**SQL ASSIGNMENT**

**Exercise 1**

**The exercise requires SQL Server AdventureWorks OLTP database which can be found at Codeplex. Download and attach a copy of the database to your server instance. Take some time to appreciate the entire schema of the database, and functions and stored procedures (refer AdventureWorks 2008 OLTP Schema.pdf). Using the AdventureWorks database, perform the following queries.**

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

SELECT COUNT(\*) AS NumberOfRecords

FROM Sales.SalesPerson ;

Graphical user interface, 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, text, application

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 FirstName,LastName,JobTitle

FROM HumanResources.Employee

INNER JOIN Person.Person

ON HumanResources.Employee.BusinessEntityID=Person.Person.BusinessEntityID

WHERE JobTitle IN ('Design Engineer', 'Tool Designer','Marketing Assistant');

A computer screen capture

Description automatically generated with medium confidence

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

SELECT Name,Color,Weight

FROM Production.Product

WHERE Weight=(SELECT MAX(Weight) FROM Production.Product );

Graphical user interface, text, application

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,CASE WHEN MaxQty IS NULL THEN 0.00 ELSE MaxQty END AS MaxQty

FROM Sales.SpecialOffer

**OR**

SELECT Description,CAST(ISNULL(MaxQty,0.00) AS numeric(4,2)) 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) OverAverage

FROM Sales.CurrencyRate

WHERE FromCurrencyCode = 'USD' AND ToCurrencyCode = 'GBP' AND YEAR(CurrencyRateDate)=2005

Graphical user interface, application

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 FirstName,LastName,

ROW\_NUMBER() OVER(ORDER BY FirstName DESC) AS Sequence

FROM Person.Person

WHERE FirstName LIKE '%ss%';

Graphical user interface, text, application

Description automatically generated

**8. Sales people receive various commission rates that belong to 1 of 4 bands. (Schema(s) involved: Sales)**

**Commission Pct Commission Band**

1. **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,

CASE

WHEN CommissionPct ='0.0' THEN 'CommissionBand Band 0'

WHEN CommissionPct <='0.01' THEN 'CommissionBand Band 1'

WHEN CommissionPct <='0.015' THEN 'CommissionBand Band 2'

ELSE 'CommissionBand Band 3'

END AS CommissionBand

FROM Sales.SalesPerson

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])**

DECLARE @BusinessEntityID int

SELECT @BusinessEntityID=BusinessEntityID

FROM Person.Person

WHERE FirstName='Ruth'AND Lastname='Ellerbrock' AND PersonType='EM'

EXEC uspGetEmployeeManagers @BusinessEntityID;

Graphical user interface, text, application

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)**

DECLARE @ProductID int

SELECT @ProductID=ProductID

FROM Production.Product

WHERE SafetyStockLevel=(SELECT MAX(SafetyStockLevel) FROM Production.Product)

SELECT DBO.UfnGetStock(@ProductID) AS LargestStockLevel, @ProductID AS ProductID

Graphical user interface, text, application

Description automatically generated

**Exercise 2.Write separate queries using a join, a subquery, a CTE, and then an EXISTS to list all AdventureWorks customers who have not placed an order.**

**Using a join**

SELECT c.CustomerID CustomerId\_DidNotPurchaseAnything

FROM Sales.Customer c

LEFT OUTER JOIN Sales.SalesOrderHeader s

ON c.CustomerID =s.CustomerID

WHERE s.SalesOrderID IS NULL

Graphical user interface, text, application

Description automatically generated

**USING CTE**

WITH s AS

(

SELECT SalesOrderID,CustomerID

FROM Sales.SalesOrderHeader AS h

)

SELECT c.CustomerID

FROM Sales.Customer c

LEFT OUTER JOIN s ON c.CustomerID = s.CustomerID

WHERE s.SalesOrderID IS NULL

A computer screen capture

Description automatically generated with medium confidence

**USING SUBQUERY**

SELECT c.CustomerID CustomerId\_DidNotPurchaseAnything

FROM Sales.Customer c

WHERE c.CustomerID NOT IN

(SELECT s.CustomerID FROM Sales.SalesOrderHeader s)

Graphical user interface, text, application

Description automatically generated

**USING EXISTS**

SELECT C.CustomerID

FROM Sales.Customer c

WHERE NOT EXISTS(

SELECT c.CustomerID

FROM Sales.SalesOrderHeader s

WHERE c.customerID = s.customerID)

Graphical user interface, text, application

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 s.AccountNumber, s.OrderDate,(

SELECT Name

FROM Production.Product

WHERE ProductID=s.ProductID) AS ProductName

FROM (

SELECT soh.\*,sod.ProductID,

TotalAmount = SUM(sod.LineTotal) OVER (PARTITION BY soh.AccountNumber),

RN = ROW\_NUMBER() OVER (PARTITION BY soh.AccountNumber ORDER BY soh.OrderDate DESC)

FROM Sales.SalesOrderHeader soh

JOIN Sales.SalesOrderDetail sod

ON soh.SalesOrderID = sod.SalesOrderID) AS s

WHERE s.TotalAmount > 70000

AND s.RN <= 5;

Graphical user interface, text, application

Description automatically generated

**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)**

CREATE FUNCTION fn\_salesorderdetail(@SalesOrderID int, @ToCurrencyCode nvarchar(5), @ModifiedDate date)

RETURNS TABLE

AS

RETURN (SELECT OrderQty, ProductID, UnitPrice,ConvertedPrice=UnitPrice\*(SELECT EndOfDayRate

FROM Sales.CurrencyRate

WHERE ToCurrencyCode=@ToCurrencyCode AND ModifiedDate=@ModifiedDate)

FROM Sales.SalesOrderDetail

WHERE SalesOrderID=@SalesOrderID AND ModifiedDate=@ModifiedDate);

SELECT\*

FROM fn\_salesorderdetail(43659, 'AUD', '20050701')

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)**

CREATE PROCEDURE nameinformation @FirstName nvarchar(30)

AS

SELECT FirstName,MiddleName, LastName

FROM Person.Person

WHERE FirstName=@FirstName

EXEC nameinformation Ken

Graphical user interface, text, application

Description automatically generated

**AFTER ALTER**

ALTER PROCEDURE nameinformation @FirstName nvarchar(30)='Ben'

AS

SELECT FirstName,MiddleName, LastName

FROM Person.Person

WHERE FirstName=@FirstName

EXEC nameinformation

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)**

CREATE TRIGGER Production.tr\_tblProduct\_ForCantRaiseListPrice

ON Production.Product

FOR UPDATE

AS

IF EXISTS (

SELECT \* FROM inserted i

JOIN deleted d

ON i.ProductID = d.ProductID

WHERE i.ListPrice > (d.ListPrice +d.ListPrice\*0.15)

)

BEGIN

PRINT ('Price increase cannot be greater than 15%')

ROLLBACK TRAN

END

UPDATE Production.Product

SET ListPrice=10

WHERE ProductID=710

Graphical user interface, text, application

Description automatically generated

**AFTER ALTER**

ALTER TRIGGER Production.tr\_tblProduct\_ForCantRaiseListPrice

ON Production.Product

FOR UPDATE

AS

IF UPDATE(ListPrice)

BEGIN

IF EXISTS (

SELECT \* FROM inserted i

JOIN deleted d

ON i.ProductID = d.ProductID

WHERE i.ListPrice > (d.ListPrice +d.ListPrice\*0.15)

)

BEGIN

PRINT ('Price increase cannot be greater than 15%')

ROLLBACK TRAN

END

END

Graphical user interface, text, application

Description automatically generated