```
SQL>
SQL>
SQL> --*****************************************************
SQL> --UCS1412                                  B.Senthil Kumar
SQL> --Database LabAsst. Prof
SQL> --        Computer Science Department
SQL> --                  SSN College of Engineering
SQL> --                        senthil@ssn.edu.in
SQL> --*****************************************************
SQL> --           PIZZA ORDERING DATASET
SQL> --                 Version 1.0
SQL> --                February 05, 2015
SQL> --*****************************************************
SQL> --Sources:
SQL> --         This dataset is prepared for the assignment
SQL> --on DML, PL/SQL blocks in Database Programming.
SQL> --This is a test dataset - pizza ordered on 28 & 29th Jun 2015.
SQL> --Do NOT MODIFY the instances.
SQL> --
SQL> --*****************************************************
SQL>
SQL>
SQL> REM customer(cust_id, cust_name, address, phone)
SQL> REM pizza (pizza_id, pizza_type, unit_price)
SQL> REM orders(order_no, cust_id, order_date ,delv_date)
SQL> REM order_list(order_no, pizza_id, qty)
SQL>
SQL> DROP TABLE order_list;

Table dropped.

SQL> DROP TABLE orders;

Table dropped.

SQL> DROP TABLE pizza;

Table dropped.

SQL> DROP TABLE customer;

Table dropped.

SQL>
SQL>
SQL> CREATE TABLE customer(
  2  cust_id VARCHAR(7),
  3  cust_name VARCHAR(25),
  4  address VARCHAR(75),
  5  phone NUMBER(10),
  6  CONSTRAINT pk_customer PRIMARY KEY(cust_id));

Table created.

SQL>
SQL> CREATE TABLE pizza(
  2  pizza_id VARCHAR(6),
  3  pizza_type VARCHAR(15),
  4  unit_price NUMBER(5),
  5  CONSTRAINT pk_pizza PRIMARY KEY(pizza_id));

Table created.

SQL>
```

```
SQL> CREATE TABLE orders(
  2  order_no VARCHAR(6),
  3  cust_id VARCHAR(6),
  4  order_date DATE,
  5  delv_date DATE,
  6  CONSTRAINT pk_orders PRIMARY KEY(order_no),
  7  CONSTRAINT fk_custid FOREIGN KEY(cust_id) REFERENCES customer(cust_id));

Table created.

SQL>
SQL> CREATE TABLE order_list(
  2  order_no VARCHAR(6),
  3  pizza_id VARCHAR(6),
  4  qty NUMBER,
  5  CONSTRAINT pk_orderlist PRIMARY KEY(order_no, pizza_id),
  6  CONSTRAINT fk_orderno FOREIGN KEY(order_no) REFERENCES orders(order_no),
  7  CONSTRAINT fk_pizzaid FOREIGN KEY(pizza_id) REFERENCES pizza(pizza_id));

Table created.

SQL>
SQL> DESC customer;
 Name
 Null?    Type

 ------------------------------------------------------------------------------
 -------- ------------------------------------------------------------
 CUST_ID
 NOT NULL VARCHAR2(7)
 CUST_NAME
 VARCHAR2(25)
 ADDRESS
 VARCHAR2(75)
 PHONE
 NUMBER(10)

SQL> DESC pizza;
 Name
 Null?    Type

 ------------------------------------------------------------------------------
 -------- ------------------------------------------------------------
 PIZZA ID
 NOT NULL VARCHAR2(6)
 PIZZA_TYPE
 VARCHAR2(15)
 UNIT_PRICE
 NUMBER(5)

SQL> DESC orders;
 Name
 Null?    Type

 ------------------------------------------------------------------------------
 -------- ------------------------------------------------------------
 ORDER_NO
 NOT NULL VARCHAR2(6)
 CUST_ID
 VARCHAR2(6)
 ORDER_DATE
 DATE
 DELV_DATE
 DATE
```

```
SQL> DESC order_list;
 Name
 Null?    Type

 ----------------------------------------------------------------------------
 -------- ------------------------------------------------------
 ORDER_NO
 NOT NULL VARCHAR2(6)
 PIZZA_ID
 NOT NULL VARCHAR2(6)
 QTY
 NUMBER

SQL>
SQL>
SQL> REM
------------------------------------------------------------------------------
------------
>
SQL> REM customer(cust_id, cust_name,address,phone)
SQL>
SQL> insert into customer values('c001','Hari','32 RING
ROAD,ALWARPET',9001200031);

1 row created.

SQL> insert into customer values('c002','Ashok','42 bull
ROAD,numgambakkam',9444120003);

1 row created.

SQL> insert into customer values('c003','Raj','12a RING
ROAD,ALWARPET',9840112003);

1 row created.

SQL> insert into customer values('c004','Raghu','P.H
ROAD,Annanagar',9845712993);

1 row created.

SQL> insert into customer values('c005','Sindhu','100 feet
ROAD,vadapalani',9840166677);

1 row created.

SQL> insert into customer values('c006','Brinda','GST ROAD, TAMBARAM',
9876543210);

1 row created.

SQL>
SQL>
SQL>
SQL> REM pizza (pizza_id, pizza_type, unit_price)
SQL>
SQL> insert into pizza values('p001','pan',130);

1 row created.

SQL> insert into pizza values('p002','grilled',230);

1 row created.
```

```
SQL> insert into pizza values('p003','italian',200);

1 row created.

SQL> insert into pizza values('p004','spanish',260);

1 row created.

SQL>
SQL> REM insert into pizza values('p005','supremo',250);
SQL>
SQL>
SQL>
SQL> REM orders(order_no, cust_id, order_date ,delv_date)
SQL>
SQL> insert into orders values('OP100','c001','28-JUN-2015','30-JUN-2015');

1 row created.

SQL> insert into orders values('OP200','c002','28-JUN-2015','30-JUN-2015');

1 row created.

SQL> insert into orders values('OP300','c003','29-JUN-2015','01-JUL-2015');

1 row created.

SQL> insert into orders values('OP400','c004','29-JUN-2015','01-JUL-2015');

1 row created.

SQL> insert into orders values('OP500','c001','29-JUN-2015','01-JUL-2015');

1 row created.

SQL> insert into orders values('OP600','c002','29-JUN-2015','01-JUL-2015');

1 row created.

SQL>
SQL>
SQL>
SQL> REM order_list(order_no, pizza_id, qty)
SQL>
SQL> insert into order_list values('OP100','p001',3);

1 row created.

SQL> insert into order_list values('OP100','p002',2);

1 row created.

SQL> insert into order_list values('OP100','p003',1);

1 row created.

SQL> insert into order_list values('OP100','p004',5);

1 row created.

SQL>
SQL> insert into order_list values('OP200','p003',2);
```

```
1 row created.

SQL> insert into order_list values('OP200','p001',6);

1 row created.

SQL> insert into order_list values('OP200','p004',8);

1 row created.

SQL>
SQL> insert into order_list values('OP300','p003',3);

1 row created.

SQL>
SQL> insert into order_list values('OP400','p001',3);

1 row created.

SQL> insert into order_list values('OP400','p004',1);

1 row created.

SQL>
SQL> insert into order_list values('OP500','p003',6);

1 row created.

SQL> insert into order_list values('OP500','p004',5);

1 row created.

SQL> insert into order_list values('OP500','p001',null);

1 row created.

SQL>
SQL> insert into order_list values('OP600','p002',3);

1 row created.

SQL>
SQL> --*****************************************************
SQL> set echo on:
SP2-0158: unknown SET option ":"
SQL> set serveroutput on format wrapped
SQL> -- To prevent dbms_output from trimming leading spaces
SQL>
SQL> @z:/Pizza DB.sql
SP2-0310: unable to open file "z:/Pizza_DB.sql"
SQL>
SQL> REM: ************************************************************Ex6 -
STORED PROCEDURES AND STORED
FUNCTIONS*****************************************************************
SQL> REM: PIZZA ORDERING SYSTEM
SQL>
SQL>
SQL> REM: Consider the following relations for Pizza Ordering System:
SQL> REM: CUSTOMER ( cust_id , cust_name, address, phone, cust_friend)
SQL> REM: PIZZA (pizza_id, pizza_type, unit_price)
SQL> REM: ORDERS (order_no, cust_id, order_date ,delv_date, total_amt,
discount, bill_amt)
SQL> REM: ORDER_LIST (order_no, pizza_id, qty)
```

```
SQL>
SQL> REM: Write a PL/SQL stored procedure / stored function for the following:
SQL> REM: Note:
SQL> REM: a. Use implicit/explicit cursor wherever required.
SQL> REM: b. Handle the error and display appropriate message if the data is
non-available.
SQL> REM: c. Add necessary attributes to ORDERS.
SQL>
SQL> ALTER TABLE orders ADD total_amt NUMBER;

Table altered.

SQL> ALTER TABLE orders ADD discount NUMBER;

Table altered.

SQL> ALTER TABLE orders ADD bill_amt NUMBER;

Table altered.

SQL>
SQL> REM: 1. Write a stored function to display the total number of pizza's
ordered
SQL> REM:    by the given  order number.
SQL>
SQL> CREATE OR REPLACE FUNCTION total_pizza(ordernum orders.order_no%TYPE)
  2  RETURN NUMBER
  3  AS
  4  tot_qty NUMBER;
  5  BEGIN
  6  SELECT SUM(ol.qty)
  7  INTO tot_qty
  8  FROM orders o, customer c, pizza p, order_list ol
  9  WHERE c.cust_id=o.cust_id
 10  AND o.order_no=ol.order_no
 11  AND ol.pizza_id=p.pizza_id
 12  AND o.order_no=ordernum;
 13
 14  RETURN tot_qty;
 15  END;
 16  /

Function created.

SQL>
SQL> VAR tot NUMBER
SQL> EXECUTE :tot := total_pizza('OP100');

PL/SQL procedure successfully completed.

SQL> PRINT tot

       TOT
----------
        11

SQL>
SQL> REM: 2. Write a PL/SQL block to calculate the total amount, discount and
billable amount
SQL> REM:(Amount to be paid) as given below:
SQL> REM:For total amount > 2000 and total amount < 5000:  Discount=5%
SQL> REM:For total amount > 5000 and total amount < 10000:  Discount=10%
SQL> REM:For total amount > 10000:  Discount=20%
SQL> REM:Calculate the billable amount (after the discount) and
```

```
SQL> REM:update the same in orders  table.
SQL> REM:Bill Amount = Total - Discount.
SQL>
SQL> SELECT * FROM orders;

ORDER_ CUST_I ORDER_DAT DELV_DATE  TOTAL_AMT   DISCOUNT   BILL_AMT
------ ------ --------- --------- ---------- ---------- ----------
OP100  c001   28-JUN-15 30-JUN-15
OP200  c002   28-JUN-15 30-JUN-15
OP300  c003   29-JUN-15 01-JUL-15
OP400  c004   29-JUN-15 01-JUL-15
OP500  c001   29-JUN-15 01-JUL-15
OP600  c002   29-JUN-15 01-JUL-15

6 rows selected.

SQL>
SQL> CREATE OR REPLACE PROCEDURE calc_bill(ordernum orders.order_no%TYPE)
  2  AS
  3  CURSOR c_orders(o_num orders.order_no%TYPE)
  4  IS
  5  SELECT o.order_no, p.pizza_id, p.pizza_type, ol.qty, p.unit_price
  6  FROM orders o, pizza p, order_list ol
  7  WHERE o.order_no=ol.order_no
  8  AND ol.pizza id=p.pizza_id
  9  AND o.order_no=o_num;
 10
 11  r_orders orders%ROWTYPE;
 12
 13  BEGIN
 14  UPDATE orders SET total_amt=0 WHERE order_no=ordernum;
 15
 16  FOR record IN c_orders(ordernum)
 17  LOOP
 18  IF record.qty IS NULL THEN
 19  record.qty := 0;
 20  END IF;
 21
 22  UPDATE orders
 23  SET total_amt=total_amt+(record.qty * record.unit_price)
 24    WHERE order_no=ordernum;
 25  END LOOP;
 26
 27  SELECT *
 28  INTO r_orders
 29  FROM orders
 30  WHERE order_no=ordernum;
 31
 32  IF r_orders.total_amt BETWEEN 2001 AND 4999 THEN
 33  UPDATE orders
 34  SET discount=5
 35  WHERE order_no=ordernum;
 36
 37  ELSIF r_orders.total_amt BETWEEN 5001 AND 9999 THEN
 38  UPDATE orders
 39  SET discount=10
 40  WHERE order_no=ordernum;
 41
 42  ELSIF r_orders.total_amt > 10000 THEN
 43  UPDATE orders
 44  SET discount=20
 45  WHERE order_no=ordernum;
 46
 47  ELSE
```

```
 48  UPDATE orders
 49  SET discount=0
 50  WHERE order_no=ordernum;
 51  END IF;
 52
 53  UPDATE orders
 54  SET bill_amt=total_amt-(discount*0.01*total_amt)
 55  WHERE order_no=ordernum;
 56
 57  END;
 58  /

Procedure created.

SQL>
SQL> EXEC calc_bill('OP500');

PL/SQL procedure successfully completed.

SQL> SELECT * FROM orders;

ORDER_ CUST_I ORDER_DAT DELV_DATE  TOTAL_AMT   DISCOUNT   BILL_AMT
------ ------ --------- --------- ---------- ---------- ----------
OP100  c001   28-JUN-15 30-JUN-15
OP200  c002   28-JUN-15 30-JUN-15
OP300  c003   29-JUN-15 01-JUL-15
OP400  c004   29-JUN-15 01-JUL-15
OP500  c001   29-JUN-15 01-JUL-15       2500          5       2375
OP600  c002   29-JUN-15 01-JUL-15

6 rows selected.

SQL>
SQL> REM: 3. For the given order number,
SQL> REM: write a PL/SQL block to print the order as shown below:
SQL> REM: Hint: Use the PL/SQL blocks created in 1 and 2.
SQL>
SQL> REM: ************************************************************
SQL> REM: Order Number:OP104Order Date :29-Jun-2015
SQL> REM: Customer Name: HariPhone: 9001200031
SQL> REM: ************************************************************
SQL> REM: SNoPizzaTypeQtyPriceAmount
SQL> REM: 1. Italian6 200 1200
SQL> REM: 2. Spanish5 260 1300
SQL> REM: ------------------------------------------------------
> REM: Total =  11 2500
SQL> REM: ------------------------------------------------------
> REM: Total Amount  :Rs.2500
SQL> REM: Discount (5%) :Rs. 125
SQL> REM: ------------------------- ---
> REM: Amount to be paid :Rs.2375
SQL> REM: ------------------------- ---
> REM: Great Offers! Discount up to 25% on DIWALI Festival Day...
SQL> REM: ************************************************************
SQL>
SQL> CREATE OR REPLACE PROCEDURE print_bill(ordernum orders.order_no%TYPE)
  2  AS
  3  CURSOR c_orders(o_num orders.order_no%TYPE)
  4  IS
  5  SELECT o.order_no, o.order_date, c.cust_name, c.phone, p.pizza_type,
  ol.qty, p.unit_price
  6  FROM orders o, pizza p, order_list ol, customer c
  7  WHERE c.cust_id=o.cust_id
  8  AND o.order_no=ol.order_no
```

```
 9  AND ol.pizza_id=p.pizza_id
10  AND o.order_no=ordernum;
11
12  CURSOR c_cust_details(o_num orders.order_no%TYPE)
13  IS
14  SELECT o.order_no, o.order_date, c.cust_name, c.phone
15  FROM orders o, customer c
16  WHERE c.cust_id=o.cust_id
17  AND o.order_no=ordernum;
18
19  r_orders c_orders%ROWTYPE;
20  r_cust_details c_cust_details%ROWTYPE;
21  r_table_order orders%ROWTYPE;
22  counter NUMBER;
23  BEGIN
24
25  OPEN c_cust_details(ordernum);
26  LOOP
27  FETCH c cust details INTO r cust details;
28  EXIT WHEN c_cust_details%NOTFOUND;
29
30
dbms_output.Put_line('**************************************************
*****');
31  dbms output.Put line('Order Number: '||r cust details.order_no||
LPAD('Customer Name: ', 36)||r_cust_details.cust_name);
32  dbms_output.Put_line('Order Date: '||r_cust_details.order_date||
LPAD('Phone: ', 28)||r_cust_details.phone);
33
dbms_output.Put_line('**************************************************
*****');
34  dbms_output.Put_line('SNo     Pizza Type     Qty   Price     Amount');
35
36  END LOOP;
37  CLOSE c_cust_details;
38
39  counter := 0;
40
41  OPEN c_orders(ordernum);
42  LOOP
43  FETCH c_orders INTO r_orders;
44  EXIT WHEN c_orders%NOTFOUND;
45
46  IF r orders.qty IS NULL THEN
47  CONTINUE;
48  END IF;
49
50  counter:=counter+1;
51  dbms_output.Put_line(RPAD(counter||'.', 3)||LPAD(r_orders.pizza_type,
15)||LPAD(r orders.qty, 8)||LPAD(r orders.unit_price, 8)||
LPAD(r_orders.qty*r_orders.unit_price, 11));
52  END LOOP;
53  CLOSE c_orders;
54
55  calc_bill(ordernum);
56
57  SELECT *
58  INTO r_table_order
59  FROM orders
60  WHERE order_no=ordernum;
61
62
dbms_output.Put_line('-------------------------------------------------');
63  dbms_output.Put_line(LPAD('   ', 3)||LPAD('Total = ', 15)||
```

```
 LPAD(total_pizza(ordernum), 8)||LPAD('     ', 8)||
 LPAD(r_table_order.total_amt, 11));
 64
 dbms_output.Put_line('-------------------------------------------------');
 65  dbms_output.Put_line('Total Amount      :Rs.'||r_table_order.total_amt);
 66  dbms_output.Put_line('Discount ('||r_table_order.discount||'%)
 :Rs.'||r_table_order.total_amt*r_table_order.discount*0.01);
 67  dbms_output.Put_line('---------------------------------');
 68  dbms_output.Put_line('Amount to be paid :Rs.'||r_table_order.bill_amt);
 69  dbms_output.Put_line('---------------------------------');
 70
 71  dbms_output.Put_line('Great Offers! Discount up to 25% on DIWALI Festival
 Day...');
 72
 dbms_output.Put_line('**********************************************************
 *****');
 73
 74  END;
 75  /

Procedure created.

SQL>
SQL> EXEC print_bill('OP500');
************************************************************
Order Number: OP500                      Customer Name: Hari
Order Date: 29-JUN-15                      Phone: 9001200031
************************************************************
SNo     Pizza Type     Qty   Price     Amount
1.         italian       6     200       1200
2.         spanish       5     260       1300
-----------------------------------------------
          Total =       11               2500
-----------------------------------------------
Total Amount      :Rs.2500
Discount (5%)     :Rs.125
---------------------------------
Amount to be paid :Rs.2375
---------------------------------
Great Offers! Discount up to 25% on DIWALI Festival Day...
************************************************************

PL/SQL procedure successfully completed.

SQL>
SQL>
SQL> spool off
```