# Content:

[Exercise 1: 2](#_Toc376814580)

[Excercise 2: User Defined Function (UDF) 4](#_Toc376814581)

[Excercise 3: Stored Procedure 5](#_Toc376814582)

[Excersice 4: Triggers 5](#_Toc376814583)

# Requires:

For the following exercises:

* Print out respectively the screenshots to show the query results.
* SQL scripts for the exercises.
* Pack screenshots and SQL scripts into the zip file named **Assignment4\_AccountName.zip**(for instance: Assignment4\_NamNT.zip)

# 

# Exercise 1:

You work as a Database Developer for ABC Inc. The company uses RDBMS for project management. The fragments of database schema are given in figure below. As shown, the Project database contains four tables: **Projects**, **Employee**, **Project\_Modules** and **Work\_Done**.



* Each project is assigned to an employee called Project Manager and there can be only one Project Manager for a project.
* For better planning and management, each project is divided into smaller modules and responsibility of each module is assigned to an Employee.
* Reporting is done on a daily basis and employees report the work done on their modules as well as other employees’ modules (if any).
* You remove the project information after three months of project completion.

Beside the fact that a Project Manager is an ABC Inc employee, we have some other assumptions on the table fields as below:

* Projects.ProjectCompletedOne: the date project was completed.
* Project\_Modules.ProjectModulesDate: the due date for the owner employee to complete the module
* ProjectModulesCompletedOn: the actual date that the owner employee has completed the module.
* Work\_Done.WorkDoneDate: the date when the relevant employee has completed the Work, it would be NULL if the Work is still in progress
* Employee.SupervisorID: the ID of mentioned employee’s supervisor, who is also an ABC Inc employee.

**Questions**

For the following request, print out respectively the screenshots to show test data (the table data that you create to test each query), the query results, and pack them into the file Assignment4\_AccountName.zip along with your answers, then handle to the evaluator via email ([XYZ@fsoft.com.vn](mailto:XYZ@fsoft.com.vn))

1. Write :
2. Create the tables (with the most appropriate field/column constraints & types) and add at least 3 records into each created table.
3. Write a stored procedure (without parameter) to remove the project information after three months of project completion, since the current date. Print out the number of records which are removed from each related table during the removals.
4. Write a stored procedure (with parameter) to print out the modules that a specific employee has been working on.
5. Write a user function (with parameter) that return the Works information that a specific employee has been involving though those were not assigned to him/her.
6. Write the trigger(s) to prevent the case that the end user to input invalid Project Modules information (Project\_Modules.ProjectModulesDate < Projects.ProjectStartDate, Project\_Modules.ProjectModulesCompletedOn > Projects.ProjectCompletedOn)

# Excercise 2: User Defined Function (UDF)

In the AdventureWorks2008 database

1. Writes an UDF returns ProductID, Name, ProductNumber, Color and ListPrice for products that have 'CA' in the ProductNumber and Color is 'black'

2. Writes an inline table UDF that accept an integer parameter named BusinessEntityID and returns the associated address of a business entity.

# Excercise 3: Stored Procedure

In the AdventureWorks2008 database, writes a stored proc meet below requirements:

- Accept 2 parameters: @ModifiedDate (datetime) and @UpperFlag (bit)

- Returns FirstName from Person.Person table if ModifedDate is greater than @ModifiedDate, FirstName will be uppercase if @UpperFlag is true.

# Excersice 4: Triggers

This exercise perforns in AdventureWorks2008 database.

1. In the AdventureWorks2008 database, create an audit table of **Production.ProductInventory** table name **Production.ProductInventoryAudit** as below:

**CREATE TABLE Production.ProductInventoryAudit**

**(**

**ProductID INT NOT NULL**

**, LocationID SMALLINT NOT NULL**

**, Shelf NVARCHAR(10) NOT NULL**

**, Bin TINYINT NOT NULL**

**, Quantity SMALLINT NOT NULL**

**, rowguid UNIQUEIDENTIFIER NOT NULL**

**, ModifiedDate DATETIME NOT NULL**

**, InsertOrDelete CHAR(1) NOT NULL**

**)**

Writes a trigger to ensure that each time user insert or delete records from **Production.ProductInventory** then these records will be captured in **Production.ProductInventoryAudit** table. In insert case, **InsertOrDelete** column in **Production.ProductInventoryAudit is "I",** in delete case, this column must be **"D"**