**Week 10 Day 4 Self Practice Challenges**

# 

# Question 1:

**Objective :**

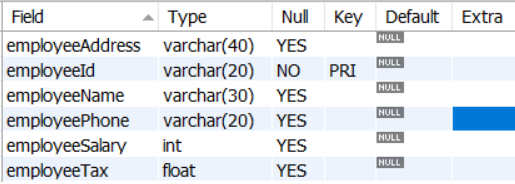
To enable the Mfg Based business to automate the Data querying activity that is carried out at the Database level.

- using Stored Procedures

In order to reduce the Network traffic, Load & bandwidth, you are asked to create a Stored Procedure which takes the EmployeeId as an Input parameter and which displays the Employee Details like

EmployeeName,EmployeeAddress, EmployeePhone & Salary details

Structure of Employee Table:



**Stub Code**

delimiter $$

// Procedure Code Here

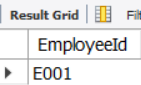
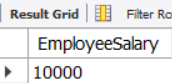
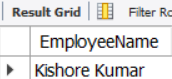
end $$

**Sample Output 1:**

**For a Procedure Call as below:**

**call fetchEmployeeDetailsMDU1('E001');**

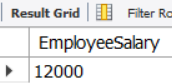
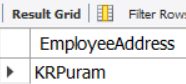
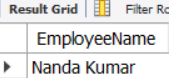
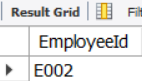
**Output:**

** **

**Sample Output 2:**   
For a call as given below:

call fetchEmployeeDetailsMDU1('E002');

OutPut is :



**Solution:**

delimiter $$

use ctsdatamdu $$

create Procedure fetchEmployeeDetailsMDU1(in empId varchar(30))

begin

declare empName varchar(50);

declare empAddress varchar(100);

declare empPhone varchar(50);

declare empSalary float;

set empName = (select employeeName from employee where employeeId = empId);

set empAddress = (select employeeAddress from employee where employeeId = empId);

set empSalary = (select employeeSalary from employee where employeeId = empId);

set empPhone = (select employeePhone from employee where employeeId = empId);

select empId as 'EmployeeId';

select empName as 'EmployeeName';

select empAddress as 'EmployeeAddress';

select empSalary as 'EmployeeSalary';

end $$

# Learning Objective:

# Learners are able to learn to create the Stored Procedures with input parameters.

# Question 2:

**Objective :**

To enable the Mfg Based business to automate the Data querying activity that is carried out at the Database level.

- using Stored Procedures

In order to reduce the Network traffic, Load & bandwidth, you are asked to create a Stored Procedure which takes the EmployeeId as an Input parameter and which returns the Employee Details like

EmployeeAddress, EmployeePhone,EMail & Salary details as Output parameters

# 

**Stub Code:**

**delimiter $$**

**//Your Code Here**

**end $$**

**Sample Output 1:**

**When procedure is invoked with the following parameters**

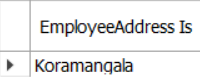
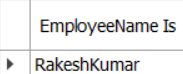
call fetchEmployeeDetailsMDU14('E006',@eName,@eAddress,@eSalary);

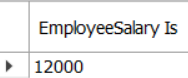
select @eName as 'EmployeeName';

select @eAddress as 'EmployeeAddress';

select @eSalary as 'EmployeeSalary';

Output is as follows:



****

**Sample Output 2:**

**When procedure is invoked with the following parameters**

call fetchEmployeeDetailsMDU14('E007',@eName,@eAddress,@eSalary);

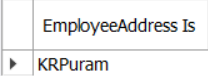
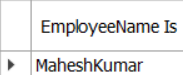
select @eName as 'EmployeeName Is';

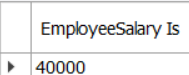
select @eAddress as 'EmployeeAddress Is';

select @eSalary as 'EmployeeSalary Is';

Output is as follows:



** **

****

**Solution:**

delimiter $$

use ctsdata $$

create Procedure fetchEmployeeDetailsMDU14(in empId varchar(20),out empName varchar(30),out empAddress varchar(40), out empSalary float)

begin

set empName =(select employeeName from employee where employeeId=empId);

set empAddress = (select employeeAddress from employee where employeeId = empId);

set empSalary = (select employeeSalary from employee where employeeId = empId);

select empId as 'EmployeeId';

select empName as 'EmployeeName';

select empAddress as 'EmployeeAddress';

select empSalary as 'EmployeeSalary';

end $$

# Learning Objective:

# Learners are able to learn to create the Stored Procedures with input and output parameters.

# Question 3:

**Objective :**

To enable the Mfg Based business to automate the Data querying activity that is carried out at the Database level to simplify the phrasing of Join statements while fetching data from multiple tables.

- using Views

In order to reduce the complexity in phrasing the join statements, you are asked to create a View for extracting the data from 4 tables , which are

Customers, Products, Orders & Suppliers.

You are required to create a view which fetches the following data by joining the above 4 tables

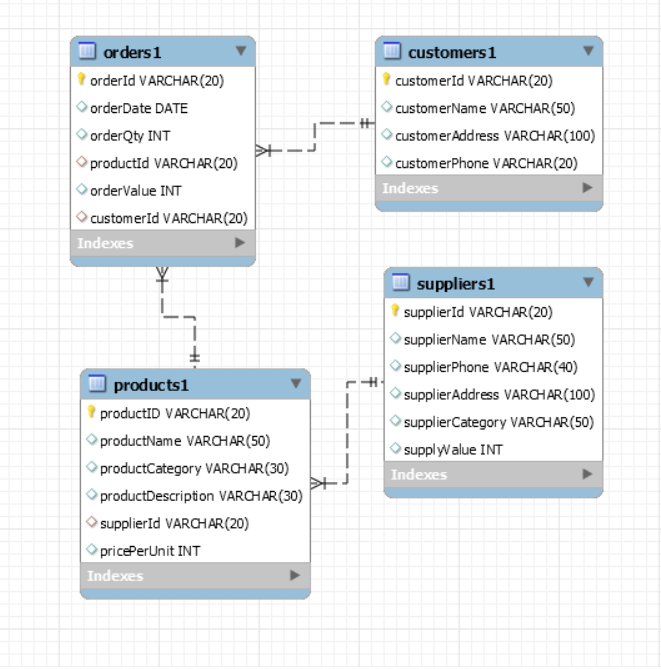
The Schema is as follows: You are expected to fetch the following column data from each table.

**Customers :** customerId, customerName & customerAddress

**Orders :** orderId, orderDate, orderQty & orderValue

**Products:** productName,productCategory,productDescription & pricePerUnit

**Suppliers:** supplierName, supplierPhone, supplierCategory & supplyValue



**Stub Code**

Create View ViewName

As

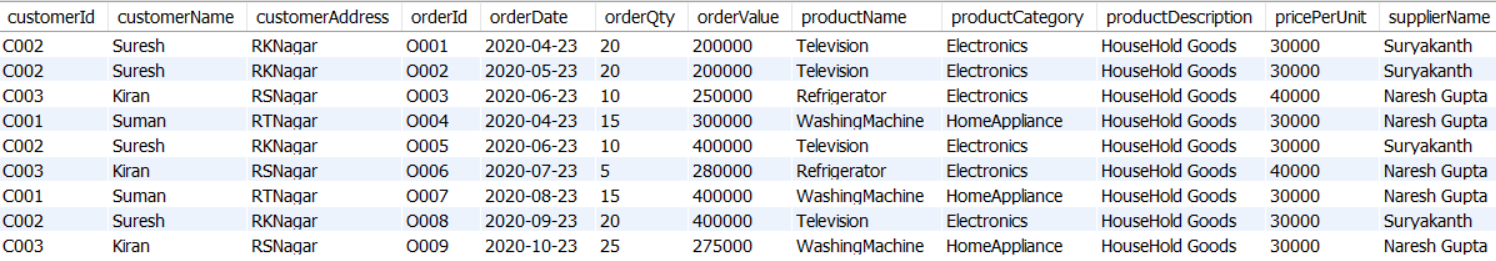
/\* Implement Code Here \*/

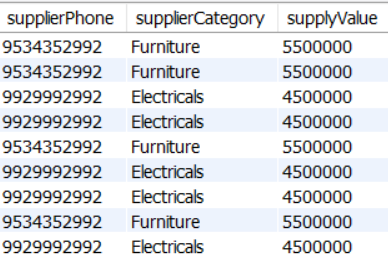
**Sample Output 1:**

When the query is given to execute the view as given below:

select \* from CustomerOrdersView;

The output is as follows:





**Solution:**

create view CustomerOrdersView

as

select c1.customerId,c1.customerName,c1.customerAddress,

o1.orderId,o1.orderDate,o1.orderQty,o1.orderValue,

p1.productName,p1.productCategory,p1.productDescription,p1.pricePerUnit,

s1.supplierName,s1.supplierPhone,s1.supplierCategory,s1.supplyValue

from Customers1 c1 join Orders1 o1

on c1.customerId = o1.customerId

join Products1 p1

on o1.productId = p1.productId

join Suppliers1 s1

on p1.supplierId = s1.supplierId;

# Learning Objective:

# Learners are able to learn to create theViews incorporating joins in it.

# 

# Question 4:

**Objective:**

To enable the Mfg Based business to automate the Data querying activity that is carried out at the Database level.

- using Stored Procedures

You are asked to create a stored procedure which fetches the Customer Details like Customer Name and the maximum OrderValue for the Customer, when CustomerId is passed as input parameter and the results should be displayed as

PremiumCustomer, NormalCustomer or GoldClassCustomer based on the OrderValues

If the OrderValue is between 100 and 1 Lakh(inclusive), the display should be NormalCustomer

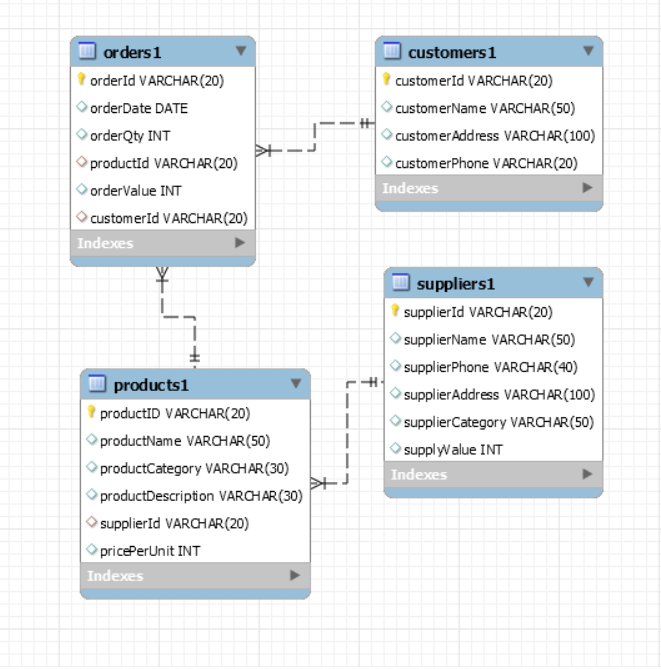
Or if the OrderValue is between 1 Lakh(greater than) and 2 Lakhs(inclusive the display should be PremiumCustomer

Else if the OrderValue is above 2 Lakhs the display should be GoldClassCustomer

**Hint:** Use Joins or Subqueries & Case Constructs in Procedures

**Stub Code:**

Schemas for Orders & Customers table are as follows:



**delimiter $$**

**/\* Implement Code Here \*/**

**end $$**

**Sample Output 1:**

When the Procedure is invoked with the following statement

call ProDec1('C003');

Following is the output:



**Sample Output 2:**

When the Procedure is invoked with the following statement

call ProDec1('C001');

Following is the output:



**Solution :**

**delimiter $$**

use `ctsdatamdu` $$

create procedure ProDec1(in custId varchar(20))

begin

declare custName varchar(50);

declare ordValue int;

declare customerType varchar(50);

set custName = (select customerName from customers1 where customerId = custId);

set ordValue = (select max(OrderValue) from Orders1 group by customerId having customerId = custId);

if ordValue >= 100 and ordValue <= 100000 then set customerType = 'Normal Customer';

elseif ordValue > 100000 and ordValue < 200000 then set customerType ='Premium Customer';

elseif ordValue > 200000 then set customerType='Gold Class Customer';

else

set customerType = 'Not Categorized';

end if;

select custName;

select ordValue;

select customerType;

**end $$**

# 