Consider the following relations in a database created for an online store

```
create table Customers
(
    cid char(4) primary key,
    name VARCHAR(50),
    phone char(10),
    country VARCHAR(20)
)

create table Employees
(
    eid char(4) primary key,
    ename varchar(50),
    phone char(10),
    birthdate date
)

create table Products
(
    productId char(4) primary key,
    productName varchar(15),
    unitPrice real,
    unitInStock int,
    ROL int
)

create table Orders
(
    oid int primary key,
    eid char(4) references Employees(eid),
    cid char(4) references Customers(cid),
    orderDate date,
    requiredDate date,
    shippedDate date,
    cost real
)

create table orderDetails
(
    oid int,
    productId char(4) references Products(productId),
    quantity int,
    discount real,
    constraint orderDetails_pk PRIMARY KEY(oid,productId)
)
```

```
insert into Customers values('C001','Saman','0772446552','Sri Lanka')
insert  into  Customers  values('C002','John','0987665446','USA')
insert into Customers values('C003','Mashato','0927665334','Japan')

insert into Employees values('E001','Kasun Weerasekara','0702994459','07-Apr-1997')
insert into Employees values('E002','Sathira Wijerathna','0760510056','05-Feb-1996')

insert into Products values('P001','Hard Disk',12000,80,50)
insert into Products values('P002','Flash Drive',3200,60,20)
insert into Products values('P003','LCD Monitor',24000,35,15)

INSERT into Orders VALUES(1,'E001','C001','01-Sep-2020','09-Sep-2020','02-Sep-2020',null)

INSERT into orderDetails VALUES(1,'P001',3,0.1);
INSERT into orderDetails VALUES(1,'P002',5,0.15);
INSERT into orderDetails VALUES(1,'P003',2,0.15);
```

Write SQL functions for the following questions

1. Write a scalar function named **calcCost** that will compute and return the total cost of an order based on an order id provided. Take note that the cost of the order is the sum of the costs associated with each product ordered, with the costs associated with each product ordered calculated by multiplying the unit price of the product by the quantity bought.
   Hint: each order may have multiple rows in the orderDetails table.

2. Write a table valued function named ***productsOfOrder*** which accepts the order id and returns all product names of the order with quantity.

3. Modify the table valued function you wrote in question 2 to display records with only product name has 'disk'.
   Hint: delete the records of the table with product name does not like 'disk' inside the function.

4. Write a table valued function named ***orderInfo*** which accepts the order id and returns the following details of the order.

```
productName   -- name of the product
qty           -- quantity ordered of the product
unitAmt       -- price of the product
totAmt        -- total price of the product
discount      -- discount of the total amount as per the discount rate in the orderDetails table
payAmt        -- total payable of the product (totAmt - discount)
```

   The following is an example of the output of the aforesaid function.

	productName	qty	unitAmt	totAmt	discount	payAmt
1	Hard Disk	3	12000	36000	3600	32400
2	Flash Drive	5	3200	16000	2400	13600
3	LCD Monitor	2	24000	48000	7200.0005	40800

5. Modify the function you wrote in question 04, that the discount is only applicable for the products ordered more than 2.
   Sample output of the modified function should be as follows.

	productName	qty	unitAmt	totAmt	discount	payAmt
1	Hard Disk	3	12000	36000	3600	32400
2	Flash Drive	5	3200	16000	2400	13600
3	LCD Monitor	2	24000	48000	0	48000