**Gateway Technical College**

DATABASES

152-080

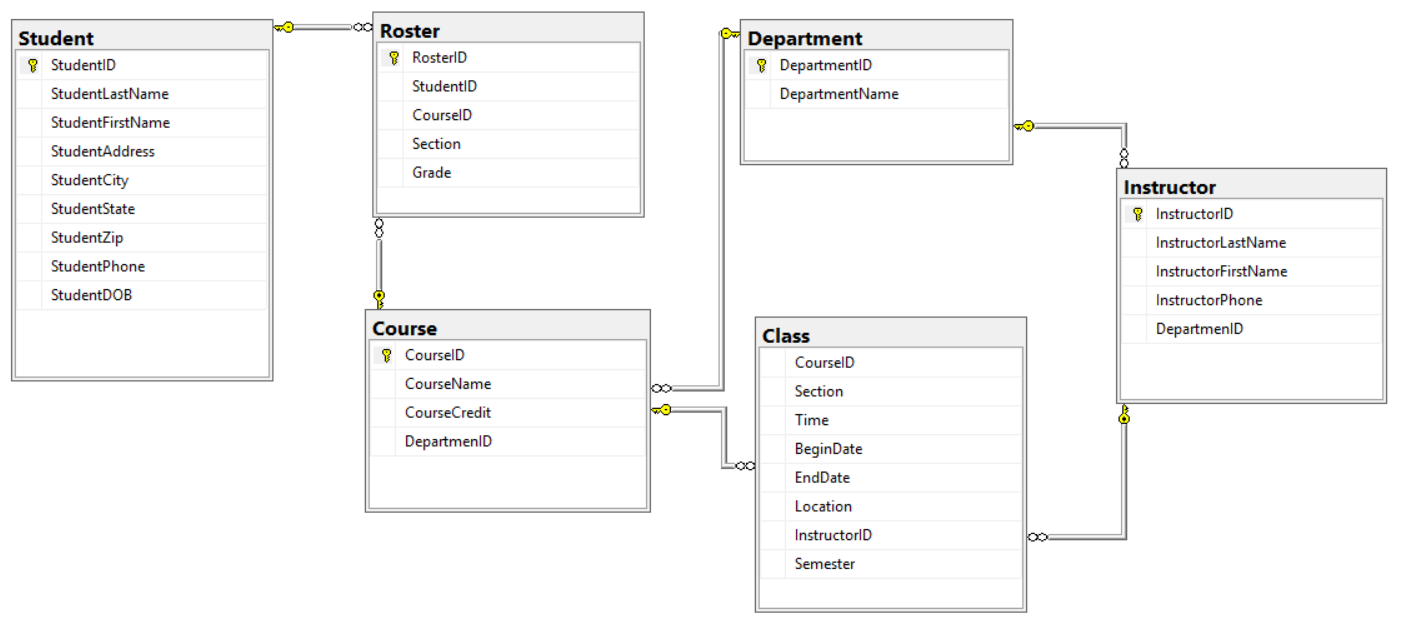
Unit 10 Assignment

# Introduction

In this lab you will CREATE, UPDATE, and ALTER views and understand how they can help reduce database complexity for end users and prevent sensitive columns from exposed to end users. You’ll also learn to create indexes on the views to speed query performance.

First, run the *Education.sql* script to create the database **Education** and all the tables and data to complete this assignment**.**

The ERD below shows all the tables and their relationships in the **Education** database.



MAKE SURE document your work and your commands work before you past them into the document.

Once completed, attach this completed word document to this assignment for grading.  Each question will be worth 5 points.

Use the **Discussion Forum** if you have any questions regarding the how to approach this assignment. You can also email your instructor directly for assistance if you have any questions.

Save your submission as ***lastnameFirstname\_assign9.docx*** and submit it in the unit *Apply* section of the course.

# Instructions

You are to complete the following actions. In order to do this assignment, you will need to run statements against the Database you created named **Education**. For each question below – paste in print screens of your progress in each step.

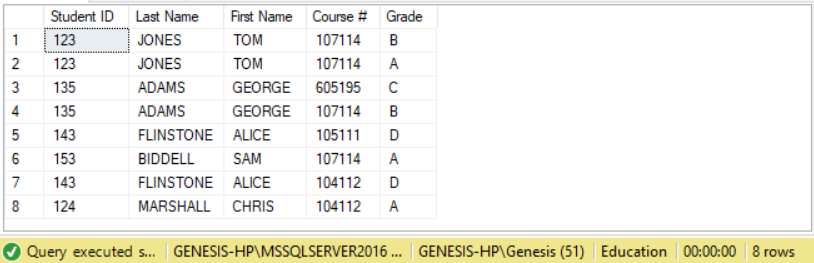
1. Create a view named **vw\_StudentGrade** to show the ***Student ID***, ***Last Name***, ***First Name***, ***Course #***, and ***Grade*** from the **Student** and **Roster** tables. (*Hint: Use JOINs).*

Paste the command you used below and run the ***sp\_helptext <viewName>*** command to show the stored view. Print screen your result below.

**YOUR COMMAND WAS:**

|  |
| --- |
| CREATE View vw\_StudentGrade  AS  SELECT s.StudentID, s.StudentLastName, s.StudentFirstName, r.CourseID, r.Grade  FROM Student as s  INNER Join Roster as r  ON s.studentID = r.StudentID |

Sample:

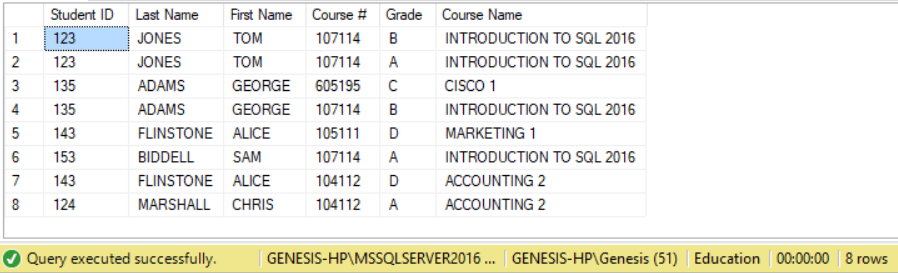


2. Modify the above view to include one additional field ***Course Name*** from the **Course** table.

Paste the command you used below and run the **sp\_helptext** command to show the stored view. Print screen your result below

**YOUR COMMAND WAS:**

|  |
| --- |
| ALTER VIEW vw\_StudentGrade  AS  SELECT s.StudentID, StudentFirstName, StudentLastName, c.CourseID, Grade, CourseName  FROM Student as s  INNER Join Roster as r  ON s.StudentID = r.StudentID  INNER Join Course as c  ON r.CourseID = c.CourseID |

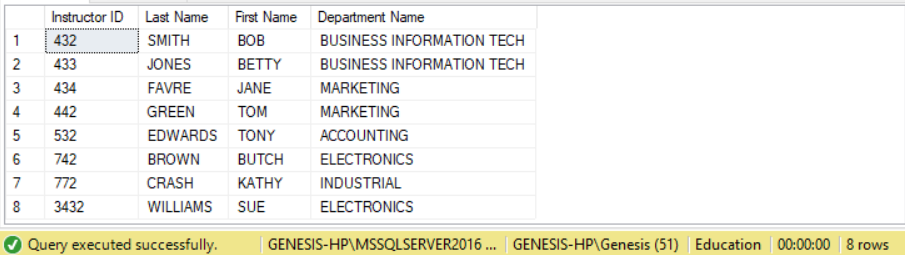


3. Create a view named **vw\_InstructorDepartment** to display the ***Instructor ID***, ***Last Name***, ***First Name***, and ***Department Name*** from the **Instructor** and **Department** tables. Use Schema Binding in this view.

Paste the command you used below and run the **sp\_helptext** command to show the stored view. Print screen your result below.

**YOUR COMMAND WAS**:

|  |
| --- |
| CREATE View vw\_InstructorDepartment  WITH SCHEMABINDING  AS  SELECT InstructorID, InstructorLastName, InstructorFirstName, DepartmentName  FROM Instructor as i  INNER Join Department as d  ON i.DepartmenID = d.DepartmentID |



4. Create a unique clustered index for the view you created in step #3. Name the index **IX\_DepartmentInstructor** and make it on the view you created the fields ***Department Name*** and ***Instructor ID***. Paste the command you used below

**YOUR COMMAND WAS**:

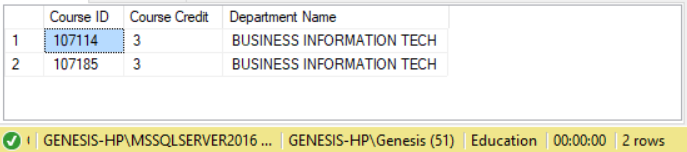
|  |
| --- |
| CREATE Unique Clustered  INDEX IX\_DepartmentInstructor  ON vw\_InstructorDepartment(DepartmentName, InstructorID) |

5. Create a view named **vw\_CourseLookup** that displays the ***Course ID***, ***Course Name***, ***Course Credit*** and ***Department Name*** from the **Department** and **Course** tables where the ***Course ID*** starts with 107. Use the Check option on this view.

Paste the command you used below and run the **sp\_helptext** command to show the stored view. Print screen your result below

**YOUR COMMAND WAS**:

|  |
| --- |
| CREATE View vw\_CourseLookup  AS  SELECT CourseID, CourseName, CourseCredit, DepartmentName  FROM Course as c  INNER Join Department as d  ON c.DepartmenID = d.DepartmentID  WHERE CourseID LIKE '107%' |



6. Using the view **vw\_CourseLookup** (which was made in Step 5), update the value for the record with **Course ID** of 107185 to be a 1 Credit.

Paste the command you used below. Print screen your result below.

**YOUR COMMAND WAS**:

|  |
| --- |
| ALTER VIEW vw\_CourseLookup  AS  SELECT CourseCredit  FROM Course  SET CourseCredit = 1  WHERE CustomerID = 107185 |

Did the update work? Why or Why Not?

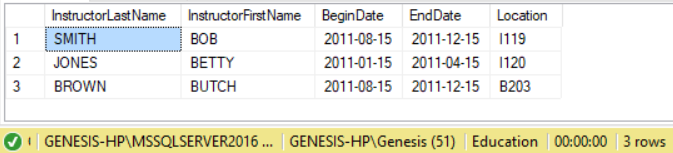
|  |
| --- |
|  |

7. Create an encrypted view named **vw\_InstructorClass** to display the Instructor’s ***Last Name***, ***First Name***, and Class’ ***Begin Date***, ***End Date***, ***Location*** for all classes that have an ***End Date*** in 2011.

Paste the command you used below and run the **sp\_helptext** command to show the stored view. Print screen your result below.

**YOUR COMMAND WAS**:

|  |
| --- |
| CREATE View vw\_InstructorClass  WITH ENCRYPTION  AS  SELECT InstructorLastName, InstructorFirstName, BeginDate, EndDate, Location  FROM Instructor as i  INNER JOIN Class as c  ON i.InstructorID = c.InstructorID  WHERE EndDate = '2011' |



Note: The following commands use the **AdventureWorks** database.

8. Create a view called **vw\_Sales\_PersonSales** in the **AdventureWorks** database that shows each salesperson’s ***FirstName***, ***LastName***, ***CustomerID***, and the total of his or her sales.

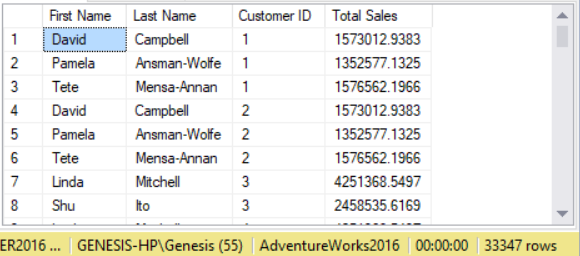
(*Hint: Use Joins, Group By, Aggregators. Think about which FK maps to which PK in the appropriate tables*)

Paste the command you used below and run the **sp\_helptext** command to show the stored view.

**YOUR COMMAND WAS**:

|  |
| --- |
| CREATE View vw\_Sales\_PersonSales  AS  SELECT p.FirstName as 'FirstName', p.LastName as 'LastName', s.CustomerID as 'CustomerID', sum(s.TotalDue) as 'Total Sales'  FROM Sales.SalesOrderHeader as s  INNER JOIN Person.Person as p  ON p.BusinessEntityID = s.SalesPersonID |

Sample Result



9. Modify the view you just created to be encrypted with schema binding.

**YOUR COMMAND WAS**:

|  |
| --- |
| ALTER View vw\_Sales\_PersonSales  WITH Schemabinding  AS  SELECT p.FirstName as 'FirstName', p.LastName as 'LastName', s.CustomerID as 'CustomerID', sum(s.TotalDue) as 'Total Sales'  FROM Sales.SalesOrderHeader as s  INNER JOIN Person.Person as p  ON p.BusinessEntityID = s.SalesPersonID |

Verify that it was success.

10. Drop the view you just created.

**YOUR COMMAND WAS**:

|  |
| --- |
| DROP View vw\_Sales\_PersonSales |

Verify that it was success.

|  |
| --- |
|  |