Read Me Document

Team 10 : SQL Samurai

Data Sources

1. Smith School Program websites
2. UMD Directory
3. Student Outcomes
4. US News
5. QS World University Rankings
6. Quantnet
7. College Factual
8. Financial Times
9. Economist
10. Linkedin
11. Faculty CVs
12. Research Publications
13. Smith School Awards 14 Faculty Grants for Innovation Research

References

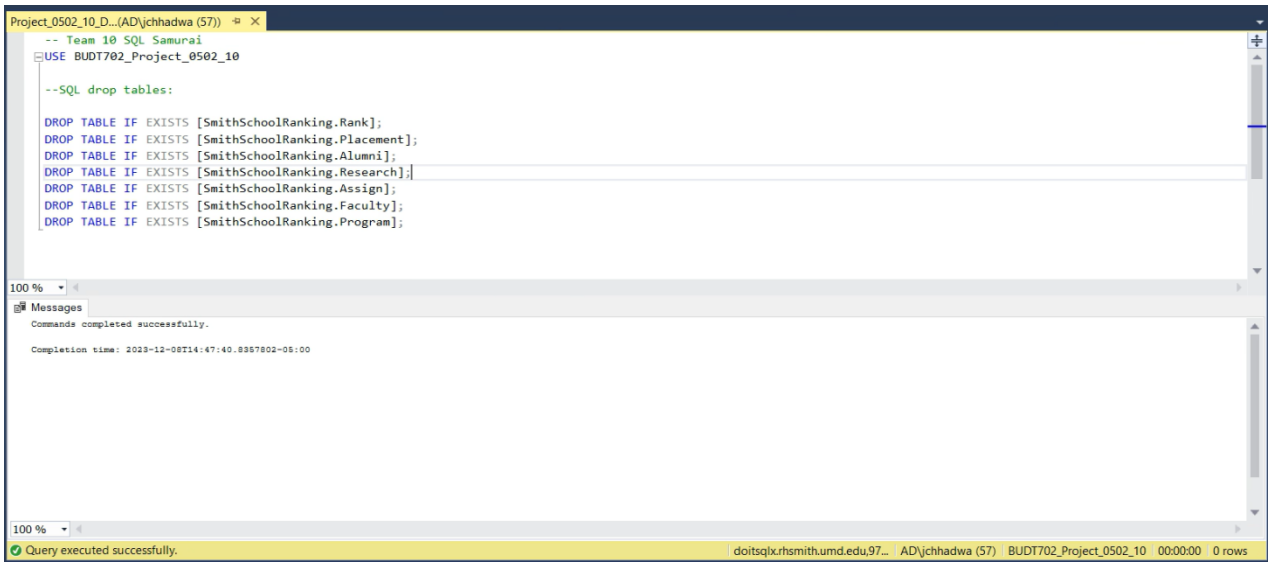
1. Modern Database Management 13th Edition' by Jeffrey A. Hoffer, V. Ramesh & Heikki Topi
2. https://learn.microsoft.com/en-us/sql/relational-databases/tutorial-getting-started-with-the-database-engine?view=sql-server-ver16
3. https://www.lucidchart.com/pages/tour

Testing The Project

- Open the Microsoft SQL Server Management Studio using Virtual Desktop. Use your UMD login credentials and server name as **desktop.rhsmith.umd.edu** and server number as **doitsqlx.rhsmith.umd.edu,9703**.

- Make sure to use the **USE BUDT702_Project_0502_10** database for this project

1. Initially, we will execute the **Project_0502_10_Drop_Table_Statements.sql** file to ensure that the database contains no duplicate tables with the same name as the tables we intend to utilize. Should they exist, we will remove them from the database. Following code execution, the following output should be obtained:



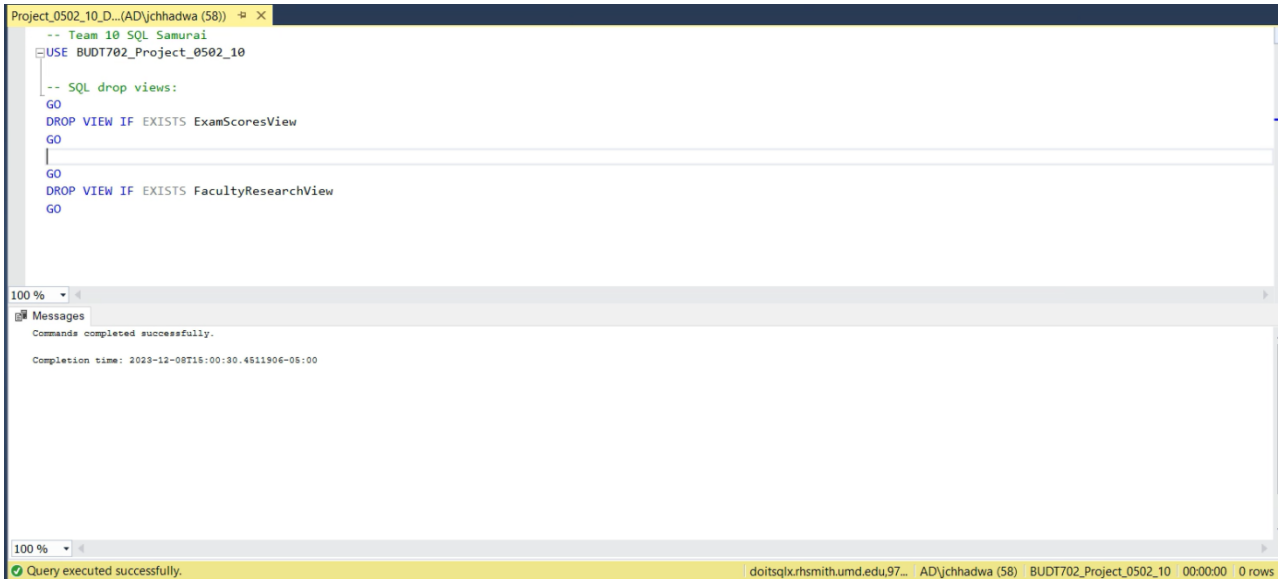
2. The file **Project_0502_10_Create_Table_Statements.sql** will then be executed in order to generate each table needed for the project. We should see the following output after running the code, indicating that the tables were successfully created.

```
Project_0502_10_Cr...(AD\jchhadwa (58))

CREATE TABLE [SmithSchoolRanking.Alumni] (
alumniID CHAR (2) NOT NULL,
alumniGraduationYear CHAR (4),
alumniName VARCHAR (75),
alumniCompanyName VARCHAR (75),
alumniDesignation VARCHAR (75),
prgmID CHAR (2),
    CONSTRAINT pk_alumniID PRIMARY KEY (alumniID),
    CONSTRAINT fk_Program_prgmID3 FOREIGN KEY (prgmID)
        REFERENCES [SmithSchoolRanking.Program] (prgmID)
        ON DELETE NO ACTION
        ON UPDATE NO ACTION
);

100 %

Messages
  Commands completed successfully.

  Completion time: 2023-12-08T14:55:49.5678373-05:00

100 %

Query executed successfully.   doitsqlx.rhsmith.umd.edu,97...   AD\jchhadwa (58)   BUDT702_Project_0502_10   00:00:00   0 rows
```

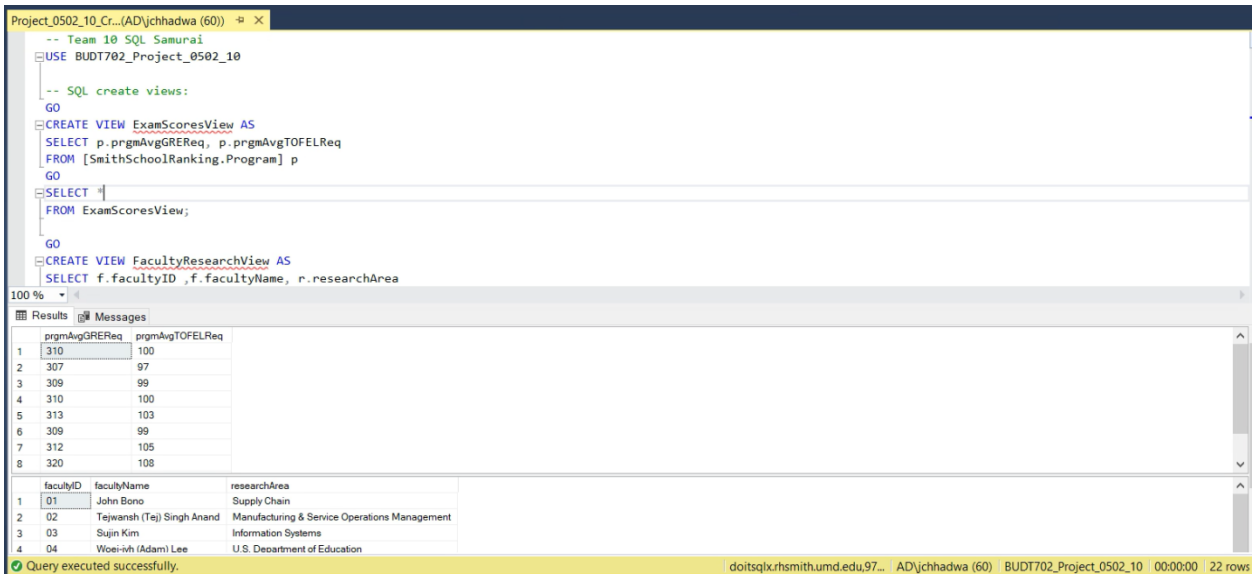
3. We use the file **Project_0502_10_Insert_Into_Statements.sql** to enter data records into each and every table after they are created. The following output will appear after the code has executed, indicating that the records have been successfully inserted.

```
Project_0502_10_In...(AD\jchhadwa (59))

    ('23','2020','Megha Komarraju','NICE CXone','Implementation Analyst','01'),
    ('24','2022','Carrie Walczak','Capital One','Finance Associate','05'),
    ('25','2022','Niani McDonald','UMD Smith School of Business','Data Analyst','04'),
    ('26','2022','Tennie (U-Tung) Chang','Wunderman Thompson','Analyst','04'),
    ('27','2016','Benjamin Eidelberg','Washington Wizards','Director, Basketball Strategy & Analytics','04'),
    ('28','2022','Farid Freyha','Employbridge','Marketing Analyst','04'),
    ('29','2019','Harriet Peng','Navy Federal Credit Union','Data Scientist','04'),
    ('30','2020','Tanisha Sabhaney','WSAudiology','Product Marketing Analyst','04');

INSERT INTO [SmithSchoolRanking.Assign]
VALUES
    ('01','01','BUDT704'),
    ('03','02','BUDT731'),
    ('02','01','BUDT721'),
    ('04','01','BUDT702'),

100 %

Messages
  (10 rows affected)

  (20 rows affected)

  (40 rows affected)

  (12 rows affected)

  (25 rows affected)

  (30 rows affected)

  (17 rows affected)

  Completion time: 2023-12-08T14:58:20.7893979-05:00

100 %

Query executed successfully.   doitsqlx.rhsmith.umd.edu,97...   AD\jchhadwa (59)   BUDT702_Project_0502_10   00:00:00   0 rows
```

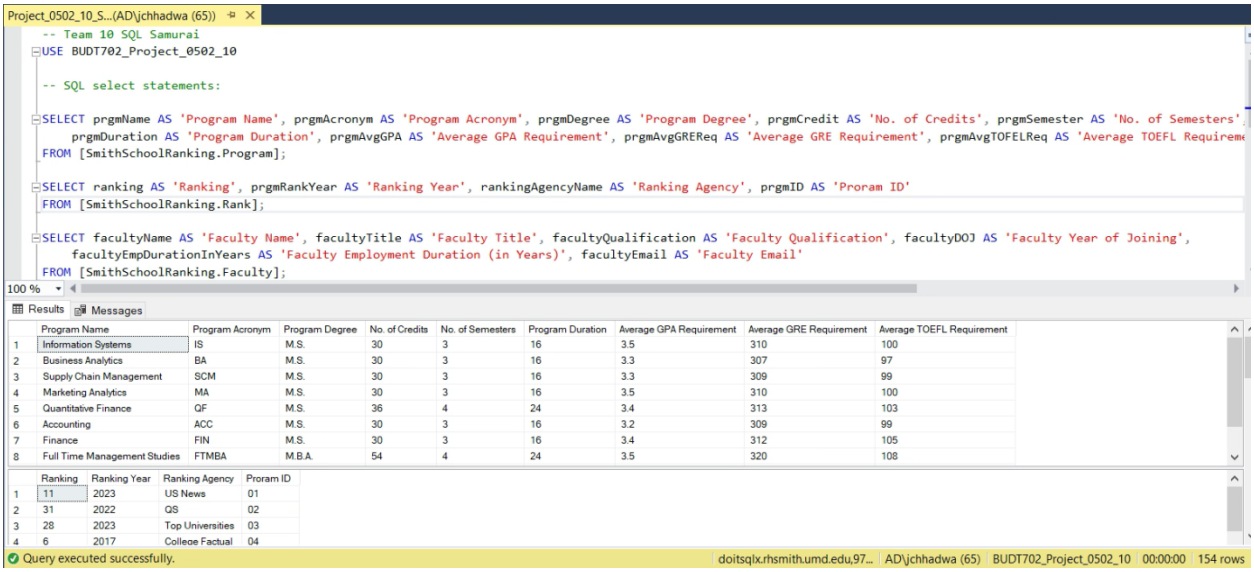
4. We will drop any views that might be present in our database with the same name once all the data records have been inserted into all the created tables by running the file **Project_0502_10_Drop_View_Statements.sql** to ensure there are no discrepancies. When we run this code, the following output should appear:



5. After the elimination of every view, we run the file **Project_0502_10_Create _View_Statements.sql**, which creates and displays views. When we run this code, the following output should appear:



6. The file **Project_0502_10_Select_Statements.sql** is then executed. We are able to see every table in our database, along with every record of data, by running this file. We should get the following result after running the code, indicating that the tables were correctly shown.



7. Finally, we will run the file **Project_0502_10_Business_Transacction_Statements.sql** to look for different business transactions. We can view different analyses by executing this file. Following code execution, the following output—which includes a number of tables and views relevant to a certain analysis—should appear. Moreover, depending on the type of analysis we like to conduct, we can execute a specific query. The file's top contains a list of several analyses.