**Niket Sahai**

**Emp Code: 3185535**

**DotNet Freshers’ Training**

**Assignment-3**

**Exercise 1**

1. Display the number of records in the [SalesPerson] table. (Schema(s) involved: Sales)

Select count(BusinessEntityID) AS 'TOTAL NO OF RECORDS' from Sales.SalesPerson;

Graphical user interface, text, application

Description automatically generated

2. Select both the FirstName and LastName of records from the Person table where the FirstName begins with the letter ‘B’.

(Schema(s) involved: Person)

select FirstName,LastName from Person.Person where FirstName LIKE 'B%';

Graphical user interface, application, Word

Description automatically generated

3. Select a list of FirstName and LastName for employees where Title is one of Design Engineer, Tool Designer or Marketing

Assistant. (Schema(s) involved: HumanResources, Person)

SELECT P.FirstName,P.LastName

FROM Person.Person AS P INNER JOIN HumanResources.Employee H ON

P.BusinessEntityID = H.BusinessEntityID

WHERE H.JobTitle = 'Design Engineer' OR

H.JobTitle = 'Tool Designer' OR

H.JobTitle = 'Marketing Assistant';

Graphical user interface, text, application

Description automatically generated

4. Display the Name and Color of the Product with the maximum weight. (Schema(s) involved: Production)

select Name,Color from Production.Product where Weight =(select max(Weight) from Production.Product);

Graphical user interface, text, application, Word

Description automatically generated

5. Display Description and MaxQty fields from the SpecialOffer table. Some of the MaxQty values are NULL, in this case display

the value 0.00 instead. (Schema(s) involved: Sales)

Select Description,COALESCE(MaxQty,0.00) as 'MaxQty'from Sales.SpecialOffer;

Graphical user interface, text, application

Description automatically generated

6. Display the overall Average of the [CurrencyRate].[AverageRate] values for the exchange rate ‘USD’ to ‘GBP’ for the year 2005 i.e. FromCurrencyCode = ‘USD’ and ToCurrencyCode = ‘GBP’. Note: The field CurrencyRate].[AverageRate] is defined as 'Average exchange rate for the day.' (Schema(s) involved: Sales)

select AVG(AverageRate) as 'Average exchange rate' from Sales.CurrencyRate

where FromCurrencyCode = 'USD' and ToCurrencyCode = 'GBP';

Graphical user interface, text, application, email

Description automatically generated

7. Display the FirstName and LastName of records from the Person table where FirstName contains the letters ‘ss’. Display an additional column with sequential numbers for each row returned beginning at integer 1. (Schema(s) involved: Person)

select ROW\_NUMBER() OVER(ORDER BY FirstName)

as 'Sequence',FirstName,LastName

from Person.Person where FirstName LIKE '%ss%';

Graphical user interface, application, Word

Description automatically generated

8. Sales people receive various commission rates that belong to 1 of 4 bands.

(Schema(s) involved: Sales)

CommissionPct Commission Band

0.00 Band 0

Up To 1% Band 1

Up To 1.5% Band 2

Greater 1.5% Band 3

Display the [SalesPersonID] with an additional column entitled ‘Commission Band’ indicating the appropriate band as above.

select BusinessEntityID as 'SalesPersonID',

case

when CommissionPct = 0.00 then 'BAND 0'

when CommissionPct > 0.00 and CommissionPct <= 0.01 then 'BAND 1'

when CommissionPct > 0.01 and CommissionPct <= 0.015 then 'BAND 2'

when CommissionPct > 0.015 then 'BAND 3'

end as 'Commission Band'

from Sales.SalesPerson

order by [Commission Band];

Graphical user interface, text, application

Description automatically generated

9. Display the managerial hierarchy from Ruth Ellerbrock (person type – EM) up to CEO Ken Sanchez. Hint: use [uspGetEmployeeManagers] (Schema(s) involved: [Person], [HumanResources])

select Person.Person.FirstName, Person.Person.LastName, HumanResources.Employee.OrganizationLevel

from HumanResources.Employee join Person.Person

on (HumanResources.Employee.BusinessEntityID=Person.Person.BusinessEntityID);

Graphical user interface, application, Word

Description automatically generated

10. Display the ProductId of the product with the largest stock level.

Hint: Use the Scalar-valued function [dbo]. [UfnGetStock]. (Schema(s) involved: Production)

select ProductID from Production.Product

where SafetyStockLevel = (select max(SafetyStockLevel) from Production.Product);

Graphical user interface, text, application

Description automatically generated

**Exercise 2**

Write separate queries to list all AdventureWorks customers who have not placed an order.

By Using Join Query

select PP.FirstName + PP.LastName AS 'Customer Name'

from Person.Person PP INNER JOIN

Sales.Customer SC on

PP.BusinessEntityID = SC.CustomerID LEFT JOIN

Sales.SalesOrderHeader SS on

SC.CustomerID = SS.CustomerID

where SS.SalesOrderID is NULL;

Graphical user interface, text, application

Description automatically generated

By Using Subquery

select FirstName + LastName as 'Customer Name'

from Person.Person

Where BusinessEntityID IN

(select CustomerID from Sales.Customer where CustomerID NOT IN

(select CustomerID from Sales.SalesOrderHeader));

Graphical user interface, text, application

Description automatically generated

By Using CTEs

;WITH UnorderProductCustomers (CustomerName)

AS (SELECT PP.FirstName + PP.LastName AS 'CustomerName'

FROM Person.Person PP INNER JOIN

Sales.Customer SC ON

PP.BusinessEntityID = SC.CustomerID LEFT JOIN

Sales.SalesOrderHeader SS ON

SC.CustomerID = SS.CustomerID

WHERE SS.SalesOrderID IS NULL)

SELECT CustomerName

FROM UnorderProductCustomers;

Graphical user interface, text, application

Description automatically generated

By Using Exists

SELECT PP.FirstName + PP.LastName AS 'Customer Name'

FROM Person.Person PP

WHERE EXISTS (SELECT SC.CustomerID

FROM Sales.Customer SC

WHERE PP.BusinessEntityID = SC.CustomerID AND

NOT EXISTS(SELECT SS.CustomerID

FROM Sales.SalesOrderHeader SS

WHERE SC.CustomerID = SS.CustomerID));

Graphical user interface, text, application, Word

Description automatically generated

**Exercise 3**

Show the most recent five orders that were purchased from account numbers that have spent more than $70,000 with AdventureWorks.

select top 5 SalesOrderID as 'Order ID',

OrderDate as 'Date Of Order',

AccountNumber as 'Account Number',

SUM(TotalDue) as 'Amount Spent'

from Sales.SalesOrderHeader

group by AccountNumber,

OrderDate,

SalesOrderID

having sum(TotalDue) > 70000

order by OrderDate desc;

Graphical user interface

Description automatically generated with medium confidence

**Exercise 4**

Create a function that takes as inputs a SalesOrderID, a Currency Code, and a date, and returns a table of all the SalesOrderDetail rows for that Sales Order including Quantity, ProductID, UnitPrice, and the unit price converted to the target currency based on the end of day rate for the date provided. Exchange rates can be found in the Sales.CurrencyRate table. ( Use AdventureWorks)

GO

CREATE FUNCTION Sales.uf\_NewFunction(@SalesOrderId int,@CurrencyCode nchar(3),@Date datetime)

RETURNS TABLE

AS

RETURN

SELECT sod.ProductID AS 'Product ID',

sod.OrderQty AS ' Order Quantity',

sod.UnitPrice As 'Unit Price',

sod.UnitPrice\*scr.EndOfDayRate AS 'Target Price'

FROM Sales.SalesOrderDetail AS sod,

Sales.CurrencyRate AS scr

WHERE scr.ToCurrencyCode = @CurrencyCode AND

scr.ModifiedDate = @Date AND

sod.SalesOrderID = @SalesOrderID

GO

Select \* from Sales.uf\_NewFunction(43659,'MXN','2005-09-05');

Graphical user interface, text, application

Description automatically generated

**Exercise 5**

Write a Procedure supplying name information from the Person.Person table and accepting a filter for the first name. Alter the above Store Procedure to supply Default Values if user does not enter any value.( Use AdventureWorks)

GO

CREATE PROCEDURE Person.up\_DisplayPersonInfo

@FirstName nvarchar(20) = 'Niket'

AS

BEGIN

SELECT BusinessEntityID AS 'ID',

FirstName + LastName AS 'NAME',

PersonType

FROM Person.Person

WHERE FirstName = @FirstName

END

EXECUTE Person.up\_DisplayPersonInfo

EXECUTE Person.up\_DisplayPersonInfo @FirstName = 'Niket'

GO

Graphical user interface, text, application

Description automatically generated

**Exercise 6**

Write a trigger for the Product table to ensure the list price can never be raised more than 15 Percent in a single change. Modify the above trigger to execute its check code only if the ListPrice column is updated (Use AdventureWorks Database).

GO

CREATE OR ALTER TRIGGER [Production].UpdateTrigger

ON Production.Product

INSTEAD OF UPDATE

AS

SET NOCOUNT ON

BEGIN

IF UPDATE(ListPrice) -- Modification A.T.Q second requirement

DECLARE @OldListPrice money

DECLARE @InsertedListPrice money

DECLARE @ID int

SELECT @OldListPrice = p.ListPrice,

@InsertedListPrice=inserted.ListPrice,

@ID = inserted.ProductID

FROM Production.Product p, inserted

WHERE p.ProductID = inserted.ProductID;

IF( @InsertedListPrice > ( @OldListPrice + (0.15\*@OldListPrice) ) )

BEGIN

RAISERROR('LIST PRICE MORE THAN 15 PERCENT, TRANSACTION FAILED',16,0)

ROLLBACK TRANSACTION

END

ELSE

BEGIN

Update Production.Product SET ListPrice=@InsertedListPrice

WHERE Production.Product.ProductID = @ID;

END

END;

SELECT Production.Product.ProductID,

Production.Product.ListPrice

FROM PRODUCTION.Product;

UPDATE PRODUCTION.Product

SET ListPrice=2

WHERE Product.ProductID=4;

Graphical user interface, text, application

Description automatically generated