Lab 06
Student Registration

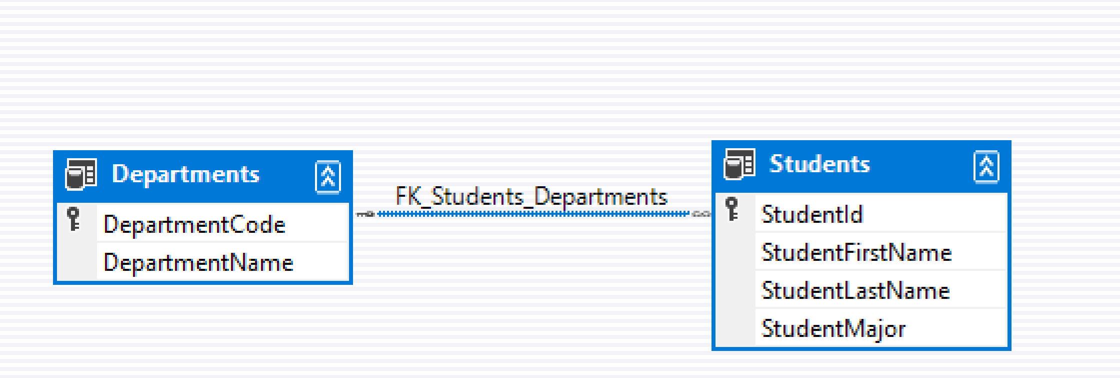
Using ADO.NET Connected Layer

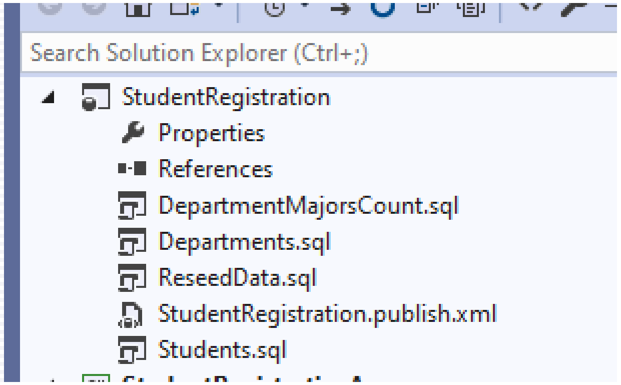
Registration Database and Application

- Unzip and open Visual Studio solution
  - Name: Lab06StudentRegistration
  - Contains
    - DataTableAccessLayer (from class exercises)
    - StudentRegistration
      - Database with only Reseed.sql and publish scripts provided.
    - StudentRegistrationApp
      - WinForms application with UI already created.
- Description
  - Given a database named StudentRegistration with students registered in a department, in the StudentRegisration SQL project
    - Create Students and Departments tables
    - Seed data
    - Create a publish script
  - Create a Windows Forms Application
    - Using DataGridView controls, displays database tables and allows for inserts, updates, and deletes of records.
    - Backs up and restores database to XML file
    - Shows a report in a DataGridView of the number of students in each major (Department)
- Project requirements
  - Create the database
  - Create the Students and Departments tables within the StudentRegistration database project (do NOT do this via the database directly)
  - Complete the code in StudentRegistrationAppForm.cs in the StudentRegistrationApp project in C# to
    - Use DataTableAccessLayer class library (see Class 06 Exercises and Notes on how to do this) and the provided form.
      - Do NOT change either of these
    - Initialize the the DataGridView controls and set DataSource to the relevant DataTable
    - Set up insert/delete/update event handlers for each DataTable
- Show instructor progress made during class. Submit completed lab using Blackboard. Make sure you follow all naming conventions.

Create the StudentRegistration Database

- Create Students.sql and Departments.sql in the StudentRegistration SQL project.
  - Use the ER Diagram below
- StudentId is Integer and a Primary Key. It should have Identity set.
- DepartmentCode and StudentMajor must be nvchar(10).
- All other columns in the tables must be nvchar(50).
- DepartmentCode is a Primary Key. It does NOT have Identity set.
- A one-to-many relationship exists between Departments and Students.
  - Create this relationship.
  - It MUST cascade updates (not deletes).
- Create a View named DepartmentMajorsCount.sql which shows the DepartmentName and NumberOfMajors
  - It should be a count of the number of Students with a major in a Department

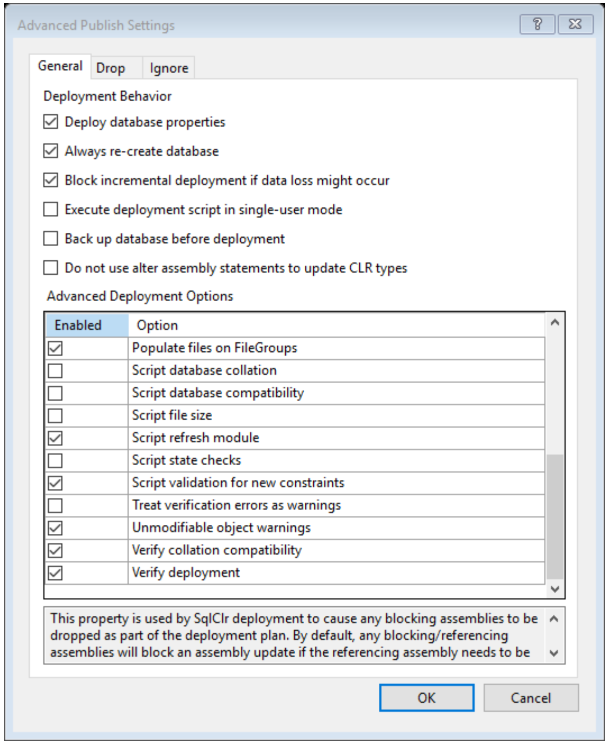



Search Solution Explorer (Ctrl+;)

- StudentRegistration
  - Properties
  - References
  - DepartmentMajorsCount.sql
  - Departments.sql
  - ReseedData.sql
  - StudentRegistration.publish.xml
  - Students.sql

Publish the database to SQL Server

- Create a StudentRegistration.publish.xml Publish profile
  - Make sure Always Recreate the Database is set in Advanced Settings
  - Make sure the script uses (localdb)\MSSQLLocalDB
- Publish the DB to SQL Server
- The ReSeedData.sql script should run and seed the database with initial data.
  - If this has errors, you have not created the tables correctly.
- Check that the database has been created and the tables have data using SQL Server Object Explorer



Advanced Publish Settings

Deployment Behavior

☑ Deploy database properties

☑ Always re-create database

☑ Block incremental deployment if data loss might occur

□ Execute deployment script in single-user mode

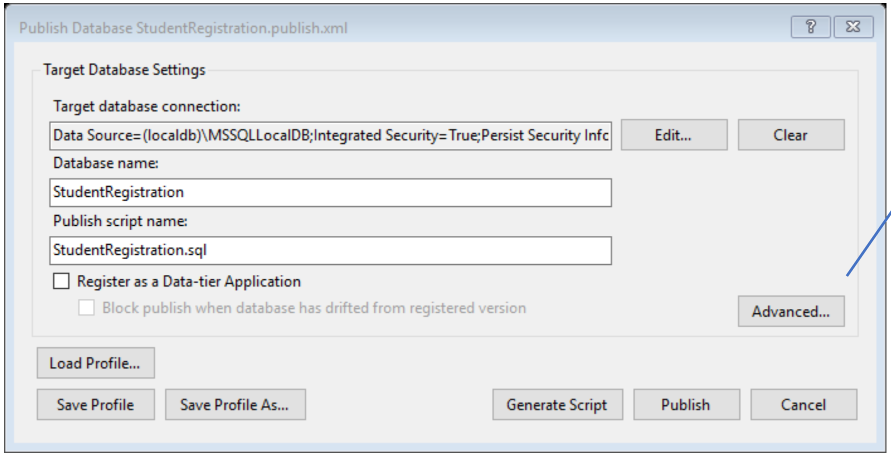
□ Back up database before deployment

□ Do not use alter assembly statements to update CLR types

Advanced Deployment Options

Enabled	Option
☑	Populate files on FileGroups
□	Script database collation
□	Script database compatibility
□	Script file size
☑	Script refresh module
□	Script state checks
☑	Script validation for new constraints
□	Treat verification errors as warnings
☑	Unmodifiable object warnings
☑	Verify collation compatibility
☑	Verify deployment

This property is used by SqlClr deployment to cause any blocking assemblies to be dropped as part of the deployment plan. By default, any blocking/referencing assemblies will block an assembly update if the referencing assembly needs to be



Publish Database StudentRegistration.publish.xml

Target Database Settings

Target database connection:

Data Source=(localdb)\MSSQLLocalDB;Integrated Security=True;Persist Security Infc

Database name:

StudentRegistration

Publish script name:

StudentRegistration.sql

□ Register as a Data-tier Application

□ Block publish when database has drifted from registered version

©Michael Hrybyk and others
NOT TO BE REDISTRIBUTED

Complete the StudentRegistrationApp

- The form in Designer view will have the following DataGridView controls
  - dataGridViewStudents
  - dataGridViewDepartments
  - dataGridViewDepartmentMajorsCount
- Each database table/view should be loaded into a DataTable using DataTableAccessLayer methods.
- dataGridViewStudents and dataGridViewDepartments MUST have
  - DataSource set to a DataTable associated with a database table.
  - AutoSizeColumnsMode set to Fill
  - MultiSelect set to True
- dataGridViewDepartmentMajorsCount MUST have
  - DataSource set to a DataTable associated with a database view.
  - AutoSizeColumnsMode set to Fill
  - ReadOnly set to true
  - user unable to edit or add rows
  - RowHeadersVisible set to false.
- dataGridViewStudents.DefaultValuesNeeded event must set Column 0 to -1
  - This is only done when rows are added
- Set controls' properties in C# code, NOT using the Designer.

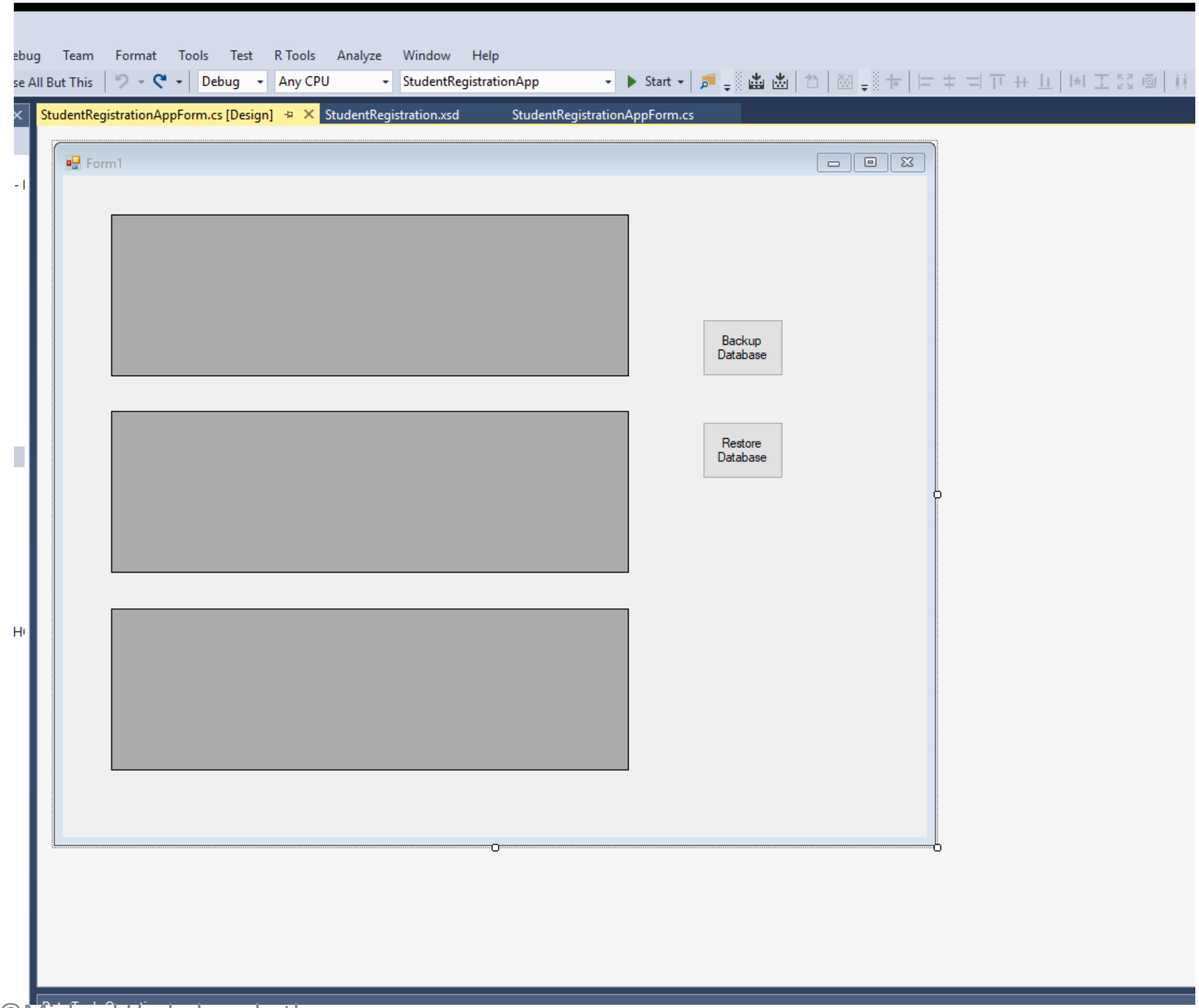
DataTable creation and event handlers

- Students and Departments DataTables must be created and added to a DataSet
- The following event handlers must be created for Students and Departments DataTables
    - RowChanged – to handle insertions
    - ColumnChanged – to handle any column changes (particular cells)
    - RowDeleted – to handle deletions
- Create the event handlers using DataTableAccessLayer methods
    - Hint: use RowState actions to determine which method to invoke.
- Handle all exceptions

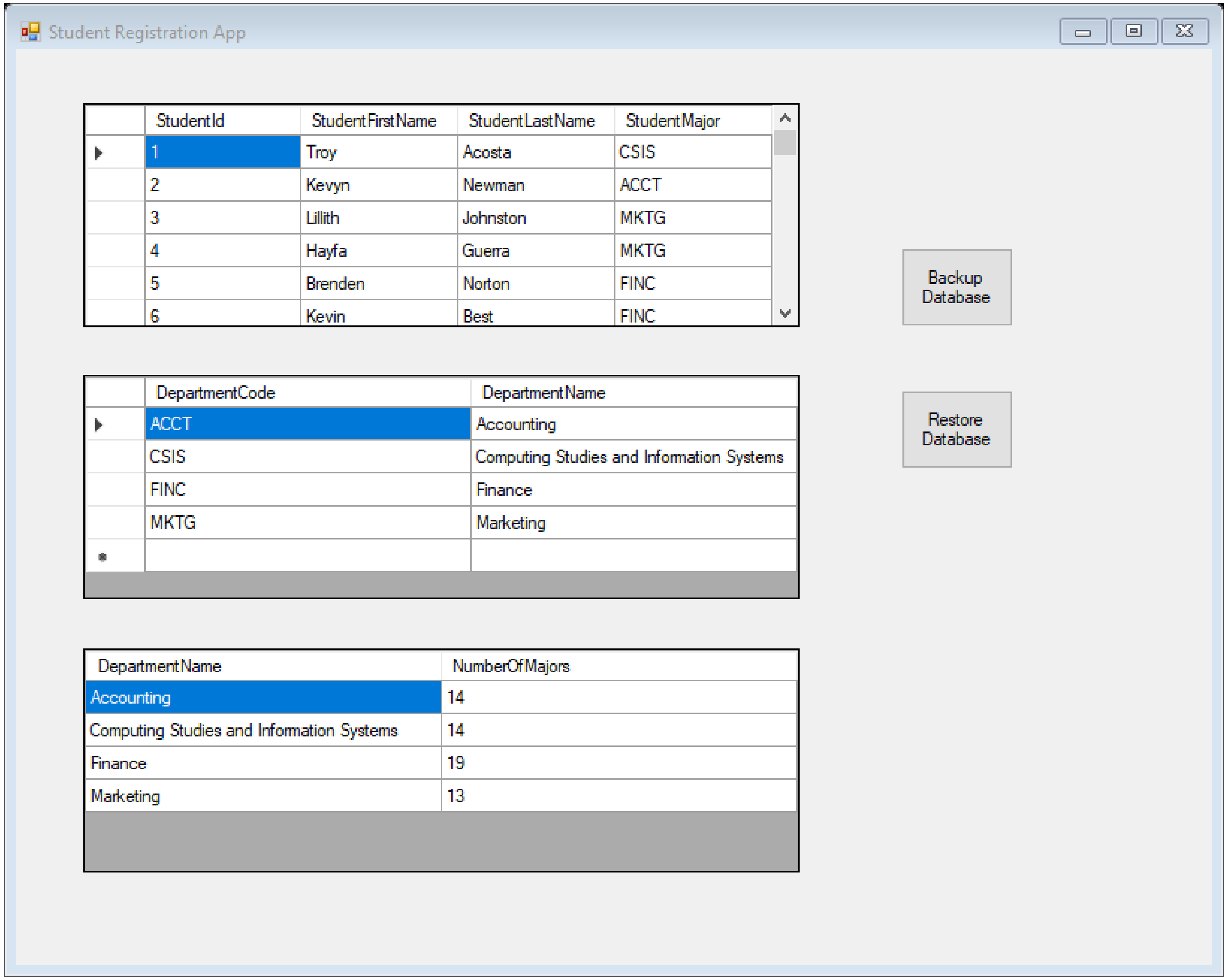
DB Backup and Restore

- Two buttons are already created in the form
    - buttonBackupDatabase
    - buttonRestoreDatabaseFromBackup
- Create Click event handlers to backup and restore the database using DataTableAccessLayer methods.
- Only backup the tables using the created DataSet
    - Contains Students and Departments DataTables.
- Check the .xml file to examine the DB backup file

Initial blank form from Designer



Completed App at startup

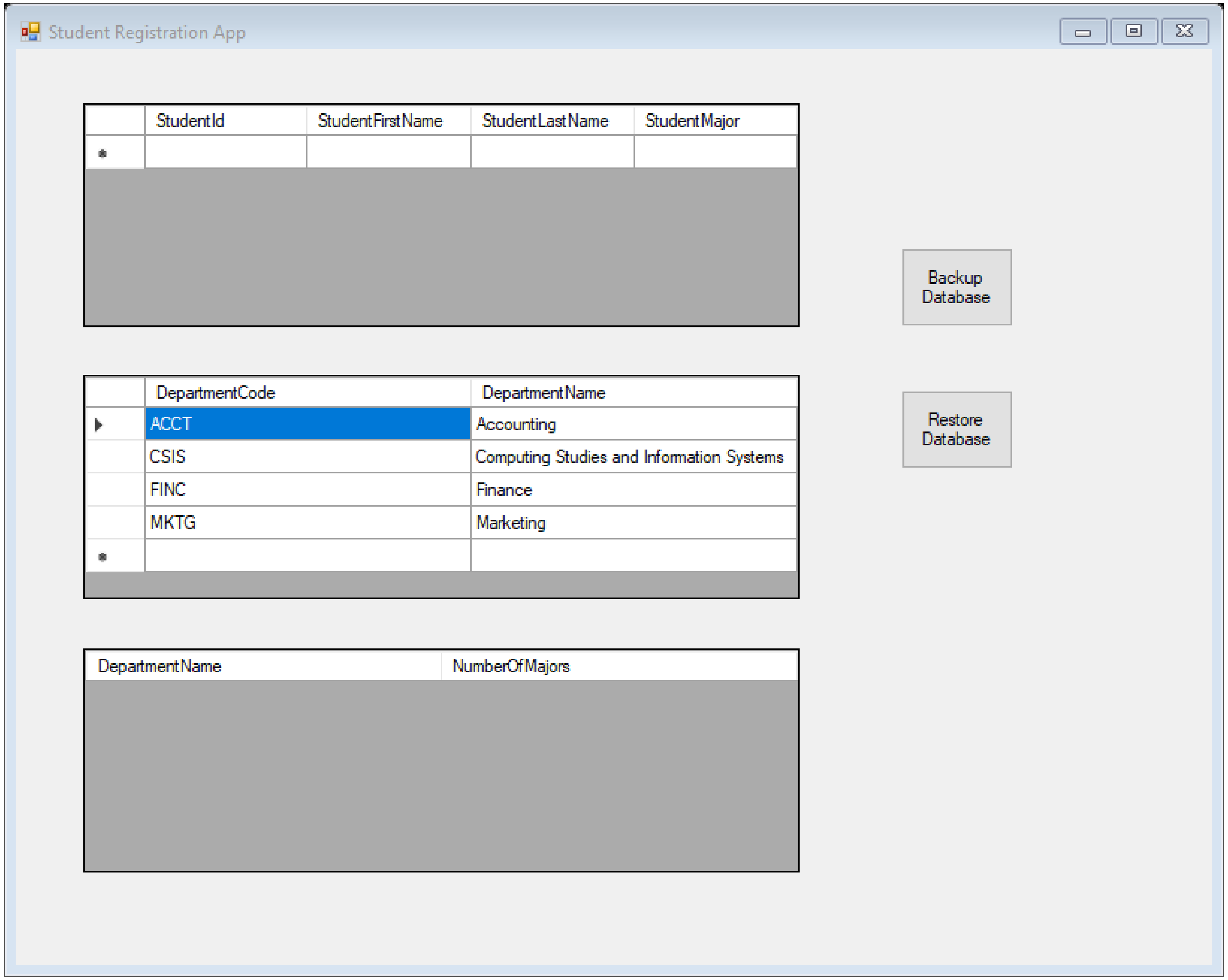


StudentId	StudentFirstName	StudentLastName	StudentMajor
1	Troy	Acosta	CSIS
2	Kevyn	Newman	ACCT
3	Lillith	Johnston	MKTG
4	Hayfa	Guerra	MKTG
5	Brenden	Norton	FINC
6	Kevin	Best	FINC

DepartmentCode	DepartmentName
ACCT	Accounting
CSIS	Computing Studies and Information Systems
FINC	Finance
MKTG	Marketing

DepartmentName	NumberOfMajors
Accounting	14
Computing Studies and Information Systems	14
Finance	19
Marketing	13

All students deleted

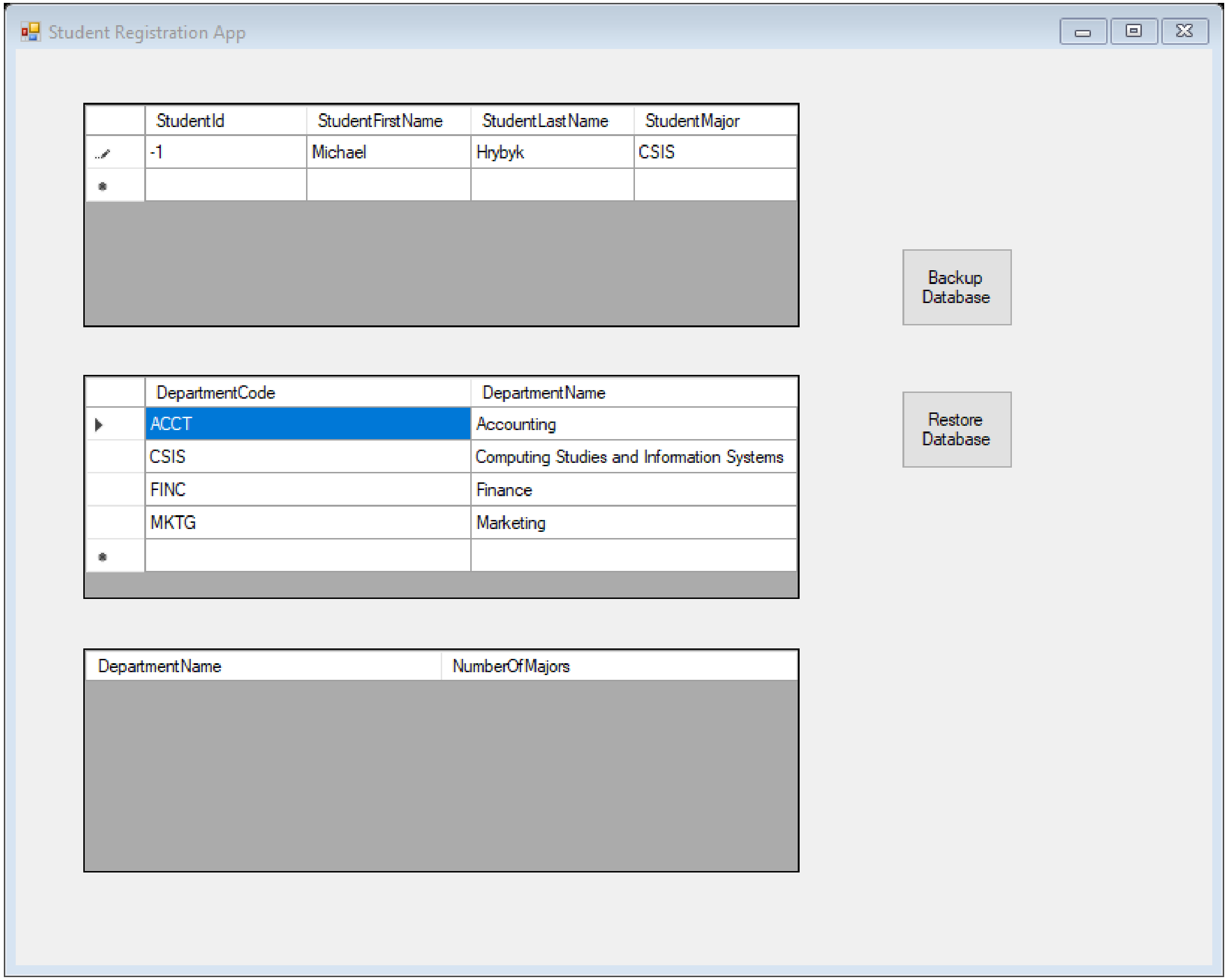


	StudentId	StudentFirstName	StudentLastName	StudentMajor
*				

	DepartmentCode	DepartmentName
▶	ACCT	Accounting
	CSIS	Computing Studies and Information Systems
	FINC	Finance
	MKTG	Marketing
*		

DepartmentName	NumberOfMajors

New student being added

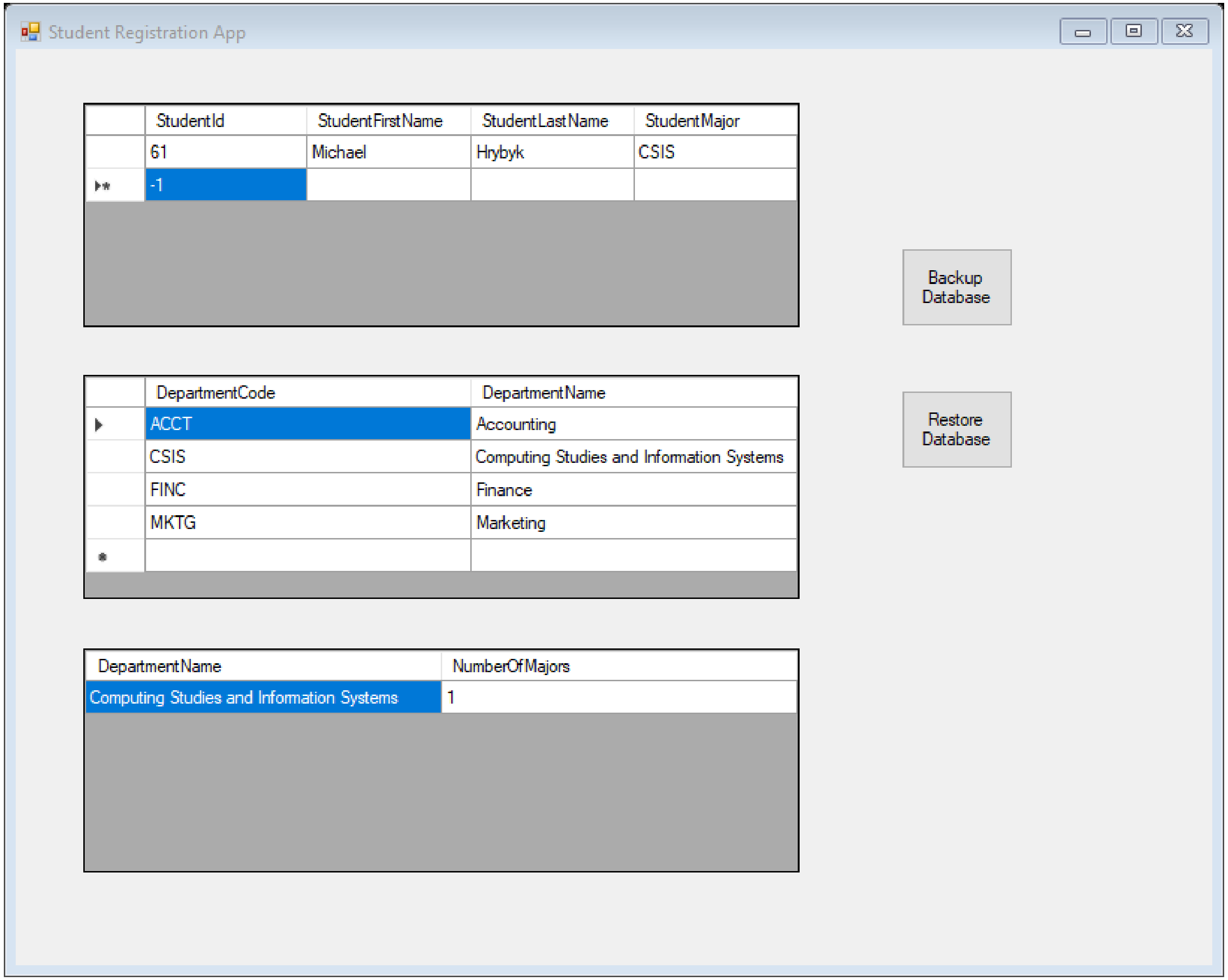


	StudentId	StudentFirstName	StudentLastName	StudentMajor
✎	-1	Michael	Hrybyk	CSIS
*				

	DepartmentCode	DepartmentName
▶	ACCT	Accounting
	CSIS	Computing Studies and Information Systems
	FINC	Finance
	MKTG	Marketing
*		

DepartmentName	NumberOfMajors

Student added

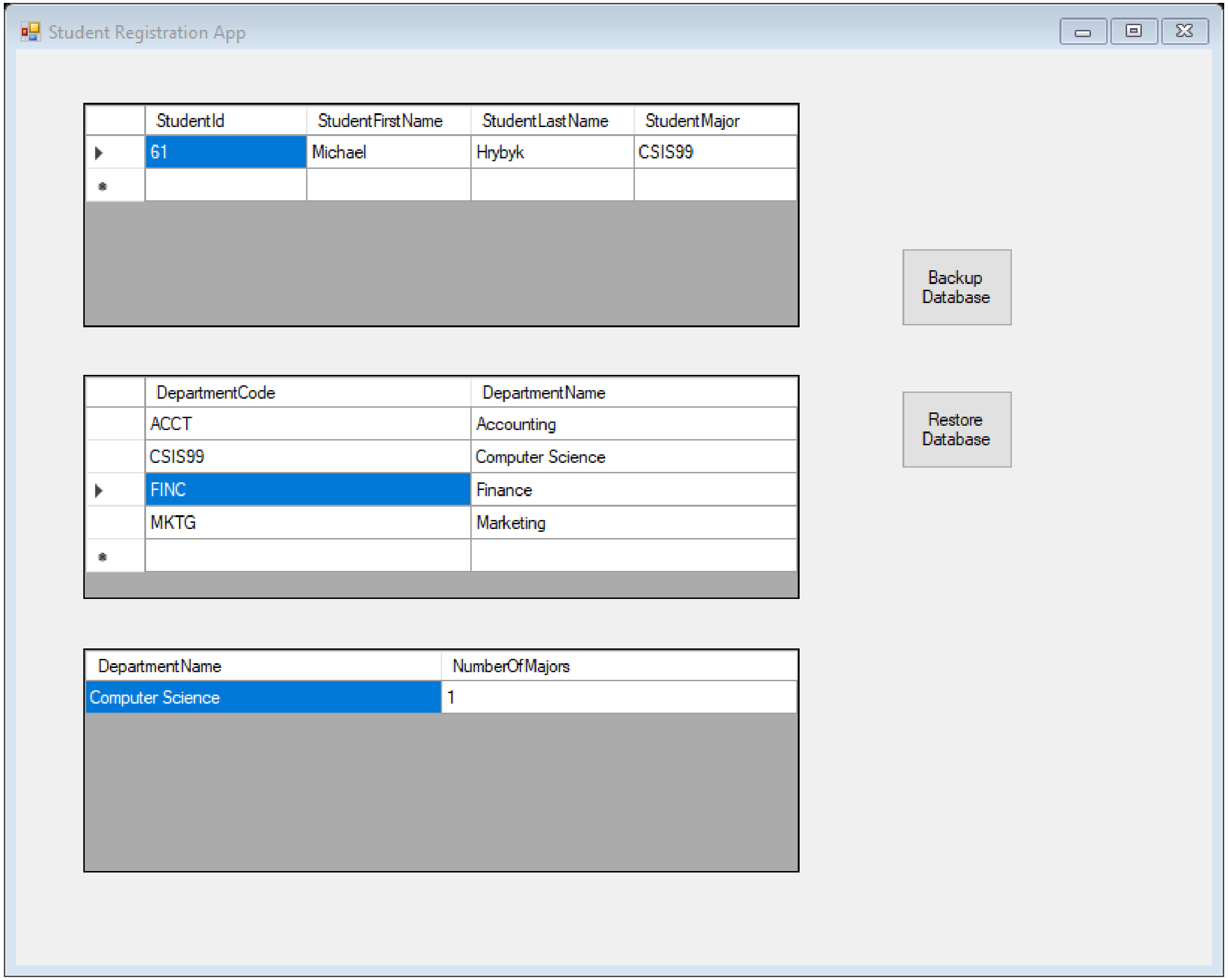


	StudentId	StudentFirstName	StudentLastName	StudentMajor
	61	Michael	Hrybyk	CSIS
▶*	-1			

	DepartmentCode	DepartmentName
▶	ACCT	Accounting
	CSIS	Computing Studies and Information Systems
	FINC	Finance
	MKTG	Marketing
*		

DepartmentName	NumberOfMajors
Computing Studies and Information Systems	1

Columns changed – notice cascade

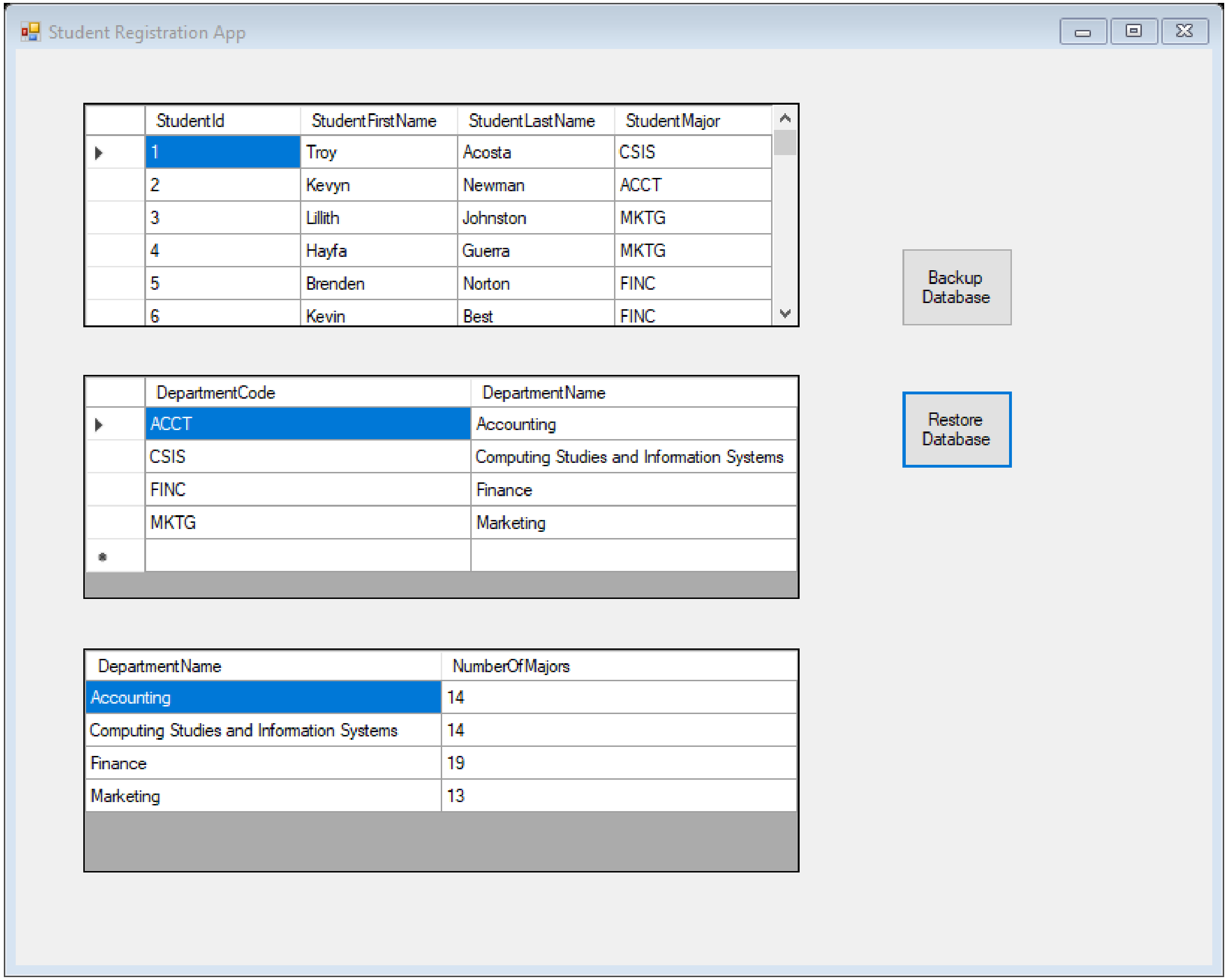


	StudentId	StudentFirstName	StudentLastName	StudentMajor
▶	61	Michael	Hrybyk	CSIS99
*				

	DepartmentCode	DepartmentName
	ACCT	Accounting
	CSIS99	Computer Science
▶	FINC	Finance
	MKTG	Marketing
*		

DepartmentName	NumberOfMajors
Computer Science	1

Database backed up and restored

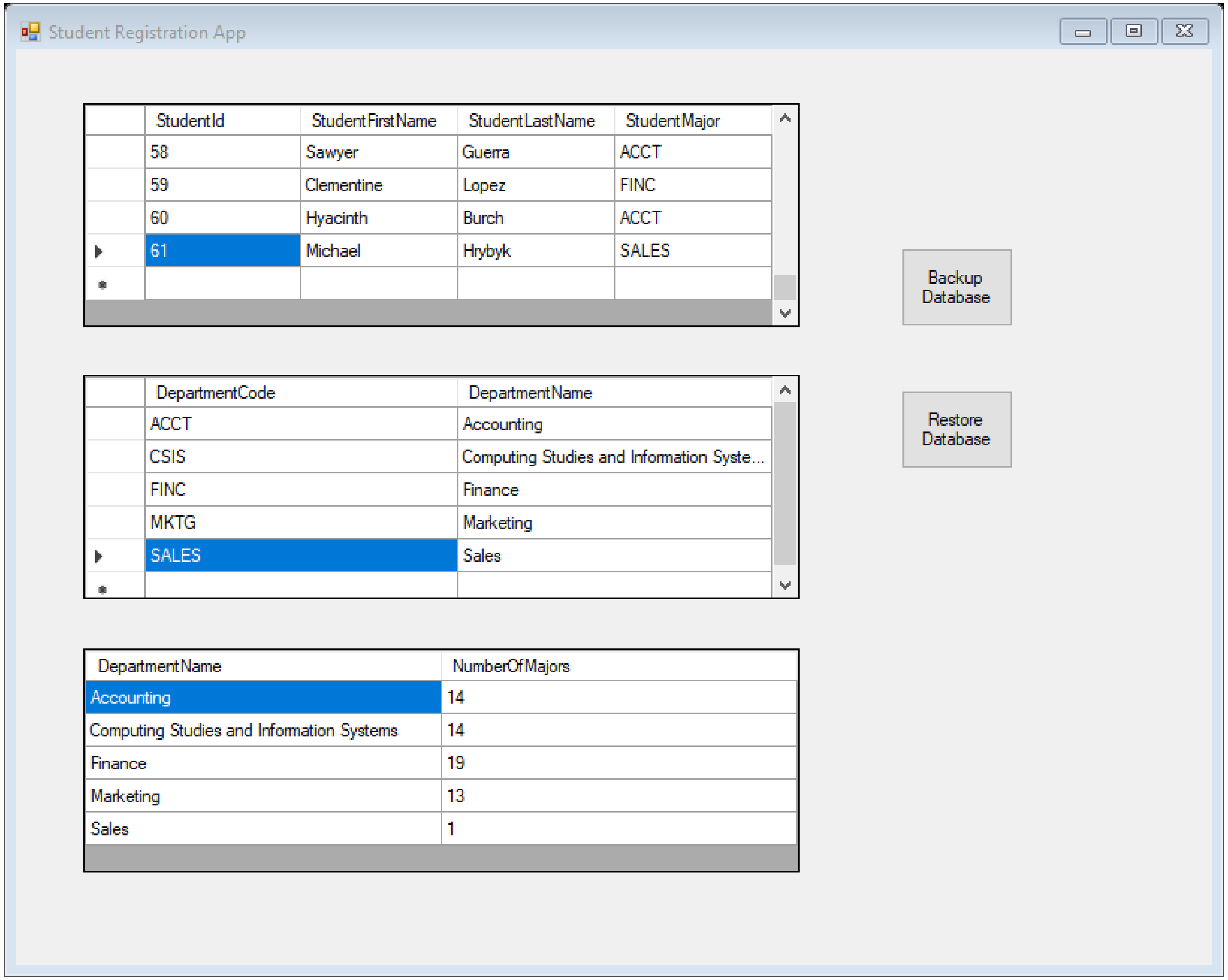


StudentId	StudentFirstName	StudentLastName	StudentMajor
1	Troy	Acosta	CSIS
2	Kevyn	Newman	ACCT
3	Lillith	Johnston	MKTG
4	Hayfa	Guerra	MKTG
5	Brenden	Norton	FINC
6	Kevin	Best	FINC

DepartmentCode	DepartmentName
ACCT	Accounting
CSIS	Computing Studies and Information Systems
FINC	Finance
MKTG	Marketing

DepartmentName	NumberOfMajors
Accounting	14
Computing Studies and Information Systems	14
Finance	19
Marketing	13

Student and Department added



Student Registration App

	StudentId	StudentFirstName	StudentLastName	StudentMajor
	58	Sawyer	Guerra	ACCT
	59	Clementine	Lopez	FINC
	60	Hyacinth	Burch	ACCT
▶	61	Michael	Hrybyk	SALES
*				

	DepartmentCode	DepartmentName
	ACCT	Accounting
	CSIS	Computing Studies and Information Syste...
	FINC	Finance
	MKTG	Marketing
▶	SALES	Sales
*		

DepartmentName	NumberOfMajors
Accounting	14
Computing Studies and Information Systems	14
Finance	19
Marketing	13
Sales	1

Notes

- Departments can not be deleted unless all students majoring in that department are first deleted (cascade delete is NOT allowed).
- Cascade update is allowed, so you must reload the Students DataTable whenever DepartmentCode is changed in Departments.
- There are 60 students and 4 departments initially.
- The DataSet name will also be the name of the xml backup file in bin/Debug. Make sure you set this.
- Do NOT use Microsoft table adapters for this lab. You must use DataTableAccessLayer library.
- Feel free to reuse the code from the Class 06 exercises.
- Comment ALL methods and code completely.
- Format ALL code correctly.