**152-080 Databases**

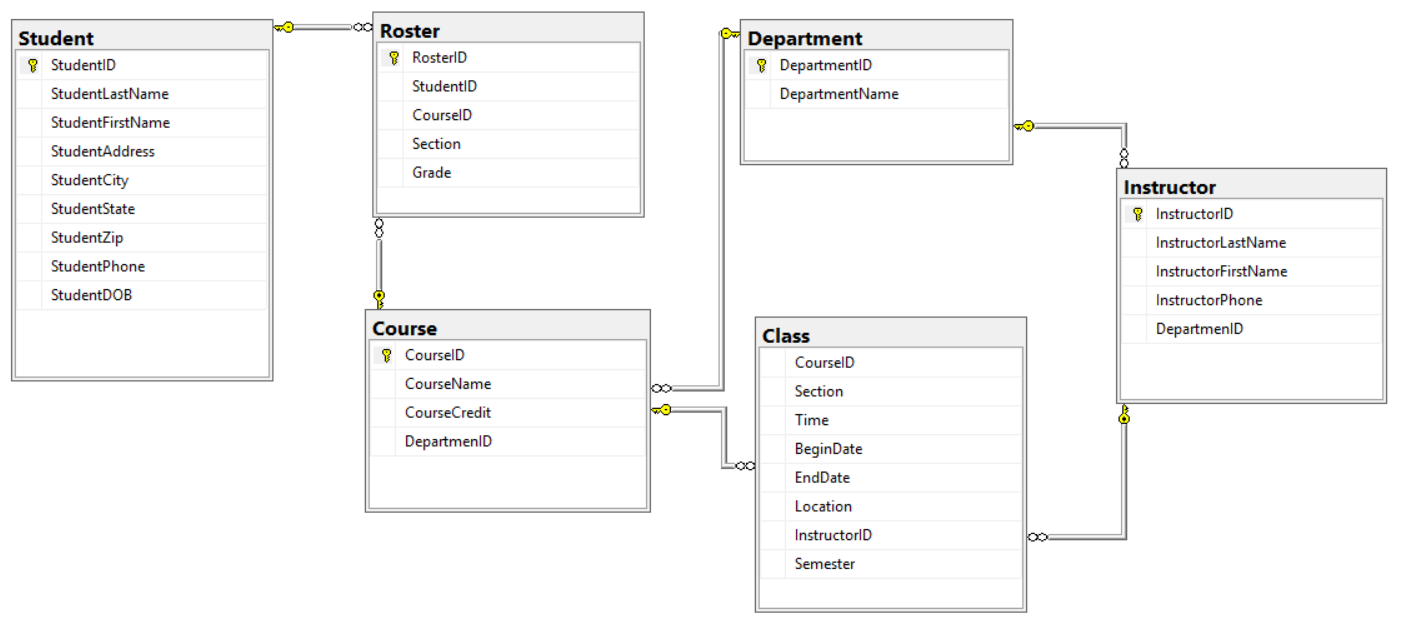
# **Unit 8: Working with Views**

# 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 *U10.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.

# 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

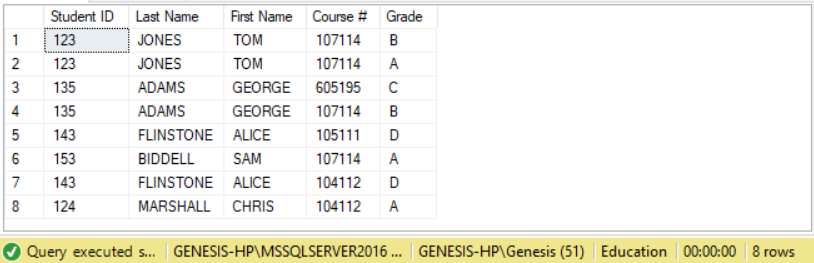
AS SELECT s.StudentID, StudentLastName, StudentFirstName, CourseID, Grade

FROM Student s, Roster r

Table

Description automatically generated

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

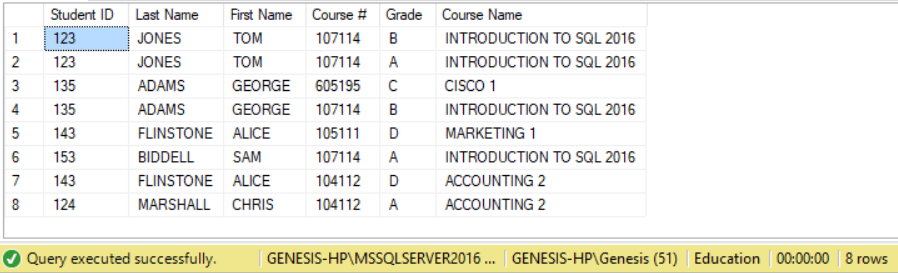
**YOUR COMMAND WAS:**ALTER VIEW vw\_StudentGrade

AS SELECT s.StudentID, StudentLastName, StudentFirstName, r.CourseID, Grade, CourseName

FROM Student s, Roster r, Course c

Table

Description automatically generated



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 i.InstructorID, i.InstructorLastName, i.InstructorFirstName, d.DepartmentName

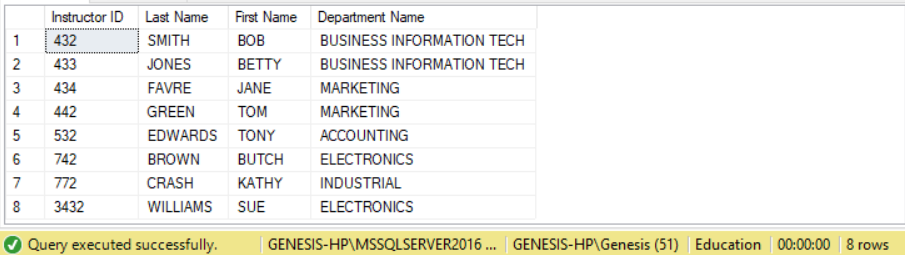
FROM dbo.Instructor i, dbo.Department d

Graphical user interface, text, application

Description automatically generated

Graphical user interface, application, table

Description automatically generated



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

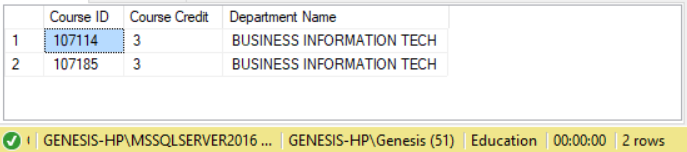
AS SELECT CourseID, CourseName, CourseCredit, DepartmentName

FROM Department d, Course c

WHERE CourseID LIKE '107%'

Table

Description automatically generated with medium confidence



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**:

UPDATE vw\_CourseLookup

SET CourseCredit = 1

WHERE CourseID = 107185

Graphical user interface

Description automatically generated

Did the update work? Why or Why Not?

It did work

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

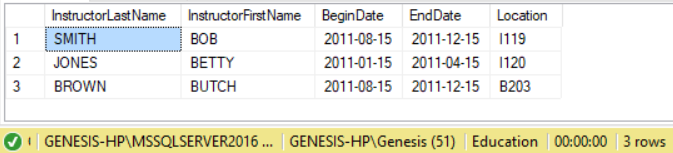
AS SELECT i.InstructorLastName, i.InstructorFirstName, c.BeginDate, c.EndDate, c.Location

FROM Instructor i, Class c

WHERE c.EndDate >= '2011' AND c.EndDate < '2012'

Graphical user interface, table

Description automatically generated with medium confidence



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 pp.FirstName,

pp.LastName,

sc.CustomerID,

SUM(ss.SalesYTD) TotalSales

FROM Sales.SalesPerson ss

JOIN Sales.Customer As sc

ON ss.TerritoryID = sc.TerritoryID

JOIN Person.Person As pp

ON ss.BusinessEntityID = pp.BusinessEntityID

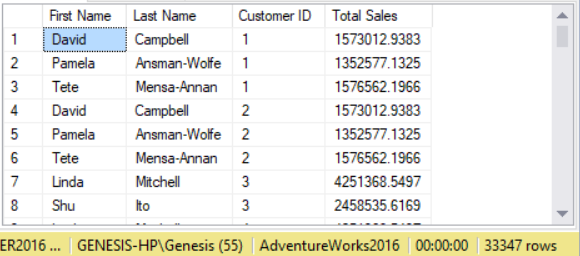
GROUP BY pp.FirstName, pp.LastName, sc.CustomerID;

GO

Graphical user interface, table

Description automatically generated

Sample Result



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

**YOUR COMMAND WAS**:

ALTER VIEW vw\_Sales\_PersonSales

WITH ENCRYPTION, SCHEMABINDING

AS SELECT pp.FirstName,

pp.LastName,

sc.CustomerID,

SUM(ss.SalesYTD) TotalSales

FROM Sales.SalesPerson ss

JOIN Sales.Customer As sc

ON ss.TerritoryID = sc.TerritoryID

JOIN Person.Person As pp

ON ss.BusinessEntityID = pp.BusinessEntityID

GROUP BY pp.FirstName, pp.LastName, sc.CustomerID;

Graphical user interface, table

Description automatically generated with medium confidence

Verify that it was success.

10. Drop the view you just created.

**YOUR COMMAND WAS**:



Verify that it was success.

Text

Description automatically generated