*** EXPERIMENT NO: 04 ***

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

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

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

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

QUERY-01 : Write a 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
2    (SELECT P_CODE AS PT_CODE, DESCRIPT AS PT_DESC, P_PRICE AS PT_PRICE, V_CODE
3                         FROM PRODUCT WHERE 1=2);

Table created.

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

17 rows created.

SELECT      TABLE_NAME, CONSTRAINT_NAME, CONSTRAINT_TYPE
2  FROM USER_CONSTRAINTS
3  WHERE TABLE_NAME IN ('PRODUCT', 'PART') ;

TABLE_NAME  CONSTRAINT_NAME                C
----------- ------------------------------ -
PART        SYS_C0012432                   C
PART        SYS_C0012433                   C
PRODUCT     PRODUCT_FK_V_CODE              R
PRODUCT     SYS_C0011457                   C
PRODUCT     SYS_C0011458                   C
PRODUCT     SYS_C0011459                   C
PRODUCT     SYS_C0011460                   C
PRODUCT     SYS_C0011461                   C
PRODUCT     SYS_C0011462                   C
PRODUCT     PRODUCT_CK_P_MIN               C
PRODUCT     PRODUCT_PK_P_CODE              P
```

```
11 rows selected.
```

/***INFERENCE** –

Entity Integrity Constraint and Referential Integrity Constraint are not applied on the table which is created using other table

*/

```
*********************************************************************************
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 the number of vendors referenced in PRODUCT who have supplied
            products with prices less than or equal to 10.
*********************************************************************************
```

```
SELECT DISTINCT V_CODE, V_NAME FROM VENDOR
2          WHERE V_CODE IN (SELECT V_CODE FROM PART);

    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.
```

```
SELECT COUNT(DISTINCT V_CODE) AS NO_OF_VENDORS FROM PRODUCT
2            WHERE P_PRICE < 11;

NO_OF_VENDORS
-------------
            4
```

```
*******************************************************************************
QUERY-03 :  Write a 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 PT_CODE, PT_DESC, PT_PRICE
              FROM PART
        WHERE PT_PRICE = (SELECT MAX(PT_PRICE) FROM PART )
                 OR
              PT_PRICE = (SELECT MIN(PT_PRICE) FROM PART )
             ORDER BY PT_PRICE DESC;
```

```
PT_CO PT_DESC                           PT_PRICE
----- ------------------------------ ----------
HC100 Hicut Chain Saw                    256.99
RF100 Rat Tail File                        4.99
```

```
SELECT P_CODE, DESCRIPT, P_PRICE
              FROM PRODUCT
        WHERE P_PRICE = (SELECT MAX(P_PRICE) FROM PRODUCT )
                 OR
              P_PRICE = (SELECT MIN(P_PRICE) FROM PRODUCT )
             ORDER BY P_PRICE DESC;
```

```
P_COD DESCRIPT                           P_PRICE
----- ------------------------------ ----------
HC100 Hicut Chain Saw                    256.99
RF100 Rat Tail File                        4.99
```

```
*******************************************************************************
QUERY-04 Write a SQL code that will retrieve the total amount owed by the customers
         As TOT_BALANCE. Also compute the total value of all items carried in
         inventory.
********************************************************************************
```

```
SELECT  C_CODE, SUM(L_PRICE) AS TOT_BALANCE
          FROM LINE, ( SELECT C_CODE, INV_NUM AS I_NUM
                         FROM INVOICE )
               WHERE I_NUM = INV_NUM
           GROUP BY C_CODE ;
```

```
    C_CODE TOT_BALANCE
---------- -----------
     10015       19.98
     10014      408.79
     10011      146.63
     10012       93.89
     10018       14.94
     10020         5.5

6 rows selected.
```

```
SELECT SUM(P_PRICE)
  2         FROM PRODUCT;


SUM(P_PRICE)
------------
      864.36

*******************************************************************************
QUERY-05 :  Write a 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 which are supplied
            by each vendor.
*********************************************************************************
```

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

P_COD DESCRIPT             P_DATE      QTY  P_MIN    P_PRICE     P_DISC     V_CODE
----- -------------------- --------- ---- ------ ---------- ---------- ----------
HC100 Hicut Chain Saw      07-FEB-12   11      5     256.99        .05      24288
JB012 Jigsaw 12in Blade    30-DEC-11    8      5     109.92        .05      24288
JB008 Jigsaw 8in Blade     24-DEC-11    6      5      99.87        .05      24288
CL025 Hrd. Cloth 1/4 in    15-JAN-12   15      8      61.74          0      23119
CD00X Cordless Drill       20-JAN-12   12      5      61.74        .05      25595
AB112 Power Painter        03-NOV-11    8      5      61.74          0      25595
SM48X Steel Malting Mesh   17-JAN-12   18      5      61.74         .1      25595
CL050 Hrd. Cloth 1/2 in    15-JAN-12   23      5      61.74          0      23119

8 rows selected.
```

```
SELECT V_CODE, COUNT(P_CODE) AS NO_OF_PRODUCTS
2       FROM PRODUCT WHERE V_CODE IS NOT NULL
3         GROUP BY(V_CODE);

    V_CODE NO_OF_PRODUCTS
---------- --------------
     25595              3
     23119              2
     24992              1
     21231              1
     21225              2
     24288              3
     21344              3

7 rows selected.
```

```
******************************************************************************
QUERY-06 :  Write a 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 500.00
            and the high value vendor is placed last.
*******************************************************************************
```

```
SELECT V_CODE, COUNT(P_CODE) AS NO_OF_PRODUCTS FROM PRODUCT
             WHERE P_PRICE < 10 AND V_CODE IS NOT NULL
                GROUP BY(V_CODE);

    V_CODE NO_OF_PRODUCTS
---------- --------------
     24992              1
     21231              1
     21225              2
     21344              1
```

```
SELECT V_CODE , TOT_COST FROM
             (SELECT V_CODE, SUM(P_PRICE*QTY)
                             AS TOT_COST FROM PRODUCT
                                      WHERE V_CODE IS NOT NULL
                            GROUP BY(V_CODE))
           WHERE TOT_COST > 500
            ORDER BY TOT_COST DESC;

    V_CODE   TOT_COST
---------- ----------
     24992       5275
     24288    4305.47
     25595    2346.12
     23119    2346.12
     21231    2002.65
     21225    1431.13
     21344    1009.07

7 rows selected.
```

****
QUERY-07 : Write a 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
2  SELECT * FROM PRODUCT;

View created.

SELECT V_CODE, SUM(P_PRICE*QTY) AS TOT_COST,
  2             MIN(P_PRICE) AS MN_QTY, MAX(P_PRICE) AS MX_QTY, AVG(QTY) AS AV_QTY
  3                         FROM PRODUCT
  4                                 WHERE V_CODE IS NOT NULL
  5             GROUP BY(V_CODE);

    V_CODE   TOT_COST     MN_QTY     MX_QTY     AV_QTY
---------- ---------- ---------- ---------- ----------
     25595    2346.12      61.74      61.74 12.6666667
     23119    2346.12      61.74      61.74         19
     24992       1150       5.75       5.75       5.75
     21231    2002.65       8.45       8.45        237
     21225    1431.13       6.99       9.95       97.5
     24288    4305.47      99.87     256.99 8.33333333
     21344    1009.07       4.99      17.49         31

7 rows selected.
```

****
QUERY-08 : Write a SQL code to count the number of invoices. Also count the number
            Of customers with a customer balance over 500.
****

```
SELECT COUNT(INV_NUM) AS LIST_OF_INVOICES
2           FROM INVOICE;

LIST_OF_INVOICES
----------------
               9
```

```
SELECT COUNT(C_CODE) AS NO_OF_CUSTOMER
             FROM CUSTOMER
       WHERE BALANCE > 500;
```

```
NO_OF_CUSTOMER
--------------
             2
```

****
QUERY-09 : 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 CUS_CODE, BALANCE, AGGREGATE_AMOUNT
     FROM CUSTOMER, (SELECT CUS_CODE, SUM(L_PRICE) AS AGGREGATE_AMOUNT
               FROM LINE, (SELECT INV_NUM AS I_NUM,C_CODE AS CUS_CODE FROM INVOICE)
                               WHERE I_NUM=INV_NUM
                      GROUP BY CUS_CODE )
         WHERE CUS_CODE=C_CODE;
```

```
  CUS_CODE    BALANCE AGGREGATE_AMOUNT
---------- ---------- ----------------
     10015          0            19.98
     10020     452.99              5.5
     10014          0           408.79
     10012     345.86            93.89
     10011          0           146.63
     10018     216.55            14.94

6 rows selected.
```

****
QUERY-10 : Modify Query-09 to include the number of individual product purchases made by each customer. (If the customer‘s invoice is based on three products, on e 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 CUS_CODE, BALANCE,NO_OF_PRODUCTS, AGGREGATE_AMOUNT
          FROM CUSTOMER, (SELECT CUS_CODE,COUNT(INV_NUM) AS
                            NO_OF_PRODUCTS,SUM(L_PRICE) AS AGGREGATE_AMOUNT
                      FROM LINE, (SELECT INV_NUM AS I_NUM,C_CODE AS CUS_CODE
                               FROM INVOICE)
6                              WHERE I_NUM=INV_NUM
7                 GROUP BY CUS_CODE )
8         WHERE CUS_CODE=C_CODE;


  CUS_CODE    BALANCE NO_OF_PRODUCTS AGGREGATE_AMOUNT
---------- ---------- -------------- ----------------
     10015          0              2            19.98
     10020     452.99              1              5.5
     10014          0              6           408.79
     10012     345.86              3            93.89
     10011          0              5           146.63
     10018     216.55              2            14.94

6 rows selected.
```

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

QUERY-11 : Write a SQL query to compute the average purchase amount per product made by each customer. (Note: the average purchase amount is equal to the total purchases divided by the number of purchases.)

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

```
SELECT C_CODE,SUM(L_PRICE)/COUNT(INV_NUM) AS AVERAGE_PURCHASE
2                      FROM LINE, (SELECT INV_NUM AS I_NUM,C_CODE  FROM INVOICE)
3                 WHERE I_NUM=INV_NUM
4            GROUP BY C_CODE ;

    C_CODE AVERAGE_PURCHASE
---------- ----------------
     10015             9.99
     10014       68.1316667
     10011           29.326
     10012       31.2966667
     10018             7.47
     10020              5.5

6 rows selected.
```

****

QUERY-12 :   Write a 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,COUNT(INV_NUM) AS  NO_OF_PRODUCTS
                     FROM LINE, (SELECT INV_NUM AS I_NUM,C_CODE AS CUS_CODE
                               FROM INVOICE)
                                WHERE I_NUM=INV_NUM
                   GROUP BY INV_NUM
              ORDER BY INV_NUM;
```

```
   INV_NUM NO_OF_PRODUCTS
---------- --------------
      1001              2
      1002              1
      1003              3
      1004              2
      1005              1
      1006              4
      1007              2
      1008              3
      1009              1

9 rows selected.
```

```
SELECT C_CODE,INV_NUM,COUNT(INV_NUM) AS  NO_OF_PRODUCTS
                     FROM LINE, (SELECT INV_NUM AS I_NUM,C_CODE
                               FROM INVOICE)
                                WHERE I_NUM=INV_NUM
                   GROUP BY C_CODE,INV_NUM
                 ORDER BY C_CODE;
```

```
    C_CODE    INV_NUM NO_OF_PRODUCTS
---------- ---------- --------------
     10011       1002              1
     10011       1005              1
     10011       1008              3
     10012       1003              3
     10014       1001              2
     10014       1006              4
     10015       1007              2
     10018       1004              2
     10020       1009              1
```

```
9 rows selected.
```

```
SELECT C_CODE,COUNT(INV_NUM) AS  NO_OF_PRODUCTS, SUM(L_PRICE) AS PRICE
                         FROM LINE, (SELECT INV_NUM AS I_NUM,C_CODE
                                   FROM INVOICE)
                                    WHERE I_NUM=INV_NUM
                       GROUP BY C_CODE ;
```

```
    C_CODE NO_OF_PRODUCTS      PRICE
---------- ------------- ----------
     10015             2      19.98
     10014             6     408.79
     10011             5     146.63
     10012             3      93.89
     10018             2      14.94
     10020             1        5.5

6 rows selected.
```

```
*********************************************************************************
QUERY-13 :  Using the results of Query-12, write a SQL code to generate the total
            Number of invoices, the invoice total for all of the invoices, the
            smallest invoice amount, the largest invoice amount, and the average of
            all of the invoices.
**********************************************************************************
```

```
SELECT COUNT(INV_NUM) AS TOTAL_NO_OF_INVOICE,
             SUM(L_PRICE) AS INVOICE_TOTAL,
                      MIN(L_PRICE) AS MINIMUM, MAX(L_PRICE) AS MAXIMUM,
                                   AVG(L_PRICE) AS AVERAGE
            FROM LINE;
```

```
TOTAL_NO_OF_INVOICE INVOICE_TOTAL    MINIMUM    MAXIMUM    AVERAGE
------------------- ------------- ---------- ---------- ----------
                 19        689.73       4.99     256.99 36.3015789
```

****
QUERY-14 :   Write a SQL code to find the customer balance summary for all customerswho have not made purchases during the current invoicing. Use this query to generate a summary of the customer balance characteristics (the output should include the minimum, maximum and average balances over across all purchases).
****

```
SELECT C_CODE, BALANCE
2           FROM CUSTOMER
3       WHERE C_CODE NOT IN (SELECT C_CODE FROM INVOICE);
```

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

```
SELECT MIN(BALANCE) AS MINIMUM_BAL,
2          MAX(BALANCE) AS MAXIMUM_BAL, AVG(BALANCE) AS AVERAGE_BAL
3                FROM CUSTOMER
4       WHERE C_CODE NOT IN (
5             SELECT C_CODE FROM INVOICE);
```

```
MINIMUM_BAL MAXIMUM_BAL AVERAGE_BAL
----------- ----------- -----------
          0      768.93     305.374
```

****
QUERY-15 : Write a SQL code to find the customer balance summary for all customers who have not made purchases during the current invoicing period (the output should include the total balances, the minimum, maximum and average balances over across all purchases). Also compute the total value of the product inventory.
****

```
SELECT MIN(BALANCE) AS MINIMUM_BAL,
2                    MAX(BALANCE) AS MAXIMUM_BAL,
3                        AVG(BALANCE) AS AVERAGE_BAL, SUM(BALANCE) AS TOTAL
4           FROM CUSTOMER
5       WHERE C_CODE NOT IN (
6             SELECT C_CODE FROM INVOICE);
```

```
MINIMUM_BAL MAXIMUM_BAL AVERAGE_BAL      TOTAL
----------- ----------- ----------- ----------
          0      768.93     305.374    1526.87
```

```
SELECT SUM(P_PRICE*QTY) AS TOTAL
  2             FROM PRODUCT;
```

```
     TOTAL
----------
  25292.32
```

Commit;

```
Commit complete.
```

************************************* **END** ***