==========

EXPERIMENT NO. 04

==========

Author : Diksha Gupta.
Roll no.: 04 [27A]
Date : 05-NOVEMBER -2022.

==========

AIM : To execute different SQL join operations, sub-queries and correlated queries on a multi-relation database .

PROBLEM STATEMENT:

*Use the **SalesCo** database established in Experiment-02 with the below mentioned schemata to execute the listed queries involving join operations, sub-queries of different kinds and correlated queries.*

CUSTOMER (C_CODE, LNAME, FNAME, C_AREA, C_PHONE, BALANCE)

INVOICE (INV_NUM, C_CODE, INV_DATE)

LINE (INV_NUM, L_NUM, P_CODE, L_UNITS, L_PRICE)

PRODUCT (P_CODE, DESCRIPT, P_DATE, QTY, P_MIN, P_PRICE, P_DISC, V_CODE)

VENDOR (V_CODE, V_NAME, V_CONTACT, V_AREA, V_PHONE, V_STATE, V_ORDER)

********** **QUERY-01** **********

Write SQL code to create a table PART without any tuple from PRODUCT such that it includes product code-PT_CODE, product description- PT_DESC, the unit price-PT_PRICE and the supplier code. Now populate PART with the tuples fetching the contents from PRODUCT. For the PART table created, compare its schema with PRODUCT for the common attributes. Observe all the constraints on PART table (use USER_CONSTRAINTS) and state your inferences.

****

```
CREATE TABLE PART
   AS SELECT P_CODE AS PT_CODE,DESCRIPT AS PT_DESC,P_PRICE AS PT_PRICE
     FROM PRODUCT
        WHERE 1=2;

Table created.


SELECT *
    FROM PART;

no rows selected


INSERT INTO PART
    (SELECT P_CODE, DESCRIPT, P_PRICE, V_CODE FROM PRODUCT);

19 rows created.


SELECT *
   FROM PART;

PT_CO PT_DESC                         PT_PRICE     V_CODE
----- ------------------------------ ---------- ----------
AB112 Power Drill                       109.99      25595
SB725 7.25in Saw Blade                   14.99      21344
SB900 9.00 in Saw Blade                  17.49      21344
CL025 Hrd. Spring 1/4in                  39.95      23119
CL050 Hrd. Spring 1/2in                  43.99      23119
JB012 Jigsaw 12in Blade                 109.92      24288
JB008 Jigsaw 8in Blade                   99.87      24288
CD00X Cordless Drill                     38.95      25595
CH10X Claw Hammer                         9.95      21225
SH100 Sledge Hammer                       14.4
RF100 Rat Tail File                       4.99      21344
HC100 Hicut Chain Saw                   256.99      24288
PP101 PVC Pipe                            5.87
MC001 Metal Screw                         6.99      21225
WC025 2.5in wide Screw                    8.45      21231
SM48X Steel Malting Mesh                119.95      25595
HW15X Hiveld Hammer                       17.5      24992
AB111 POWER DRILL                          125      24992
PP102 PVC PIPE                           15.25      24992
```

```
19 rows selected.
```

```
SELECT TABLE_NAME, CONSTRAINT_NAME
  FROM USER_CONSTRAINTS
     WHERE TABLE_NAME IN ('PART', 'PRODUCT');

TABLE_NAME                     CONSTRAINT_NAME
------------------------------ ------------------------------
PRODUCT                        SYS_C008461
PRODUCT                        SYS_C008462
PRODUCT                        SYS_C008463
PRODUCT                        SYS_C008464
PRODUCT                        SYS_C008465
PRODUCT                        SYS_C008466
PRODUCT                        SYS_C008467
PRODUCT                        PRODUCT_CK_P_MIN
PRODUCT                        PRODUCT_PK_P_CODE
PRODUCT                        PRODUCT_VENDOR_FK_V_CODE
PART                           SYS_C008565
PART                           SYS_C008566
PART                           SYS_C008567

13 rows selected.
```

```
 ALTER TABLE PART
    ADD CONSTRAINT PART_PK PRIMARY KEY(PT_CODE);

Table altered.
```

```
SELECT TABLE_NAME, CONSTRAINT_NAME
   FROM USER_CONSTRAINTS
      WHERE TABLE_NAME IN ('PART', 'PRODUCT');

TABLE_NAME                     CONSTRAINT_NAME
------------------------------ ------------------------------
PRODUCT                        SYS_C008461
PRODUCT                        SYS_C008462
PRODUCT                        SYS_C008463
PRODUCT                        SYS_C008464
PRODUCT                        SYS_C008465
PRODUCT                        SYS_C008466
PRODUCT                        SYS_C008467
PRODUCT                        PRODUCT_CK_P_MIN
PRODUCT                        PRODUCT_PK_P_CODE
PRODUCT                        PRODUCT_VENDOR_FK_V_CODE
PART                           SYS_C008565
PART                           SYS_C008566
PART                           SYS_C008567
PART                           PART_PK

14 rows selected.
```

DESC PART;

Name	Null?	Type
PT_CODE	NOT NULL	CHAR(5)
PT_DESC	NOT NULL	VARCHAR2(30)
PT_PRICE	NOT NULL	NUMBER(6,2)
V_CODE		NUMBER(5)

DESC PRODUCT;

Name	Null?	Type
P_CODE	NOT NULL	CHAR(5)
DESCRIPT	NOT NULL	VARCHAR2(30)
P_DATE	NOT NULL	DATE
QTY	NOT NULL	NUMBER(4)
P_MIN	NOT NULL	NUMBER(3)
P_PRICE	NOT NULL	NUMBER(6,2)
P_DISC	NOT NULL	NUMBER(2,2)
V_CODE		NUMBER(5)

********** **QUERY-02** **********

Write a SQL code that will list all vendors who have supplied a part (You must ensure that only unique V_CODE values are displayed). Also retrieve information on vendors referenced in PRODUCT who have supplied products with prices in excess of 10 units.

****

```
SELECT DISTINCT(V_CODE),V_NAME
    FROM PART JOIN VENDOR USING(V_CODE)
        WHERE V_CODE IS NOT NULL;
```

```
V_CODE V_NAME
---------- ----------------------
     25595 HighEnd Supplies
     21344 Gomez Sons
     23119 Blackman Sisters
     24288 Justin Stores
     21225 Bryson, Inc.
     21231 GnB Supply
     24992 INDIAN MASTERS

7 rows selected.
```

```
SELECT V_NAME , V_CODE , DESCRIPT , P_PRICE , QTY
   FROM PRODUCT JOIN VENDOR USING (V_CODE)
       WHERE P_PRICE > 10;
```

V_NAME	V_CODE	DESCRIPT	P_PRICE	QTY
HighEnd Supplies	25595	Power Drill	109.99	8
Gomez Sons	21344	7.25in Saw Blade	14.99	32
Gomez Sons	21344	9.00 in Saw Blade	17.49	18
Blackman Sisters	23119	Hrd. Spring 1/4in	39.95	15
Blackman Sisters	23119	Hrd. Spring 1/2in	43.99	23
Justin Stores	24288	Jigsaw 12in Blade	109.92	8
Justin Stores	24288	Jigsaw 8in Blade	99.87	6
HighEnd Supplies	25595	Cordless Drill	38.95	12
Justin Stores	24288	Hicut Chain Saw	256.99	11
HighEnd Supplies	25595	Steel Malting Mesh	119.95	18
INDIAN MASTERS	24992	Hiveld Hammer	17.5	60
INDIAN MASTERS	24992	POWER DRILL	125	15
INDIAN MASTERS	24992	PVC PIPE	15.25	50

13 rows selected.

********** **QUERY-03** **********

Write SQL code that will retrieve the product particulars for the parts with the highest and the lowest price. Use this query to retrieve the product particulars for the parts with the highest and the lowest inventory value (In both outputs the highest price products should be listed first).

****

```
SELECT P_CODE, P_PRICE FROM PRODUCT P
  WHERE P_CODE IN(
   SELECT PT_CODE FROM PART WHERE PT_PRICE IN(
     SELECT MAX(PT_PRICE ) FROM PART UNION
       SELECT MIN(PT_PRICE ) FROM PART)
        );
```

```
P_COD    P_PRICE
----- ----------
RF100       4.99
HC100     256.99
```

```
SELECT P_CODE, P_PRICE, P_PRICE*QTY AS INV_VAL FROM PRODUCT P
   WHERE P_CODE IN(
    SELECT PT_CODE FROM PART PT WHERE (PT.PT_PRICE * P.QTY) IN(
      SELECT MAX(PT.PT_PRICE * P.QTY) FROM PART PT UNION
        SELECT MIN(PT.PT_PRICE * P.QTY) FROM PART PT)
         )
          ORDER BY P_PRICE DESC;
```

```
P_COD    P_PRICE    INV_VAL
----- ---------- ----------
HC100     256.99    2826.89
RF100       4.99     214.57
```

********** **QUERY-04** **********

Write SQL code that will retrieve the product particulars for all products whose prices (largest first) exceed the average product price of the inventory. Also list the number of products that are supplied by each vendor.

****

```
SELECT * FROM PRODUCT
   WHERE P_PRICE >
     (SELECT AVG(P_PRICE) FROM
       PRODUCT)
         ORDER BY P_PRICE DESC;
```

```
P_COD DESCRIPT     P_DATE           QTY      P_MIN    P_PRICE     P_DISC     V_CODE
----- ------------ --------- ---------- ---------- ---------- ---------- ----------
HC100 Hicut Chain  07-FEB-20         11          5     256.99        .05      24288
      Saw

AB111 POWER DRILL  27-SEP-22         15          5        125         .1      24992

SM48X Steel Maltin 17-JAN-20         18          5     119.95         .1      25595
      g Mesh

AB112 Power Drill  03-NOV-19          8          5     109.99          0      25595

JB012 Jigsaw 12in  30-DEC-19          8          5     109.92        .05      24288
      Blade

JB008 Jigsaw 8in B 24-DEC-19          6          5      99.87        .05      24288
      lade

6 rows selected.
```

```
SELECT V.V_CODE,V.V_NAME,COUNT(P_CODE)
    FROM VENDOR V,PRODUCT
      WHERE V.V_CODE = PRODUCT.V_CODE
        GROUP BY V.V_CODE,V.V_NAME;
```

```
    V_CODE V_NAME                         COUNT(P_CODE)
---------- ------------------------------ -------------
     21225 Bryson, Inc.                               2
     21231 GnB Supply                                 1
     21344 Gomez Sons                                 3
     23119 Blackman Sisters                           2
     24288 Justin Stores                              3
```

```
     25595 HighEnd Supplies                          3
     24992 INDIAN MASTERS                            3

      7 rows selected.
```

********** **QUERY-05**********

Write SQL code to generate a listing of the number of products in the inventory supplied by each vendor that has prices average below 10. Extend this query to generate a listing of the total cost of products for each vendor - TOT_COST, such that the total cost exceeds 400.00 and the high value vendor is placed last.

****

```
SELECT V_CODE, COUNT(P_CODE) COUNT, AVG(P_PRICE) AVERAGE_PRICE
   FROM PRODUCT
    GROUP BY V_CODE
      HAVING AVG(P_PRICE)<10;

    V_CODE      COUNT AVERAGE_PRICE
---------- ---------- -------------
     21225          2          8.47
     21231          1          8.45
```

```
SELECT V_CODE,SUM(P_PRICE * QTY) TOT_COST
  FROM PRODUCT
    WHERE V_CODE IS NOT NULL
       GROUP BY V_CODE
           HAVING SUM(P_PRICE*QTY)>400;

    V_CODE   TOT_COST
---------- ----------
     25595    3506.42
     21344    1009.07
     23119    1611.02
     24288    4305.47
     21225    1431.13
     21231    2002.65
     24992     3687.5
```

7 rows selected.

********** **QUERY-06** **********

Write SQL code to create a view - PRODUCT_STATS from PRODUCT that generate a report that shows a summary of total product cost – TOT_COST, and statistics on the quantity on hand [maximum - MX_QTY, minimum – MN_QTY, average – AV_QTY] for each vendor.

****

```
CREATE OR REPLACE VIEW PRODUCT_STATS AS
  SELECT V_CODE,SUM(P_PRICE) TOT_COST, MAX(QTY) MX_QTY,MIN(QTY) MN_QTY,
    AVG(QTY)   AV_QTY
     FROM PRODUCT
       WHERE V_CODE IS NOT NULL
         GROUP BY V_CODE;
```

View created.

```
SELECT *
  FROM PRODUCT_STATS;
```

V_CODE	TOT_COST	MX_QTY	MN_QTY	AV_QTY
25595	268.89	18	8	12.6666667
21344	37.47	43	18	31
23119	83.94	23	15	19
24288	466.78	11	6	8.33333333
21225	16.94	172	23	97.5
21231	8.45	237	237	237
24992	157.75	60	15	41.6666667

7 rows selected.

********** **QUERY-07** **********

Write a SQL query that will list for each customer who has made purchases, the customer number, the customer balance and the aggregate purchase amount.

****

```
SELECT C_CODE,SUM(L_UNITS*L_PRICE) PURCHASE,BALANCE FROM INVOICE
    NATURAL JOIN LINE NATURAL JOIN CUSTOMER
       GROUP BY C_CODE,BALANCE;
```

```
    C_CODE    PURCHASE     BALANCE
---------- ---------- ----------
     10011     479.57           0
     10012     153.85      345.86
     10014     422.77           0
     10015      34.97           0
     10018      34.87      216.55
     10020        350         500

6 rows selected.
```

********** **QUERY-08**********

Modify Query-07 to include the number of individual product purchases made by each customer. (If the customer's invoice is based on three products, one per L_NUM, then count 3 product purchases. For example, customer 10011 generated 3 invoices, which contained a total of 5 lines, each representing a product purchase.).

****

```
SELECT C_CODE,SUM(L_UNITS) TOT_UNITS FROM INVOICE
   NATURAL JOIN LINE NATURAL JOIN CUSTOMER
     GROUP BY C_CODE
      ORDER BY C_CODE;

    C_CODE  TOT_UNITS
---------- ----------
     10011         23
     10012          7
     10014          8
     10015          3
     10018          5
     10020         20

6 rows selected.
```

********** **QUERY-09** **********

Write SQL query to produce the total purchase per invoice (The invoice total is the sum of the product purchases in the LINE that corresponds to the INVOICE). Further, produce a listing showing invoice numbers with corresponding invoice total identified to a customer (Use GROUP BY on C_CODE). Also generate a listing showing the number of invoices and the total purchase amounts by customer.

****

```
SELECT INV_NUM,C_CODE,SUM(L_PRICE)
  FROM INVOICE NATURAL JOIN LINE
    GROUP BY C_CODE,INV_NUM;
```

INV_NUM	C_CODE	SUM(L_PRICE)
1001	10014	24.94
1002	10011	4.99
1003	10012	93.89
1004	10018	14.94
1005	10011	5.87
1006	10014	383.85
1007	10015	19.98
1008	10011	135.77
1019	10020	17.5

9 rows selected.

```
SELECT COUNT(INV_NUM) INV_TOT, C_CODE, SUM(L_PRICE) SUM
  FROM INVOICE NATURAL JOIN LINE
   GROUP BY C_CODE
     ORDER BY C_CODE;
```

INV_TOT	C_CODE	SUM
5	10011	146.63
3	10012	93.89
6	10014	408.79
2	10015	19.98
2	10018	14.94
1	10020	17.5

6 rows selected.

********** **QUERY-10** **********

Write SQL code to find the customer balance summary for all customers who have not made purchases during the current invoicing period. Use this query to generate a summary of the customer balance characteristics (the output should include the minimum, maximum and average balances over all purchases).

****

```
SELECT C_CODE, BALANCE
  FROM CUSTOMER WHERE C_CODE IN (
    SELECT C_CODE FROM CUSTOMER
     MINUS
      SELECT C_CODE FROM INVOICE)
        GROUP BY C_CODE, BALANCE;
```

```
    C_CODE    BALANCE
---------- ----------
     10010          0
     10013     536.75
     10016     221.19
     10017     768.93
     10019          0
```

```
SELECT MAX(BALANCE) MX_BAL,MIN(BALANCE) MN_BAL,AVG(BALANCE) AV_BAL
  FROM CUSTOMER WHERE C_CODE IN (
    SELECT C_CODE FROM CUSTOMER
     MINUS
     SELECT C_CODE FROM INVOICE);
```

```
    MX_BAL     MN_BAL     AV_BAL
---------- ---------- ----------
    768.93          0    305.374
```

********** **QUERY-11** **********

Write SQL code to create a table INV_CUSTOMER that includes INV_NUM as QUOTE_ID, INV_DATE as QUOTE_DT and C_NAME combining FNAME and LNAME with embedded space. Enforce the entity integrity constraint on QUOTE_ID. (You may use subquery to create the table structure. Ensure that the created table is empty). Now, use SELECT subquery to populate INV_CUSTOMER using the information contained in INVOICE and CUSTOMER.

****

```
CREATE TABLE INV_CUSTOMER AS
 SELECT I.INV_NUM AS QUOTE_ID,
   I.INV_DATE AS QUOTE_DT,
     C.FNAME ||' '||C.LNAME AS CNAME
        FROM INVOICE I
          JOIN CUSTOMER C
            USING(C_CODE)
              WHERE 1=2;

Table created.

ALTER TABLE INV_CUSTOMER
   ADD PRIMARY KEY (QUOTE_ID);

Table altered.

INSERT INTO INV_CUSTOMER
 ( SELECT I.INV_NUM AS QUOTE_ID,I.INV_DATE AS QUOTE_DT,C.FNAME ||'
'||C.LNAME
   AS CNAME
     FROM INVOICE I
        JOIN CUSTOMER C
            USING(C_CODE));

9 rows created.

SELECT * FROM
   INV_CUSTOMER;


  QUOTE_ID QUOTE_DT  CNAME
---------- --------- ----------------------
      1005 17-JAN-20 Elena Johnson
      1008 17-JAN-20 Elena Johnson
      1002 16-JAN-20 Elena Johnson
      1003 16-JAN-20 Kathy Smith
      1006 17-JAN-20 Bill Johnson
      1001 16-JAN-20 Bill Johnson
```

```
1007 17-JAN-20 Julia Samuels
1004 17-JAN-20 Ming Lee
1019 22-JUN-20 DIKSHA GUPTA

9 rows selected.
```

********** **QUERY-12** **********

Modify Query-11 to create a view INV_CUTOMER_VW with the mentioned composition. Do not enforce entity integrity as in Query-11. Populate this view in similar manner. State the problem(s) are encountered. Try populating taking alternative approach you knew. Does that work? Now create the same view (use CREATE OR REPLACE VIEW) such that the view is populated at the creation time. Check the view contents. Now try inserting a record – 1011, Jagat Narayan, 12-Mar-2020, and observe the result.Three non-discounted products - ZZ999 & AB212 (vendor 24992) and SH200 were added to the inventory. The details are as below...

SH200, Sledge Hammer, 05-Jul-2020, 10, 3, 25.8

ZZ999, Cordless Drill, 10-Jul-2020, 200, 40, 25.5

AB212, Power Drill, 03-Aug-2020, 15, 3,275.0

****

```
CREATE OR REPLACE VIEW INV_CUSTOMER_VW AS
  SELECT I.INV_NUM AS QUOTE_ID,I.INV_DATE AS QUOTE_DT,
    C.FNAME ||' '||C.LNAME AS CNAME
      FROM INVOICE I JOIN CUSTOMER C USING(C_CODE)
        WHERE 1=2;

View created.
```

```
INSERT INTO INV_CUSTOMER_VW
   ( SELECT I.INV_NUM AS QUOTE_ID,
     I.INV_DATE AS QUOTE_DT,
      C.FNAME ||' '||C.LNAME AS CNAME
        FROM INVOICE I
          JOIN CUSTOMER C
            USING(C_CODE));
```

```
INSERT INTO INV_CUSTOMER_VW
*
ERROR at line 1:
ORA-01733: virtual column not allowed here
```

```
DROP VIEW INV_CUSTOMER_VW;
View dropped.
```

```
CREATE OR REPLACE VIEW INV_CUSTOMER_VW AS
  SELECT I.INV_NUM AS QUOTE_ID,I.INV_DATE AS QUOTE_DT,
     C.FNAME ||' '||C.LNAME AS CNAME
       FROM INVOICE I JOIN CUSTOMER CUSING(C_CODE);

View created.
```

```
SELECT * FROM INV_CUSTOMER_VW;

  QUOTE_ID QUOTE_DT  CNAME
---------- --------- --------------------
      1005 17-JAN-20 Elena Johnson
      1008 17-JAN-20 Elena Johnson
      1002 16-JAN-20 Elena Johnson
      1003 16-JAN-20 Kathy Smith
      1006 17-JAN-20 Bill Johnson
      1001 16-JAN-20 Bill Johnson
      1007 17-JAN-20 Julia Samuels
      1004 17-JAN-20 Ming Lee
      1019 22-JUN-20 DIKSHA GUPTA
```

```
INSERT INTO INV_CUSTOMER_VW
   VALUES(1011, '12-Mar-2020','Jagat Narayan');
INSERT INTO INV_CUSTOMER_VW VALUES(1011, '12-Mar-2020','Jagat Narayan')
*
ERROR at line 1:
ORA-01733: virtual column not allowed here
```

```
DELETE
   FROM PRODUCT WHERE P_CODE IN ('SH200','ZZ999','AB212')
```

```
INSERT INTO PRODUCT
    VALUES('SH200', 'Sledge hammer', '05-Jul-2020', 10,3,25.8,0,24992);

1 row created.

INSERT INTO PRODUCT
   VALUES('ZZ999', 'Cordless Drill', '10-Jul-2020', 200,40,25.5,0,24992);

1 row created.

INSERT INTO PRODUCT
   VALUES('AB212', 'Power Drill', '03-Aug-2020', 15, 3, 275.0, 0,NULL);

1 row created.

SELECT *
   FROM PRODUCT WHERE P_CODE IN ('SH200','ZZ999','AB212');
```

P_COD	DESCRIPT	P_DATE	QTY	P_MIN	P_PRICE	P_DISC	V_CODE
AB212	Power Drill	03-AUG-20	15	3	275	0	
SH200	Sledge hammer	05-JUL-20	10	3	25.8	0	24992
ZZ999	Cordless Drill	10-JUL-20	200	40	25.5	0	24992

********** **QUERY-13** **********

Write SQL code using subquery to list the supplier number and supplier name of only those suppliers who supply some products.

****

```
SELECT V_CODE, V_NAME FROM VENDOR WHERE V_CODE IN (
    SELECT DISTINCT V_CODE FROM PRODUCT
      );
    V_CODE V_NAME
---------- ------------------------------
     25595 HighEnd Supplies
     21344 Gomez Sons
     23119 Blackman Sisters
```

```
     24288 Justin Stores
     21225 Bryson, Inc.
     21231 GnB Supply
     24992 INDIAN MASTERS

7 rows selected.
```

********** **QUERY-14** **********

Write SQL code using subquery that will compute the average price of all products. Modify the query to compute the average price of all products based on the product description.

****

```
SELECT AVG(P_PRICE)
    FROM PRODUCT;

AVG(P_PRICE)
------------
  63.0359091

SELECT DESCRIPT,COUNT(DESCRIPT) COUNT,AVG(P_PRICE) AVG_PRICE
    FROM PRODUCT
      GROUP BY DESCRIPT;

DESCRIPT                           COUNT  AVG_PRICE
------------------------------ ---------- ----------
Power Drill                             2    192.495
7.25in Saw Blade                        1      14.99
9.00 in Saw Blade                       1      17.49
Hrd. Spring 1/4in                       1      39.95
Hrd. Spring 1/2in                       1      43.99
Jigsaw 12in Blade                       1     109.92
Jigsaw 8in Blade                        1      99.87
Cordless Drill                          2     32.225
Claw Hammer                             1       9.95
Sledge Hammer                           1       14.4
Rat Tail File                           1       4.99
Hicut Chain Saw                         1     256.99
PVC Pipe                                1       5.87
Metal Screw                             1       6.99
```

```
2.5in wide Screw                         1       8.45
Steel Malting Mesh                       1     119.95
Hiveld Hammer                            1       17.5
POWER DRILL                              1        125
PVC PIPE                                 1      15.25
Sledge hammer                            1       25.8

20 rows selected.
```

```
SELECT DESCRIPT,
  AVG(P_PRICE)
    FROM PRODUCT
       GROUP BY DESCRIPT;
```

```
DESCRIPT                       AVG(P_PRICE)
------------------------------ ------------
Power Drill                         192.495
7.25in Saw Blade                      14.99
9.00 in Saw Blade                     17.49
Hrd. Spring 1/4in                     39.95
Hrd. Spring 1/2in                     43.99
Jigsaw 12in Blade                    109.92
Jigsaw 8in Blade                      99.87
Cordless Drill                       32.225
Claw Hammer                            9.95
Sledge Hammer                          14.4
Rat Tail File                          4.99
Hicut Chain Saw                      256.99
PVC Pipe                               5.87
Metal Screw                            6.99
2.5in wide Screw                       8.45
Steel Malting Mesh                   119.95
Hiveld Hammer                          17.5
POWER DRILL                             125
PVC PIPE                              15.25
Sledge hammer                          25.8
20 rows selected.
```

********** **QUERY-15** **********

Write SQL code using subquery that will list product code, product description and unit product price for all products having the unit price higher than or equal to the average product price.

****

```
SELECT P_CODE, DESCRIPT, P_PRICE
   FROM PRODUCT
     WHERE P_PRICE >= (
       SELECT AVG(P_PRICE)
         FROM PRODUCT);
```

```
P_COD DESCRIPT                         P_PRICE
----- ------------------------------ ----------
AB112 Power Drill                         109.99
JB012 Jigsaw 12in Blade                   109.92
JB008 Jigsaw 8in Blade                     99.87
HC100 Hicut Chain Saw                     256.99
SM48X Steel Malting Mesh                  119.95
AB111 POWER DRILL                            125
AB212 Power Drill                            275

7 rows selected.
```

********** **QUERY-16** **********

Write SQL code that will list supplier number, name and contact person for suppliers who do not supply any product in current season.

****

```
SELECT V_CODE,V_NAME,V_CONTACT
  FROM VENDOR WHERE V_CODE IN(
     SELECT V_CODE FROM VENDOR
       MINUS
         SELECT V_CODE FROM PRODUCT);
```

```
   V_CODE V_NAME                        V_CONTACT
---------- ------------------------------ --------------------
     21226 SuperLoo, Inc.                 Ching Ming
     22587 Downing, Inc.                  Simon Singh
     24004 Almeda House                   Almeda Brown
```

```
25443 Super Systems                Ted Hwang
25501 Silvermines Ltd.             Anne White
```

********** **QUERY-17** **********

Write SQL code using subquery to update the product price to the average product price, but only for the products that are supplied by vendors not belonging to the state 'TN' and 'KY'.

Add a line for invoice number 1003 to include 10 items of the product named ZZ999 - 1003, 4, ZZ999, 10, 25.5

****

```
UPDATE PRODUCT SET P_PRICE=(SELECT AVG(P_PRICE) FROM PRODUCT)
  WHERE V_CODE IN(
    SELECT V_CODE FROM VENDOR
       WHERE V_STATE NOT IN ('TN', 'KY'));

5 rows updated.

INSERT INTO LINE VALUES(1003,4,'ZZ999',10,25.5);

1 row created.

SELECT COUNT(*)
   FROM LINE;

  COUNT(*)
----------
        20
```

********** **QUERY-18** **********

Write SQL code using subquery to find all the customers (include customer numbers, first name and last name) who have ordered some kind of a blade. Now find the customers who have ordered the part "Power Drill".

****

```
SELECT C_CODE,FNAME,LNAME FROM CUSTOMER
    WHERE C_CODE IN(
       SELECT C_CODE FROM INVOICE
         WHERE INV_NUM IN(
           SELECT INV_NUM FROM LINE
             WHERE P_CODE IN (
               SELECT P_CODE FROM PRODUCT
                 WHERE LOWER(DESCRIPT) LIKE '%blade')));
```

```
    C_CODE FNAME      LNAME
---------- ---------- ----------
     10014 Bill       Johnson
     10012 Kathy      Smith
     10015 Julia      Samuels
```

```
SELECT C_CODE,FNAME,LNAME
   FROM CUSTOMER
      WHERE C_CODE IN(
        SELECT C_CODE FROM INVOICE
          WHERE INV_NUM IN(
            SELECT INV_NUM FROM LINE
              WHERE P_CODE IN(
               SELECT PT_CODE FROM PART
                   WHERE LOWER(PT_DESC) LIKE '%power drill'
           )
         )
      );
```

```
no rows selected
```

********** **QUERY-19** **********

Write SQL code using subquery to find all the customers who have purchased a drill or a hammer or a saw.

****

```
SELECT C_CODE,FNAME,LNAME
   FROM CUSTOMER
     WHERE C_CODE IN(
       SELECT C_CODE FROM INVOICE
         WHERE INV_NUM IN (
          SELECT INV_NUM FROM LINE
            WHERE P_CODE IN(
             SELECT P_CODE FROM PRODUCT
               WHERE UPPER(DESCRIPT) LIKE '%HAMMER' OR
                UPPER(DESCRIPT) LIKE '%DRILL' OR
                 UPPER(DESCRIPT) LIKE '%SAW%'
                        )
                    )
                  );
```

```
    C_CODE FNAME      LNAME
---------- ---------- ----------
     10011 Elena      Johnson
     10012 Kathy      Smith
     10014 Bill       Johnson
     10015 Julia      Samuels
     10018 Ming       Lee
     10020 DIKSHA     GUPTA

6 rows selected.
```

********** **QUERY-20** **********

Write SQL code using subquery to list all products with the total quantity sold greater than the average quantity sold.

****

```
SELECT * FROM PRODUCT WHERE P_CODE IN (
   SELECT P_CODE FROM (
     SELECT P_CODE,SUM(L_UNITS)
      FROM LINE
       GROUP BY P_CODE
        HAVING SUM(L_UNITS)>AVG(L_UNITS)

         ) );
```

```
P_COD DESCRIPT            P_DATE           QTY      P_MIN    P_PRICE     P_DISC     V_CODE
----- ------------------- --------- ---------- ---------- ---------- ---------- ----------
SB725 7.25in Saw Blade    13-DEC-19         32         15      14.99        .05      21344
CD00X Cordless Drill      20-JAN-20         12          5      63.04        .05      25595
CH10X Claw Hammer         20-JAN-20         23         10       9.95         .1      21225
RF100 Rat Tail File       15-DEC-19         43         20       4.99          0      21344
PP101 PVC Pipe            20-FEB-20        188         75       5.87          0
```

********** **QUERY-21** **********

Write SQL code using subquery to list all customers who have purchased products HC100 and JB012.

****

```
SELECT C_CODE,FNAME,LNAME FROM CUSTOMER NATURAL JOIN INVOICE
  WHERE INV_NUM IN(
    SELECT INV_NUM FROM LINE
     WHERE P_CODE IN ('HC100','JB012')
     );
```

```
    C_CODE FNAME      LNAME
---------- ---------- ----------
     10014 Bill       Johnson
```

********** **QUERY-22** **********

Write SQL code using subquery that will for all products list the product price and the difference between each product's price and the average product price. Ensure that the average product price is also displayed

****

```
SELECT P_CODE,P_PRICE,(
   P_PRICE-(
     SELECT AVG(P_PRICE)
       FROM PRODUCT)
        ) DIFF,
         (SELECT AVG(P_PRICE) FROM PRODUCT) AVERAGE
           FROM PRODUCT;
```

```
P_COD    P_PRICE       DIFF    AVERAGE
----- ---------- ---------- ----------
AB112      63.04 1.71454545 61.3254545
SB725      14.99 -46.335455 61.3254545
SB900      17.49 -43.835455 61.3254545
CL025      63.04 1.71454545 61.3254545
CL050      63.04 1.71454545 61.3254545
JB012     109.92 48.5945455 61.3254545
JB008      99.87 38.5445455 61.3254545
CD00X      63.04 1.71454545 61.3254545
CH10X       9.95 -51.375455 61.3254545
SH100       14.4 -46.925455 61.3254545
RF100       4.99 -56.335455 61.3254545
HC100     256.99 195.664545 61.3254545
PP101       5.87 -55.455455 61.3254545
MC001       6.99 -54.335455 61.3254545
WC025       8.45 -52.875455 61.3254545
SM48X      63.04 1.71454545 61.3254545
HW15X       17.5 -43.825455 61.3254545
AB111        125 63.6745455 61.3254545
```

```
PP102      15.25 -46.075455 61.3254545
SH200       25.8 -35.525455 61.3254545
ZZ999       25.5 -35.825455 61.3254545
AB212        275 213.674545 61.3254545

22 rows selected.
```

********** **QUERY-23** **********

Write SQL code using correlated query to list all product sales in which the units sold value is greater than the average units sold value for that product (as opposed to the average for all products).

****

```
SELECT P_CODE, SUM(L_UNITS) FROM LINE L
   GROUP BY P_CODE
     HAVING SUM(L.L_UNITS)>(
      SELECT AVG(L_UNITS) FROM LINE LA
        WHERE LA.P_CODE=L.P_CODE);
```

```
P_COD SUM(L_UNITS)
----- ------------
SB725             8
CH10X             5
RF100             6
CD00X             2
PP101            17
```

********** **QUERY-24** **********

Write SQL code using correlated query to list all customers who have placed an order. (Use EXISTS clause in SELECT statement).

****

```
SELECT * FROM CUSTOMER C
    WHERE EXISTS(
      SELECT * FROM INVOICE I
        WHERE I.C_CODE=C.C_CODE
         );
```

C_CODE	LNAME	FNAME	C_AREA	C_PHONE	BALANCE
10011	Johnson	Elena	713	2753455	0
10012	Smith	Kathy	615	2873453	345.86
10014	Johnson	Bill	615	2455533	0
10015	Samuels	Julia	713	2345432	0
10018	Lee	Ming	713	2323234	216.55
10020	GUPTA	DIKSHA	904	3562098	500

6 rows selected.

==========

INFERENCES OF THE EXPERIMENT

==========

Hence , we have successfully execute different SQL join operations, sub-queries and correlated queries on a multi-relation database .