the result by using windows function in the query
```

```
SELECT
      p_subject
FROM
    (SELECT *, MAX(ct) OVER (PARTITION BY m_gender,P_subject) mc
    FROM
        (SELECT
            m_gender, p_subject, COUNT(*) AS ct
        FROM
            mentors me INNER JOIN research_mentor rm  -- Inner join of tables
            ON me.m_id = rm.m_id
            INNER JOIN researchers r
            ON r.r_id = rm.r_id
            INNER JOIN research_paper rp
            ON rp.r_id = r.r_id
            INNER JOIN papers p
            ON p.p_id = rp.p_id
        WHERE
            p_subject LIKE '%b%'    -- subject with b letter
        GROUP BY m_gender, p_subject) a
        ) b
    WHERE
        m_gender = 'F'        -- filtering gender
    AND
        ct = mc         -- filtering the paper
```

VARMA PRASAD S