Part-A

1. **Create a trigger that fires on INSERT, UPDATE and DELETE operation on the Person table to display a message "Record is Affected."**

```
CREATE TRIGGER tr_Person_Record_Affected
on Person
AFTER Insert, Update, Delete
AS
BEGIN
        PRINT 'Record is Affected.'
END
```

2. **Create a trigger that fires on INSERT, UPDATE and DELETE operation on the Person table. For that, log all operations performed on the person table into PersonLog.**
   a. **Insert**

```
CREATE TRIGGER tr_Person_after_Insert
ON Person
AFTER INSERT
AS
BEGIN
    DECLARE @PersonID AS INT
    DECLARE @PersonName AS VARCHAR(50)
    SELECT @PersonID=PersonID from inserted
    SELECT @PersonName = PersonName from inserted

    INSERT INTO PersonLog
    VALUES(@PersonID, @PersonName, 'INSERT', GETDATE())
END
```

   b. **Update**

```
CREATE TRIGGER tr_Person_after_Update
ON Person
AFTER UPDATE
AS
BEGIN
    DECLARE @PersonID AS INT
    DECLARE @PersonName AS VARCHAR(50)
    SELECT @PersonID=PersonID from inserted
    SELECT @PersonName = PersonName from inserted

    INSERT INTO PersonLog
    VALUES(@PersonID, @PersonName, 'UPDATE', GETDATE())
END
```

   c. **Delete**

```
CREATE TRIGGER tr_Person_after_Delete
ON Person
AFTER DELETE
```

```
AS
BEGIN
    DECLARE @PersonID AS INT
    DECLARE @PersonName AS VARCHAR(50)
    SELECT @PersonID=PersonID from deleted
    SELECT @PersonName = PersonName from deleted

    INSERT INTO PersonLog
    VALUES(@PersonID,@PersonName,'DELETE',GETDATE())
END
```

3. **Create an INSTEAD OF trigger that fires on INSERT, UPDATE and DELETE operation on the Person table. For that, log all operations performed on the person table into PersonLog.**
   a. **Insert**

```
CREATE TRIGGER tr_Person_InsteadOf_Insert
ON Person
INSTEAD OF INSERT
AS
BEGIN
    DECLARE @PersonID int
    DECLARE @PersonName VARCHAR(50)
    SELECT @PersonID = PersonID from inserted
    SELECT @PersonName = PersonName from inserted

    INSERT INTO PersonLog
    VALUES(@PersonID, @PersonName, 'INSERT', GETDATE())
END
```

   b. **Update**

```
CREATE TRIGGER tr_Person_InsteadOf_Update
ON Person
INSTEAD OF UPDATE
AS
BEGIN
    DECLARE @PersonID int
    DECLARE @PersonName VARCHAR(50)
    SELECT @PersonID = PersonID from inserted
    SELECT @PersonName = PersonName from inserted

    INSERT INTO PersonLog
    VALUES(@PersonID, @PersonName, 'UPDATE', GETDATE())
END
```

   c. **Delete**

```
CREATE TRIGGER tr_Person_InsteadOf_Delete
ON Person
INSTEAD OF DELETE
```

```
AS
BEGIN
    DECLARE @PersonID int
    DECLARE @PersonName VARCHAR(50)
    SELECT @PersonID = PersonID from DELETED
    SELECT @PersonName = PersonName from DELETED

    INSERT INTO PersonLog
    VALUES(@PersonID, @PersonName, 'DELETE', GETDATE())
END
```

4. **Create a trigger that fires on INSERT operation on the Person table to convert person name into uppercase whenever the record is inserted.**

```
CREATE TRIGGER tr_Person_NameUpper_Inset
ON Person
AFTER INSERT
AS
BEGIN
        DECLARE @Uname VARCHAR(50)
        DECLARE @PersonID int
        select @Uname=PersonName from inserted
        select @PersonID=PersonID from inserted

        UPDATE Person
        SET PersonName=Upper(@Uname)
        WHERE PersonID=@PersonID
END
```

Part-B

5. **Create a trigger that fires on INSERT operation on person table, which calculates the age and update that age in Person table.**

```
CREATE TRIGGER tr_Person_Update_Age
ON Person
FOR INSERT
AS
BEGIN
        DECLARE @PersonID int;
        DECLARE @Age int;
        DECLARE @BIRTHDATE DATETIME;
        SELECT @PersonID = PersonID from inserted
        SELECT @BIRTHDATE = Birthdate from inserted
        SET @Age=DATEDIFF(YEAR,@BIRTHDATE,GETDATE())

        UPDATE PERSON
        SET Age=@Age
        WHERE PersonID=@PersonID
```

```
END
```

Part-C

6. **Create DELETE trigger on PersonLog table, when we delete any record of PersonLog table it prints 'Record deleted successfully from PersonLog'.**

```
CREATE TRIGGER tr_Log_Delete
on PersonLog
FOR DELETE
AS
BEGIN
        PRINT 'Record deleted successfully from PersonLog'
END
```