PES UNIVERSITY
ELECTRONIC CITY CAMPUS

(Established under Karnataka Act No. 16 of 2013)
100 Feet Ring Road, BSK III Stage, Bengaluru-560 085
Department of Computer Science and Engineering
Session: Aug – Dec 2021
SEMESTER – 5

Database Management System(UE19CS301)

Assignment - 4

Team Details

Team Number-5

	NAME	SRN
1	Prachi Sengar	PES2UG19CS285
2	Raeesa Tanseen	PES2UG19CS310
3	Ria Singh	PES2UG19CS326

Problem Statement

Our aim is to use database management software and build a relational database for efficiently organizing, storing, managing, and using the data kept by a placement cell with details about students, colleges, and companies in order to make the placement process hassle-free.

Language of Choice

We have decided to use PostgreSQL for the database and for the frontend we have employed python. Using the psycopg2 library we have connected the database to the frontend because it is a postgreSQL adapter for python programming language. It allows us to connect the capabilities of the Python language and libraries to obtain, manipulate, input, and update data stored in a PostgreSQL database. Using tkinter we made the frontend UI of our Student Placement Application as it is available for cross platform, has lighter framework and is easy to use.

Dependencies for DB connectivity

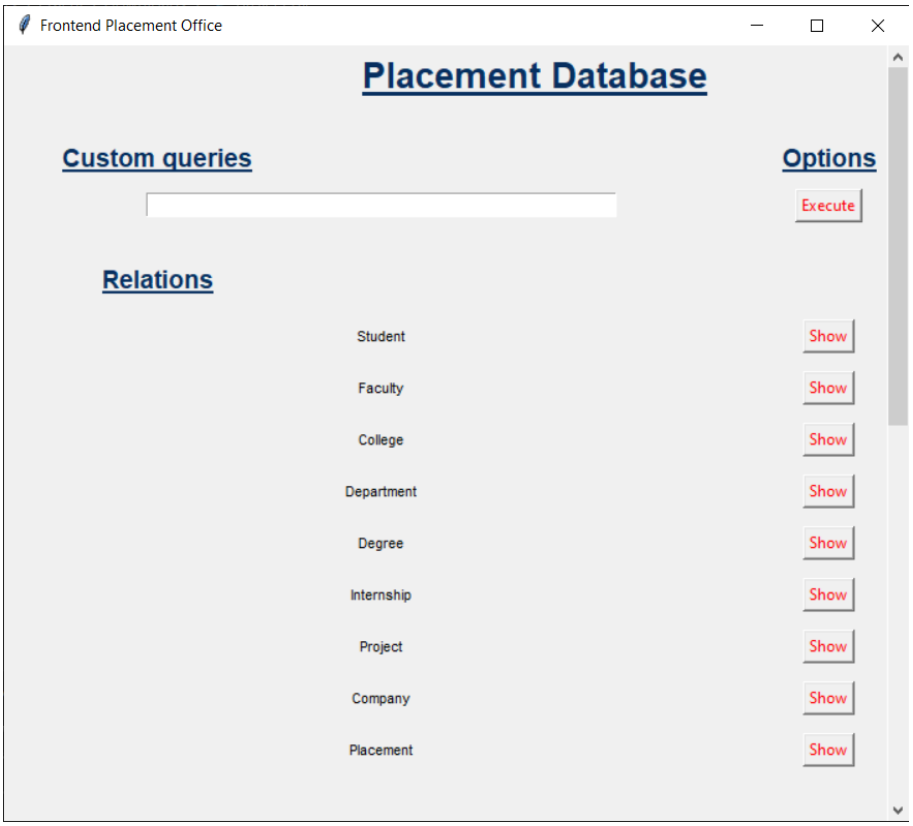
We have employed python language for both frontend and backend purposes to be connected to our Student Placement database.

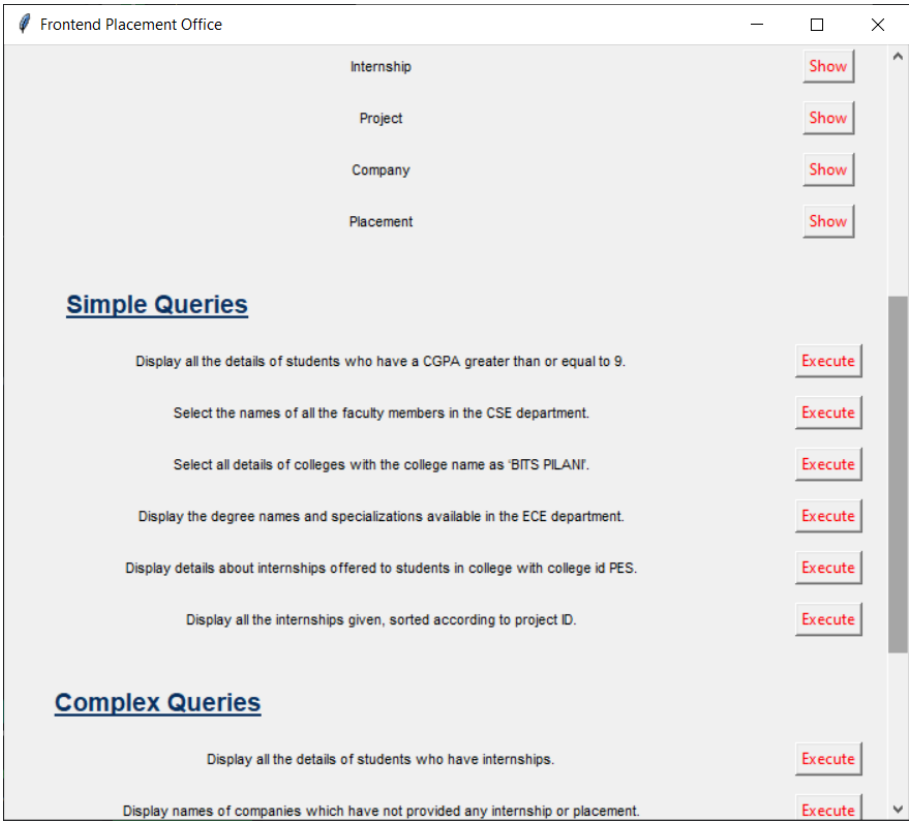
Using psycopg2 library we connected to the database. The most widely used PostgreSQL database adapter for Python programming language is Psycopg. It was created for heavy multi-threaded programmes that construct and delete a lot of cursors and perform a lot of "INSERT"s or "UPDATE"s at the same time. Psycopg 2 is mostly written in C as a libpq wrapper, which makes it both efficient and safe. Client-side and server-side cursors are supported, as well as asynchronous communication and notifications and "COPY TO/COPY FROM" functionality. Many Python types are out-of-the-box supported and adapted to PostgreSQL data types; adaptation may be expanded and altered owing to a flexible objects adaptation framework.

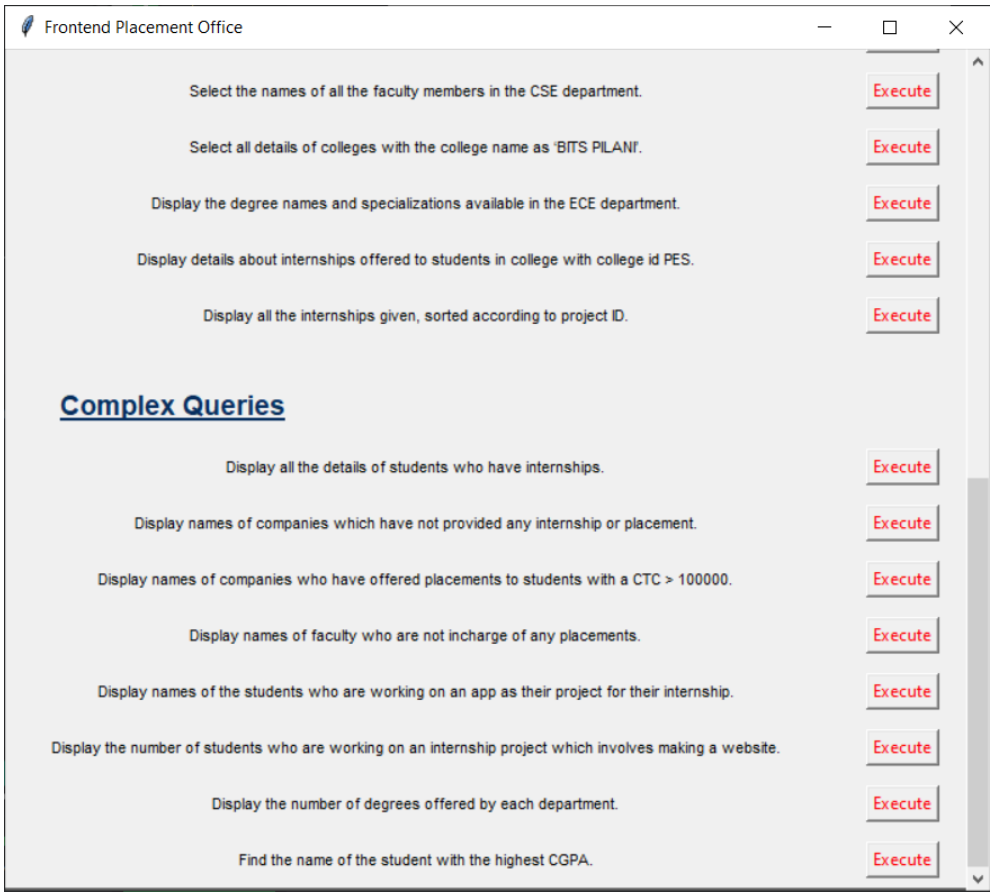
For frontend we used the tkinter library. Although Python has numerous GUI frameworks, Tkinter is the only one that is included in the standard library. Tkinter provides a number of advantages. The code is cross-platform, so it works on Windows, Mac OS X, and Linux. Tkinter programmes seem like they belong on the platform they're running on because visual components are produced using native operating system elements. Also Tkinter is a lightweight framework that is reasonably easy to use in comparison to other frameworks. This makes it an attractive, alternative for creating GUI apps in Python, especially for projects where a current gloss isn't required and the main goal is to quickly construct something that works on several platforms.

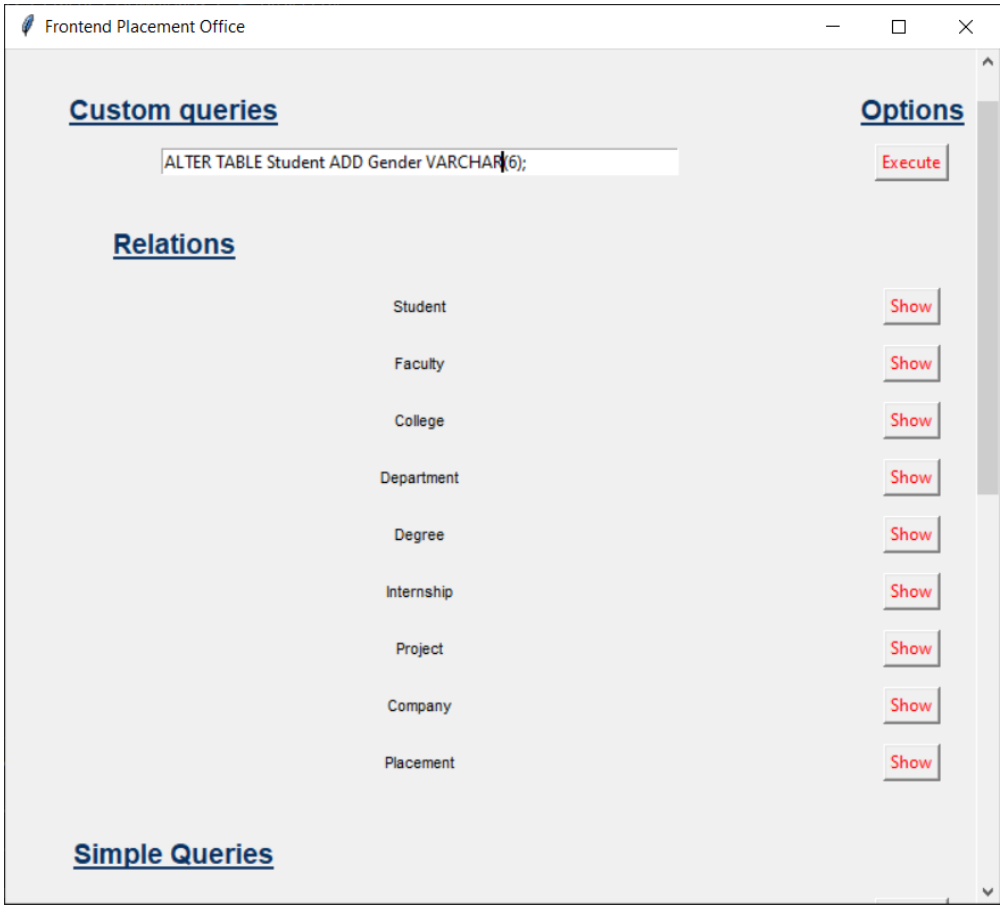
Screenshots

Statements executed from the front-end





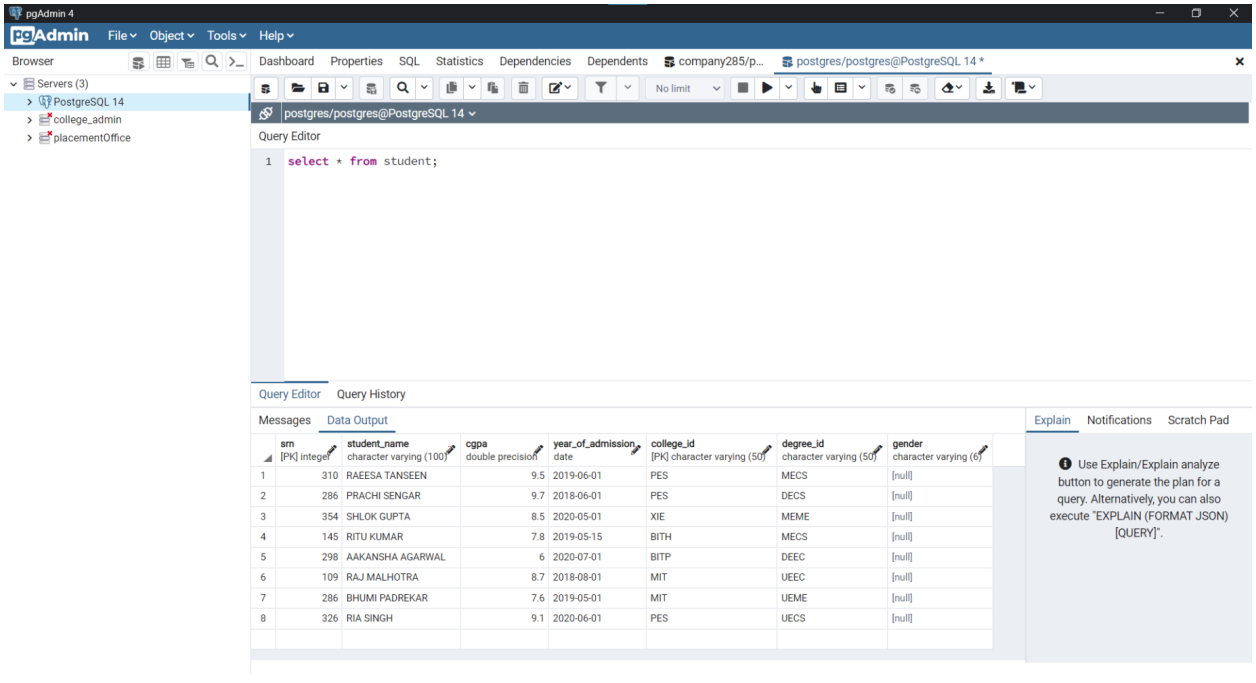

Frontend Placement Office
Select the names of all the faculty members in the CSE department.
Execute
Select all details of colleges with the college name as 'BITS PILANI'.
Execute
Display the degree names and specializations available in the ECE department.
Execute
Display details about internships offered to students in college with college id PES.
Execute
Display all the internships given, sorted according to project ID.
Execute
Complex Queries
Display all the details of students who have internships.
Execute
Display names of companies which have not provided any internship or placement.
Execute
Display names of companies who have offered placements to students with a CTC > 100000.
Execute
Display names of faculty who are not incharge of any placements.
Execute
Display names of the students who are working on an app as their project for their internship.
Execute
Display the number of students who are working on an internship project which involves making a website.
Execute
Display the number of degrees offered by each department.
Execute
Find the name of the student with the highest CGPA.
Execute


Frontend Placement Office
Custom queries
Options
ALTER TABLE Student ADD Gender VARCHAR(6);
Execute
Relations
Student
Show
Faculty
Show
College
Show
Department
Show
Degree
Show
Internship
Show
Project
Show
Company
Show
Placement
Show
Simple Queries

Result

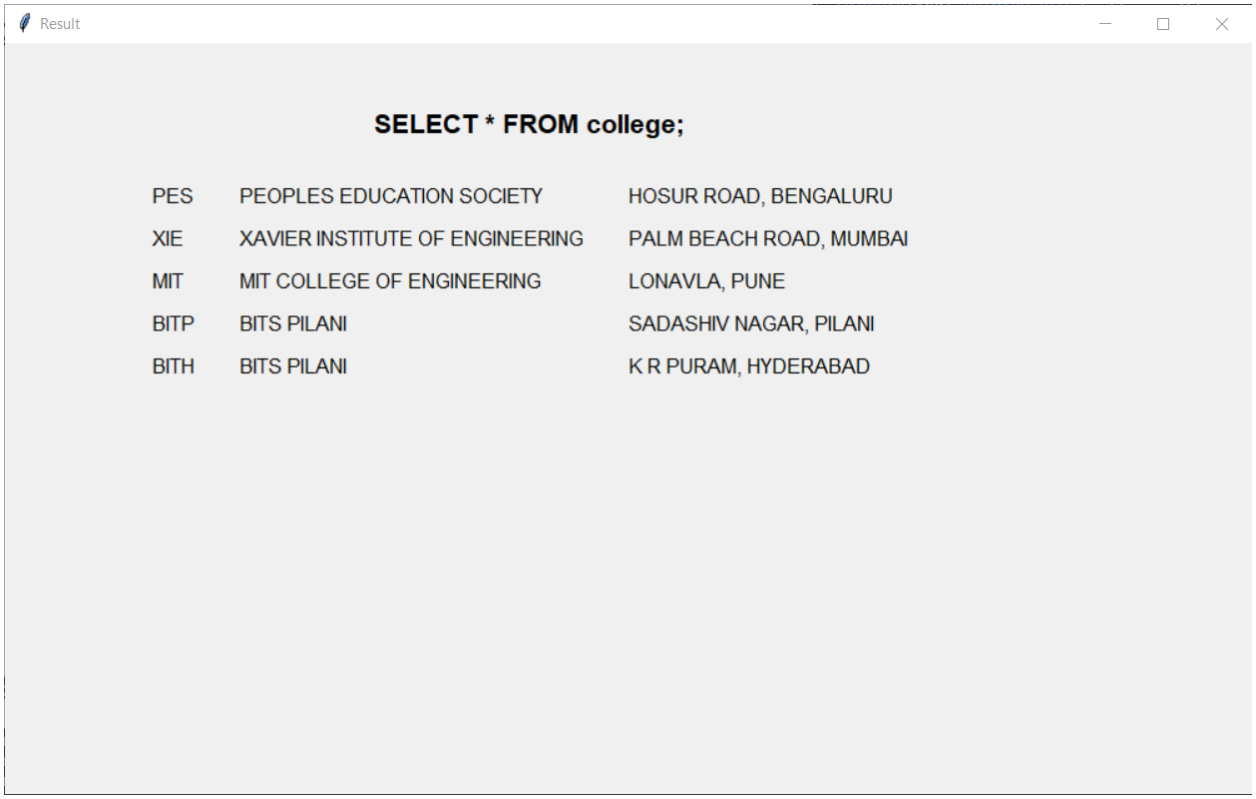
SELECT * FROM student;

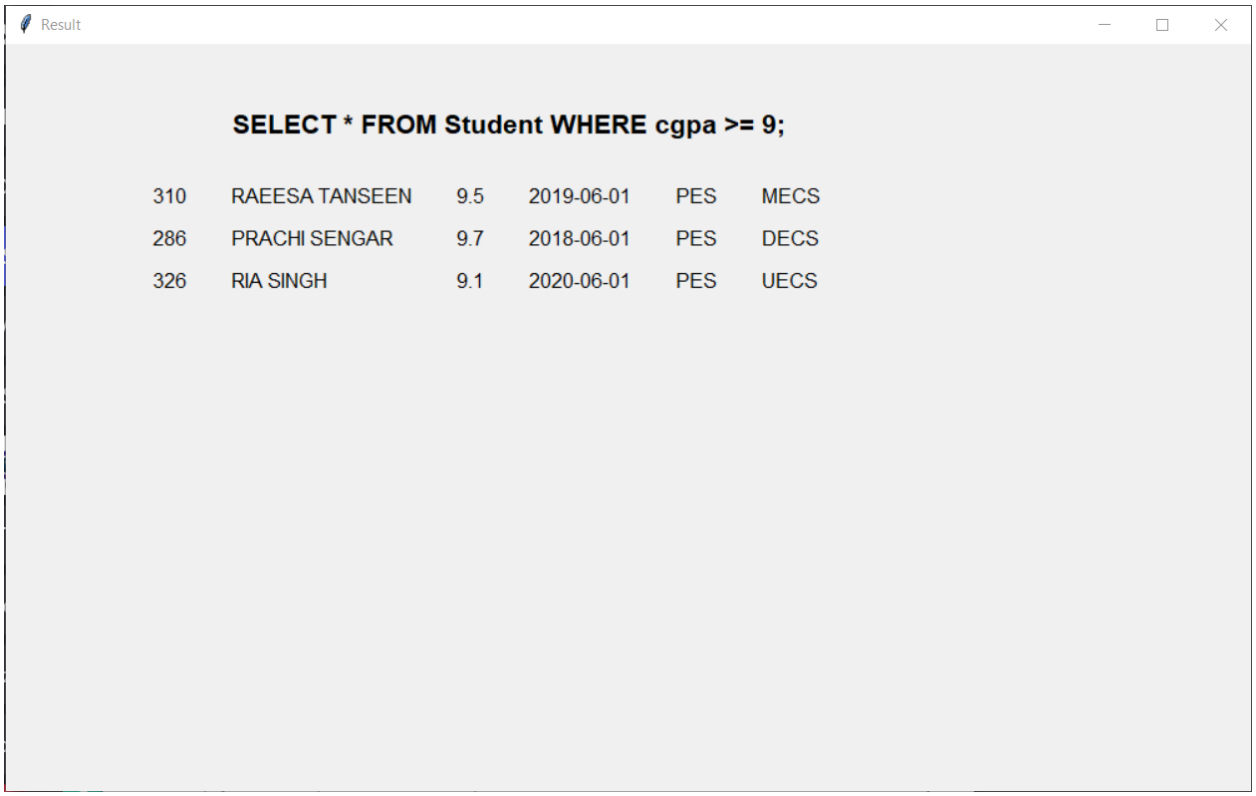
310	RAEESA TANSEEN	9.5	2019-06-01	PES	MECS
286	PRACHI SENGAR	9.7	2018-06-01	PES	DECS
354	SHLOK GUPTA	8.5	2020-05-01	XIE	MEME
145	RITU KUMAR	7.8	2019-05-15	BITH	MECS
298	AAKANSHA AGARWAL	6.0	2020-07-01	BITP	DEEC
109	RAJ MALHOTRA	8.7	2018-08-01	MIT	UEEC
286	BHUMI PADREKAR	7.6	2019-05-01	MIT	UEME
326	RIA SINGH	9.1	2020-06-01	PES	UECS



	srn [PK] integer	student_name character varying (100)	cgpa double precision	year_of_admission date	college_id [PK] character varying (50)	degree_id character varying (50)	gender character varying (6)
1	310	RAEESA TANSEEN	9.5	2019-06-01	PES	MECS	[null]
2	286	PRACHI SENGAR	9.7	2018-06-01	PES	DECS	[null]
3	354	SHLOK GUPTA	8.5	2020-05-01	XIE	MEME	[null]
4	145	RITU KUMAR	7.8	2019-05-15	BITH	MECS	[null]
5	298	AAKANSHA AGARWAL	6	2020-07-01	BITP	DEEC	[null]
6	109	RAJ MALHOTRA	8.7	2018-08-01	MIT	UEEC	[null]
7	286	BHUMI PADREKAR	7.6	2019-05-01	MIT	UEME	[null]
8	326	RIA SINGH	9.1	2020-06-01	PES	UECS	[null]

Simple Queries





Result

SELECT * FROM College WHERE college_name = 'BITS PILANI';

BITP	BITS PILANI	SADASHIV NAGAR, PILANI
BITH	BITS PILANI	K R PURAM, HYDERABAD

Result

SELECT degree_name,specialization FROM Degree WHERE dept_id = 'ECE';

BACHELORS	IMAGE PROCESSING
MASTERS	SIGNAL PROCESSING
DOCTORATE	VLSI

Result

SELECT * FROM Internship WHERE college_id = 'PES';

326	PESCS115	A1	PES	1008
286	PESCS115	B3	PES	1003
310	PESCS115	B3	PES	1005

Result

SELECT * FROM Internship ORDER BY project_id;

354	XIEME111	D7	XIE	1001
109	MITEC123	C6	MIT	1002
286	PESCS115	B3	PES	1003
286	MITME023	A2	MIT	1004
310	PESCS115	B3	PES	1005
354	XIEME111	D8	XIE	1006
145	BITHCS105	C6	BITH	1007
326	PESCS115	A1	PES	1008

Complex Queries

Result

select * from student where student.SRN in (
select internship.SRN from internship);

286	BHUMI PADREKAR	7.6	2019-05-01	MIT	UEME
286	PRACHI SENGAR	9.7	2018-06-01	PES	DECS
310	RAEESA TANSEEN	9.5	2019-06-01	PES	MECS
109	RAJ MALHOTRA	8.7	2018-08-01	MIT	UEEC
326	RIA SINGH	9.1	2020-06-01	PES	UECS
354	SHLOK GUPTA	8.5	2020-05-01	XIE	MEME
145	RITU KUMAR	7.8	2019-05-15	BITH	MECS

Result

select company_name from company where company.company_id in (select placement.company_id
from placement join (select * from placement where (placement.CTC > 100000))
student on student.SRN = placement.SRN);

XORIANT

AMAZON

MAQ SOFTWARE

Result

select faculty_name from faculty where faculty.faculty_id in (select faculty_id from faculty except select faculty_id from placement);

Aishwarya S

Archana S

Sriram K

Satish HM

Vandana M

Result

select count(*) from project where upper(project_name) like '%WEBSITE';

2

Result

select student_name, max from student join (select max(cgpa) from student) new_table on new_table.max=student.cgpa;

PRACHI SENGAR 9.7

Business/Application changes/expansion

Enterprises are rapidly migrating from Oracle databases to PostgreSQL, an open-source database. There are several reasons to switch from Oracle to PostgreSQL. Here are a few of the advantages:
Cost- Using Oracle databases incurs additional fees for features like partitioning and high availability, in addition to Oracle licencing prices, and these costs can soon pile up. PostgreSQL is an open-source database that is free to download and use.

Flexibility- PostgreSQL is open-source and available from a variety of public cloud providers, including Amazon Web Services (AWS). You won't be locked onto a vendor using PostgreSQL.

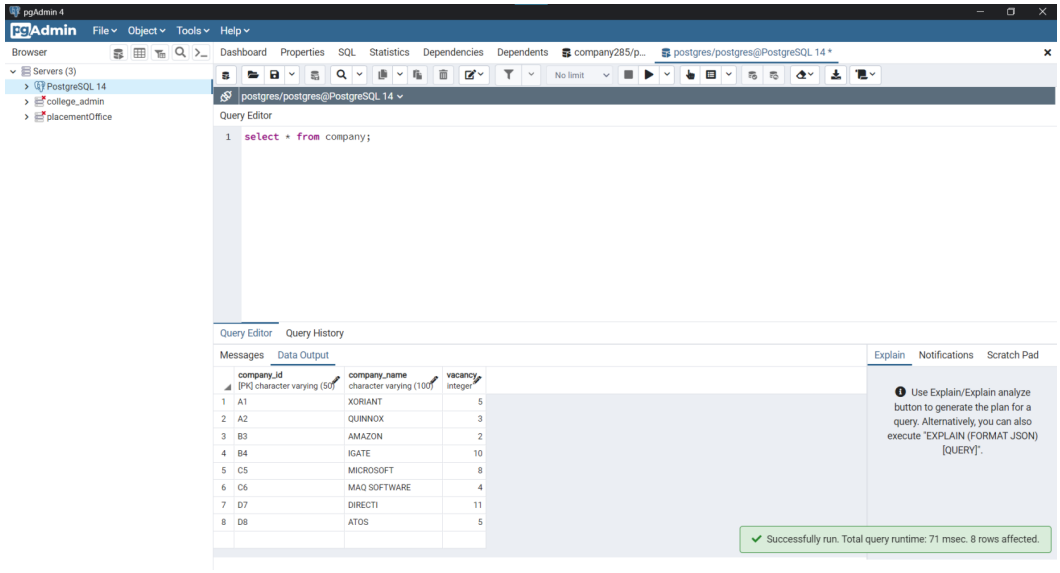
Customizability- Because PostgreSQL is an open-source database, there are a plethora of extensions and add-ons available, many of which are free to use. Similar functionalities in Oracle easily add up in price.

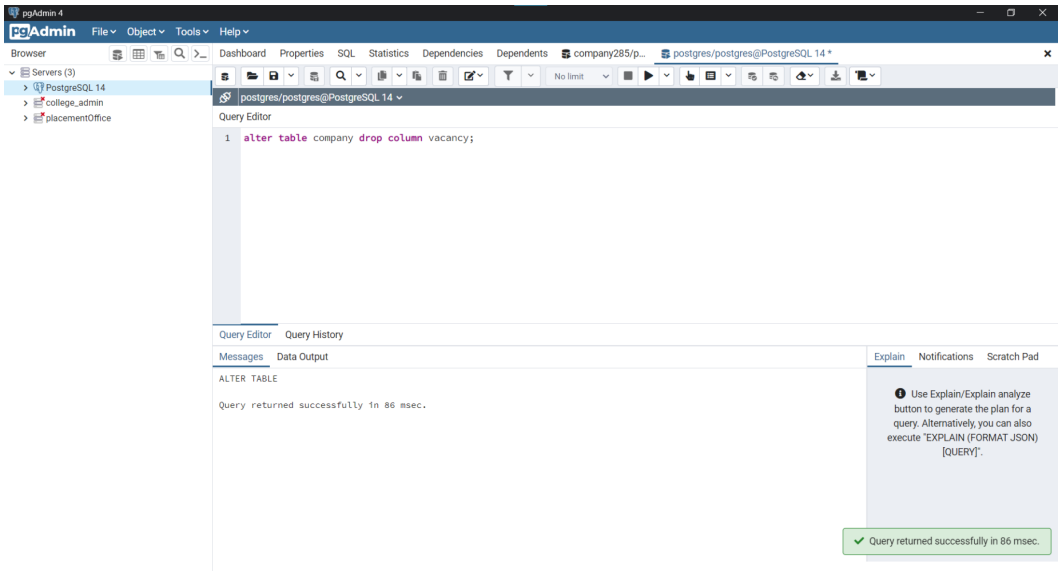
schema changes

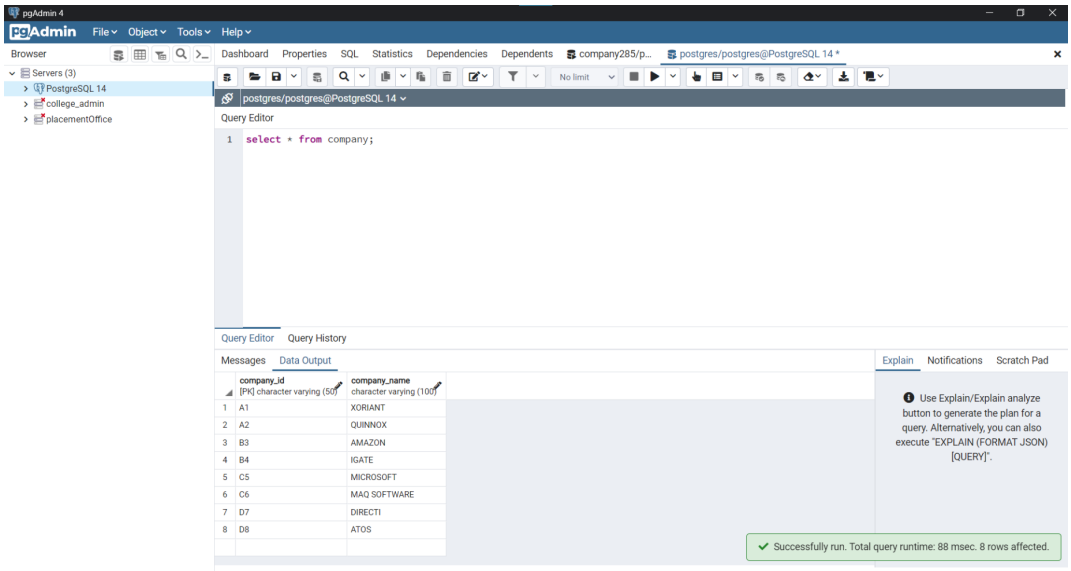
In Oracle, a schema is referred to as a "User" and has the same name as the user. Each Oracle user has their own schema by default. These are not the same in PostgreSQL, and new objects will be assigned to a public schema by default if you don't select one explicitly.

- Create a user with the same name as the schema.
- By default, $user is the first component in the schema search path.

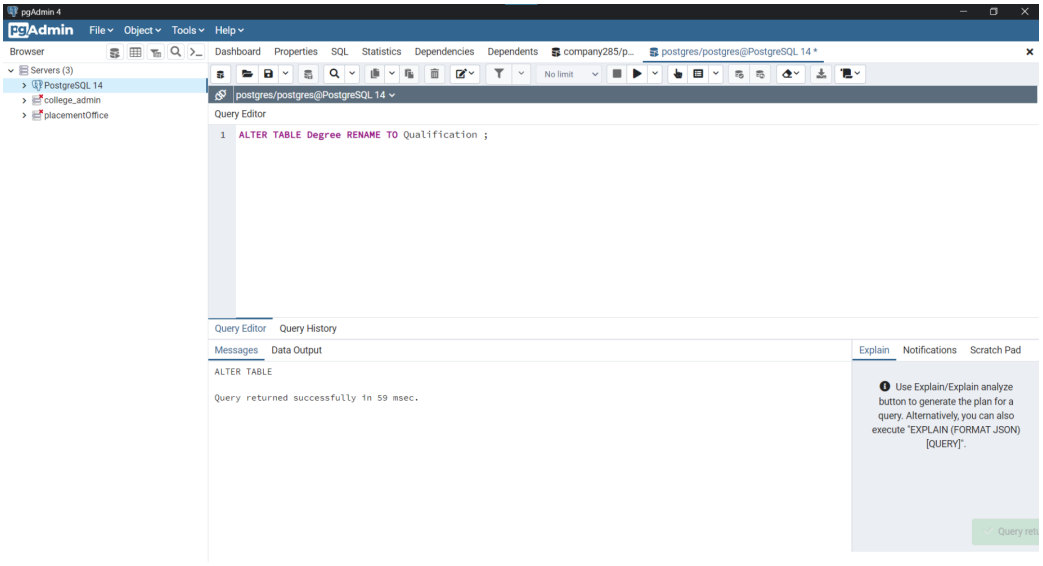
One benefit of the PostgreSQL setup is that a user may establish several schemas without having to create separate users, and provide access to others to create objects in those schemas.

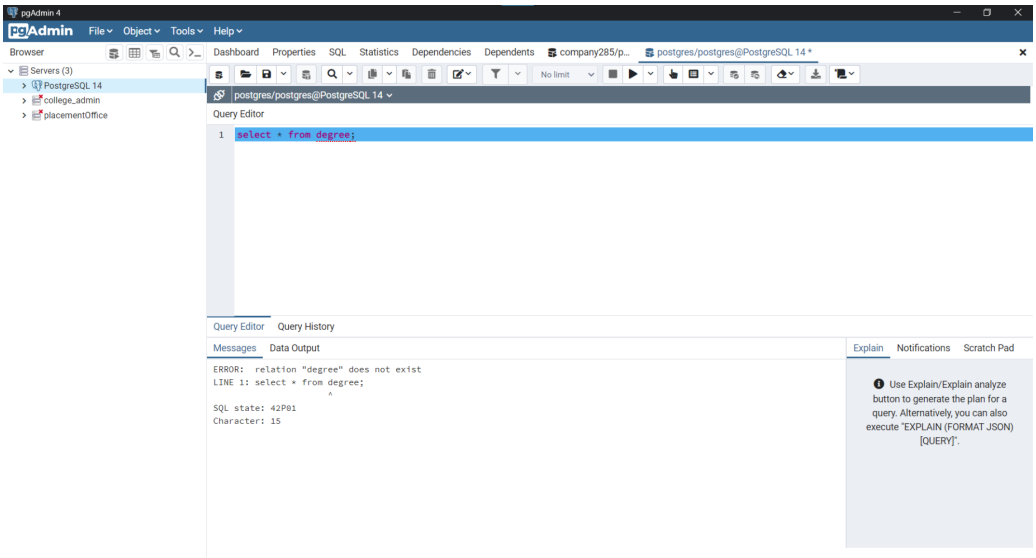


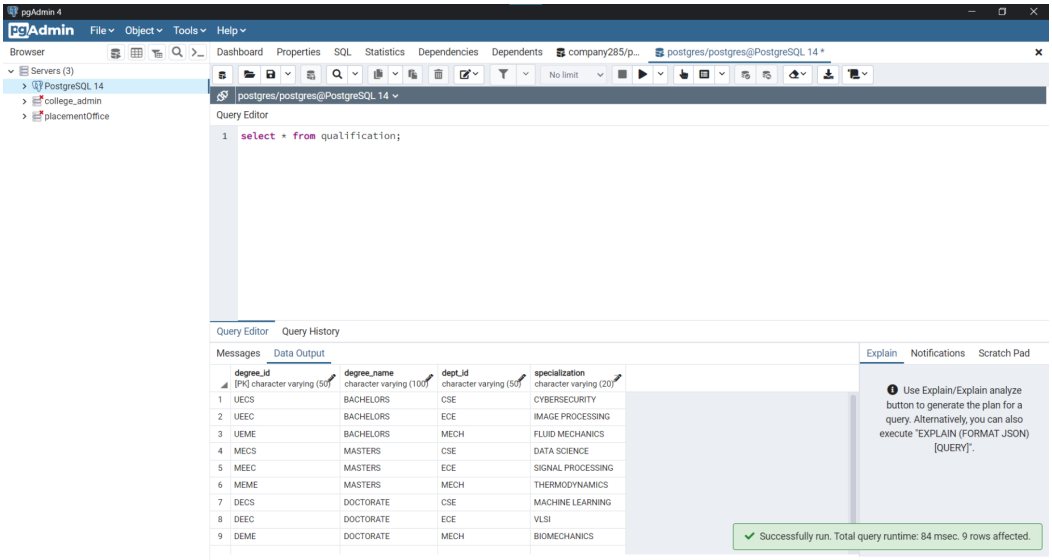




Altering company table

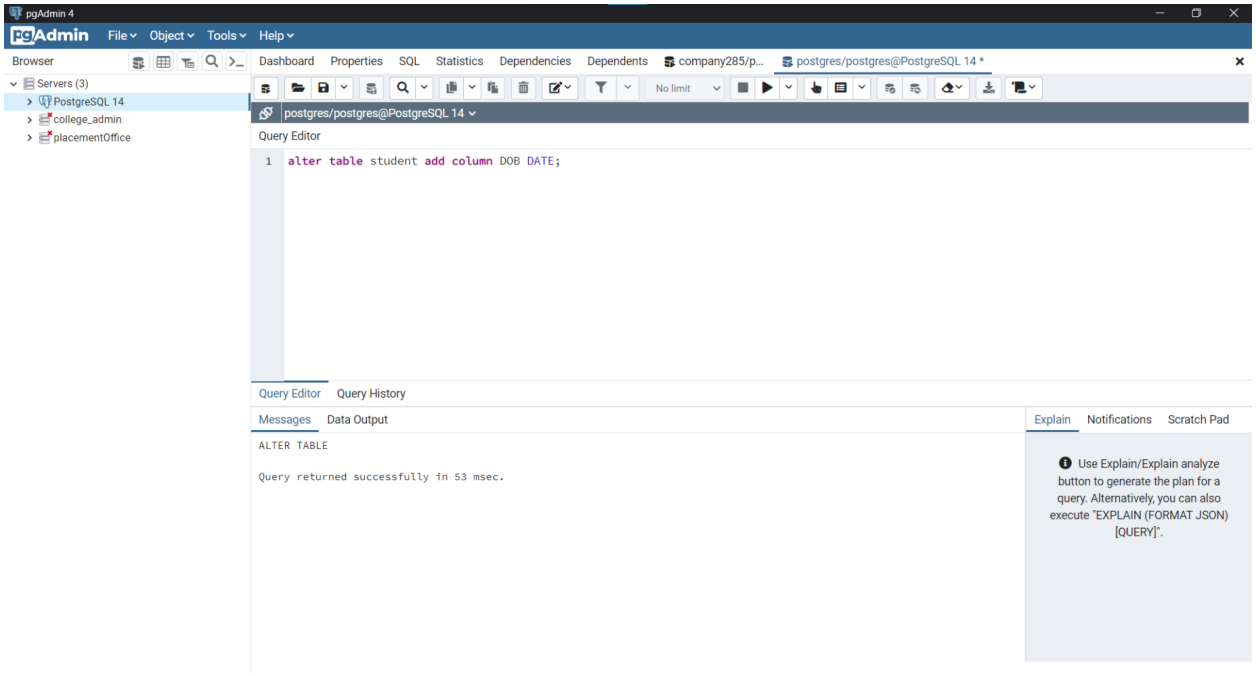


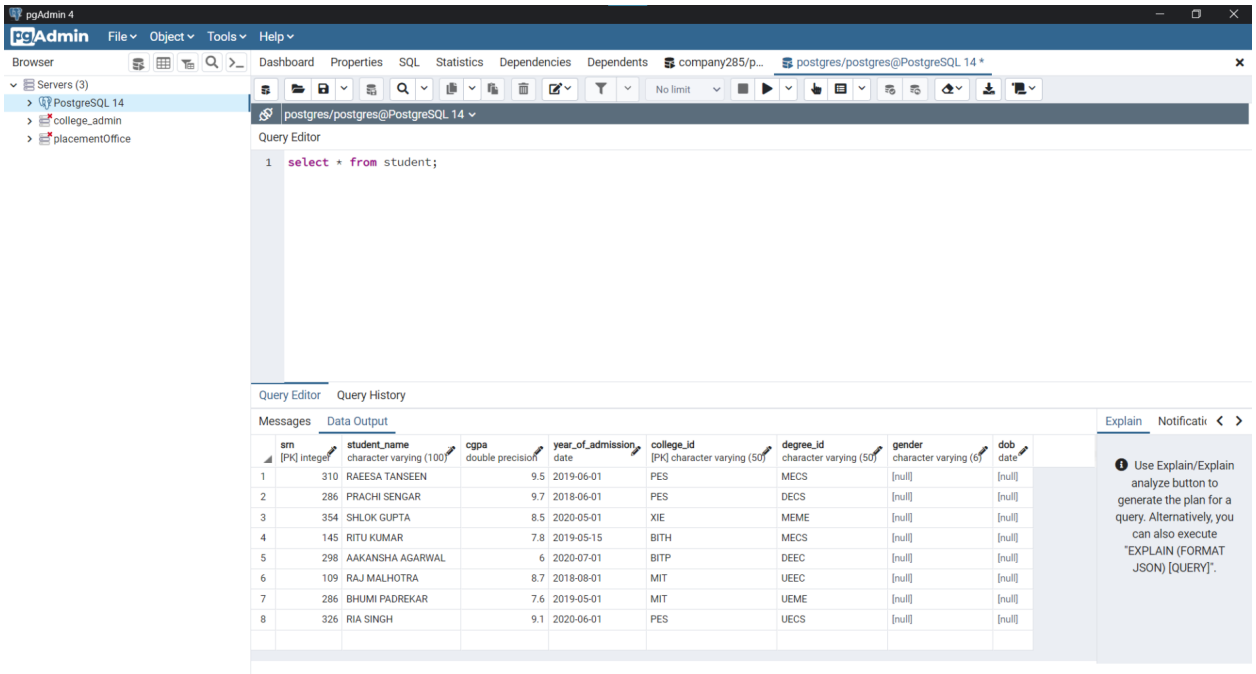




	degree_id [PK] character varying (50)	degree_name character varying (100)	dept_id character varying (50)	specialization character varying (20)
1	UECS	BACHELORS	CSE	CYBERSECURITY
2	UEEC	BACHELORS	ECE	IMAGE PROCESSING
3	UEME	BACHELORS	MECH	FLUID MECHANICS
4	MECS	MASTERS	CSE	DATA SCIENCE
5	MEEC	MASTERS	ECE	SIGNAL PROCESSING
6	MEME	MASTERS	MECH	THERMODYNAMICS
7	DECS	DOCTORATE	CSE	MACHINE LEARNING
8	DEEC	DOCTORATE	ECE	VLSI
9	DEME	DOCTORATE	MECH	BIOMECHANICS

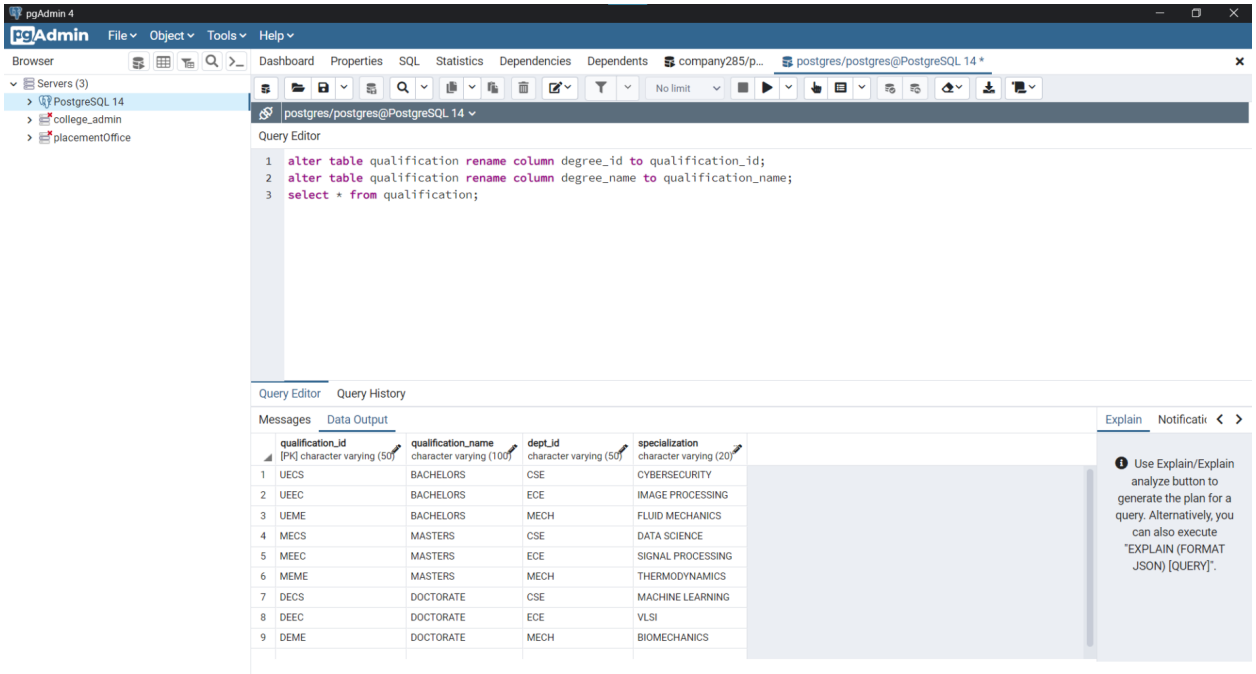
Since students can have a degree or diploma or any other qualification, we change the 'Degree' relation name to 'Qualification'.





	srn [PK] integer	student_name character varying (100)	cgpa double precision	year_of_admission date	college_id [PK] character varying (50)	degree_id character varying (50)	gender character varying (6)	dob date
1	310	RAEESA TANSEEN	9.5	2019-06-01	PES	MECS	[null]	[null]
2	286	PRACHI SENGAR	9.7	2018-06-01	PES	DECS	[null]	[null]
3	354	SHLOK GUPTA	8.5	2020-05-01	XIE	MEME	[null]	[null]
4	145	RITU KUMAR	7.8	2019-05-15	BITH	MECS	[null]	[null]
5	298	AAKANSHA AGARWAL	6	2020-07-01	BITP	DEEC	[null]	[null]
6	109	RAJ MALHOTRA	8.7	2018-08-01	MIT	UEEC	[null]	[null]
7	286	BHUMI PADREKAR	7.6	2019-05-01	MIT	UEME	[null]	[null]
8	326	RIA SINGH	9.1	2020-06-01	PES	UECS	[null]	[null]

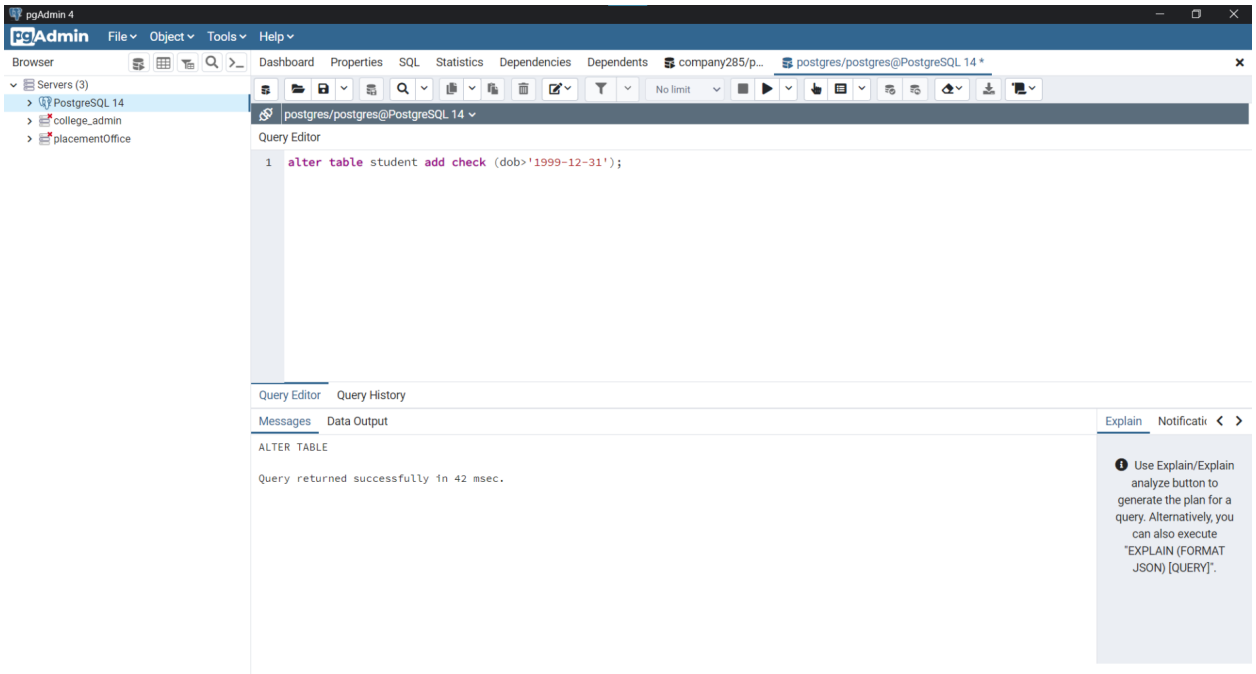
Adding 'DOB' column for students in 'Student' relation in order to determine their age later.



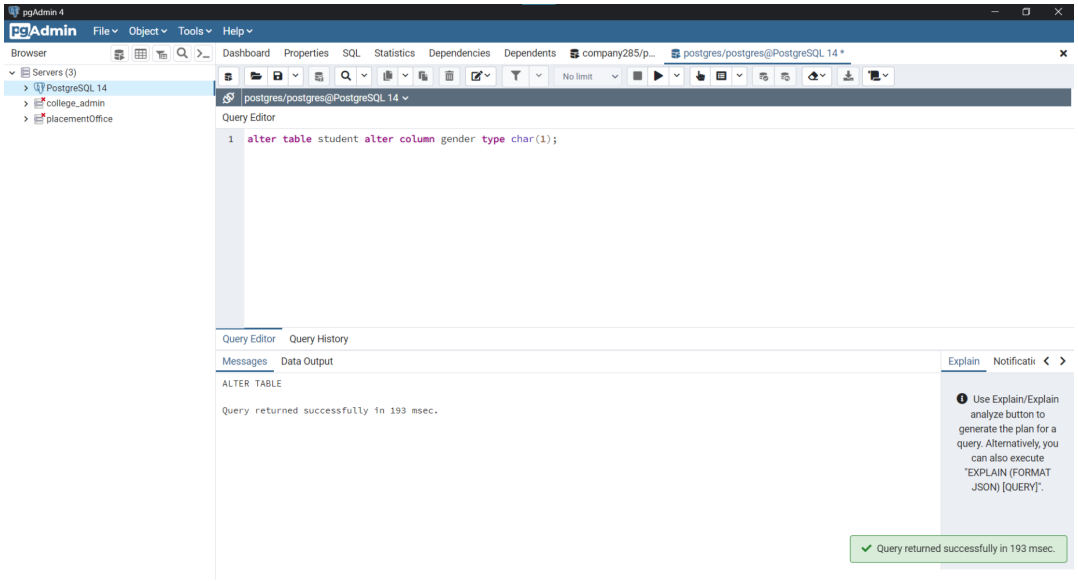
Changing the 'degree_id' attribute in 'Qualification' relation to 'qualification_id' and changing the 'degree_name' attribute to 'qualification_name' in the same relation.

constraint changes

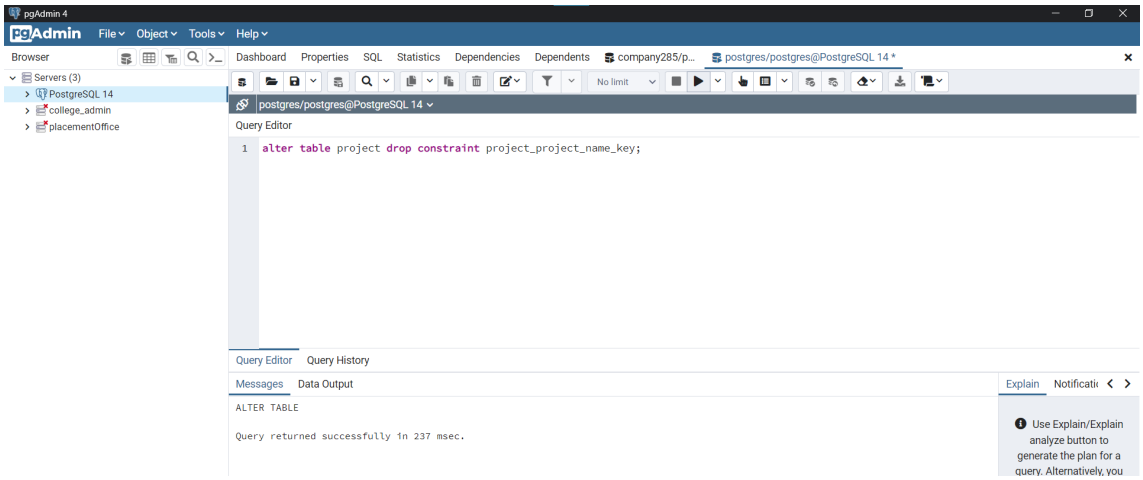
In both database systems the Primary and Foreign Key, Check, Not-Null, and Unique, constraints all operate more or less the same way.



To ensure that all the students eligible for placements and internships have no backlogs and are in their final two years, we add a 'CHECK' constraint to the 'DOB' column of 'Student ' relation



Adding a 'Gender' attribute in 'Student' relation and changing its data type from VARCHAR(6) TO CHAR(1) so as to store 'M' for Male and 'F' for female



Dropping the UNIQUE constraint on the 'project_name' column in the 'Project' relation because two students can be working on the same project as their internship

DBMS migration (from SQL based to No-SQL)

MongoDB is a NoSQL document database that is scalable and adaptable. MongoDB stores information in collections of BSON (Binary JSON) documents rather than tables. MongoDB is capable of handling both structured and unstructured data. This allows you to begin developing your application without first defining the schema.

The PostgreSQL is a relational database management system (RDBMS). Data is stored in tables in an RDBMS, and the database's schema must be defined at the time of construction. While it is feasible to alter tables after they have been created to meet the demands of an application, this process can be time-consuming and error-prone.

If our data model and schema change evolve frequently in an Agile environment, MongoDB, with its flexible schema, is the ideal solution. We can alter the structure of documents quickly with MongoDB, without having to go through application code to update queries and table references. Furthermore, it enables us to swiftly create and scale MongoDB clusters.

An RDMBS like PostgreSQL, on the other hand, may be suitable for our application needs provided we have a stable, relational structure that does not alter over time. It may already be strongly tied with SQL-based clients and frameworks, thus deferring a migration and instead rebuilding the application as a long-term solution is a viable option.

While data models and schema flexibility are crucial, scalability should also be considered. Thus MongoDB is the greatest option for quick querying and scalability. MongoDB scales well in both vertical and horizontal directions.

Contributions

Prachi Sengar - Frontend and query execution
Raeesa Tanseen - Frontend and schema changes
Ria Singh - Report Writing and constraint changes

Time Spent- 12 hrs