Task 3- Data Retrieval Language, SELECT from several logical linked tables

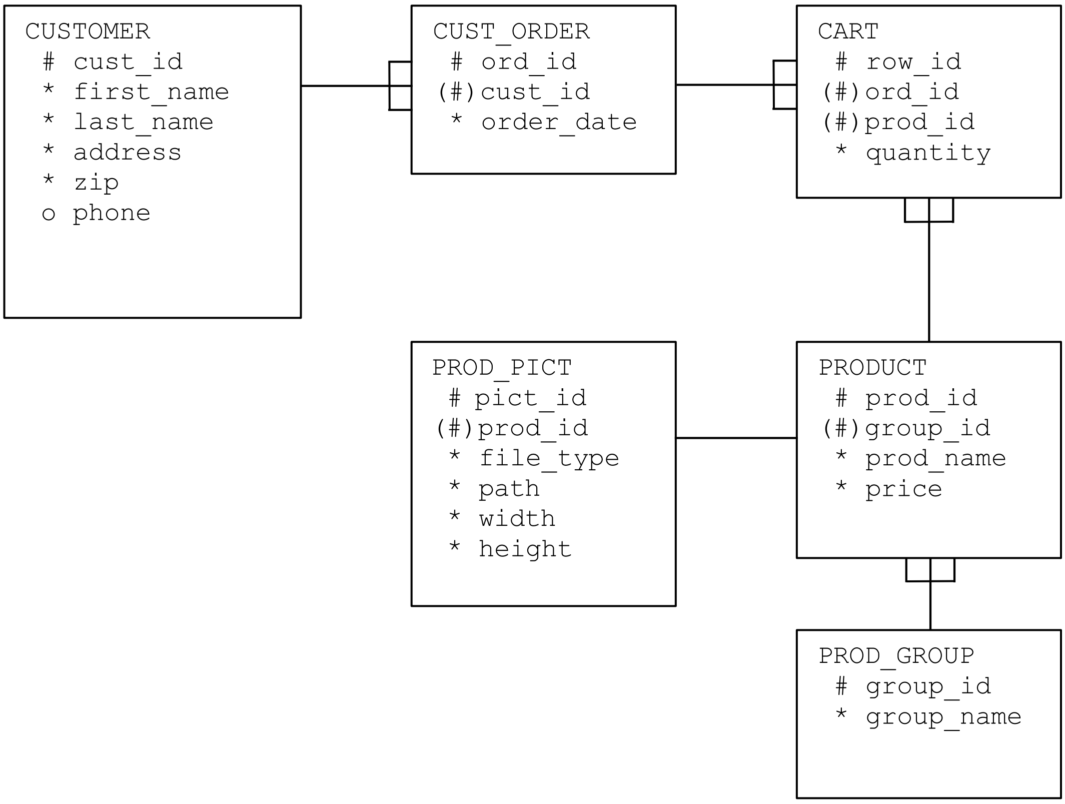
In this Task you will work with SELECT statements against several logical linked tables.

Create the tables for this lab

```
------------------------Create tables START COPY------------------------------------------------------------
CREATE TABLE customer(
cust_id NUMBER(6) PRIMARY KEY,
first_name VARCHAR2(20),
last_name VARCHAR2(25),
address VARCHAR2(30),
zip_code VARCHAR2(8),
city VARCHAR2(20),
area_code VARCHAR2(6),
telephone VARCHAR2(12));
CREATE TABLE cust_order(
ord_id NUMBER(9) PRIMARY KEY,
cust_id REFERENCES customer(cust_id),
order_date DATE);
CREATE TABLE prod_group(
group_id NUMBER(4) PRIMARY KEY,
group_name VARCHAR2(30));
CREATE TABLE product(
prod_id NUMBER(8) PRIMARY KEY,
group_id REFERENCES prod_group(group_id),
prod_name VARCHAR2(25),
price NUMBER(9,2));
CREATE TABLE cart(
row_id NUMBER(9) PRIMARY KEY,
ord_id REFERENCES cust_order(ord_id),
prod_id REFERENCES product(prod_id),
quantity NUMBER(6));
CREATE TABLE prod_pict(
pict_id NUMBER(9) PRIMARY KEY,
prod_id REFERENCES product(prod_id),
file_type VARCHAR2(5),
path VARCHAR2(80),
width NUMBER(4),
height NUMBER(4));
------------------------Create tables END COPY--------------------------------------------------------------
```

Now we have a table structure representing some kind of sales activities. If you look at the next page you will see a data model of the tables with primary- and foreign keys.

Data model



Explanation of notation

```
 #  = Primary key
(#) = Foreign key
 *  = Mandatory (must contain a value => NOT NULL)
 o  = Optional (must not contain a value can be NULL)
```

Next step is to fill the tables with data. Do that by copy and paste the following into SQL Live, and hit Run.

```
--------------------------Fill the tables with data START COPY-------------------------------------------------
INSERT INTO customer
VALUES(1,'olof','andersson','box144','79100','falun','023','225478');
INSERT INTO customer VALUES(2,'maria','andersson','storgatan
23',79123,'falun','023','445599');
INSERT INTO customer VALUES(3,'tomas','kvist','box1','54784','gagnef','0246','11122');
INSERT INTO customer VALUES(4,'hans','rosenboll','sommarvagen
36','78458','borlange','0243','228869');
INSERT INTO customer VALUES(5,'yvette','porpoix','sadelgatan
10','79100','falun','023','147858');
INSERT INTO customer
VALUES(6,'gustav','moller','box33','78547','gustafs','0243','122099');
INSERT INTO customer VALUES(7,'zoltan','habbervic','paradisvagen
12','78523','borlange','0243','45877');
INSERT INTO customer VALUES(8,'lena','larsson','sandgatan
13','73100','sater','0225','43251');
INSERT INTO customer VALUES(9,'ollas','bullas','korkhuvudvagen
1','79100','falun','023','11477');
INSERT INTO customer VALUES(10,'roger','nyberg','soldatvagen
25','79100','falun','023','225499');
INSERT INTO cust_order VALUES(100,1,TO_DATE('2001-02-14','YYYY-MM-DD'));
INSERT INTO cust_order VALUES(101,4,TO_DATE('2001-02-14','YYYY-MM-DD'));
INSERT INTO cust_order VALUES(289,4,TO_DATE('2003-03-04','YYYY-MM-DD'));
INSERT INTO cust_order VALUES(125,2,TO_DATE('2001-05-24','YYYY-MM-DD'));
INSERT INTO cust_order VALUES(147,3,TO_DATE('2001-12-11','YYYY-MM-DD'));
INSERT INTO cust_order VALUES(152,5,TO_DATE('2001-12-15','YYYY-MM-DD'));
INSERT INTO cust_order VALUES(458,6,TO_DATE('2004-05-08','YYYY-MM-DD'));
INSERT INTO cust_order VALUES(489,6,TO_DATE('2004-06-10','YYYY-MM-DD'));
INSERT INTO cust_order VALUES(324,10,TO_DATE('2003-08-22','YYYY-MM-DD'));
INSERT INTO cust_order VALUES(198,9,TO_DATE('2002-01-12','YYYY-MM-DD'));
INSERT INTO cust_order VALUES(348,1,TO_DATE('2004-07-17','YYYY-MM-DD'));
INSERT INTO prod_group VALUES(1,'beard care');
INSERT INTO prod_group VALUES(2,'hunting');
INSERT INTO prod_group VALUES(3,'farmhouse');
INSERT INTO prod_group VALUES(4,'leisure');
INSERT INTO product VALUES(1434,1,'trimmer deluxe',189.50);
INSERT INTO product VALUES(1724,1,'fungicides',198.5);
INSERT INTO product VALUES(113,2,'hatchet',795);
INSERT INTO product VALUES(1447,2,'knife',349.5);
INSERT INTO product VALUES(5896,3,'pig feed',240);
INSERT INTO product VALUES(5542,3,'potato fertilizer',128);
INSERT INTO product VALUES(1333,4,'dartboard',49.50);
INSERT INTO product VALUES(1888,4,'peasant trap',788.50);
INSERT INTO product VALUES(1141,4,'hammock',181.50);
INSERT INTO prod_pict VALUES(1,1434,'jpg','/images/1/',480,640);
INSERT INTO prod_pict VALUES(2,113,'jpg','/images/2/',480,640);
INSERT INTO prod_pict VALUES(3,5896,'jpg','/images/3/',480,640);
INSERT INTO prod_pict VALUES(4,1888,'gif','/images/4/',480,640);
INSERT INTO cart VALUES(1,100,1141,1);
INSERT INTO cart VALUES(2,101,1434,3);
INSERT INTO cart VALUES(3,101,1724,4);
INSERT INTO cart VALUES(4,289,1434,1);
INSERT INTO cart VALUES(5,289,1724,5);
INSERT INTO cart VALUES(6,125,1333,1);
INSERT INTO cart VALUES(7,125,1141,1);
INSERT INTO cart VALUES(8,147,5896,4);
INSERT INTO cart VALUES(9,147,5542,4);
INSERT INTO cart VALUES(10,152,113,2);
INSERT INTO cart VALUES(11,458,5896,3);
INSERT INTO cart VALUES(12,458,1447,1);
INSERT INTO cart VALUES(13,489,5542,3);
INSERT INTO cart VALUES(14,324,113,3);
INSERT INTO cart VALUES(15,324,1447,3);
INSERT INTO cart VALUES(16,324,1888,1);
INSERT INTO cart VALUES(17,198,1141,7);
INSERT INTO cart VALUES(18,348,113,3);
COMMIT;
-------------------------- Fill the tables with data END COPY-------------------------------------------------
```

Answer the following questions by writing appropriate SQL statements.

Task 1

Show **cust_id**, **first_name**, **last_name** and the **number** of customer orders that each customer has.

```
SELECT
    c.cust_id,
    c.first_name,
    c.last_name,
    COUNT(o.ord_id) AS number_of_orders
FROM
    customer c
    JOIN cust_order o ON c.cust_id = o.cust_id
GROUP BY
    c.cust_id,
    c.first_name,
    c.last_name
ORDER BY
    c.cust_id;
```

Correct answer:

```
CUST_ID  FIRST_NAME  LAST_NAME       NUMBER_OF_ORDERS
-------  ----------  --------------  ----------------
      1  olof        andersson                      2
      2  maria       andersson                      1
      3  tomas       kvist                          1
      4  hans        rosenboll                      2
      5  yvette      porpoix                        1
      6  gustav      moller                         2
      9  ollas       bullas                         1
     10  roger       nyberg                         1
```

Task 2

Show **cust_id**, **first_name**, **last_name** for those customers who have bought products that belong to the product groups: 'farmhouse' or 'beard care'. **Solve this task by using nested search** (i.e. using sub queries)

```
SELECT cust_id, first_name, last_name
FROM customer
WHERE cust_id IN (
    SELECT cust_id
    FROM cust_order
    WHERE ord_id IN (
        SELECT ord_id
        FROM cart
        WHERE prod_id IN (
            SELECT prod_id
            FROM product
            WHERE group_id IN (
                SELECT group_id
                FROM prod_group
                WHERE group_name IN ('farmhouse', 'beard care')
            )
        )
    )
)
ORDER BY cust_id;
```

Correct answer:

```
CUST_ID FIRST_NAME           LAST_NAME
------- -------------------- ---------
      3 tomas                kvist
      4 hans                 rosenboll
      6 gustav               moller
```

Task 3

Show **cust_id**, **first_name**, **last_name** for those customers who have bought products that belong to the product groups: 'farmhouse' and 'beard care'. **Solve this task by using join search** (i.e. using equi-join)

```
SELECT DISTINCT
    customer.cust_id,
    customer.first_name,
    customer.last_name
FROM customer
JOIN cust_order ON customer.cust_id = cust_order.cust_id
JOIN cart ON cust_order.ord_id = cart.ord_id
JOIN product ON cart.prod_id = product.prod_id
JOIN prod_group ON product.group_id = prod_group.group_id
WHERE prod_group.group_name IN ('farmhouse', 'beard care')
ORDER BY customer.cust_id;
```

Correct answer:

```
CUST_ID FIRST_NAME           LAST_NAME
------- -------------------- ---------
      3 tomas                kvist
      4 hans                 rosenboll
      6 gustav               moller
```

Task 4
Show **cust_id**, **first_name**, **last_name** and the **total order value** that customers have shopped for.

```
SELECT c.cust_id,
       c.first_name,
       c.last_name,
       SUM(ca.quantity * p.price) AS total
FROM customer c,
     cust_order o,
     cart ca,
     product p
WHERE c.cust_id = o.cust_id
  AND o.ord_id = ca.ord_id
  AND ca.prod_id = p.prod_id
GROUP BY c.cust_id, c.first_name, c.last_name
ORDER BY c.cust_id;
```

Correct answer:

```
CUST_ID FIRST_NAME         LAST_NAME         TOTAL
------- ------------------ ----------------- ---------
      1 olof               andersson         2566,5
      2 maria              andersson            231
      3 tomas              kvist               1472
      4 hans               rosenboll         2544,5
      5 yvette             porpoix             1590
      6 gustav             moller            1453,5
      9 ollas              bullas            1270,5
     10 roger              nyberg              4222
```

Task 5
Show **cust_id**, **first_name**, **last_name** and the **total order value** that customers have shopped for. Sort the result, so the customer with the highest total order value comes first.
Show the total order value without any decimals. **Hint!** The `ROUND()Function.`

```
SELECT c.cust_id,
       c.first_name,
       c.last_name,
       ROUND(SUM(ca.quantity * p.price)) AS total
FROM customer c,
     cust_order o,
     cart ca,
     product p
WHERE c.cust_id = o.cust_id
  AND o.ord_id = ca.ord_id
  AND ca.prod_id = p.prod_id
GROUP BY c.cust_id, c.first_name, c.last_name
```

```
ORDER BY total DESC;
```

Correct Answer:

```
CUST_ID FIRST_NAME                  LAST_NAME      TOTAL
------- --------------------------- -------------- -------
     10 roger                       nyberg            4222
      1 olof                        andersson         2567
      4 hans                        rosenboll         2545
      5 yvette                      porpoix           1590
      3 tomas                       kvist             1472
      6 gustav                      möller            1454
      9 ollas                       bullas            1271
      2 maria                       andersson          231
```

Task 6
Same as in task 5, but show only customers who have a total over 1500.

```
SELECT c.cust_id,
       c.first_name,
       c.last_name,
       ROUND(SUM(ca.quantity * p.price)) AS total
FROM customer c,
     cust_order o,
     cart ca,
     product p
WHERE c.cust_id = o.cust_id
  AND o.ord_id = ca.ord_id
  AND ca.prod_id = p.prod_id
GROUP BY c.cust_id, c.first_name, c.last_name
HAVING ROUND(SUM(ca.quantity * p.price)) > 1500
ORDER BY total DESC;
```

Correct answer:

```
CUST_ID FIRST_NAME                  LAST_NAME      TOTAL
------- --------------------------- -------------- -------
     10 roger                       nyberg            4222
      1 olof                        andersson         2567
      4 hans                        rosenboll         2545
      5 yvette                      porpoix           1590
```

Task 7
Show **first_name** and **last_name capitalized**, for those customers who **have no** orders.

```
SELECT INITCAP(first_name) AS first_name,
       INITCAP(last_name)  AS last_name
FROM customer
WHERE cust_id NOT IN (
      SELECT cust_id
      FROM cust_order
);
```

Correct answer:

```
FIRST_NAME                     LAST_NAME
------------------------------ ---------
Zoltan                         Habbervic
Lena                           Larsson
```

Task 8
Show **group_name** and the **price** for the most expensive product in that product group

```
SELECT *
FROM (
       SELECT g.group_name,
              p.most_expensive
       FROM prod_group g,
            (SELECT group_id, MAX(price) AS most_expensive
             FROM product
             GROUP BY group_id) p
       WHERE p.group_id = g.group_id
     );
```

Correct answer:

```
GROUP_NAME                       MOST_EXPENSIVE
-------------------------------- --------------
farmhouse                                   240
leisure                                   788,5
hunting                                     795
beard care                                198,5
```

Task 9
Show **prod_id**, and the **full path** to the image of the products that have an image. You get the full path by **concatenate** *path*, *pict_id* and *file_type*.

```
SELECT
    prod_id,
    path || pict_id || '.' || file_type AS full_path
FROM prod_pict
```

Correct answer:

```
PROD_ID FULL_PATH
------- ------------------
 1434   /images/1/1.jpg
  113   /images/2/2.jpg
 5896   /images/3/3.jpg
 1888   /images/4/4.gif
```

Task 10
Show **first name** and **last name** for those customers who owns a customer order that was created during 2004.

```
SELECT DISTINCT
       c.first_name,
       c.last_name
FROM customer c,
     cust_order o
WHERE c.cust_id = o.cust_id
  AND TO_CHAR(o.order_date, 'YYYY') = '2004';
```

Correct answer:

```
FIRST_NAME                LAST_NAME
------------------------- -----------
olof                      andersson
gustav                    moller
```

Optional task (not necessary for the lab)
Show ord_id and total order value for the most expensive customer order.

```
SELECT ord_id, total
FROM (
    SELECT o.ord_id AS ord_id,
           SUM(ca.quantity * p.price) AS total
    FROM cust_order o,
         cart ca,
         product p
    WHERE o.ord_id = ca.ord_id
      AND ca.prod_id = p.prod_id
    GROUP BY o.ord_id
    ORDER BY total DESC
)
WHERE ROWNUM = 1;
```

Correct answer:

```
ORD_ID             TOTAL
--------- --------------
      324           4222
```