MS – 203: SQL

Introduction	**This course aims to quickly ramp the trainees in T-SQL. In this course you will learn what are the objects that make up a SQL Server database and which tools are provided by SQL Server. Also, you will also learn** how to retrieve, insert, update and delete data in Microsoft SQL Server 2012. The course will finish off by exploring programming in SQL in form of stored procedures and user defined functions.	
Target Audience	**Trainees/Beginners**	
Duration	**Full**	**Partial**
	40 Hours	**20 Hours**
Pre-requisite	**No pre-requisite.**	

Course Material	**Trainees might find few overlapping areas between the reading material and online video tutorials, but it is by design and in principle they complete each other. It is expected that trainees will make the best use of the entire study material provided.**
	• **Books**
	  ○ **#1 Beginning Microsoft SQL Server 2012 Programming**
	• **Videos**
	  ○ **#2** https://www.youtube.com/playlist?list=PLIoX3-mcY80ipygQHDYN_w_F1lAl2m2JS
	  ○ **#3** https://www.youtube.com/playlist?list=PL08903FB7ACA1C2FB
	  ○ #4 https://www.youtube.com/playlist?list=PLU9JMEzjCv14f3cWDhubPaddxRvx1reKR	

Pathway		

** Mandatory

Topic	Description	Hours	Book	Media
Introduction to SQL	• RDBMS Basics • SQL Server Data Types • SQL Server Tools • Select,Update,Delete,Insert	4	**#1 Chapter 1 - 3**	#2 Introduction to SQL Server 2012 #2 Introduction to Relational Databases #2 Introduction to SELECT Statements #2 **Filtering Results with WHERE Statements** **#2 Modifying Data**
Retrieving, sorting and filtering data**	• Joins • **Creating Tables** • **Altering Tables** • **Keys and Constraints** • **Subquery** • **Correlated Subquery** • **Exists Operator** • **CTE'S** • **Cast and Convert**	4	**#1 Chapter 4 – 7**	#2 **Utilizing Joins** **#2 Executing Subqueries and UNION Statements** **#2 Aggregating Data** **#2 Advanced Data Aggregations** **#2 Built-in Functions** **#2 Advance Data Modification** **#3 Data Validation with Constraints** **#3 Creating and Altering Tables**

Normalization, Indexes and Views	• **Normalization** • **Denormalization** • Creating and Altering Indexes • Maintaining your Index's • **Editing Views** • **Dropping Views**	4	**#1 Chapter 8-10**	#2 Query Optimization #3 **Creating and Modifying Views**
Writing Scripts and Batches, User Defined Functions, Store Procedures**	• Writing Scripts and Batches • Dynamic SQL • Understanding Stored Procedures • **Understanding User-Defined functions**	4	**#1 Chapter 11 - 13**	#3 **Creating and Modifying User Defined Functions** **#3 Creating and Modifying Procedures** **#3 Implementing Error Handling**
Triggers and transactions **	• Understanding Transaction Processing • Understanding Locks and Concurrency • Dealing with Deadlocks • **Triggers** • **Dropping Triggers** • **Debugging Triggers**	4	**#1 Chapter 14 & 15**	**#3 Creating and Modifying Triggers** **#3 Managing Transactions**

Assignment

Exercise 1

(3 Hours)

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

9. **Display the managerial hierarchy from Ruth Ellerbrock (person type – EM) up to CEO Ken Sanchez. Hint: use [uspGetEmployeeManagers] (*Schema(s) involved: [Person], [HumanResources]*)**
10. **Display the ProductId of the product with the largest stock level. Hint: Use the Scalar-valued function [dbo]. [UfnGetStock]. (*Schema(s) involved: Production*)**

Exercise 2 (3 Hours)

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.

Exercise 3 (2 Hours)

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

Exercise 4 (2 Hours)

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)**

Exercise 5 (2 Hours)

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)**

Exercise 6 (2 Hours)

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**).