**Course Selection Database Project Report**

**Marie Hasegawa**

**Alexei Harris**

**Cole Young**

**Florida Polytechnic University**

**November 2020**

**Table of Contents**

1. **Cover Page**……………………………………….……...........……………….…………….…...1
2. **Table of Contents**…………………………….………...........………………….………….…....2
3. **Project Overview**……………………………..……….....………………..….……….…………3
4. **ERM (Entity Relationship Model) Diagram**……………………………..….…………………3

**2.1. ERM (Entity Relationship Model) Diagram** ….……………………..….……………......3

1. **PostgreSQL Database**………....…………………….……………………..….……………......5

**3.1. Major Enumerated Type** …...........................………….....…………..….……………......5

**3.2. Student\_Type Table** …..........................................……………………..….……………......6

**3.3. Graduate Table** ………………...............................................………...….…………….......6

**3.4. Undergraduate Table** ………......................................………………..….……………........7

**3.5. Student Table** ……………………....................................................…..….……………......7

**3.6. Course Table** ……...................................................…………………..….……………........8

**3.7. Department Table** ……............................................…………………..….…………….......8

**3.8. Course\_Offering Table** ………………….....................................……..….……………......9

**3.9. Professor Table** ………………………..….……………......................................................10

**3.10. Student\_Course\_Of Table**………………...........................................….……………......10

1. **MongoDB Database** …………....…….……………………………………..………........……11

**4.1 Student\_Info Collection** ……....…….…………………………………..………..........…11

**4.2 Student\_Advisor\_Comment Collection** ……....…….…………………………….......…12

**4.3 Pending\_Completed Collection**...…….………………........................…………….....……13

**4.4 Pending\_Schedule Collection** …….…………….....…........................…………….....……14

1. **Conclusion**……………………....………………………...…………..……….……………......15
2. **References** …………...............……………………………....………..……….…………….....16

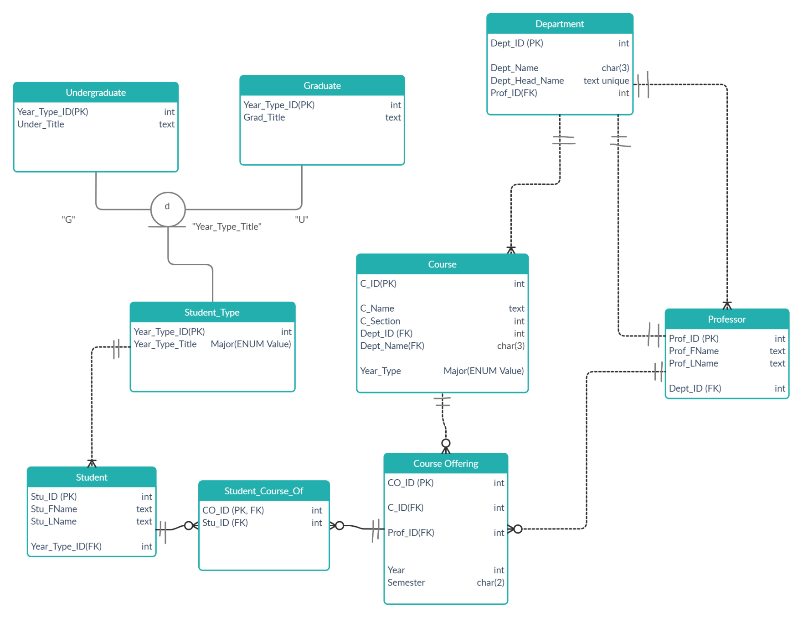
1. **Project Overview**

The goal of this project was to use two separate databases a relational PostgreSQL database and a nonrelational MongoDB database to create a working database system that could easily be implemented into a Course Registry online application for students attending higher-level education.

The PostgreSQL database act as a collection of relational tables that is used to organize data regarding a student’s academic information that would determine the student’s eligibility to a certain course that correlate to their degree and major.

The MongoDB database act as a nonrelational, document-driven database that stores the data in collections and documents that correlate to the course registration by indexing pending course registrations, pending course schedules, and recorded conversations between students and advisors concerning which courses would suit the students’ need.

1. **ERM (Entity Relationship Model) Diagram**



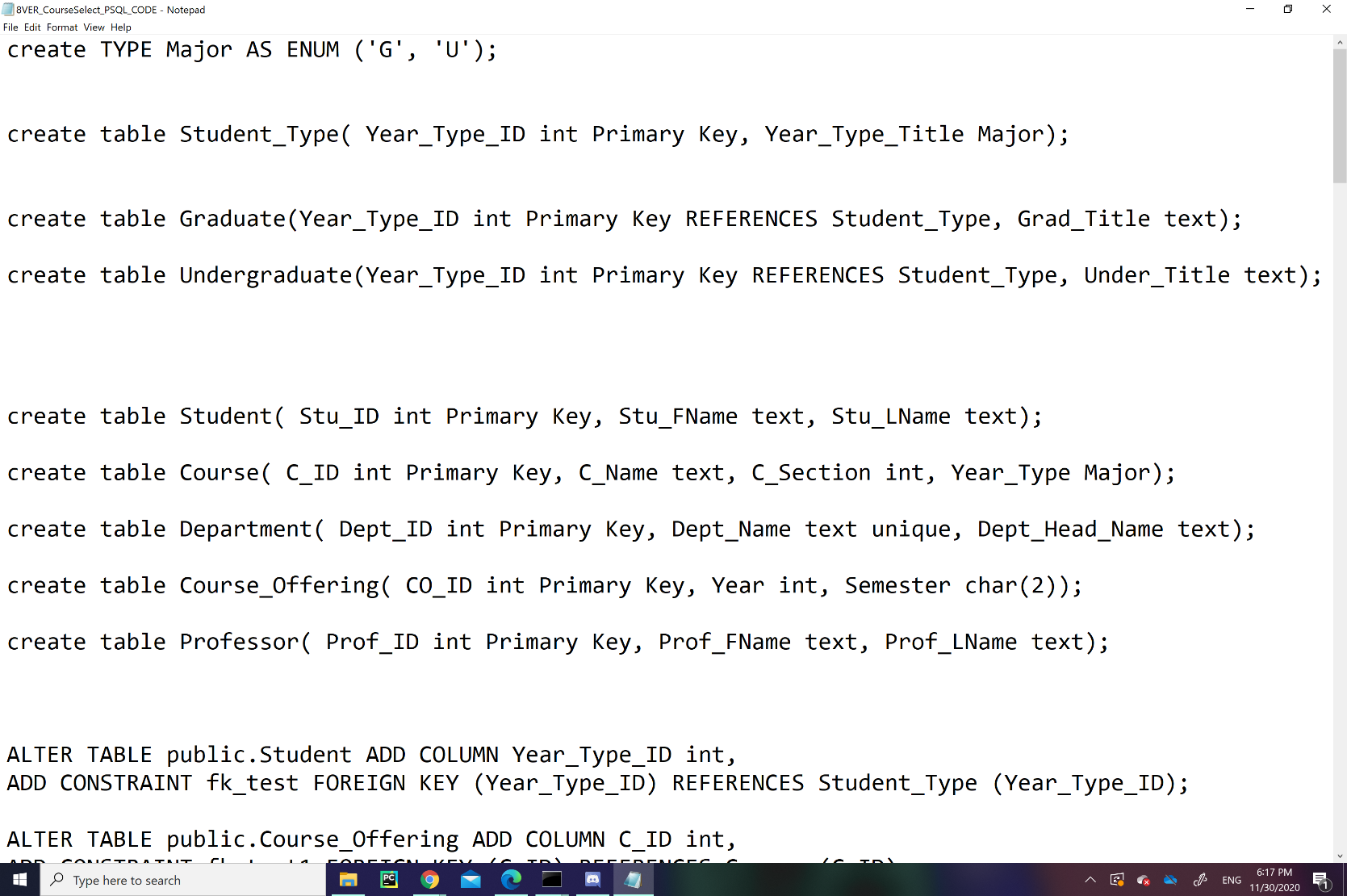
*2.1 Course Selection ERM Diagram*

The Entity-Relationship Model, shown above, was used to develop the Course Selection PostgreSQL database, which will be further explained in the next section. The table, “Student\_Type,” act as a subtype discriminator that determines the student as being a Graduate or Undergraduate, which are correlated to the subtype tables, “Undergraduate” and “Graduate.” The table, “Student\_Course\_Of ” act as a mediator between “Student” and “Course\_Offering,” so that the variables “CO\_ID” and “Stu\_ID” can be contained in the same table. This has also led to having strong relationship between tables “Student,” “Student\_Course\_Of,” and “Course\_Offering" since the “CO\_ID” act as the primary key on the table, “Student\_Course\_Of.”

The tables “Student\_Type” and “Student” have a weak, one to many relationship, since one “Student\_Type” and can have many “Students” but a “Student” can only have one “Student\_Type.” The tables “Course\_Offering” and “Course” have a weak, many to one relationship, since a “Course” can have many “Course Offerings,” but a “Course” can have only one “Course\_Offering.” The “Department” and “Course” tables have a weak one to many relationships, since a “Department” can have many “Courses,” but a “Course” can have only one “Department.” The tables, “Course\_Offering” and “Professor” have a weak, many to one relationship, since a “Professor” can have many “Course\_Offerings” but one “Course\_Offering” can have only one “Professor.” The 1st relationship between tables “Department” and “Professor” show a weak, one-to-many relationship, since a “Department” can have many “Professors”, but one “Professor” can only have one “Department”. The 2nd relationship between tables “Department” and “Professor” show a weak, one-to-one relationship, since one “Department” can only have one Head-of-Department and a Head-of-Department can only be head of one “Department”.

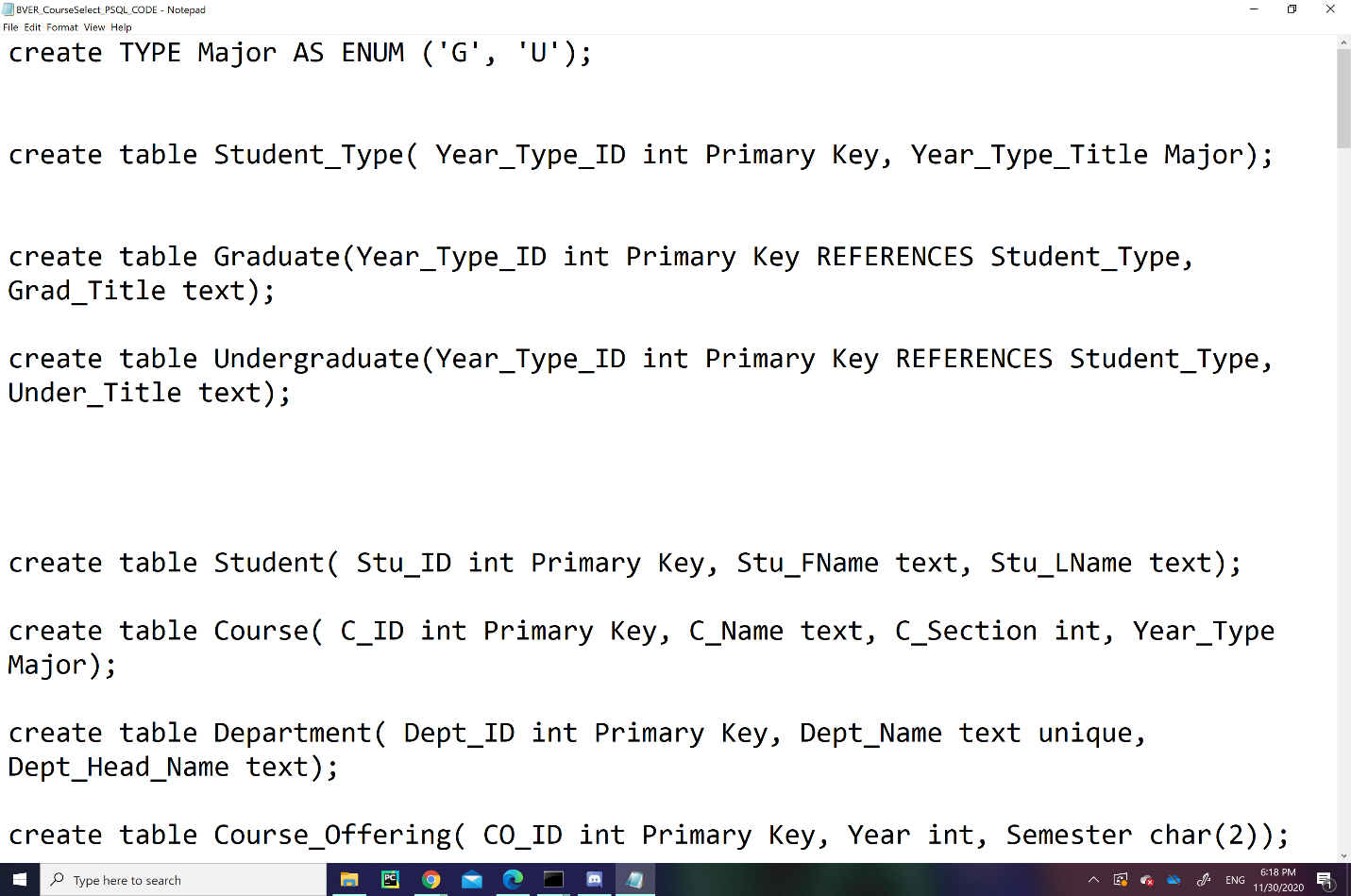
1. **PostgreSQL Database**

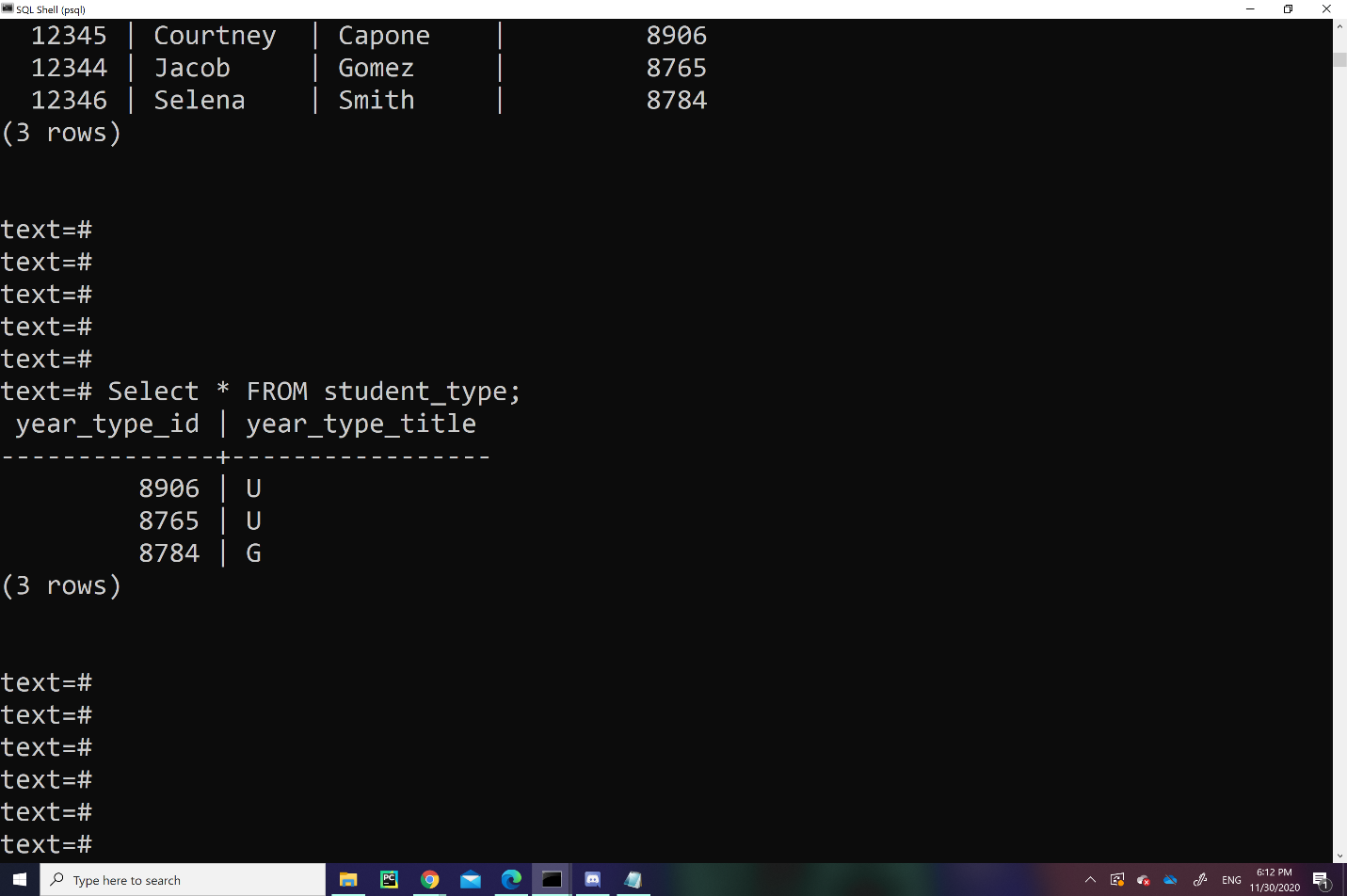
PostgreSQL is a relational database that corresponds to the ERM diagram shown in the previous section. In other words, the foreign keys, subtype discriminator, and the weak and strong relationships, and the one to one and many to many relationships will play a crucial role in the PostgreSQL portion of the “Course Selection” project. The database is used as a collection of relational tables that can be used to collect data about a student’s academic information which would determine the student’s eligibility to a course and if they can register in future semesters.



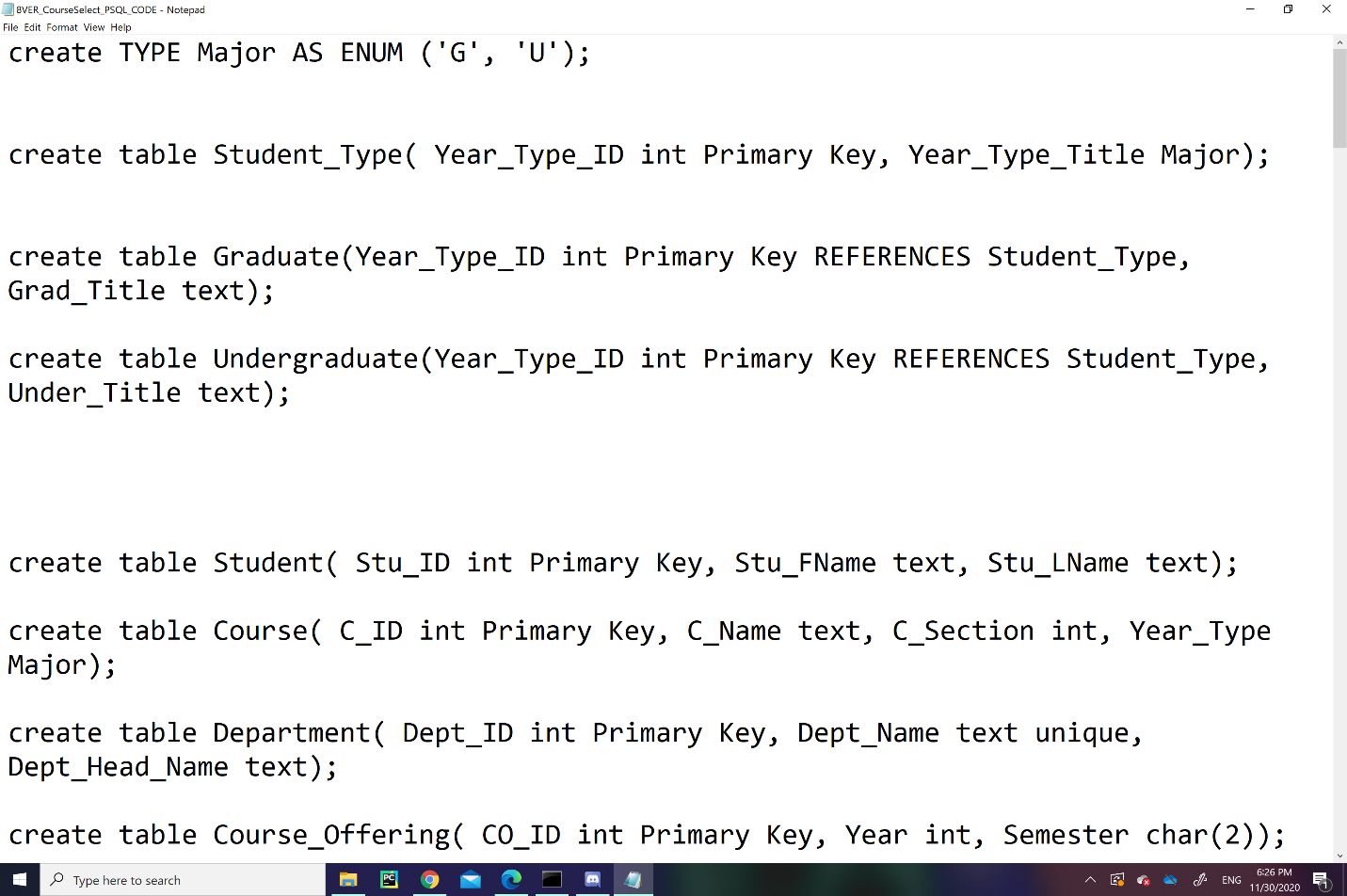
*3.1 Major Enumerated Type*

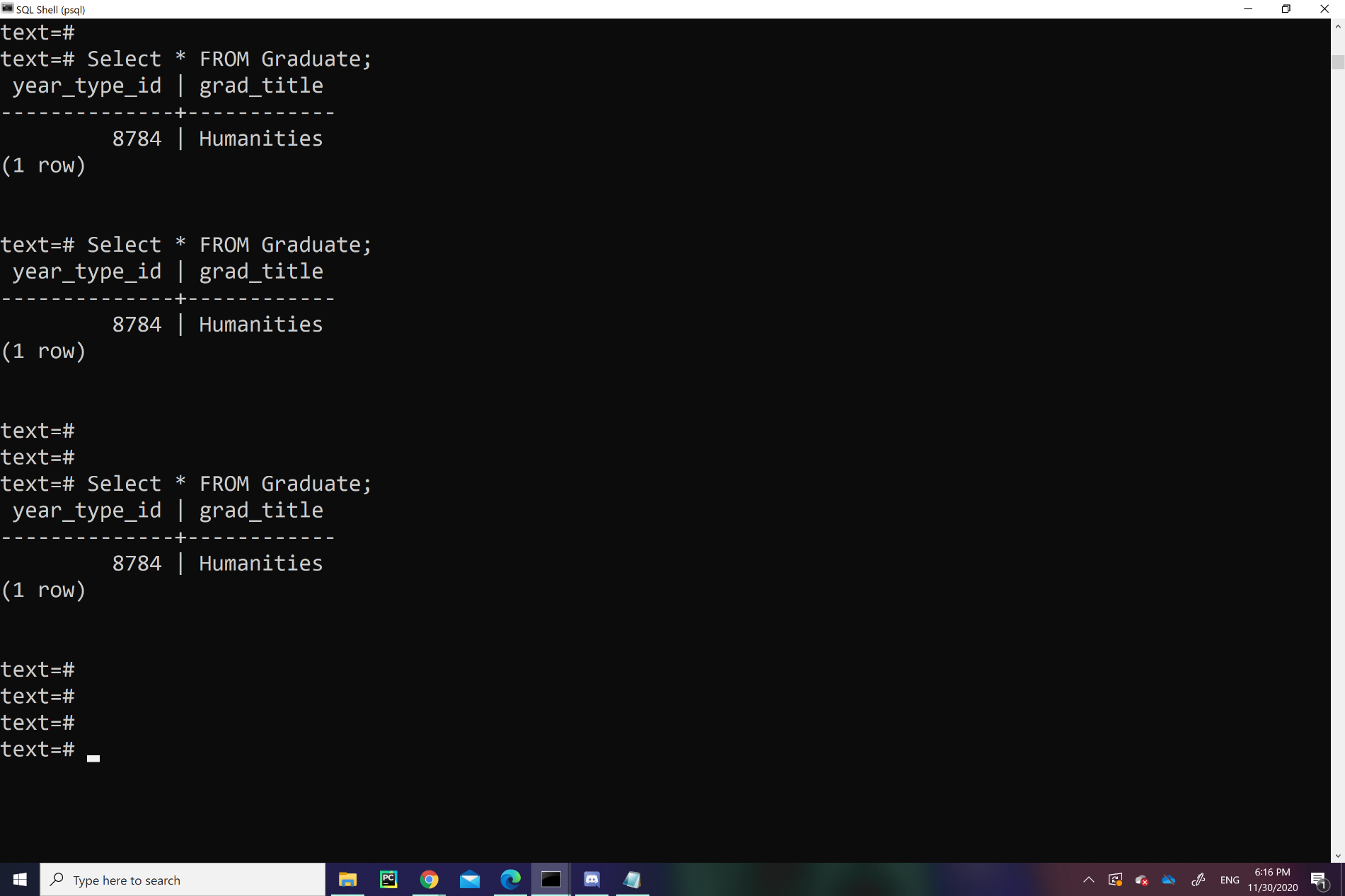
The code above creates an enumerated type called “Major” which plays as an identifier for Graduates (G) and Undergraduates (U). This enumerated type is used in the variable “Year\_Type\_Title,” which identifies a student being either a Graduate or an Undergraduate under the table “Student\_Type.” In the table, “Course,” it has a column, “Year\_Type,” that uses the enumerated type, “Major,” as an identifier of which Course is for Graduate students or Undergraduate Students.





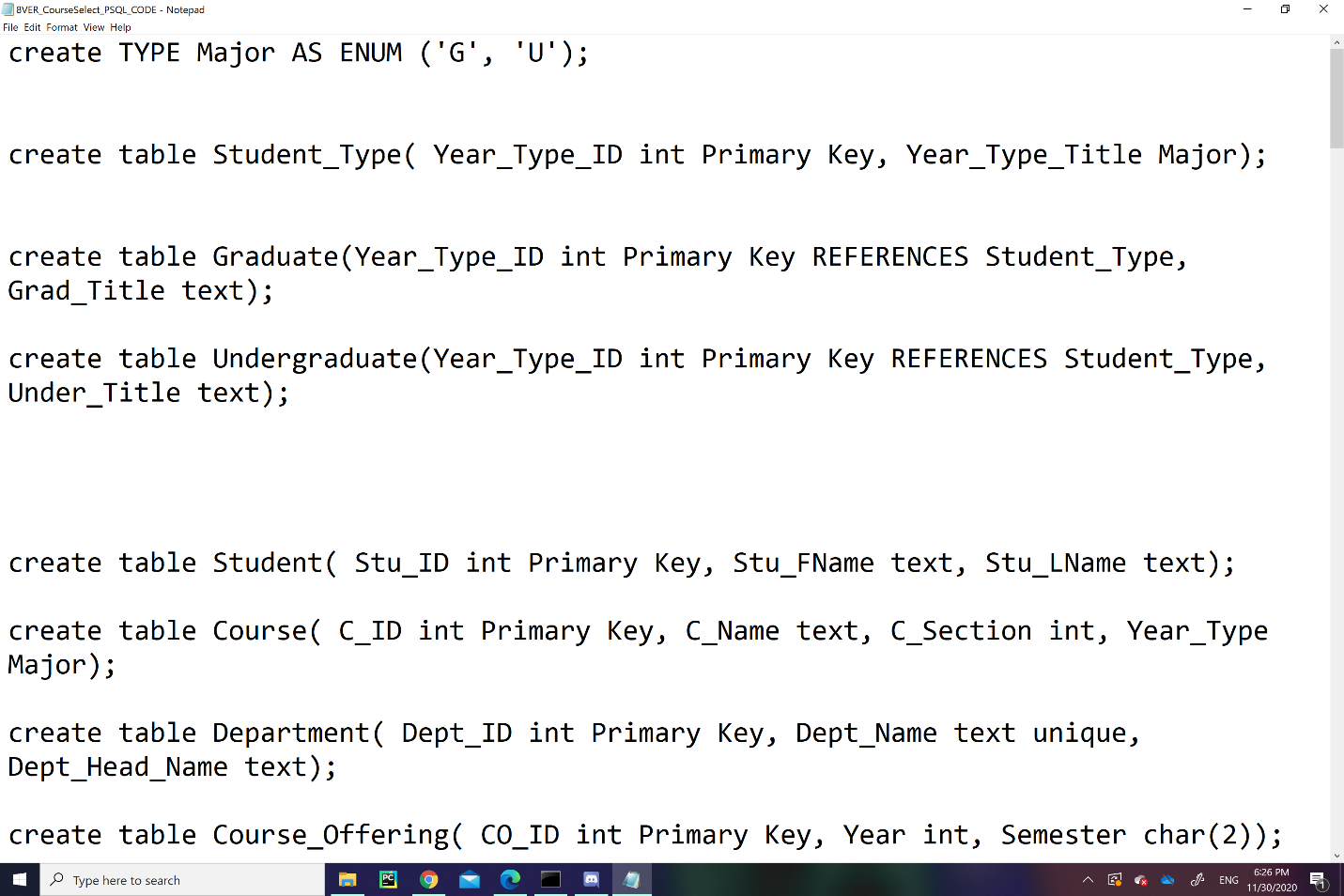
*3.2 Student\_Type Table*

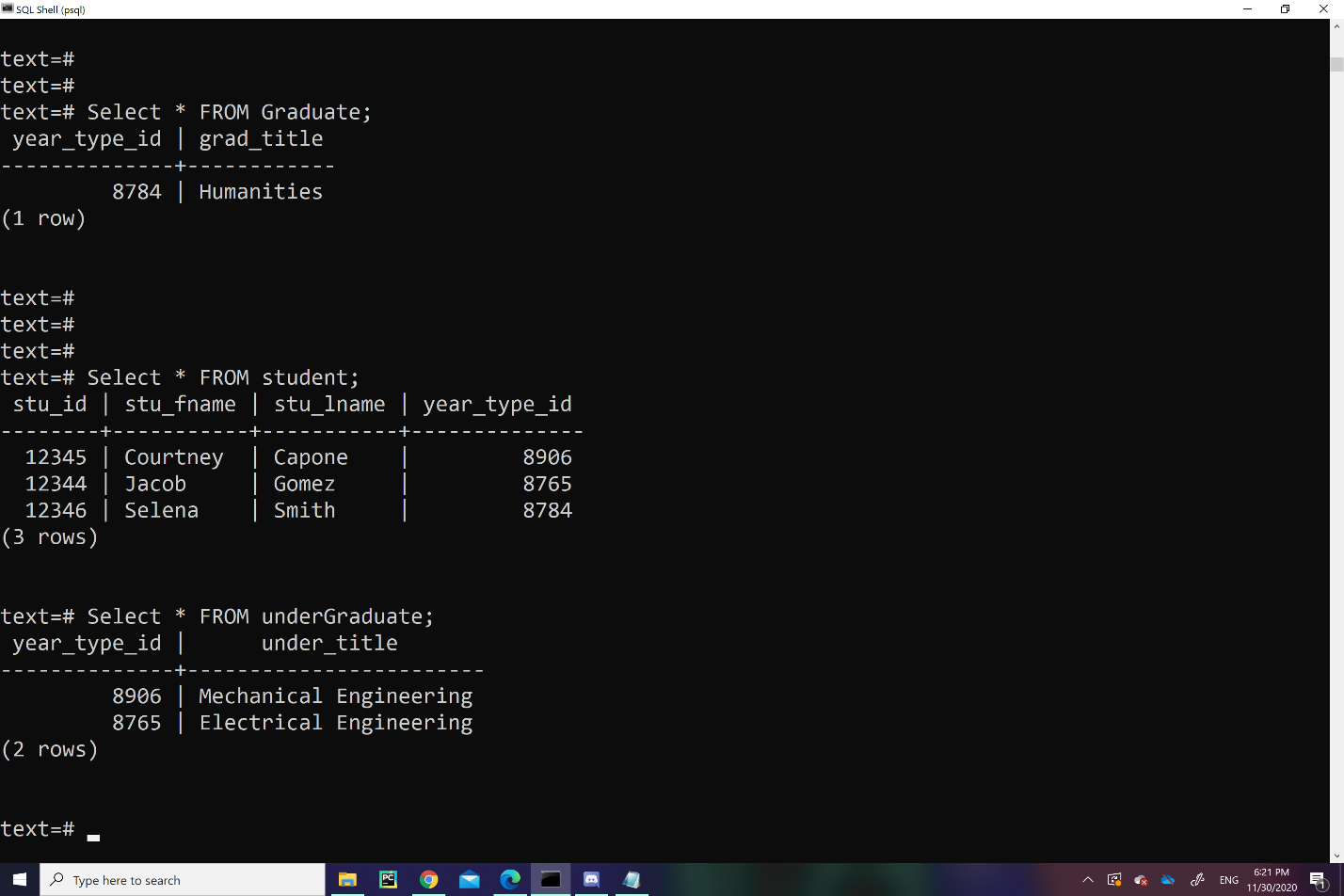
The table, “Student\_Type,” acts as subtype discriminator that subtypes students as either a Graduate or Undergraduate, which correlates to the tables shown below, “Graduate” and “Undergraduate,” that act as subtype tables. The “Student\_Type” table is a collection of rows that contain the information of a student’s “Year\_Type\_ID,” which acts as a primary key and ID number, and “Year\_Type\_Title", which is a “Major” enumerated type variable that identifies the student being a Graduate or an Undergraduate with character initials: “G” and “U.” 



*3.3 Graduate Table*

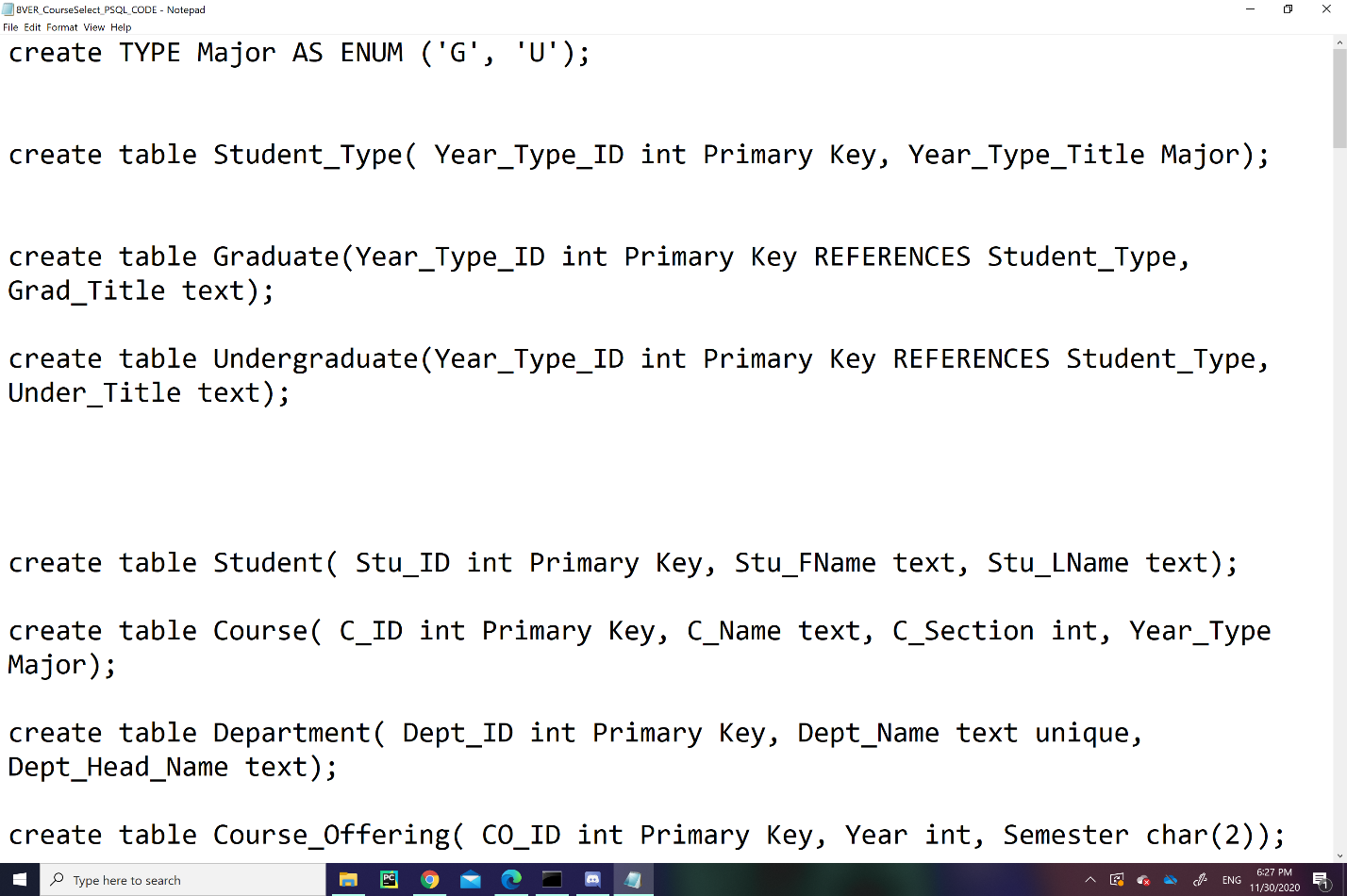
The “Graduate” table is a subtype for the table “Student\_Type,” which stores information regarding Graduate students’ major. It contains the Graduate’s “Year\_Type\_ID” and the string variable, “Grad\_Title,” which acts as the Student’s major.

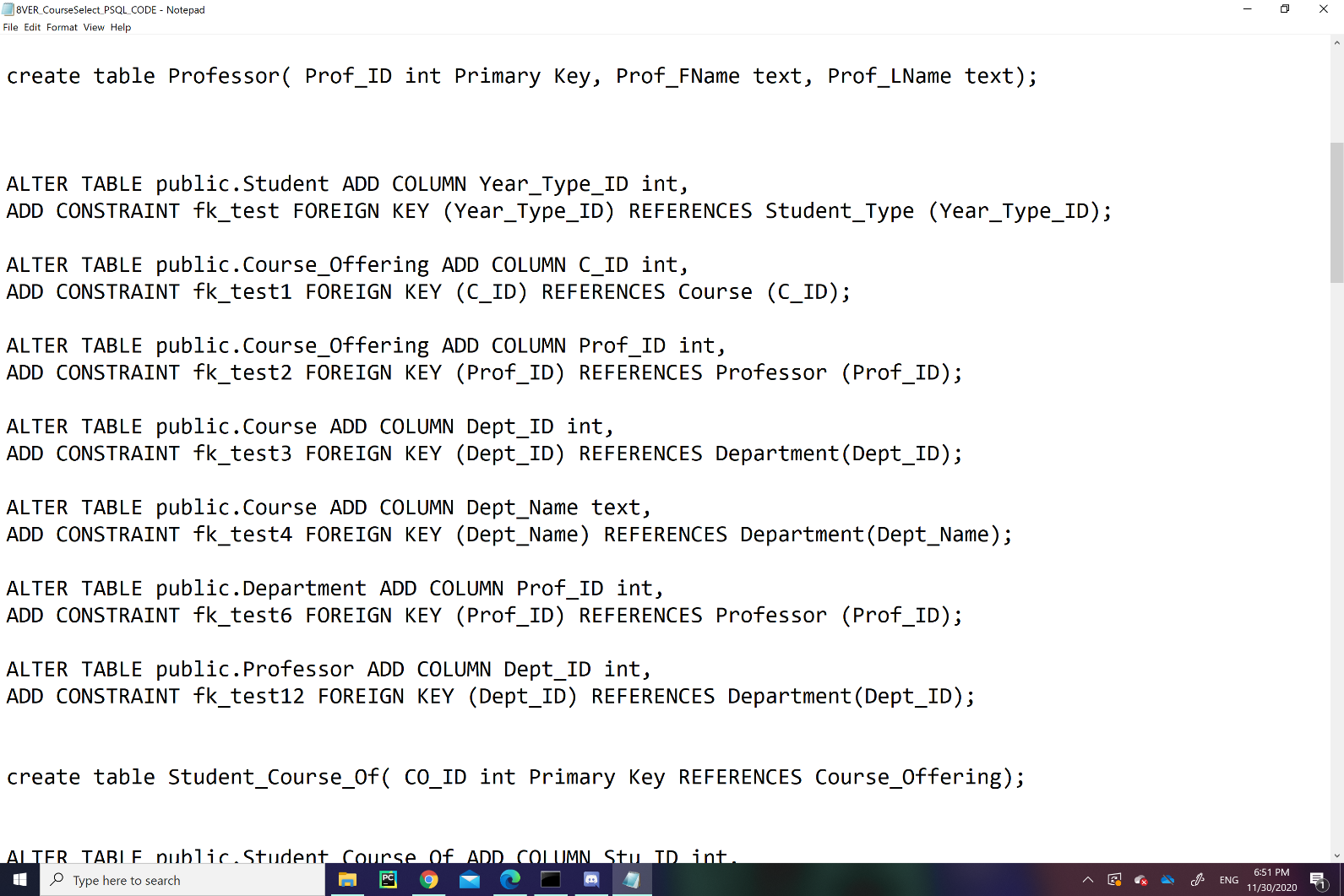


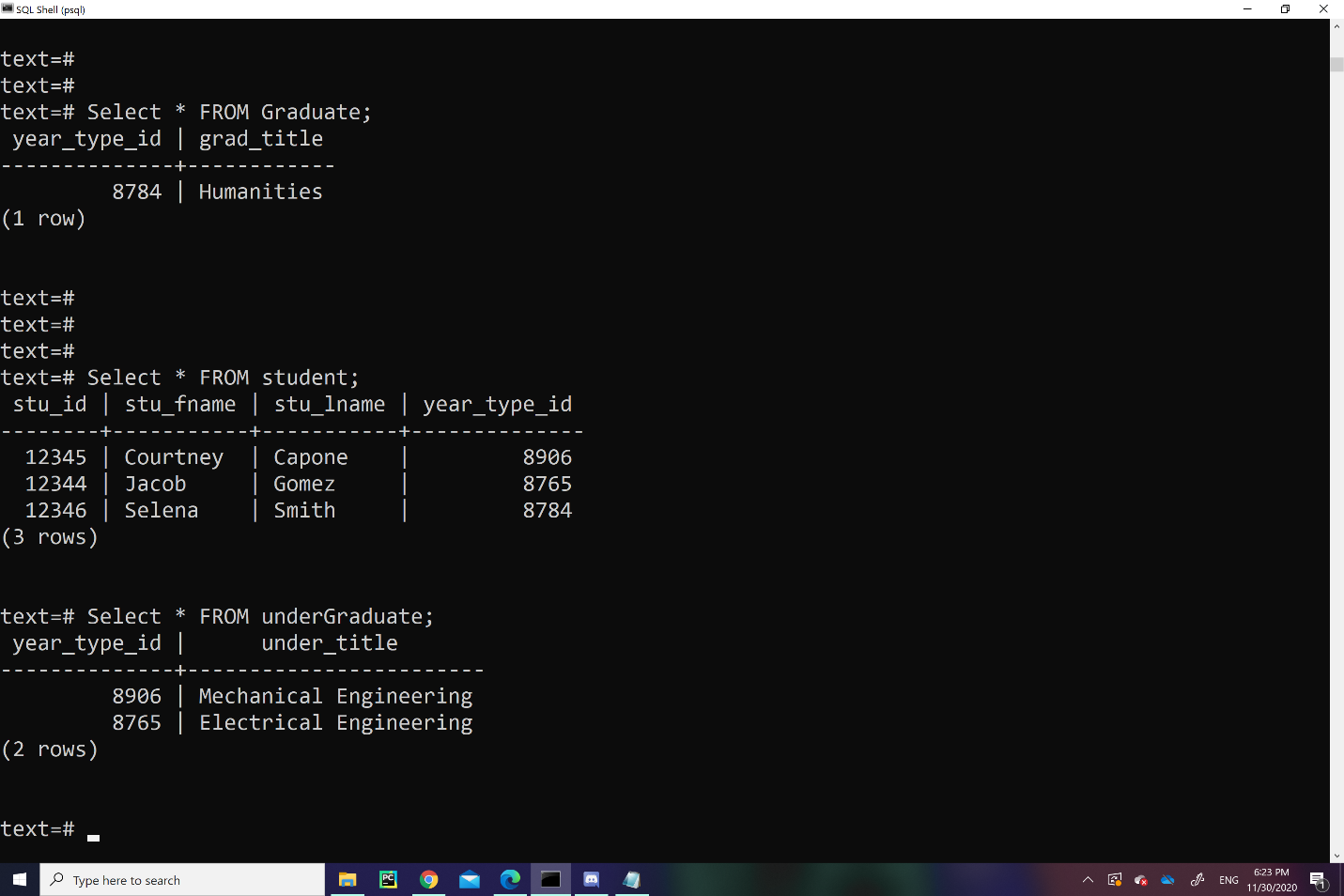


*3.4 Undergraduate Table*

The “Undergraduate” table is a subtype for the table “Student\_Type,” which stores information regarding Undergraduate students’ major. It contains the Undergraduate’s “Year\_Type\_ID” and the string variable, “Grad\_Title,” which acts as the Student’s major.

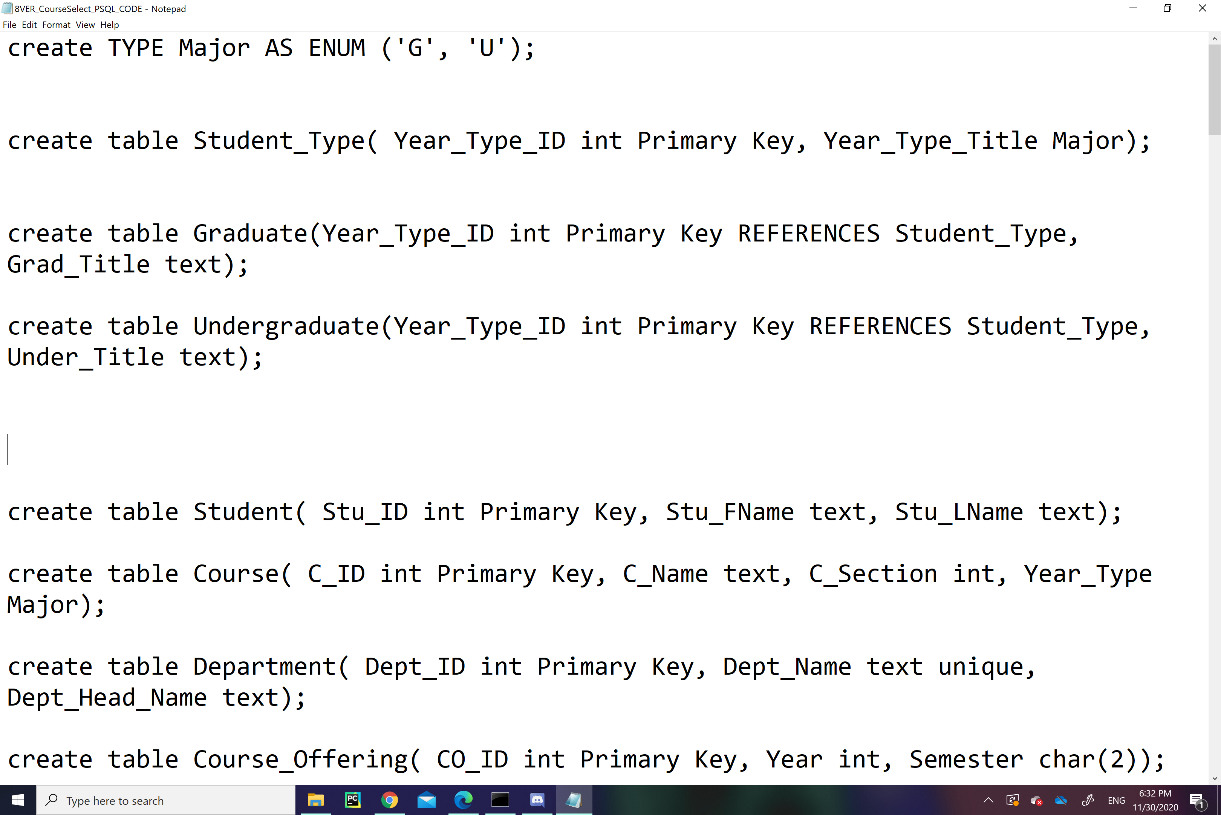


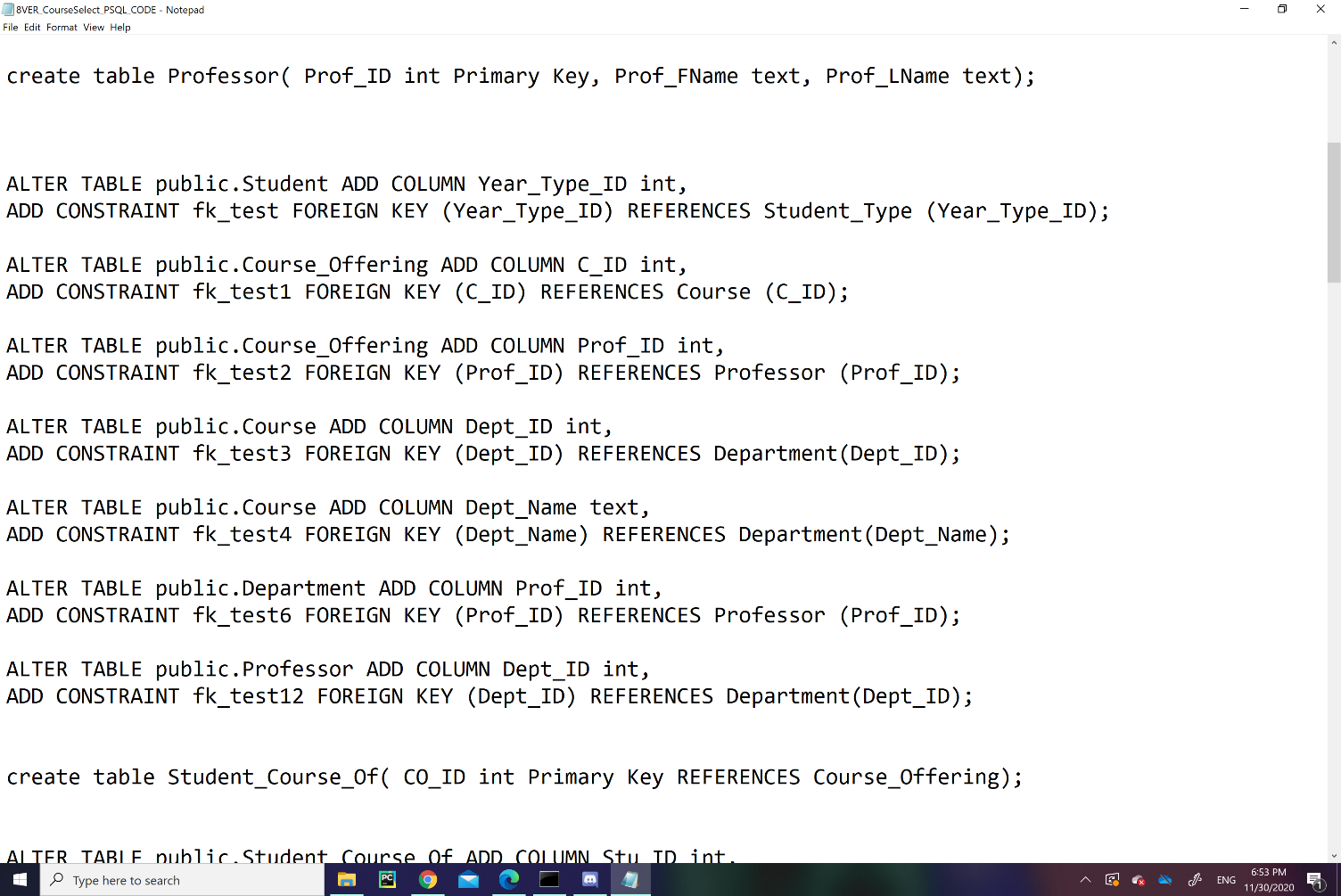


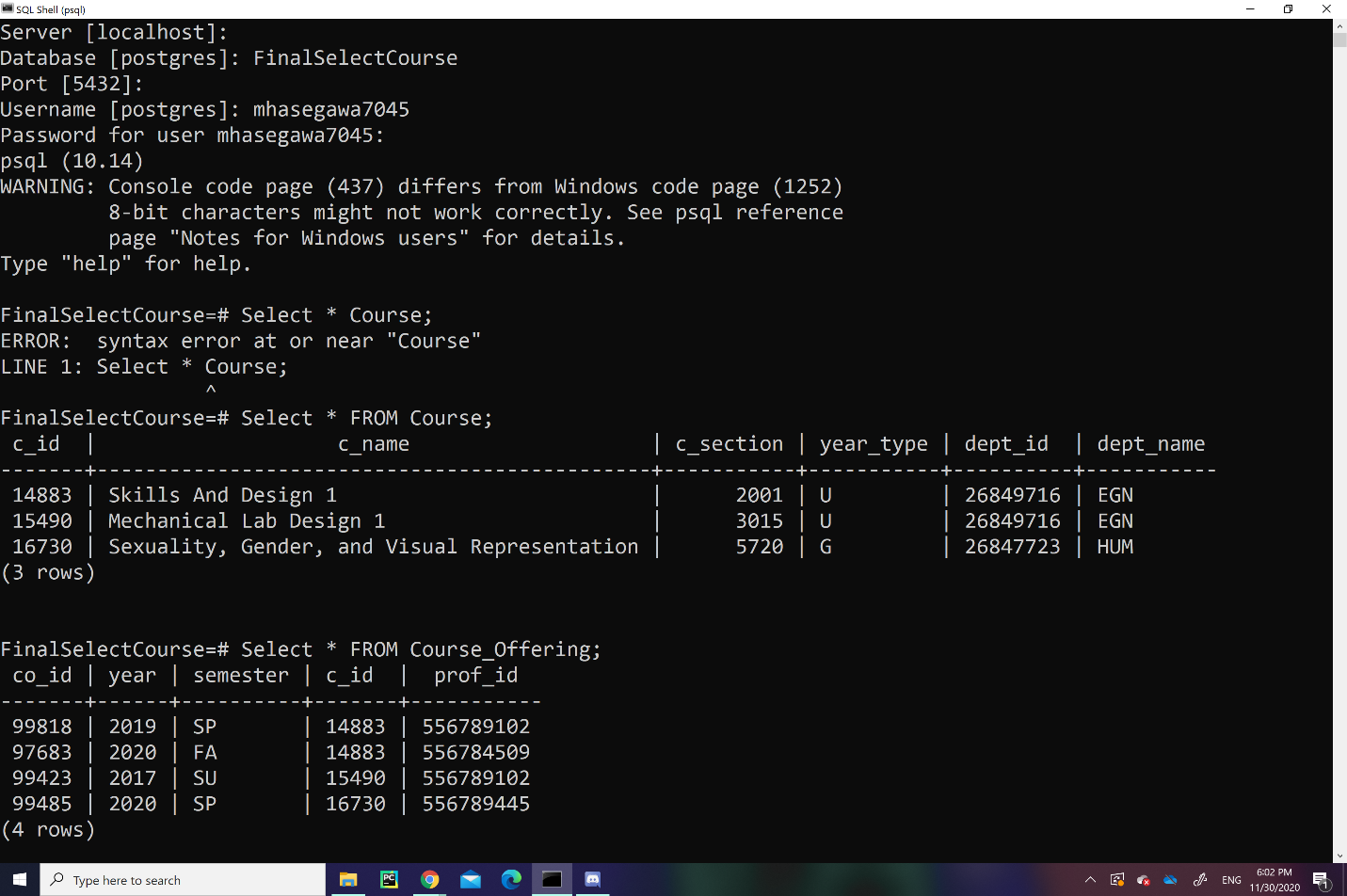


*3.5 Student Table*

The “Student” table stores data regarding student information. It contains the students “Stu\_Id” which is the primary key of the table and uniquely identifies each student with an ID number. “Stu\_Fname,” and “Stu\_Lname” are the string columns of the first and last name of the student. The variable, “Year\_Type\_Id” is a foreign key from the table “Year\_Type.”

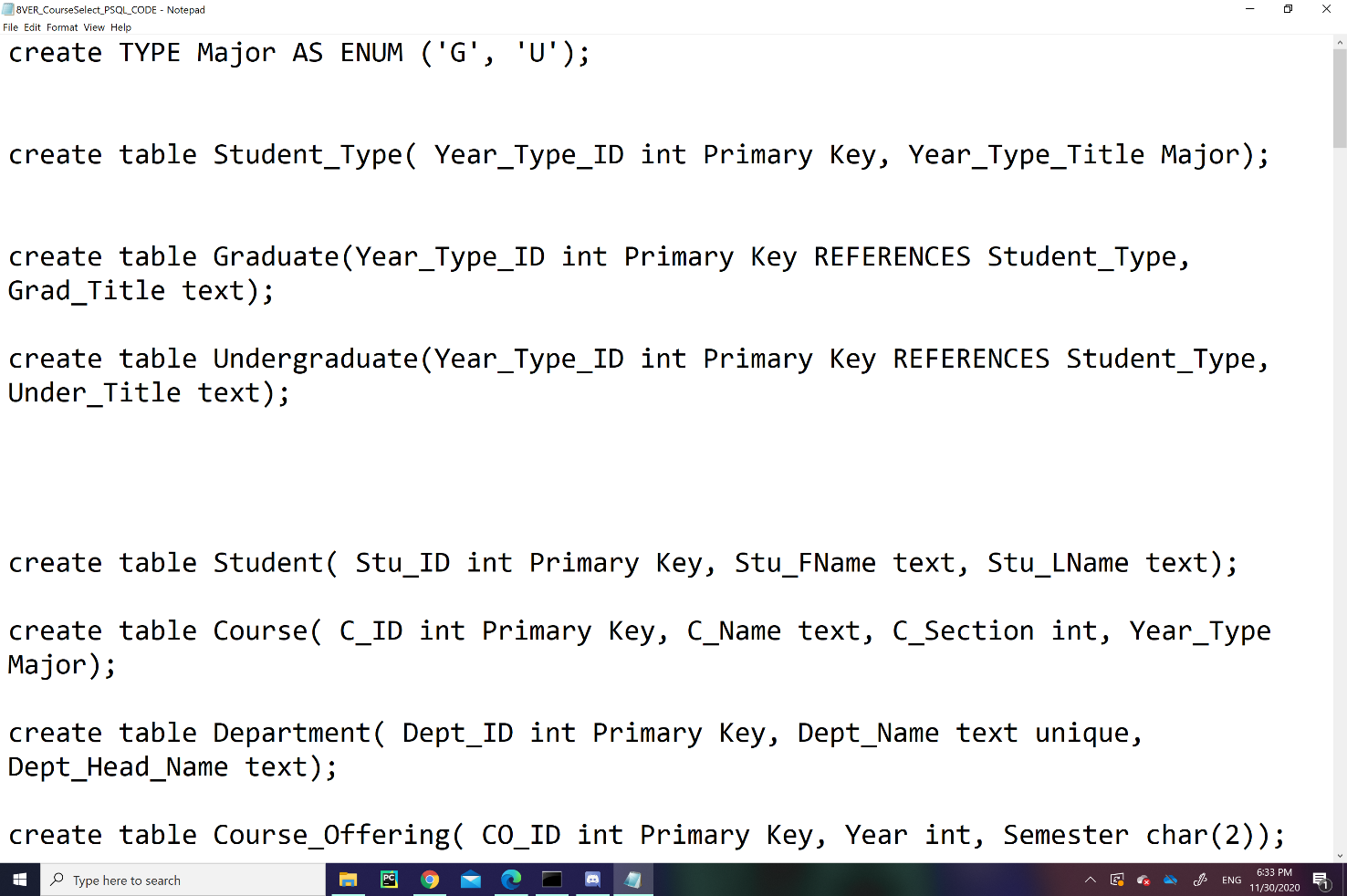


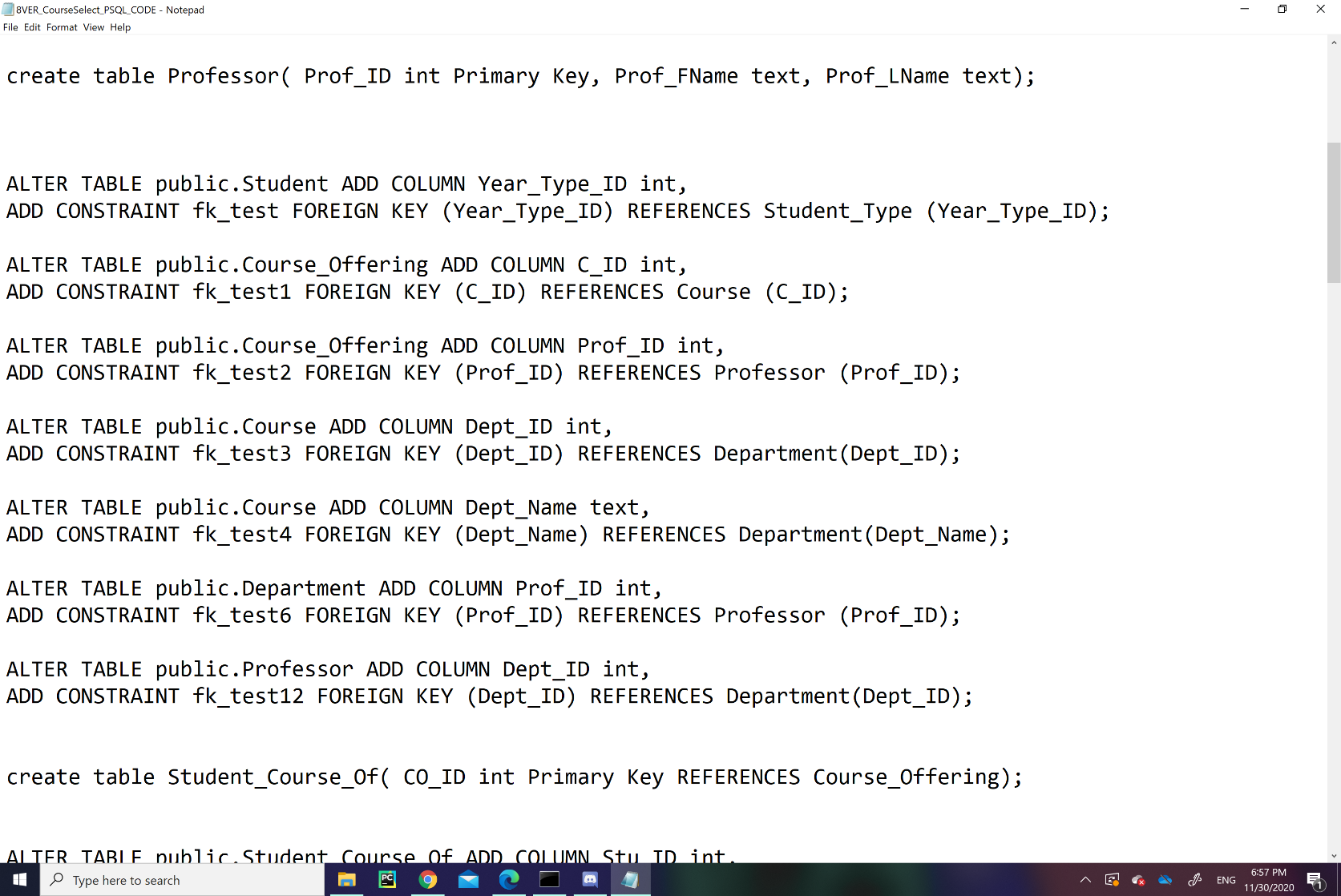


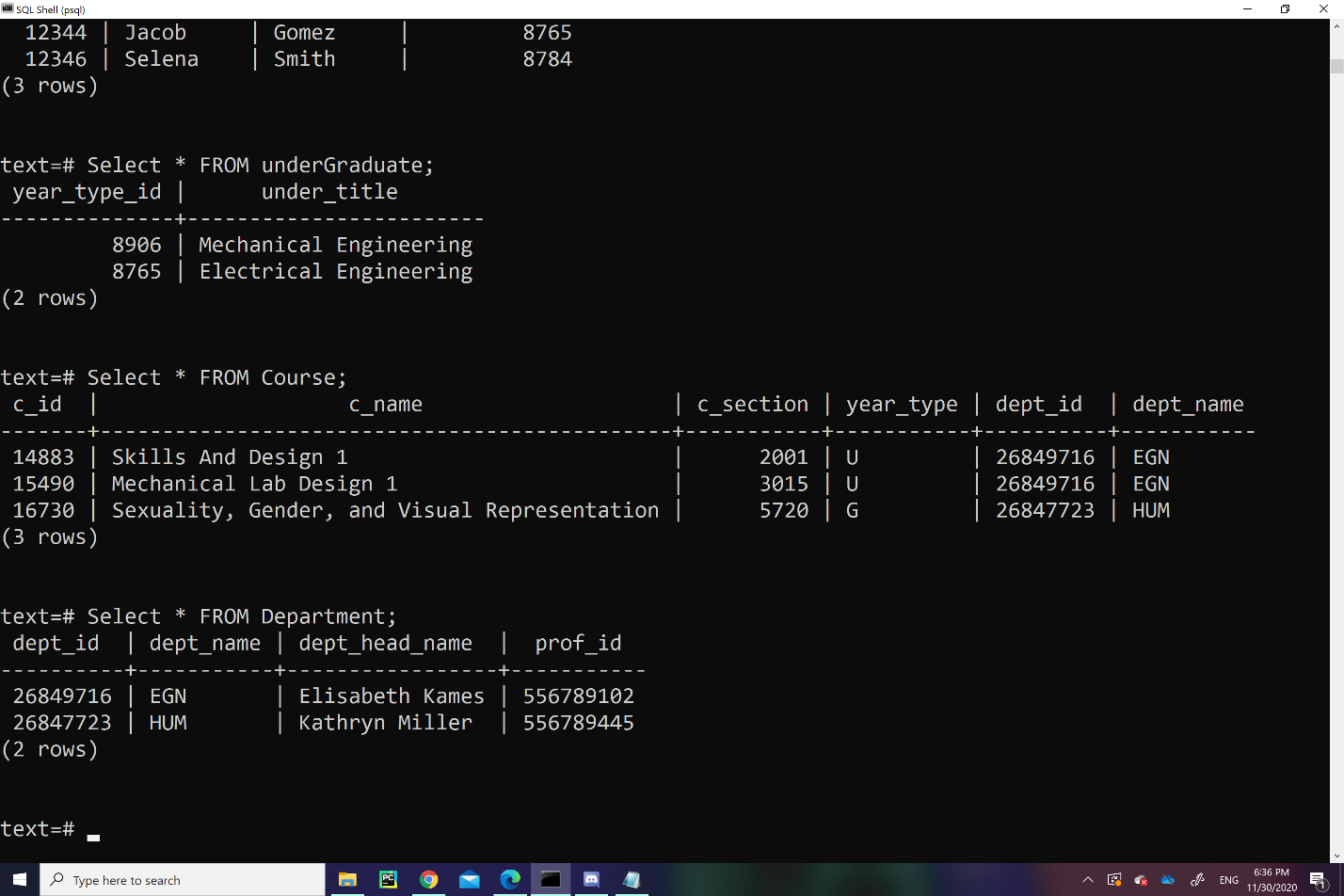


*3.6 Course Table*

The “Course Table” stores data regarding a Course’s information with 6 columns: “C\_ID”, “C\_Name”, “C\_Section”, “Year\_Type”, “Dept\_ID”, and “Dept\_Name”. The variables “C\_ID”, “C\_Name”, and “C\_Section” are the Course’s id number, name, and section number. The variable “Year\_Type” is an enumerated type, “Major”, which means that it identifies the “Courses” as either “Undergraduate” or “Graduate”. The “Dept\_ID” and the “Dept\_Name” are foreign keys from the table Department, which determines the Course’s “Department” ID and name.



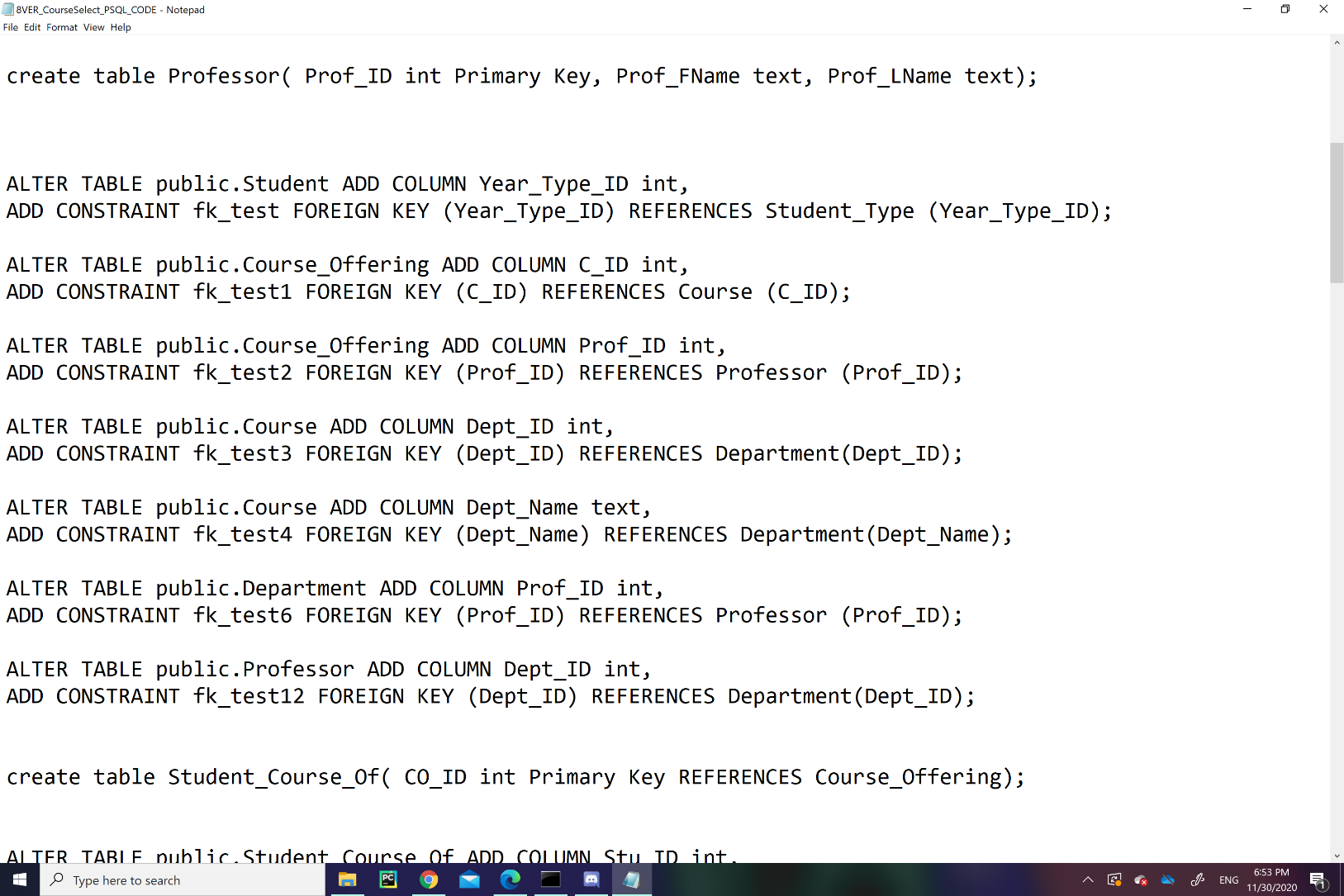


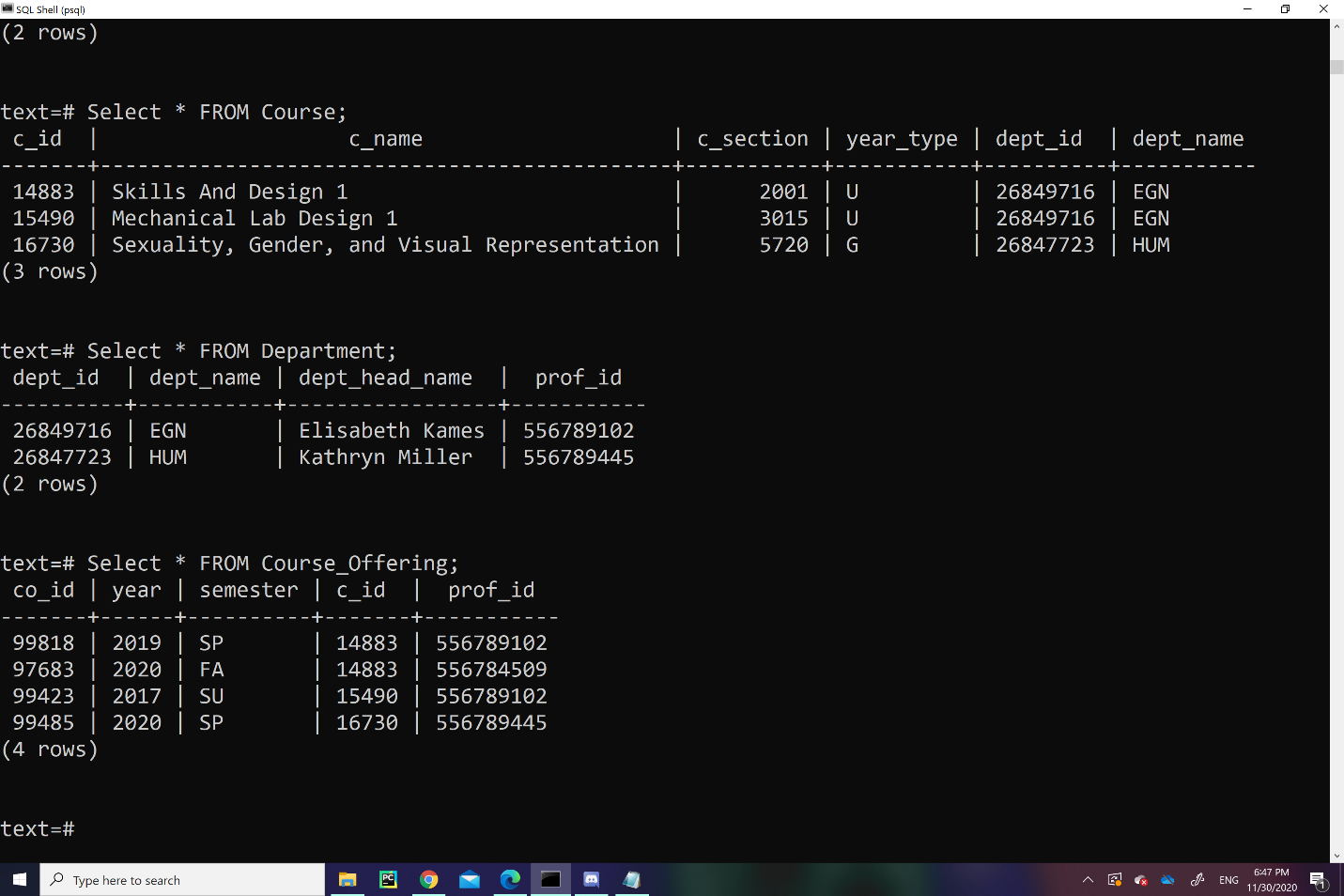


*3.7 Department Table*

The “Department” table contains information regarding departments. The “Dept\_ID” and “Dept\_Name” attributes are the ID number and name of the department. The “Dept\_Head\_Name” attribute has a string value of the Head-of-Department's full name. The “Prof\_ID” is a foreign key from table “Professor” which refers to the Head-of-Department's “Professor” ID.

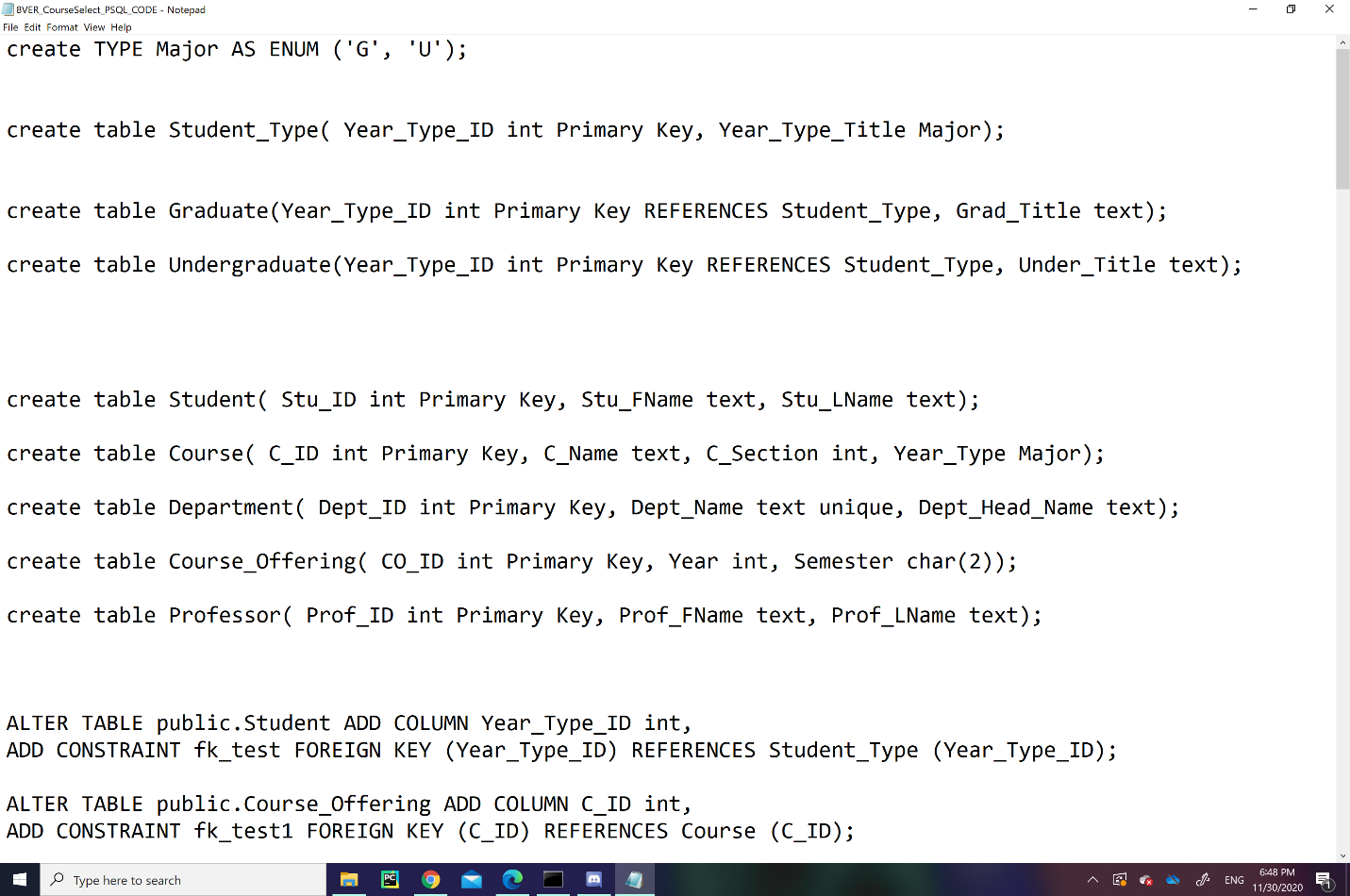


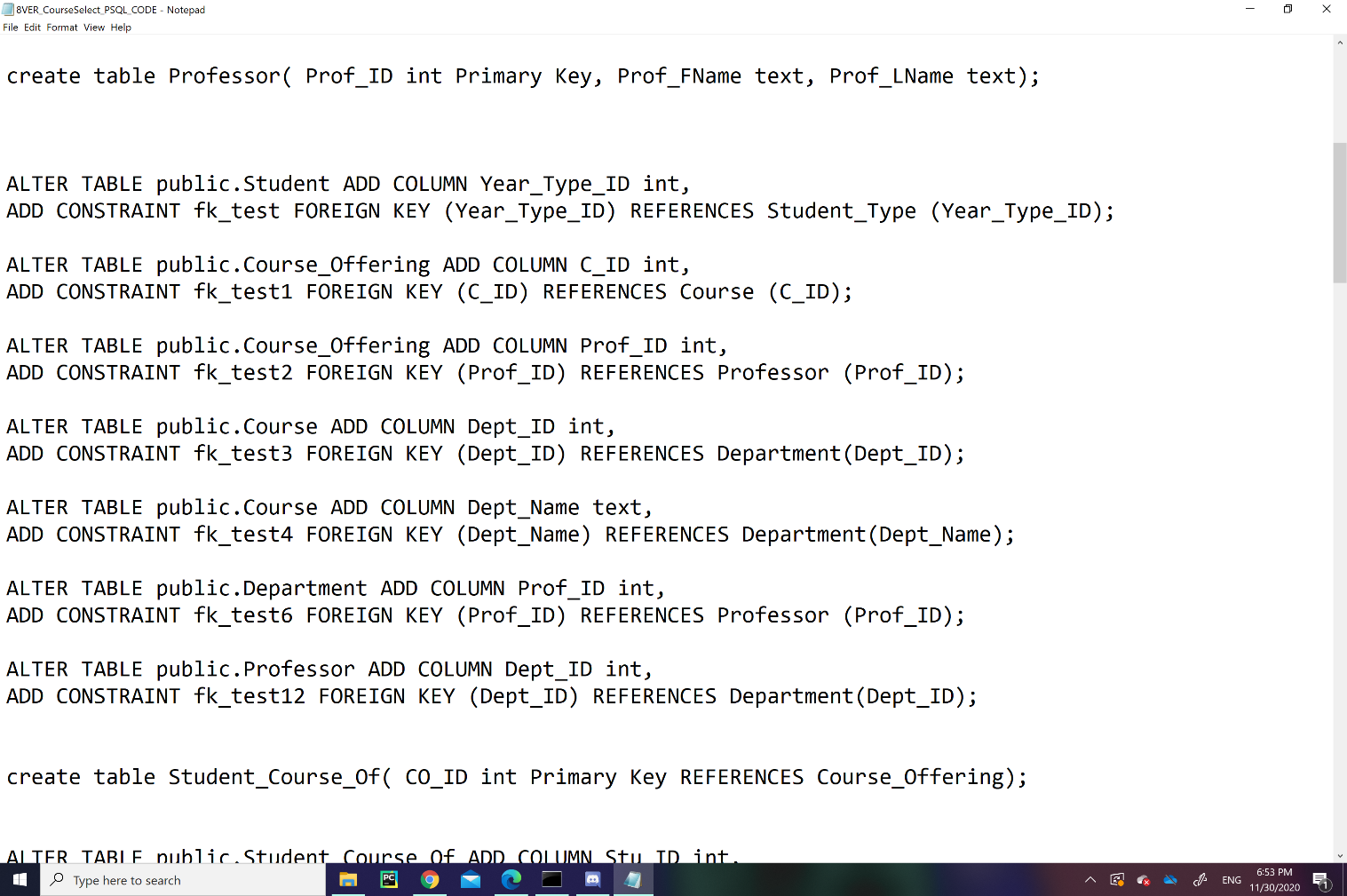


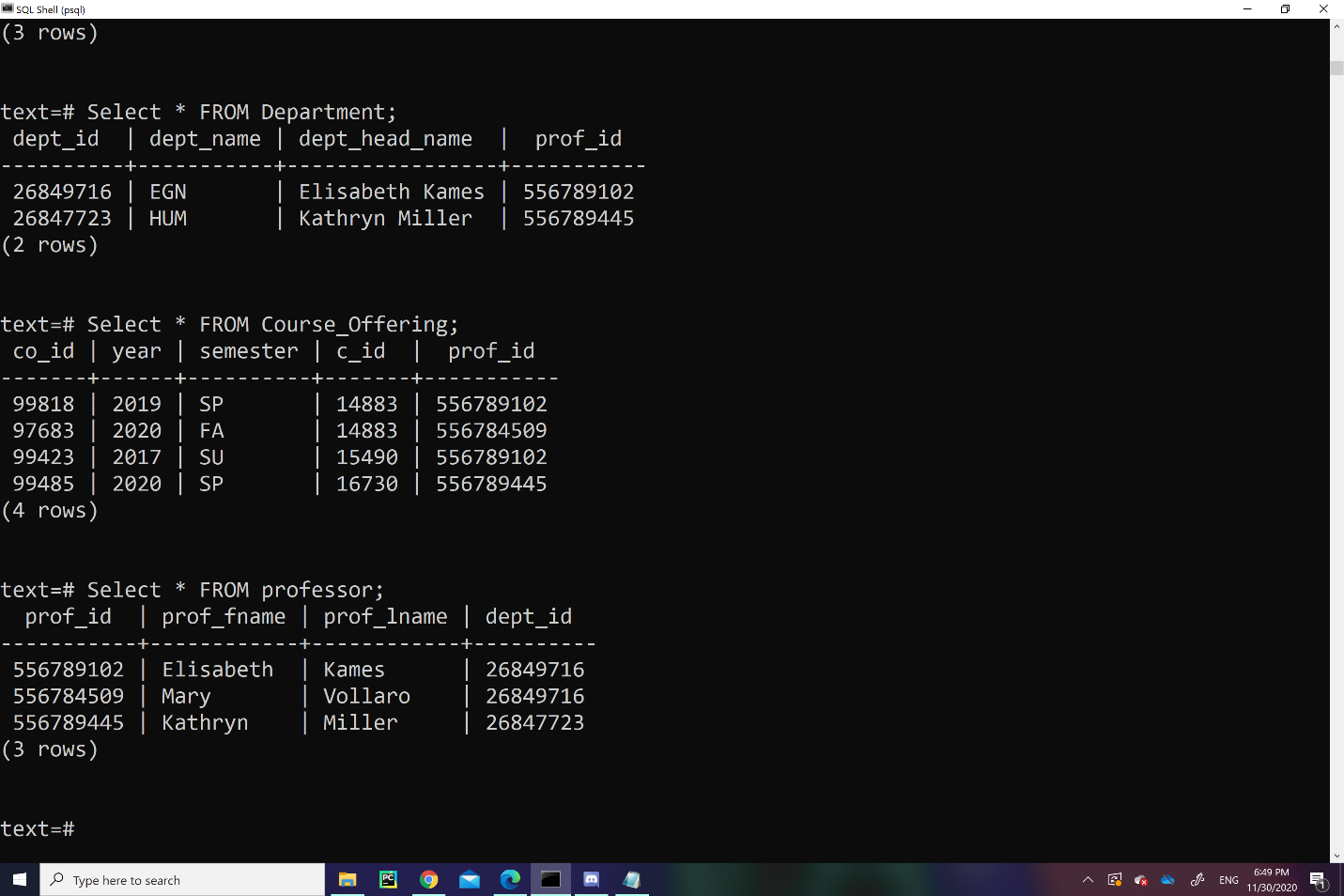


*3.8 Course\_Offering Table*

The table, “Course\_Offering Table”, is a personal collection of information regarding a “Course” offered to a particular Student for future semesters in the Course Registry. The attributes, “Co\_ID”, “Year”, and “Semester” is an ID number, year, and semester of the “Course” offer. The variables, “C\_ID” and “Prof\_ID”, are foreign keys from the tables “Course” and “Professor.”

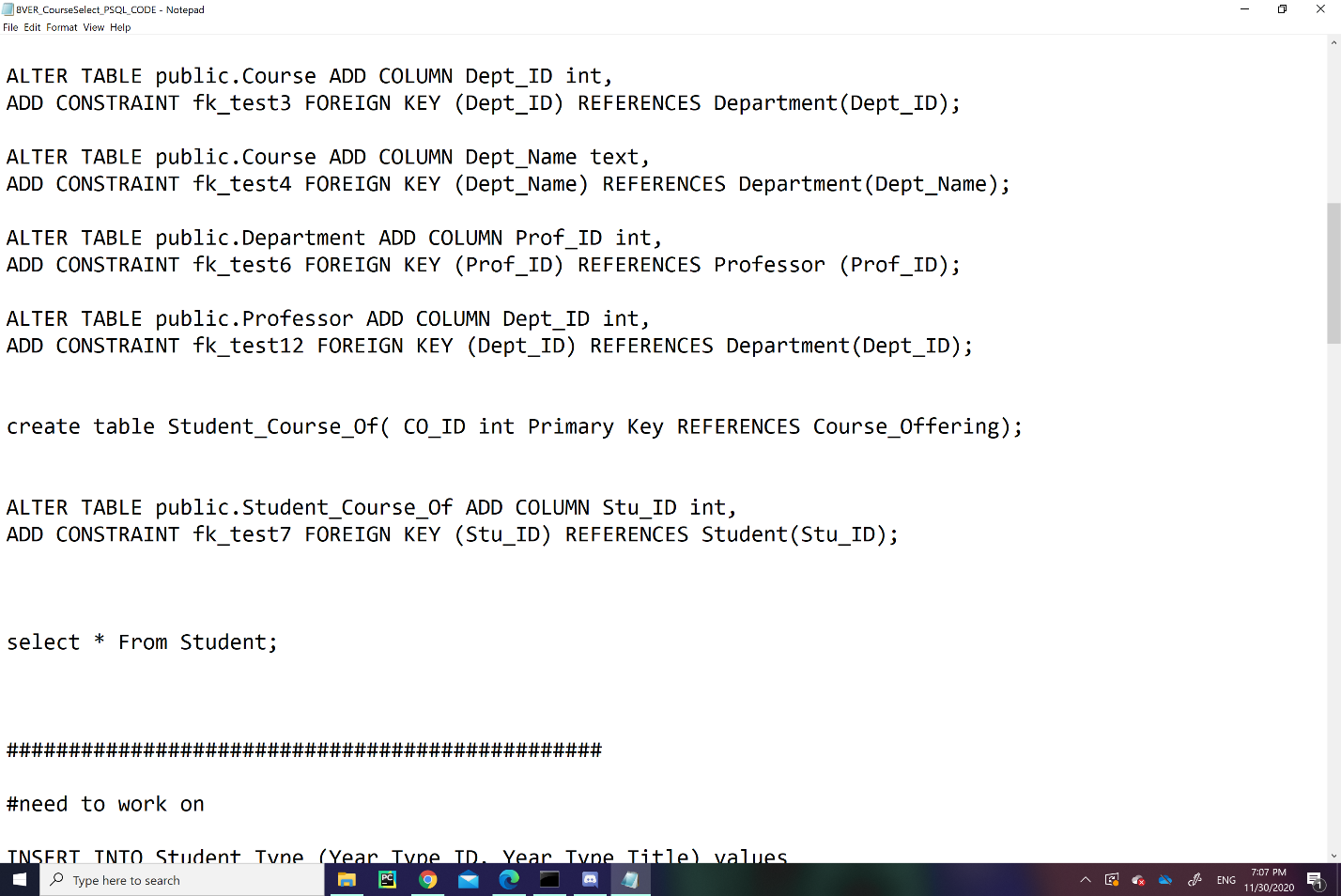


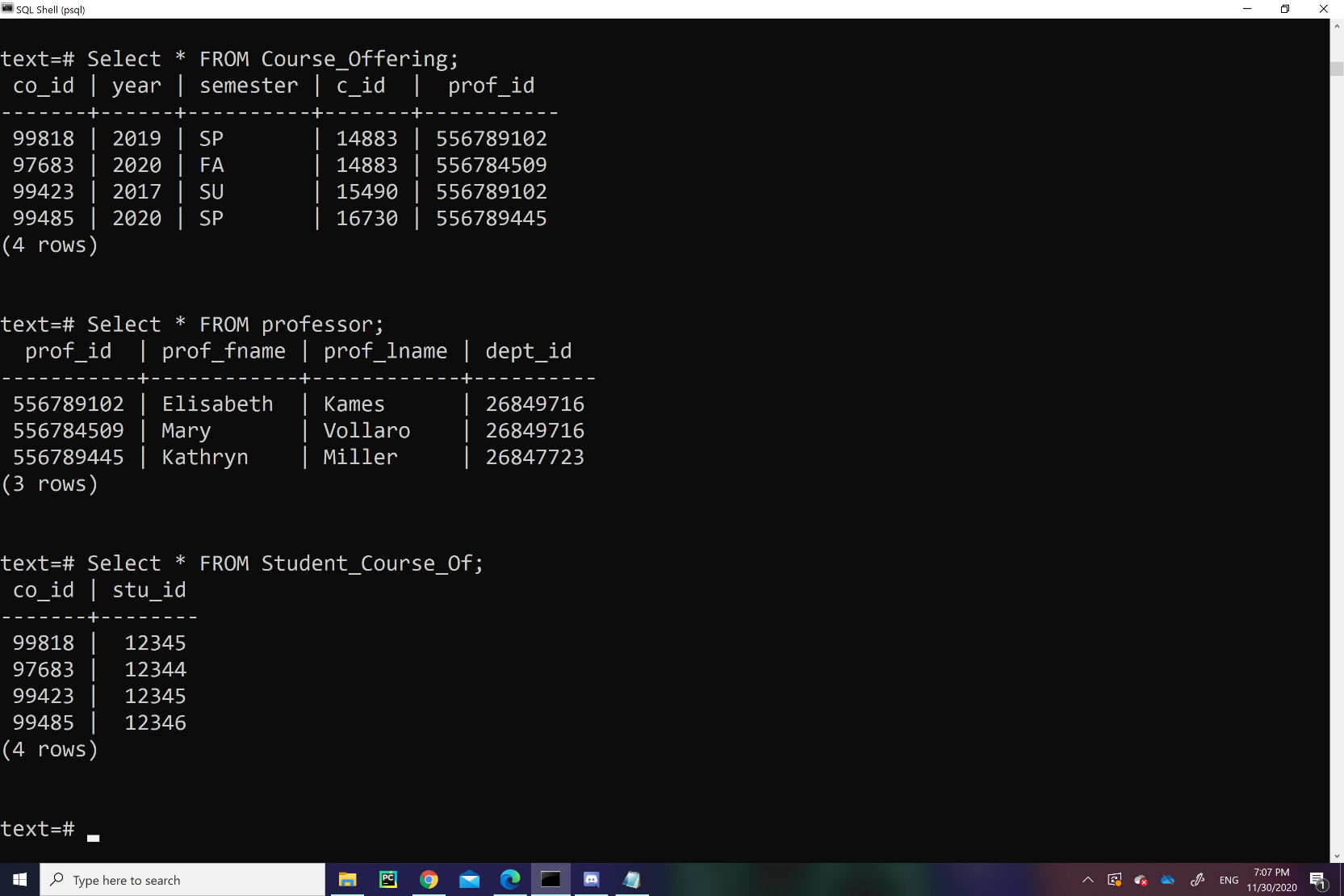




*3.9 Professor Table*

The table, “Professor”, has a collection of rows that hold academic and personal information of the “Professors”. The attributes, “Prof\_ID”, “Prof\_FName”, and “Prof\_LName” indicates the Professor’s ID number, first name, and last name. The column, “Dept\_ID”, is a foreign key from the table, “Department”, which acts as the Professor’s “Department” ID number.



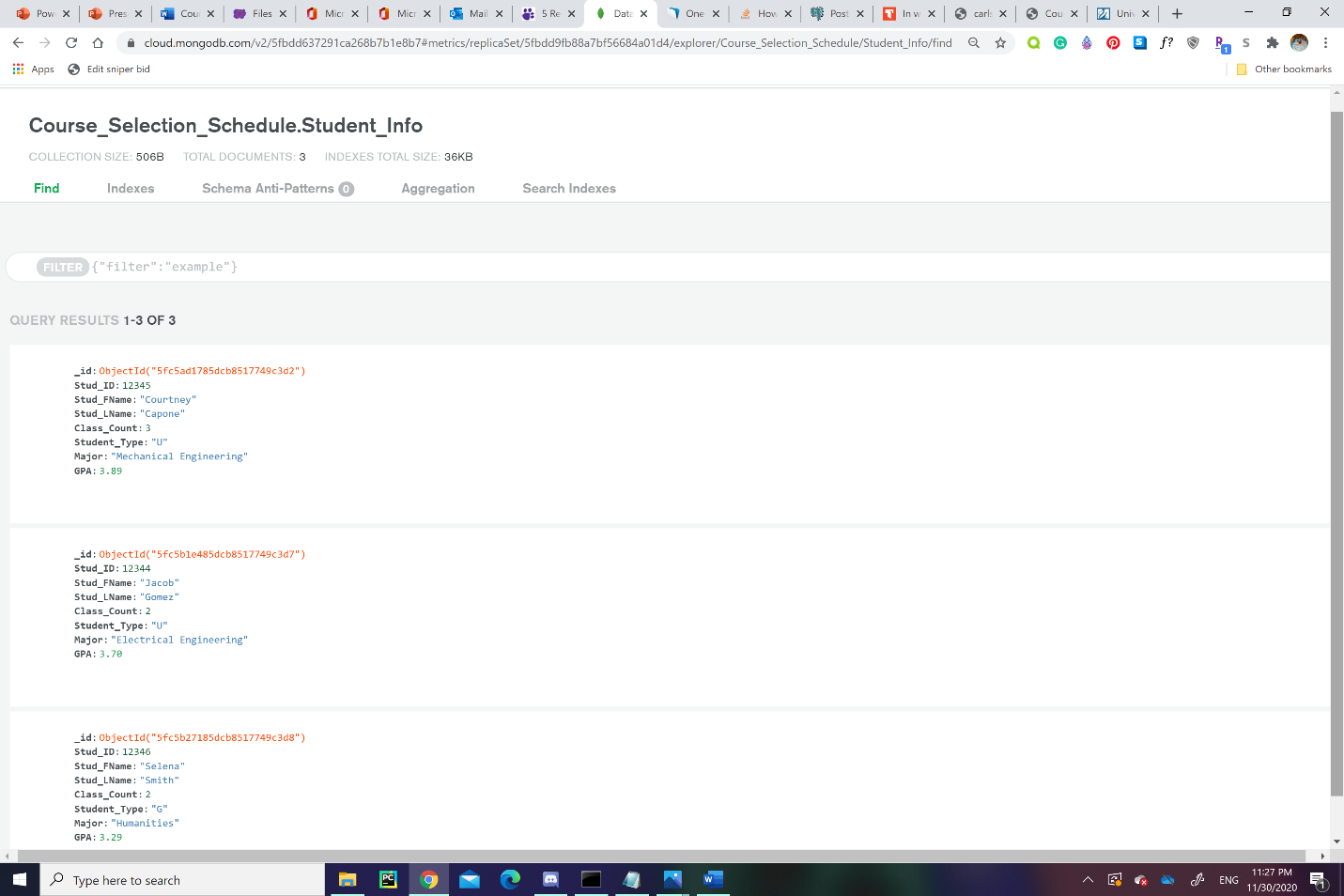


*3.10 Student\_Course\_Of Table*

The table, “Student\_Course\_Of”, act as a mediator between “Student” and “Course\_Offering,” so that the variables “CO\_ID” and “Stu\_ID” can be contained in the same table. The variable “CO\_ID” act as a primary key, but it is also a foreign key from “Course\_Offering”, so the tables “Course\_Offering” and “Student\_Course\_Of” has a strong relationship.

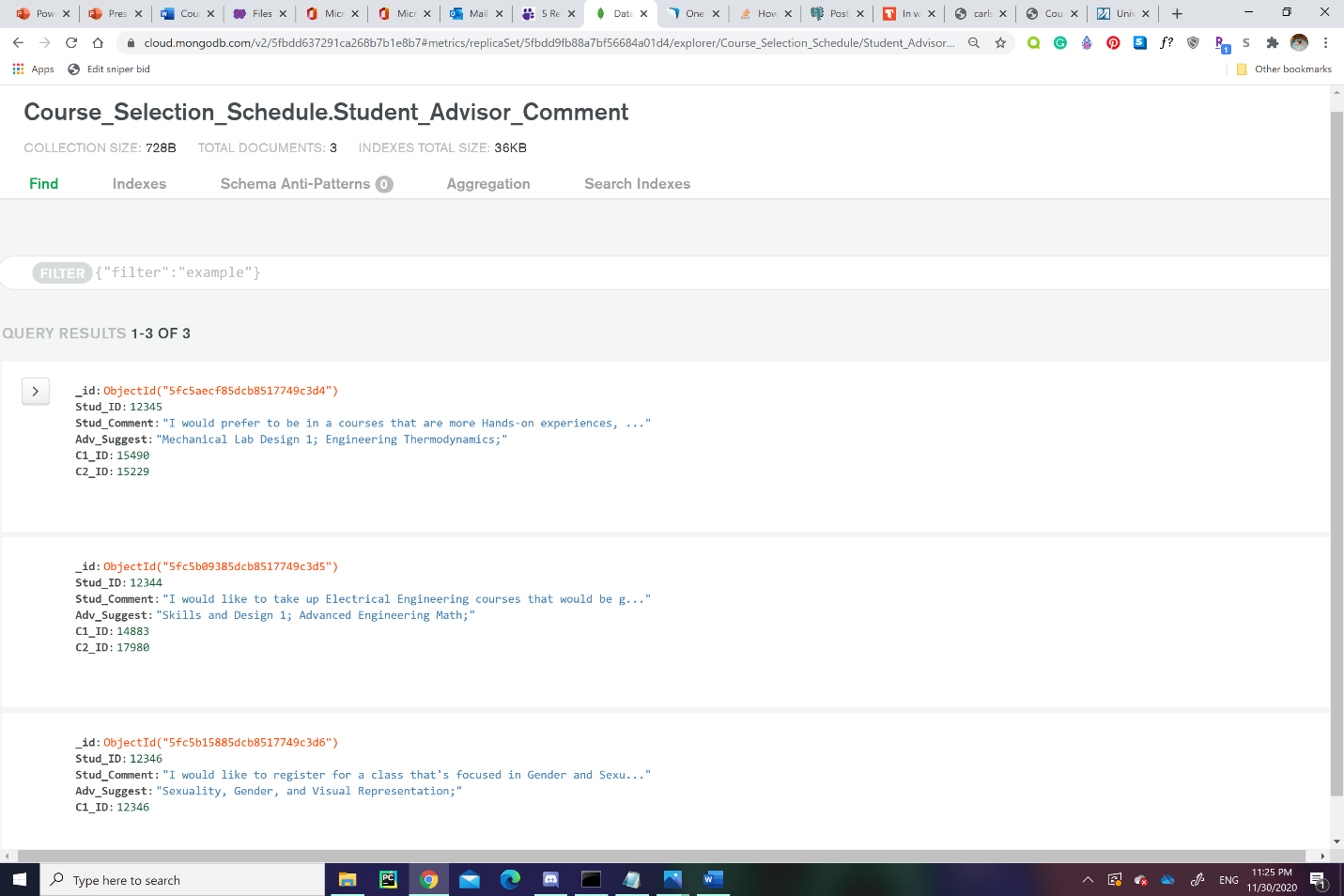
1. **MongoDB Database**

MongoDB is a nonrelational, document storing database that we used to store the information that we had relating to the individual aspects of our database. Our MongoDB Database was made to store documents and collections regarding the course registration with aspects of pending course registrations, pending course schedules, and recorded conversations between students and advisors concerning which courses would suit the students’ need. This type of database can be used for the pending schedule webpage in a Course registration website and create a private online chat room between students and advisors where it can collect their conversations regarding course selection. This is especially more useful as a MongoDB database, since indexing conversations through a relational database would be impractical since all conversations would be unique.



*4.1 Student\_Info Collection*

The “Student\_Info” collection has documents regarding a Student’s academic information to be used for Course Registration. The documents have 8 attributes: “\_id”, “Stud\_ID”, “Stud\_FName”, “Stud\_LName”, “Class\_Count”, “Student\_Type”, “Major”, and “GPA”. The attribute, “\_id,” acts as the primary key or id key since every document contains one in MongoDB. The “Stud\_ID” attribute is the id number of a student, while “Stu\_FName” and “Stu\_LName” are the first and last names of a student. The attribute, “Class\_Count,” act as an integer variable that counts the number of a student’s pending and completely registered courses. The attribute, “Major,” act as a string variable that identifies a student’s major, while the “GPA” attribute is a decimal variable that clarifies a student’s GPA.



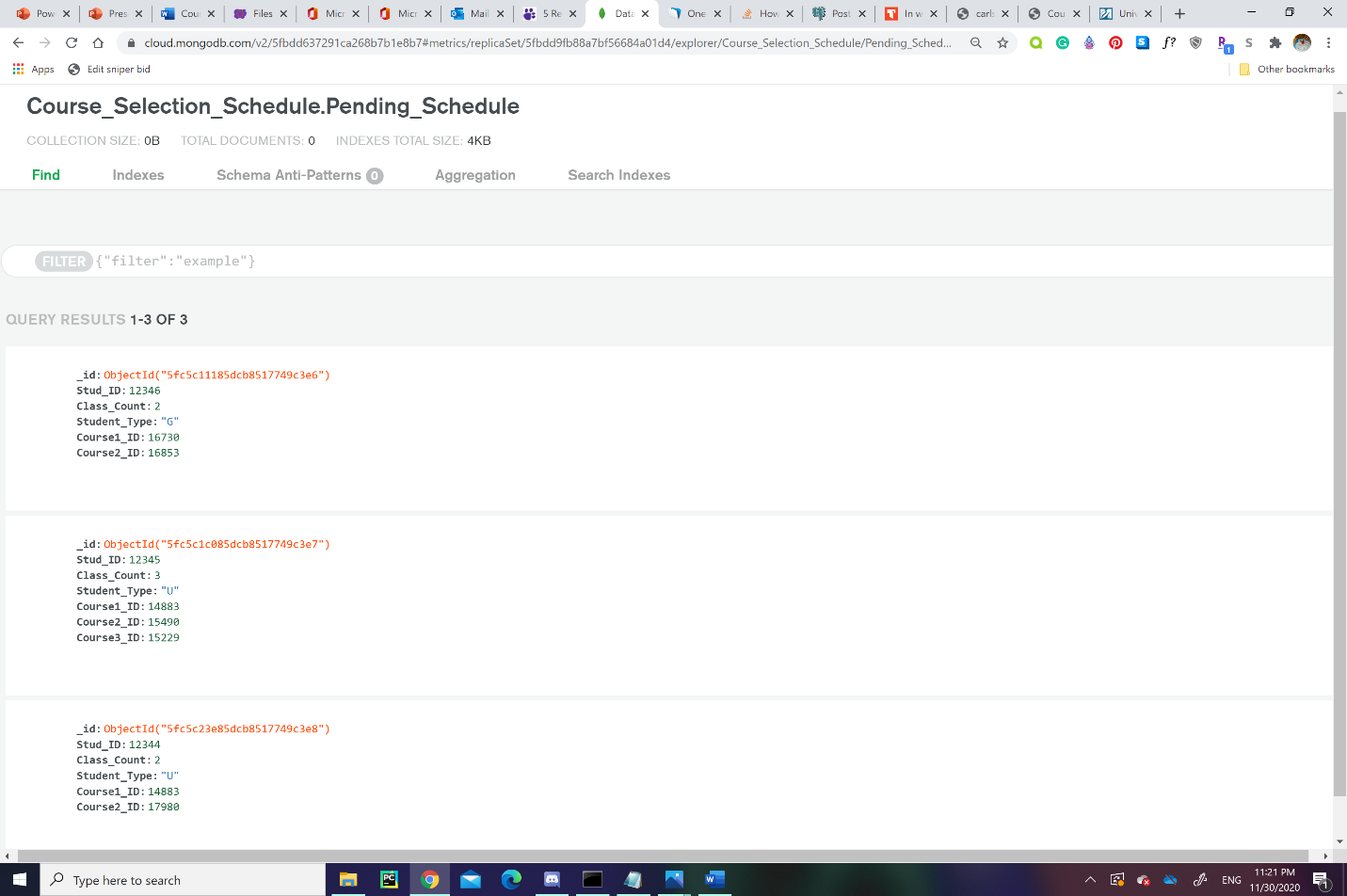
*4.2 Student\_Advisor\_Comment Collection*

The “Student\_Advisor\_Comment” Collection is a table of conversations between students and advisors. A document from this table has 5 attributes that contain details about the conversation: “\_id”, “Stud\_ID”, “Adv\_Suggest”, and “C#\_ID”. The “Stud\_Comment” attribute is the string variable that contains the student’s statement of what preference they want for the course, while the string attribute, “Adv\_Suggest", acts as the Advisor’s course suggestion to the student’s preferences. The attribute group, “C#\_ID,” is a collection of integer variables that identify suggested courses’ ID, and the amount of “C#\_ID” is dependent on the number of suggestions under the variable, “Adv\_Suggest.”



*4.3 Pending\_Completed Collection*

The “Pending\_Completed” Collection is a table of pending and completely registered courses of specific students. The variables, “Course\_ID” and “Course\_Name,” act as a course id and name of a registered course, while the “Student\_ID” attribute act as the id number of the student who is registering for that specific course. The attribute, “Course\_Status,” acts as the status of the registering course as either “PENDING” or “COMPLETED.”



*4.4 Pending\_Schedule Collection*

The “Pending\_Status” Collection has documents that act as a student’s undetermined schedule of pending and completely registered courses for future semesters. The collection has 6 attributes: “\_id”, “Stud\_ID”, “Student\_Type”, and “Course#\_ID”. The attributes, “Stu\_ID” and “Student\_Type,” refers to the student’s id number and degree level (Undergraduate or Graduate). The “Class\_Count” variable counts the number of pending and completely registered courses the student has applied for. The variable group, “Course#\_ID,” is a list of variables that list all the id numbers of pending and completely registered courses.

1. **Conclusion**

The project has successfully implemented a Course Selection PostgreSQL database and a Scheduling Advice MongoDB database which could be used for a Course Registry online application for students attending higher-level education.

The PostgreSQL database organizes academic data about “Student”, “Course”, “Department”, and “Professor” information in order to help implement a theoretical Course Registry Application. It would do this by organizing the course registration data of a student in a relational manner.

The MongoDB database is document-driven that indexes collections and documents that focus on pending course registrations, pending course schedules, and recorded conversations between advisors and students regarding which courses suit the student’s preferences.

1. **References**

# Vats, R (2020) “15 Exciting SQL Project Ideas & Topics for Beginners” UpGrad Blog

Retrieved from: <https://www.upgrad.com/blog/sql-project-ideas-topics-for-beginners/>

99 Unique Database Project Ideas for Final Year Students. (2020, September 29). Retrieved from <https://instanteduhelp.com/unique-database-project-ideas/>

Domínguez, C., & Jaime, A. (2010, June 16). Database design learning: A project-based approach organized through a course management system. Retrieved from <https://www.sciencedirect.com/science/article/abs/pii/S0360131510001600>