***** EXPERIMENT NO: 05 *****

Author: Pradyot Patil
Roll No : 53 [6B]
Date: 25-March-2018

****

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. (You may use sub query to create the table structure. Do not Fail to ensure that the created table is empty). Now, use SELECT sub query to populate INV_CUSTOMER using the information contained in INVOICE and CUSTOMER.

****

```
CREATE TABLE INV_CUSTOMER AS
2         (SELECT INV_NUM AS QUOTE_ID, INV_DATE AS QUOTE_DT,
3                   FNAME||' '||LNAME AS C_NAME FROM INVOICE,CUSTOMER
4                   WHERE 1=2);

Table created.


DESC INV_CUSTOMER;

 Name       Null?    Type
 ---------- -------- -------
 QUOTE_ID            NUMBER(4)
 QUOTE_DT   NOT NULL DATE
 C_NAME              VARCHAR2(21)



SELECT * FROM INV_CUSTOMER;

no rows selected.



ALTER TABLE INV_CUSTOMER
2         ADD CONSTRAINT INV_CUSTOMER_PK_QUOTE_ID PRIMARY KEY (QUOTE_ID);

Table altered.



INSERT INTO INV_CUSTOMER (SELECT INV_NUM , INV_DATE, FNAME||' '||LNAME
2                                   FROM INVOICE,CUSTOMER
3                                      WHERE INVOICE.C_CODE=CUSTOMER.C_CODE);

9 rows created.
```

```
********************************************************************************
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.
/* The inventory has been added with three more non-discounted products –ZZ999 and
   AB212 (supplied by 24992) and SH200. The details are as below...
            SH200, Sledge Hammer, 02-Jun-2012, 10, 3, 25.8
            ZZ999, Cordless Drill, 12-Jun-2011, 200, 40, 25.5
            AB212, Power Painter, 11-Apr-2011, 15, 3, 275.0                    */
********************************************************************************

CREATE OR REPLACE VIEW INV_CUTOMER_VW     AS
2            (SELECT INV_NUM AS QUOTE_ID, INV_DATE AS QUOTE_DT,
3                    FNAME||' '||LNAME AS C_NAME FROM INVOICE,CUSTOMER
4                                            WHERE 1=2);


View created.



INSERT INTO INV_CUTOMER_VW (SELECT INV_NUM , INV_DATE, FNAME||' '||LNAME
2                               FROM INVOICE,CUSTOMER
3                                    WHERE INVOICE.C_CODE=CUSTOMER.C_CODE);

INSERT INTO INV_CUTOMER_VW (SELECT INV_NUM , INV_DATE, FNAME||' '||LNAME
                     FROM INVOICE,CUSTOMER
*
ERROR at line 1:
ORA-01779: cannot modify a column which maps to a non key-preserved table



CREATE OR REPLACE VIEW INV_CUTOMER_VW  AS
2             (SELECT INV_NUM AS QUOTE_ID, INV_DATE AS QUOTE_DT,
3                       FNAME||' '||LNAME AS C_NAME  FROM INVOICE,CUSTOMER
4                                    WHERE INVOICE.C_CODE=CUSTOMER.C_CODE);

View created.



SELECT * FROM INV_CUTOMER_VW  ;

  QUOTE_ID QUOTE_DT  C_NAME
---------- --------- --------------------
      1005 17-JAN-12 Elena kurtis
      1008 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
      1009 22-JAN-13 Grace Dawson

9 rows selected.
```

```
INSERT INTO  INV_CUTOMER_VW VALUES(1011,'12-Jan-1992',     'Jagat Narayan')  ;

INSERT INTO  INV_CUTOMER_VW VALUES(1011,'12-Jan-1992',  'Jagat Narayan')
*
ERROR at line 1:
ORA-01733: virtual column not allowed here
```

```
INSERT INTO  PRODUCT
2        VALUES('SH200','Sledge Hammer', '02-Jun-2012', 10, 3, 25.8, 0.00, NULL)  ;

1 row created.
```

```
INSERT INTO  PRODUCT
2       VALUES('ZZ999', 'Cordless Drill', '12-Jun-2011', 200, 40, 25.5, 0.00,NULL);

1 row created.
```

```
INSERT INTO  PRODUCT
2        VALUES('AB212', 'Power Painter', '11-Apr-2011',15, 3, 275.0, 0.00,24992);

1 row created.
```

```
********************************************************************************
QUERY-03 :  Write a SQL code using sub query to list the supplier number and
            supplier name of only those suppliers who supply some products.
********************************************************************************

SELECT V_CODE, V_NAME FROM VENDOR
2         WHERE V_CODE IN (SELECT V_CODE FROM PRODUCT );


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

7 rows selected.

****
QUERY-04 : Write a SQL code using sub query 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) AS AVERAGE_PRICE FROM PRODUCT;

AVERAGE_PRICE
-------------
       59.533
```

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

DESCRIPT             AVERAGE_PRICE
-------------------- -------------
Hicut Chain Saw             256.99
9.00 in Saw Blade            17.49
Jigsaw 12in Blade           109.92
Jigsaw 8in Blade             99.87
Hrd. Cloth 1/2 in            61.74
Metal Screw                   6.99
2.5in wide Screw              8.45
Rat Tail File                 4.99
Power Painter               168.37
7.25 in Saw Blade            14.99
Sledge Hammer                 20.1
PVC Pipe                      5.87
Hrd. Cloth 1/4 in            61.74
Hanging Hook                  5.75
Claw Hammer                   9.95
Steel Malting Mesh           61.74
Cordless Drill               43.62

17 rows selected.
```

****
QUERY-05 : Write a SQL code using sub query 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
2               FROM PRODUCT
3           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
AB212 Power Painter                         275

6 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.

****

```
SELECT V_CODE,V_NAME, V_CONTACT
2                FROM VENDOR
3           WHERE V_CODE NOT IN (SELECT V_CODE FROM PRODUCT
4                                    WHERE V_CODE IS NOT NULL);

    V_CODE V_NAME                         V_CONTACT
---------- ------------------------------ --------------------
     21226 Super. Loo.                     Ching-Hun
     22587 Downing. Inc.                   Simon Singh
     24004 Almeda House                    Almeda
     25443 Super Systems                   Ted Hwang
     25501 SilverminesLtd.                 Anne White

5 rows selected.
```

****

QUERY-07 : Write a SQL code using sub query 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 a 10 items of the product named ZZ999 – 1003, 4, ZZ999, 10, 25.5 */

****

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

```
5 rows updated.

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

1 row created.
```

****

QUERY-08 : Write a SQL code using sub query 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".

****

```
SELECT C_CODE, FNAME||' '||LNAME AS C_NAME FROM CUSTOMER
  2      WHERE C_CODE IN (SELECT C_CODE FROM INVOICE
  3                  WHERE INV_NUM IN (SELECT INV_NUM FROM LINE
  4                       WHERE P_CODE IN (SELECT P_CODE FROM PRODUCT
  5                                WHERE DESCRIPT LIKE '%Blade%')) );



    C_CODE C_NAME
---------- --------------------
     10012 Kathy Smith
     10014 Bill Johnson
     10015 Julia Samuels


SELECT C_CODE, FNAME||' '||LNAME AS C_NAME FROM CUSTOMER
2      WHERE C_CODE IN (SELECT C_CODE FROM INVOICE
3                  WHERE INV_NUM IN (SELECT INV_NUM FROM LINE
4                       WHERE P_CODE IN (SELECT P_CODE FROM PRODUCT
5                                WHERE DESCRIPT LIKE 'Jigsaw 12in Blade')) );

    C_CODE C_NAME
---------- --------------------
     10014 Bill Johnson
```

```
********************************************************************************
QUERY-09 :  Write a SQL code using sub query to find all the customers who have
            purchased a drill or a hammer or a saw.
********************************************************************************

SELECT C_CODE FROM
2      INVOICE WHERE INV_NUM IN (
3           SELECT INV_NUM FROM LINE WHERE P_CODE IN (
4                   SELECT P_CODE FROM PRODUCT WHERE DESCRIPT LIKE
5         '%Saw%' OR  DESCRIPT LIKE '%Drill%' OR  DESCRIPT LIKE '%Hammer%' ) ) ;

    C_CODE
----------
     10014
     10012
     10018
     10014
     10015
     10011

6 rows selected.


********************************************************************************
QUERY-10 :  Write a SQL code using sub query to list all products with the total
            quantity sold greater than the average quantity sold.
********************************************************************************

SELECT PR_CODE AS PRODUCT_CODE,UNITS,UNITS/QTY AS AVERAGE_QTY FROM
2            PRODUCT,(SELECT P_CODE AS PR_CODE,SUM(L_UNITS) AS UNITS
3                                    FROM LINE
4                                GROUP BY P_CODE)
5                 WHERE PR_CODE=P_CODE AND UNITS> UNITS/QTY;


PRODUCT_CODE      UNITS AVERAGE_QTY
------------ ---------- -----------
CD00X                 1  .083333333
CH10X                 5  .217391304
CL025                 1  .066666667
HC100                 1  .090909091
HH15X                20          .1
JB012                 1        .125
MC001                 3   .01744186
PP101                17  .090425532
RF100                 6  .139534884
SB725                 8         .25
SM48X                 3  .166666667
ZZ999                10         .05

12 rows selected.
```

```
*********************************************************************************
```

QUERY-11 : Write a SQL code using sub query to list all customers who have purchased products HC100 and JB012.

```
*********************************************************************************
```

```
SELECT C_CODE
2       FROM INVOICE ,
3           (SELECT I1 FROM
4            (SELECT  INV_NUM AS I1
5                    FROM LINE
6                                WHERE P_CODE LIKE 'JB012' ),
7            (SELECT  INV_NUM AS I2
8                    FROM LINE
9                                WHERE P_CODE LIKE 'HC100')
10            WHERE I1=I2)
11   WHERE INV_NUM= I1;
```

```
    C_CODE
----------
     10014
```

```
*********************************************************************************
```

QUERY-12 : Write a SQL code using sub query 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_PRICE AS PRICE_OF_PRODUCT,
2     ABS(P_PRICE-(SELECT AVG(P_PRICE) FROM PRODUCT)) PRICE_DIFFERENCE,
3                (SELECT AVG(P_PRICE) FROM PRODUCT) AS AVERAGE
4                        FROM PRODUCT;
```

PRICE_OF_PRODUCT	PRICE_DIFFERENCE	AVERAGE
61.74	2.207	59.533
14.99	44.543	59.533
17.49	42.043	59.533
61.74	2.207	59.533
61.74	2.207	59.533
109.92	50.387	59.533
99.87	40.337	59.533
61.74	2.207	59.533
9.95	49.583	59.533
14.4	45.133	59.533
5.87	53.663	59.533
4.99	54.543	59.533
256.99	197.457	59.533
6.99	52.543	59.533

```
           8.45          51.083     59.533
          61.74           2.207     59.533
           25.8          33.733     59.533
           5.75          53.783     59.533
           25.5          34.033     59.533
            275         215.467     59.533

20 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).

****

```
SELECT DISTINCT P_CODE FROM
  2           LINE L
  3          WHERE
  4              (SELECT SUM(L_UNITS) FROM LINE
  5                  WHERE L.P_CODE=P_CODE
  6                                GROUP BY P_CODE)
  7                                >
  8          ( SELECT SUM(L_UNITS)/COUNT(L_UNITS)  FROM LINE
  9                                        WHERE L.P_CODE=P_CODE
 10                                                    GROUP BY P_CODE) ;


P_COD
-----
SB725
RF100
CH10X
PP101
```

****

QUERY-14 : Write a SQL code using correlated query to list all customers who have placed an order. (Use EXISTS clause in SELECT statement).

****

```
SELECT C_CODE , FNAME||' '||LNAME AS C_NAME
2                  FROM CUSTOMER C
3                         WHERE EXISTS (SELECT C_CODE FROM INVOICE I
4                                              WHERE C.C_CODE=I.C_CODE) ;

    C_CODE C_NAME
---------- --------------------
     10011 Elena kurtis
     10012 Kathy Smith
     10014 Bill Johnson
```

```
     10015 Julia Samuels
     10018 Ming Lee
     10020 Grace Dawson

6 rows selected.


********************************** END ****************************************
```