```
 --JOURNAL COMPILATION
 /*
        AIM:To establish database characterizing data-extensive system and
            execute different SQL join operations on it. To execute subqueries
            and correlated queries on the database.

        PROBLEM STATEMENT:Use the SalesCo database established in Experiment-02
                          with the below mentioned schemata to execute the listed
                          queries involving subqueries 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 a 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.Now, use SELECT
 subquery to populate INV_CUSTOMER using the information contained in INVOICE and
CUSTOMER.
 */
 ------------------------------------------------------------------------------
SQL> CREATE TABLE INV_CUSTOMER AS
  2  (SELECT I.INV_NUM AS QUOTE_ID,I.INV_DATE AS QUOTE_DT,(C.FNAME||' '||C.LNAME) AS
C_NAME FROM
  3  INVOICE I JOIN CUSTOMER C ON I.C_CODE=C.C_CODE WHERE 1=0);

Table created.

SQL> ALTER TABLE INV_CUSTOMER  ADD PRIMARY KEY(QUOTE_ID);

Table altered.

SQL> DESC INV_CUSTOMER;
 Name                          Null?    Type
 ----------------------------- -------- --------------
 QUOTE_ID                      NOT NULL NUMBER(4)
 QUOTE_DT                      NOT NULL DATE
 C_NAME                                 VARCHAR2(21)

SQL> SELECT CONSTRAINT_NAME,CONSTRAINT_TYPE FROM USER_CONSTRAINTS WHERE TABLE_NAME LIKE
'INV_CUSTOMER';

CONSTRAINT_NAME           C
------------------------- -
SYS_C0012332              P
SYS_C0012331              C
SYS_C0012330              C

3 rows selected.

SQL> INSERT INTO INV_CUSTOMER (QUOTE_ID,QUOTE_DT,C_NAME)
  2  (SELECT INV_NUM,INV_DATE,FNAME||' '||LNAME FROM  INVOICE JOIN CUSTOMER ON
INVOICE.C_CODE=CUSTOMER.C_CODE);
```

```
8 rows created.

SQL> SELECT * FROM INV_CUSTOMER;

  QUOTE_ID QUOTE_DT  C_NAME

---------- --------- --------------------

      1008 17-JAN-12 Elena Kurtis

      1005 17-JAN-12 Elena Kurtis

      1002 16-JAN-12 Elena Kurtis

      1003 16-JAN-12 Kathy Smith

      1006 17-JAN-12 Bill Johnson

      1001 16-JAN-12 Bill Johnson

      1007 17-JAN-12 Julia Samuels

      1004 17-JAN-12 Ming Lee


8 rows selected.

 ----------------------------------- QUERY-02 -------------------------------
 /*
 Modify Query-01 to create a view INV_CUTOMER_VW with the mentioned composition. Do not
enforce entity integrity
 as in Query-01. 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-Jan-1992, and observe the result.
 */
 ------------------------------------------------------------------------------
SQL> 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 C_NAME FROM
  2  INVOICE I JOIN CUSTOMER C ON I.C_CODE=C.C_CODE WHERE 1=0);

View created.

SQL> INSERT INTO INV_CUSTOMER_VW (QUOTE_ID,QUOTE_DT,C_NAME)
  2  (SELECT INV_NUM,INV_DATE,FNAME||' '||LNAME FROM  INVOICE JOIN CUSTOMER ON
INVOICE.C_CODE=CUSTOMER.C_CODE);
INSERT INTO INV_CUSTOMER_VW (QUOTE_ID,QUOTE_DT,C_NAME)
                                                    *
ERROR at line 1:
ORA-01733: virtual column not allowed here

SQL> INSERT INTO INV_CUSTOMER_VW
  2  VALUES (1011,'Jagat Raman','12-Jan-1992');
INSERT INTO INV_CUSTOMER_VW
*
ERROR at line 1:
```

```
ORA-01733: virtual column not allowed here

SQL> INSERT INTO INV_CUSTOMER_VW
  2  SELECT INV_NUM, NAME INV_DATE
  3  FROM INVOICE I JOIN
  4  (SELECT C_CODE, FNAME||' '||LNAME AS NAME FROM CUSTOMER) C
  5  USING (C_CODE);
INSERT INTO INV_CUSTOMER_VW
            *
ERROR at line 1:
ORA-00947: not enough values

SQL> CREATE OR REPLACE VIEW INV_CUSTOMER_VW
  2  AS SELECT I.INV_NUM AS QUOTE_ID, FNAME||' '||LNAME AS C_NAME,
  3  I.INV_DATE AS QUOTE_DT
  4  FROM INVOICE I JOIN CUSTOMER C
  5  ON C.C_CODE = I.C_CODE;

View created.

SQL> SELECT * FROM INV_CUSTOMER_VW;

  QUOTE_ID C_NAME                QUOTE_DT

---------- --------------------- ---------

      1008 Elena Kurtis          17-JAN-12

      1005 Elena Kurtis          17-JAN-12

      1002 Elena Kurtis          16-JAN-12

      1003 Kathy Smith           16-JAN-12

      1006 Bill Johnson          17-JAN-12

      1001 Bill Johnson          16-JAN-12

      1007 Julia Samuels         17-JAN-12

      1004 Ming Lee              17-JAN-12


8 rows selected.

 ----------------------------------- QUERY-03 ------------------------------
 /*
 Write a SQL code using subquery to list the supplier number and supplier name of only
those suppliers who supply some products.
 */
 ------------------------------------------------------------------------------
SQL> SELECT DISTINCT VENDOR.V_CODE,V_NAME
  2  FROM VENDOR JOIN PRODUCT ON VENDOR.V_CODE = PRODUCT.V_CODE
  3  WHERE VENDOR.V_CODE IN (SELECT  PRODUCT.V_CODE FROM PRODUCT);

    V_CODE V_NAME

---------- ------------------------------

     24288 Justin Stores
```

21344 Gomez Sons

25595 HighEnd Supplies

21225 Bryson, Inc.

23119 Blackman Sisters

21231 GnB Supply

6 rows selected.

```
 ----------------------------------- QUERY-04 -----------------------------
 /*
 Write a 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.
 */
 ------------------------------------------------------------------------------
SQL> SELECT P_CODE,DESCRIPT,AVG(P_PRICE) FROM PRODUCT GROUP BY P_CODE,DESCRIPT;

P_COD DESCRIPT                       AVG(P_PRICE)

----- ------------------------------ ------------

SB725 7.25in Saw Blade                      14.99

JB012 Jigsaw 12in Blade                    109.92

SM48X Steel Malting Mesh                   119.95

SH100 Sledge Hammer                          14.4

CL025 Hrd. Cloth 1/4in                      39.95

JB008 Jigsaw 8in Blade                      99.87

MC001 Metal Screw                            6.99

WC025 2.5in wide Screw                       8.45

CL050 Hrd. Cloth 1/2in                      43.99

RF100 Rat Tail File                          4.99

PP101 PVC Pipe                               5.87


P_COD DESCRIPT                       AVG(P_PRICE)

----- ------------------------------ ------------

AB112 Power Painter                        109.99

SB900 9.00 in Saw Blade                     17.49

CD00X Cordless Drill                        38.95
```

```
CH10X Claw Hammer                              9.95

HC100 Hicut Chain Saw                        256.99

16 rows selected.

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

DESCRIPT                       AVG(P_PRICE)

------------------------------ ------------

Hicut Chain Saw                       256.99

7.25in Saw Blade                       14.99

9.00 in Saw Blade                      17.49

Jigsaw 12in Blade                     109.92

Jigsaw 8in Blade                       99.87

Metal Screw                             6.99

2.5in wide Screw                        8.45

Hrd. Cloth 1/2in                       43.99

Rat Tail File                           4.99

Power Painter                         109.99

Sledge Hammer                           14.4

DESCRIPT                       AVG(P_PRICE)

------------------------------ ------------

PVC Pipe                                5.87

Hrd. Cloth 1/4in                       39.95

Claw Hammer                             9.95

Steel Malting Mesh                    119.95

Cordless Drill                         38.95

16 rows selected.

 ---------------------------------- QUERY-05 ------------------------------
 /*
 Write a 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.
 */
 ------------------------------------------------------------------------
```

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

P_COD DESCRIPT                         P_PRICE

----- ------------------------------ ----------

AB112 Power Painter                     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


5 rows selected.

 ----------------------------------- QUERY-06 -------------------------------
 /*
 Write a SQL code that will list supplier number, name and contact person for suppliers
 who do not supply any product in current season.
 */
 ------------------------------------------------------------------------------
SQL> SELECT V_CODE,V_NAME,V_CONTACT FROM VENDOR
  2  WHERE V_CODE NOT IN (SELECT V_CODE FROM PRODUCT WHERE V_CODE IS NOT NULL);

    V_CODE V_NAME                         V_CONTACT

---------- ------------------------------ --------------------

     21226 Superloo, Inc.                 Ching-Hun

     24004 Almeda House                   Almeda

     22587 Dowing, Inc.                   Simon Singh

     25501 SilverminesLtd.                Anne White

     25443 Super Systems                  Ted Hwang


5 rows selected.

 ----------------------------------- QUERY-07 -------------------------------
 /*
 Write a 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'.
 */
 ------------------------------------------------------------------------------
SQL> SELECT P_CODE,P_PRICE FROM PRODUCT;

P_COD    P_PRICE

----- ----------
```

```
AB112     109.99

SB725      14.99

SB900      17.49

CL025      39.95

CL050      43.99

JB012     109.92

JB008      99.87

CD00X      38.95

CH10X       9.95

SH100       14.4

RF100       4.99


P_COD    P_PRICE

----- ----------

HC100     256.99

PP101       5.87

MC001       6.99

WC025       8.45

SM48X     119.95


16 rows selected.

SQL> UPDATE PRODUCT
  2  SET P_PRICE=(SELECT AVG(P_PRICE) FROM PRODUCT)
  3  WHERE V_CODE NOT IN
  4  (SELECT V_CODE FROM VENDOR WHERE VENDOR.V_STATE IN ('TN','KY') );

5 rows updated.

SQL> SELECT P_CODE,P_PRICE FROM PRODUCT;

P_COD    P_PRICE

----- ----------

AB112      56.42

SB725      14.99

SB900      17.49

CL025      56.42
```

```
CL050      56.42

JB012     109.92

JB008      99.87

CD00X      56.42

CH10X       9.95

SH100       14.4

RF100       4.99


P_COD    P_PRICE

----- ----------

HC100     256.99

PP101       5.87

MC001       6.99

WC025       8.45

SM48X      56.42


16 rows selected.

SQL> ROLLBACK;

Rollback complete.

---------------------------------- QUERY-08 ------------------------------
 /*
 Write a 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
“Jigsaw 12in Blade”.
 */
 ------------------------------------------------------------------------
SQL> SELECT DISTINCT CUSTOMER.C_CODE,LNAME,FNAME FROM CUSTOMER
  2  JOIN INVOICE ON INVOICE.C_CODE = CUSTOMER.C_CODE
  3  WHERE INVOICE.INV_NUM IN
  4  (SELECT INV_NUM FROM LINE JOIN PRODUCT ON PRODUCT.P_CODE = LINE.P_CODE
  5  WHERE UPPER(DESCRIPT) LIKE '%BLADE%');

    C_CODE LNAME      FNAME

---------- ---------- ----------

     10012 Smith      Kathy

     10014 Johnson    Bill

     10015 Samuels    Julia
```

```
3 rows selected.

SQL> SELECT DISTINCT CUSTOMER.C_CODE,LNAME,FNAME FROM CUSTOMER
  2  JOIN INVOICE ON INVOICE.C_CODE = CUSTOMER.C_CODE
  3  WHERE INVOICE.INV_NUM IN
  4  (SELECT INV_NUM FROM LINE JOIN PRODUCT ON PRODUCT.P_CODE = LINE.P_CODE
  5  WHERE UPPER(DESCRIPT) LIKE '%JIGSAW 12IN BLADE%');

    C_CODE LNAME      FNAME

---------- ---------- ----------

     10014 Johnson    Bill


1 row selected.

  ------------------------------------ QUERY-09 -------------------------------
 /*
 Write a SQL code using subquery to find all the customers who have purchased a drill or
a hammer or a saw.
 */
 ---------------------------------------------------------------------------
SQL> SELECT DISTINCT CUSTOMER.C_CODE,LNAME,FNAME FROM CUSTOMER
  2  JOIN INVOICE ON INVOICE.C_CODE = CUSTOMER.C_CODE
  3  WHERE INVOICE.INV_NUM IN
  4  (SELECT INV_NUM FROM LINE JOIN PRODUCT ON PRODUCT.P_CODE = LINE.P_CODE
  5  WHERE UPPER(DESCRIPT) LIKE '%HAMMER%' OR UPPER(DESCRIPT) LIKE '%DRILL%' OR
UPPER(DESCRIPT)LIKE '%SAW%');

    C_CODE LNAME      FNAME

---------- ---------- ----------

     10012 Smith      Kathy

     10018 Lee        Ming

     10011 Kurtis     Elena

     10014 Johnson    Bill

     10015 Samuels    Julia


5 rows selected.

 ------------------------------------ QUERY-10 -------------------------------
 /*
 Write a SQL code using subquery to list all products with the total quantity sold
greater than the average quantity sold.
 */
 ---------------------------------------------------------------------------
SQL> SELECT P_CODE,SUM(L_UNITS) AS TOT_QTY FROM LINE JOIN INVOICE
  2  ON LINE.INV_NUM = INVOICE.INV_NUM
  3  GROUP BY LINE.P_CODE
  4  HAVING SUM(L_UNITS)>(SELECT AVG(L_UNITS) FROM LINE);
```

```
P_COD    TOT_QTY

----- ----------

SB725          8

SM48X          3

RF100          6

CH10X          5

PP101         17

MC001          3


6 rows selected.

 ------------------------------------ QUERY-11 ------------------------------
 /*
 Write a SQL code using subquery to list all customers who have purchased products HC100
and JB012.
 */
 ---------------------------------------------------------------------------
SQL> SELECT CUSTOMER.C_CODE,CUSTOMER.LNAME FROM CUSTOMER
  2  WHERE CUSTOMER.C_CODE IN
  3  (SELECT DISTINCT C_CODE FROM INVOICE JOIN LINE
  4  ON LINE.INV_NUM = INVOICE.INV_NUM
  5  WHERE P_CODE IN ('HC100','JB012'));

    C_CODE LNAME

---------- ----------

     10014 Johnson


1 row selected.

 ------------------------------------ QUERY-12 ------------------------------
 /*
 Write a 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.
 */
 ---------------------------------------------------------------------------
SQL> COLUMN AV_PRICE FORMAT 999.99;
SQL> COLUMN DF_PRICE FORMAT 999.99;

SQL> SELECT P_CODE,P_PRICE,(SELECT AVG(P_PRICE) FROM PRODUCT) AV_PRICE,
  2  P_PRICE - (SELECT AVG(P_PRICE) FROM PRODUCT) AS DF_PRICE
  3  FROM PRODUCT;

P_COD    P_PRICE AV_PRICE DF_PRICE

----- ---------- -------- --------

AB112     109.99    56.42    53.57
```

```
SB725      14.99    56.42   -41.43

SB900      17.49    56.42   -38.93

CL025      39.95    56.42   -16.47

CL050      43.99    56.42   -12.43

JB012     109.92    56.42    53.50

JB008      99.87    56.42    43.45

CD00X      38.95    56.42   -17.47

CH10X       9.95    56.42   -46.47

SH100        14.4   56.42   -42.02

RF100       4.99    56.42   -51.43


P_COD    P_PRICE AV_PRICE DF_PRICE

----- ---------- -------- --------

HC100     256.99    56.42   200.57

PP101       5.87    56.42   -50.55

MC001       6.99    56.42   -49.43

WC025       8.45    56.42   -47.97

SM48X     119.95    56.42    63.53


16 rows selected.

 ----------------------------------- QUERY-13 -------------------------------
 /*
 Write a 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).
 */
 ------------------------------------------------------------------------------
SQL> COLUMN AV_PRICE FORMAT 999.99
SQL> SELECT INV_NUM,P_CODE,L_UNITS,
  2  (SELECT AVG(L_UNITS) FROM LINE LP WHERE LP.P_CODE=LX.P_CODE) AS AV_PRICE FROM LINE
LX
  3  WHERE LX.L_UNITS>(SELECT AVG(L_UNITS) FROM LINE LZ
  4  WHERE LX.P_CODE=LZ.P_CODE);

   INV_NUM P_COD    L_UNITS AV_PRICE

---------- ----- ---------- --------

      1003 SB725          5     2.67
```

```
      1004 RF100          3     2.00

      1004 CH10X          2     1.25

      1005 PP101         12     8.50


4 rows selected.

 ------------------------------------ QUERY-14 -------------------------------
 /*
 Write a SQL code using correlated query to list all customers who have placed an order.
 (Use EXISTS clause in SELECT statement).
 */
 ---------------------------------------------------------------------------
SQL> SELECT C_CODE,FNAME,LNAME
  2  FROM CUSTOMER C
  3  WHERE EXISTS(SELECT C_CODE FROM INVOICE I WHERE I.C_CODE=C.C_CODE);

    C_CODE FNAME      LNAME

---------- ---------- ----------

     10014 Bill       Johnson

     10011 Elena      Kurtis

     10012 Kathy      Smith

     10018 Ming       Lee

     10015 Julia      Samuels


5 rows selected.

 ---------------------------------- END OF QUERIES----------------------------

SQL> SET FEEDBACK OFF
SQL> SPOOL OFF
```