--1. Create University (UniversityId, UniversityName and Address) and Student (studId, UniversityId, studName, gpa) tables.

--2. Establish primary foreign key relationship between these two tables.

--3. Add some rows to both tables.

--4. Write the joint statement of these two tables and show the query execution plan of these two tables.

--5. Write the SQL statement that aggregates (groups by) the gpa data based on UniversityName and displays those departments that shows the

--departments in which the average gpa is more than 3.

--6. Write an SQL command that will retrieve the names of students whose gpa is higher than the average gpa.

-- Change the SQL command so that it outputs the names of the students along with the names of the universities where those students’ study.

--7. Create the procedures addstudent and adduniversity, which should add new rows to the universities table, and the students table based on the passed parameters.

--8. Write a function that returns the average gpa of the students in the students’ table. Give an example of a function call.

--9. Create a separate table UniAverageGpa which will have the following Columns UniversityId, University and averageGpa.

-- Write a trigger for the students table that will change the averageGpa data when a new record is added or changed in the students table.

1,2,3>

CREATE TABLE University (

UniversityId INT PRIMARY KEY,

UniversityName VARCHAR(100),

Address VARCHAR(100)

);

CREATE TABLE Student (

StudId INT PRIMARY KEY,

UniversityId INT,

StudName VARCHAR(100),

GPA FLOAT,

FOREIGN KEY (UniversityId) REFERENCES University(UniversityId)

);

INSERT INTO University (UniversityId, UniversityName, Address) VALUES

(1, 'University of Example', '123 Example Street'),

(2, 'Another University', '456 University Avenue')

INSERT INTO Student (StudId, UniversityId, StudName, GPA) VALUES

(1, 1, 'John Doe', 3.8),

(2, 1, 'Jane Smith', 3.5),

(3, 2, 'Michael Johnson', 3.9),

(4, 1, 'Jane Dakota', 3.5),

(5, 2, 'James Bond', 5),

(6, 2, 'John Wick', 5),

(7, 1, 'Cane', 4.7);

4>

SELECT University.UniversityId, StudName, UniversityName, GPA FROM University

INNER JOIN Student ON University.UniversityId = Student.UniversityId;

5>

SELECT UniversityName, GPA FROM University

GROUP BY UniversityName, GPA

HAVING AVG(GPA) > 3;

6.1>

SELECT StudName, GPA FROM Student

WHERE GPA > (SELECT AVG(GPA) FROM Student);

6.2>

SELECT UniversityName, StudName, GPA FROM University

INNER JOIN Student ON University.UniversityId = Student.UniversityId

WHERE GPA > (SELECT AVG(GPA) FROM Student)

7>

CREATE OR ALTER PROCEDURE addstudent (@StudId INT,@UniversityId INT, @StudName VARCHAR, @GPA FLOAT)

AS

BEGIN

INSERT INTO Student values(@StudId,@UniversityId,@StudName,@GPA);

END;

CREATE OR ALTER PROCEDURE adduniversity (@UniversityId INT, @UniversityName VARCHAR, @Address VARCHAR)

AS

BEGIN

INSERT INTO University values(@UniversityId,@UniversityName,@Address);

END;

8>

CREATE OR ALTER FUNCTION avg\_gpa()

RETURNS FLOAT

AS

BEGIN

DECLARE @avg FLOAT

SELECT @avg = AVG(GPA) FROM Student;

RETURN @avg

END;

9>

CREATE TABLE UniAverageGpa (

UniversityId INT,

University VARCHAR(100),

averageGpa FLOAT

);

CREATE OR ALTER TRIGGER avgGPA

ON Student

AFTER INSERT

AS

BEGIN

DECLARE @id INT, @avg\_gpa FLOAT, @name VARCHAR(255);

SELECT @id = UniversityId FROM inserted;

SELECT @avg\_gpa = AVG(GPA), @name = UniversityName FROM Student, University

WHERE University.UniversityId = @id

GROUP BY UniversityName;

IF EXISTS (SELECT 1 FROM UniAverageGpa WHERE UniversityId = @id)

BEGIN

UPDATE UniAverageGpa

SET averageGpa = @avg\_gpa

WHERE UniversityId = @id;

END

ELSE

INSERT INTO UniAverageGpa values(@id,@name, @avg\_gpa);

END;