Practice Exam 1

This is a practice exam from one of my earlier courses. Please note that the format of your exam will be different from this exam. Please use the questions in this exam to review some of the concepts that we have covered in the class. I have also provided answers to the questions.

PART I: True or False. (8 points)

State whether the statement is true of false. You must **justify why** this statement is true or false (you will not get any credit by just -- accidentally -- getting true or false correct). You may use hypothetical examples to articulate your thoughts. Correcting or re-writing the statement is not justification. You must say WHY.

1. An attribute, which is a foreign key in a table, can take a NULL value under all situations. T / F (2 points)

False. Foreign key cannot take a NULL value when it is part of a composite primary key since no part of a primary key can take NULL value as it violates entity integrity rule.
(Note – some stated when we put NO NULL constraint then it cannot take NULL value. But, this applies to any attribute. Also, many are confusing FK of one table and PK of another table. I DISCUSSED THE SAME IN CLASS)

2. Consider the following two sets of table. The underlined attribute is the primary key (PK). Both sets of table are equivalent since they represent one-to-many relationship. (2 points)
Table Set A:
Customer (
CustID (PK),
Custname,
CustAddress
)

Account (
AcctID (PK),
AcctBalance,
CustID (FK)
)

Table Set B:
Customer (
CustID (PK),
Custname,
CustAddress
)

Account (
AcctID,
CustID (FK),
AcctBalance
)

****NOTE : AcctID and CustID are a composite PK**

FALSE: Set A suggests, an account has only ONE owner but the owner can have many accounts. Set B suggests that an account can be owned by many customers while a customer can own many accounts (many-to-many).
(Note: Some stated, in Set A CustID can take NULL but not in Set B. But that is by definition. The question is about being similar and have 1-many relationship.)

3. Consider the following table definitions:

```
CREATE Table Category (
        CategoryID char(5) PRIMARY KEY,
        CatName varchar(25)
);

CREATE Table Item (
        ItemID char(5) PRIMARY KEY,
        ItemName varchar(25),
        CategoryID REFERENCES Category (CategoryID) ON DELETE CASCADE
);

CREATE table Time (
        TimeID char(5) PRIMARY KEY,
        Year number(4),
        Month varchar(10)
);

CREATE Table Fact (
        ItemID REFERENCES Item (ItemID),
        TimeID REFERENCES Time (TimeID),
        Amount Number
);
```

Answer the following based the above DDL:

a) If you delete a CategoryID from Category table then it will result in deletion of all corresponding records in the table Item. (T/F) (3 points)

False – Cascading delete from Category to Item table implies there may be some ItemID that can be deleted that are linked to Fact table. **However, if there are records in Fact table related to those ItemID, that delete is restricted.**
(Note: Sadly only one student got this correct in the past. We discussed this with an example in class).

b) If you delete a record from Fact table, it will delete all corresponding records in the Item table. (T/F) (1 point)

FALSE – Deleting a child record has no bearing on the parent.

Part II: Table Design (worth 3 points)

Consider the following four tables. Underlined attribute is the primary key in each table.

Table 1: Product(ProductID, ProdName, QtyonHand)
Table 2: Customer(CustID, CustName, CustAddress)
Table 3: Orders(OrderID, CustomerID (FK), ProductID (FK), OrderDate, NumOfUnits, Amount)
Table 4: ServiceRequest(RequestID, OrderID (FK), ServiceDate, ServiceCharge)

a. The above set of tables assumes that a customer can place an order for only one product at a time. How would you change the tables to let a customer place an order with multiple products and identify service requests accordingly? (3 points)

No changes to Tables 1 and 2
Table 3: Orders(OrderID, CustomerID, OrderDate)
Table 4: OrderDetails (OrderID(FK), ProductID(FK), NumOfUnits, Amount) Alternatively: OrderDetails (OrdDetailID, OrderID(FK), ProductID(FK), NumofUnits, Amount)
Table 5: ServiceRequest(RequestID, OrderID, ProductID,ServiceDate, ServiceCharge)
Alternatively: ServiceRequest(RequestID, OrdDetailID (FK),ServiceDate, ServiceCharge)

Part III: SQL Queries (worth 39 points)

All queries are based on the following set of tables discussed in part II. Underlined attribute is the primary key in each table.

Table 1: Product(ProductID, ProdName, QtyonHand)
Table 2: Customer(CustID, CustName, CustAddress)
Table 3: Orders(OrderID, CustomerID (FK), ProductID (FK), OrderDate, NumOfUnits, Amount)
Table 4: ServiceRequest(RequestID, OrderID (FK), ServiceDate, ServiceCharge)

1. Identify the top three products with the most number of service requests. Please note that there can be multiple products with similar number of service requests. (5 points)

```
SELECT productID, COUNT(RequestID) as SumRequest
FROM Orders O, ServiceRequest S
WHERE O.orderID = S.OrderID
GROUP BY productid
ORDER BY SumRequest DESC
FETCH TOP 3 with TIES;
```

2. Find all products where the total service charges are greater than the average of total order amounts of all products. (6 points)

```
SELECT B.productID, A.AvgProdAmount, B.SumCharges FROM
        (SELECT AVG(SUM(Amount) AvgProdAmount
        FROM Orders
        GROUP BY productid) A,
        (SELECT ProductID, SUM(Servicecharge) SumCharges FROM Orders, ServiceRequest
        WHERE orders.orderid = ServiceRequest.orderid GROUP BY ProductID) B
WHERE B.SumCharges > A.AvgProdAmount;
```

(NOTE: Some just took average of servicecharge or amount from ServiceRequest or Orders, respectively. You have to SUM and then take average for Servicecharge.

3. Find the total revenue from each product (i.e., total amount + total service charges) in each quarter. Display the quarter in columns. (7 points).
Hint – To get the quarter, use to_char(date_field, 'Q')*

```
With SumOrder as (SELECT ProductID, to_char(OrderDate, 'Q') as Quart, SUM(Amount) as SumAmt
FROM Orders
GROUP BY ProductID),
```

```
Charges as (SELECT ProductID, to_char(OrderDate, 'Q') as Quart, SUM(Servicecharges) as SumCharges
FROM orders, servicerequest
WHERE ServiceRequest.OrderID = Orders.OrderID
GROUP BY ProductID)
SELECT * FROM (
SELECT ProductID, Quart, (SumAmt + SumCharges) as SumTotal
FROM SumOrder LEFT Outer JOIN Charges ON Sumorder.productD = Charges.ProductID AND
sumorder.quart = charges.quart)
PIVOT (
Sum(SumTotal)
FOR Quart in (1 as 'Q1', 2 as 'Q2', 3 as 'Q3', 4 as 'Q4'));
```

IMPORTANT: many have lost points in the past on Q4 because you always assumed every order will have service charges. What if there are NO SERVICE CHARGES? You have to use LEFT OUTER JOIN