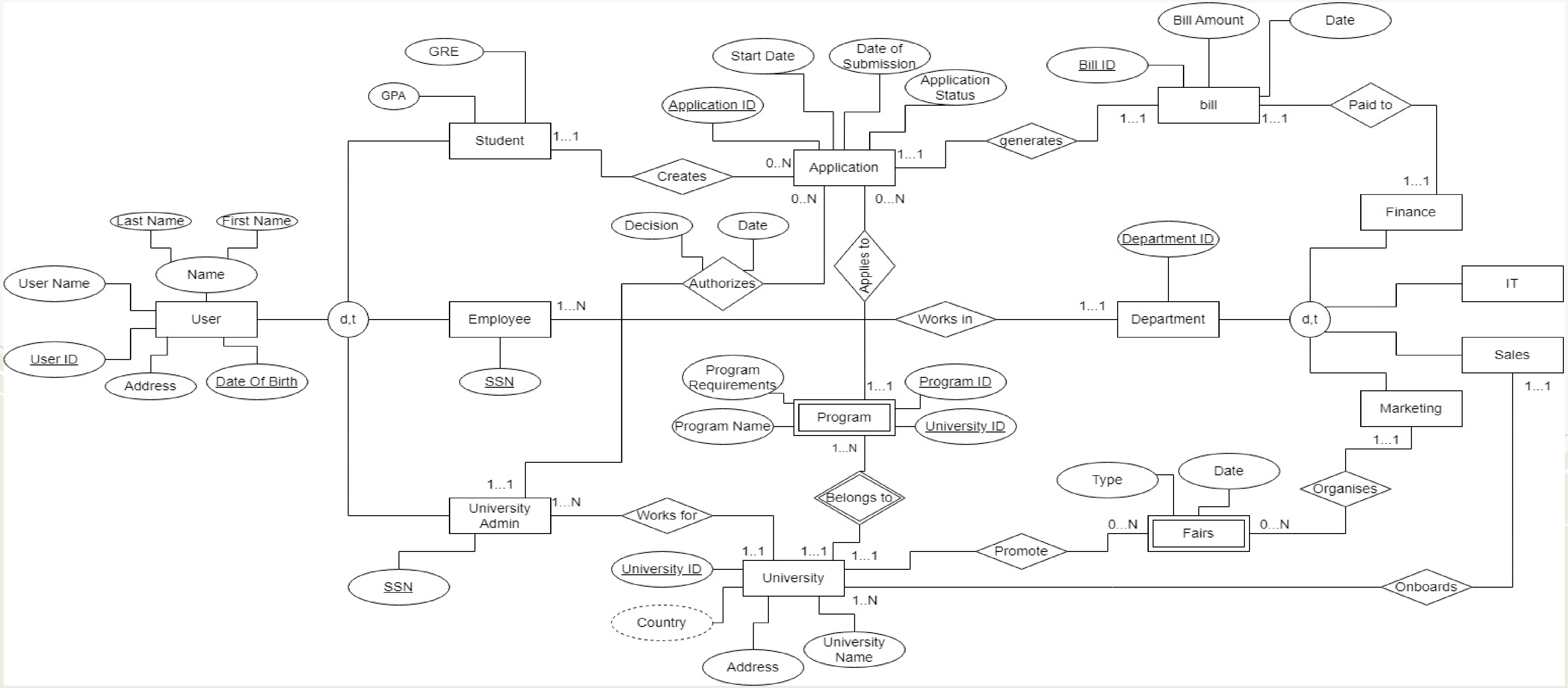
VOYAGE

A University Application Portal

BUSINESS OBJECTIVE

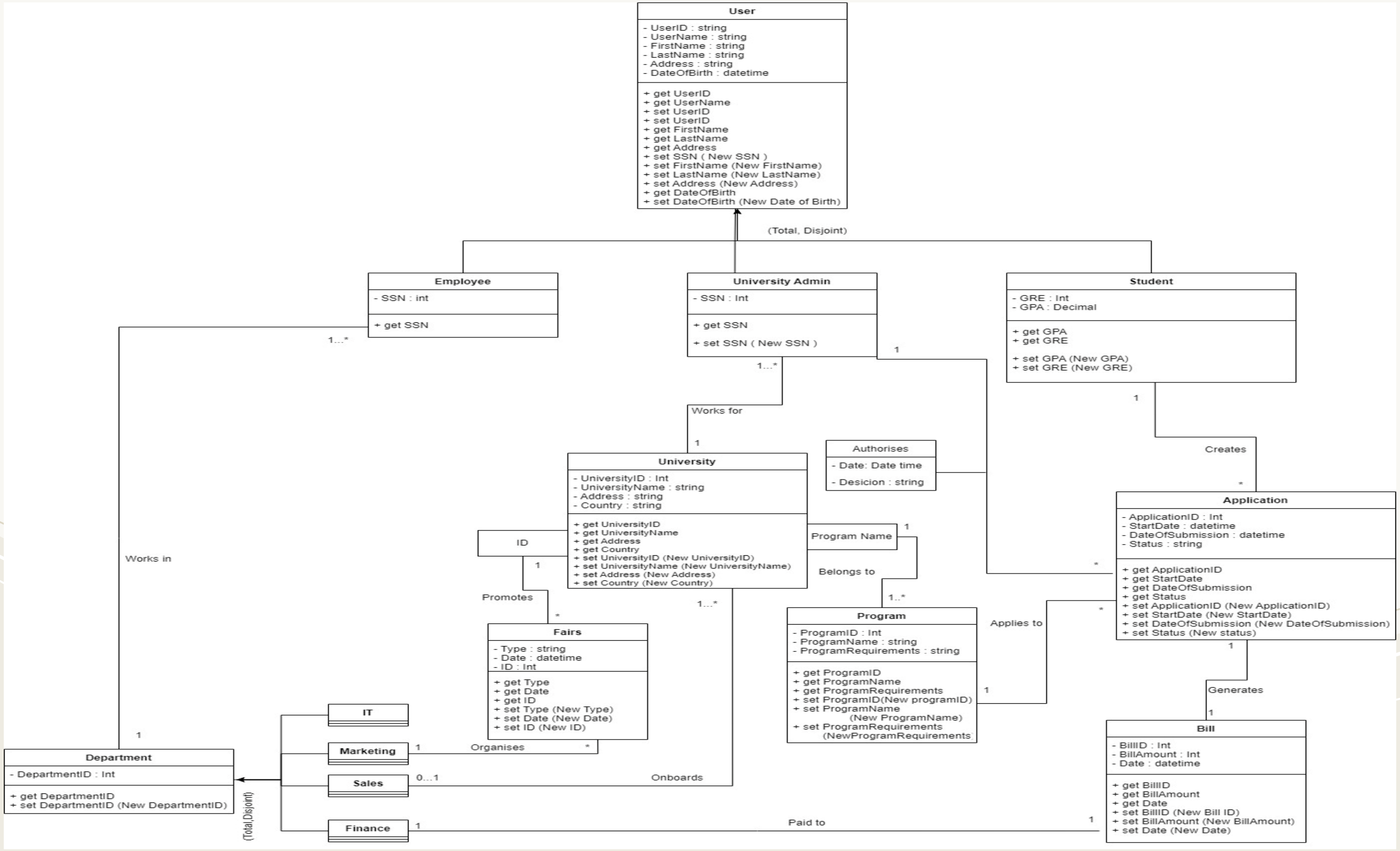
- To provide consultancy services for students to apply for universities across many regions like the US(United States), UK(United Kingdom), and Canada.
- Since, we know to apply for a university a student needs to go to every university’s website and apply over there after applying we keep checking the portal to see the application status it is pretty tiresome for every student. So to simplify this procedure, we partner with universities across the globe to provide a one-stop solution.
- In our application a student can submit and track all their university applications in an easier and more efficient way.
- The universities will appoint their representative on our platform to give their decision on applications based on student credentials.
- We also organize career fairs, providing a platform for universities to promote and highlight their USPs

EER DIAGRAM



GRE
GPA
Student
1...1
Creates
0..N
Start Date
Date of Submission
Application Status
Application ID
Application
1...1
0..N
0...N
generates
Bill Amount
Date
Bill ID
bill
1...1
1...1
Paid to
1...1
Finance
Last Name
First Name
Name
User Name
User
User ID
Address
Date Of Birth
d,t
Decision
Date
Authorizes
Applies to
Employee
1...N
SSN
Works in
1...1
Department ID
Department
d,t
IT
Sales
Marketing
1...1
1...1
Program Requirements
Program ID
1...1
Program
Program Name
University ID
1...N
Belongs to
1...1
University Admin
1...N
Works for
SSN
1..1
1...1
1...1
University ID
University
Country
Address
University Name
1..N
Promote
Type
Date
Fairs
0...N
0...N
Organises
1...1
Onboards

UML CLASS DIAGRAM



RELATIONAL MODEL

User (<u>UserID</u>, UserName, Address, First_Name, Last_Name, Date_Of_Birth)

Primary Key: UserID

User_Employee (<u>*UserID*</u>, SSN, *DepartmentID*)

Primary Key: UserID; Foreign Key: UserID (NOT NULL), DepartmentID (NOT NULL)

User_Student (<u>*UserID*</u>, GPA, GRE)

Primary Key: UserID; Foreign Key: UserID (NOT NULL)

User_UniversityAdmin (<u>*UserID*</u>, SSN, *UniversityID*)

Primary Key: UserID; Foreign Key: UserID (NOT NULL), UniversityID (NOT NULL)

University (<u>UniversityID</u>, UniveristyName, Address, Country, *Sales_DepartmentID*)

Primary Key: UniversityID; Foreign Key: Sales_departmentID (CAN BE NULL)

Program (<u>ProgramID</u>, ProgramName, <u>*UniveristyID*</u>)

Primary Key: (ProgramID, UniversityID); Foreign Key: UniversityID (NOT NULL)

Department (<u>Department_ID</u>, Department_name)

Primary Key: Department_ID

RELATIONAL MODEL CONTD.

Application (<u>ApplicationID</u>, StartDate, DateOfSubmission, ApplicationStatus, Authorize_Decision, Authorize_Date, Student_*UserID, ProgramID, UniversirtyID, UniversityAdmin_UserID*)

Primary Key: ApplicationID; Foreign Key: Student_UserID (NOT NULL), ProgramID (NOT NULL), UniversityID (NOT NULL), UniversityAdmin_UserID (NOT NULL)

Bill (<u>Bill_ID</u>, BillAmount, Date, *ApplicationID, Finance_DepartmentID*)

Primary Key: Bill_ID; Foreign Key: *ApplicationID (NOT NULL), Finance_DepartmentID (NOT NULL)*

Fairs (<u>FairID</u>, Type, Date*, <u>UniversityID,</u> Marketing_DepartmentID*)

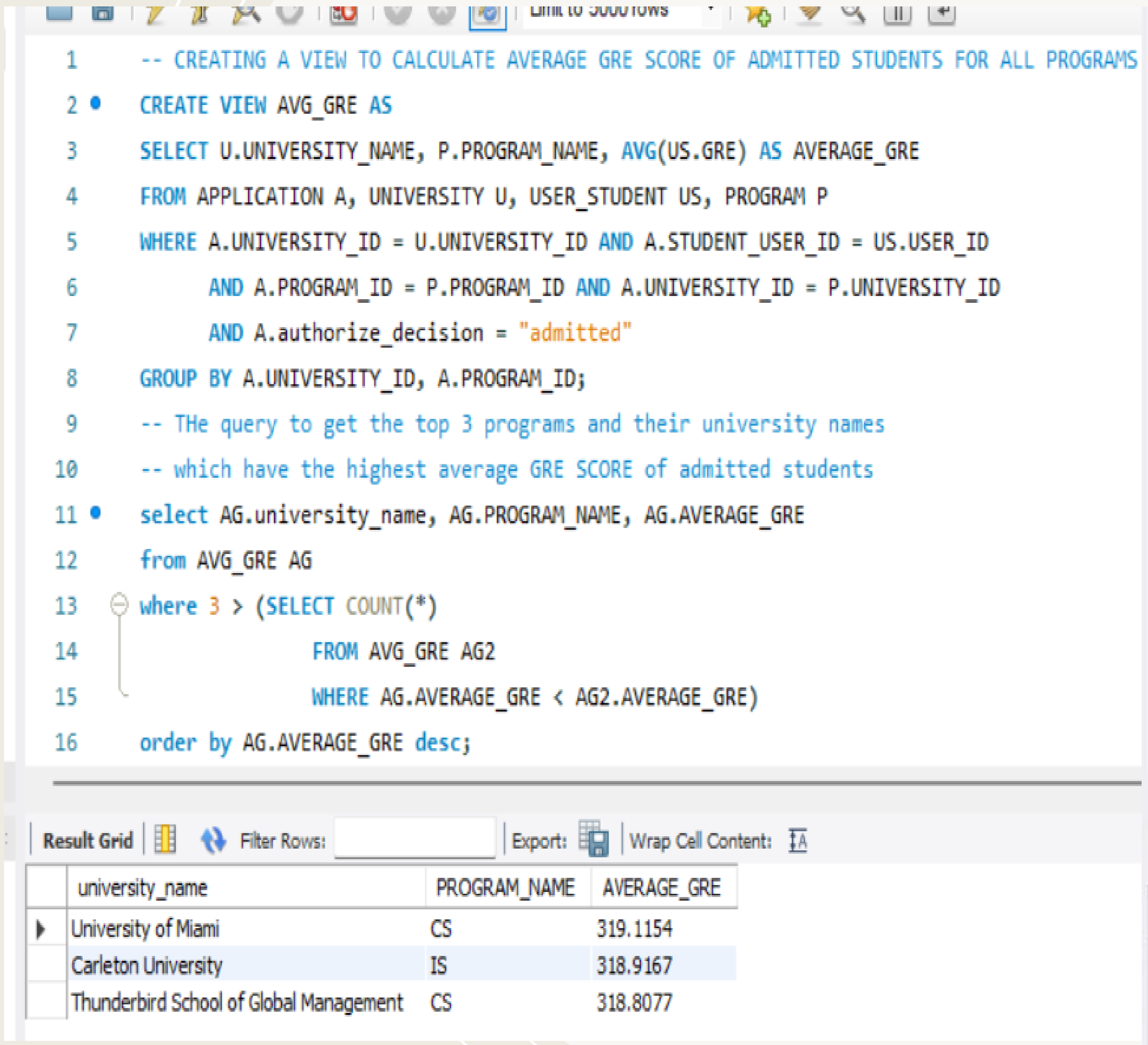
Primary Key: (FairID, UniversityID) Foreign Key: UniversityID (NOT NULL), Marketing_DepartmentID (NOT NULL)

DATA SOURCE: https://mockaroo.com/

SCOPE FOR ANALYTICS

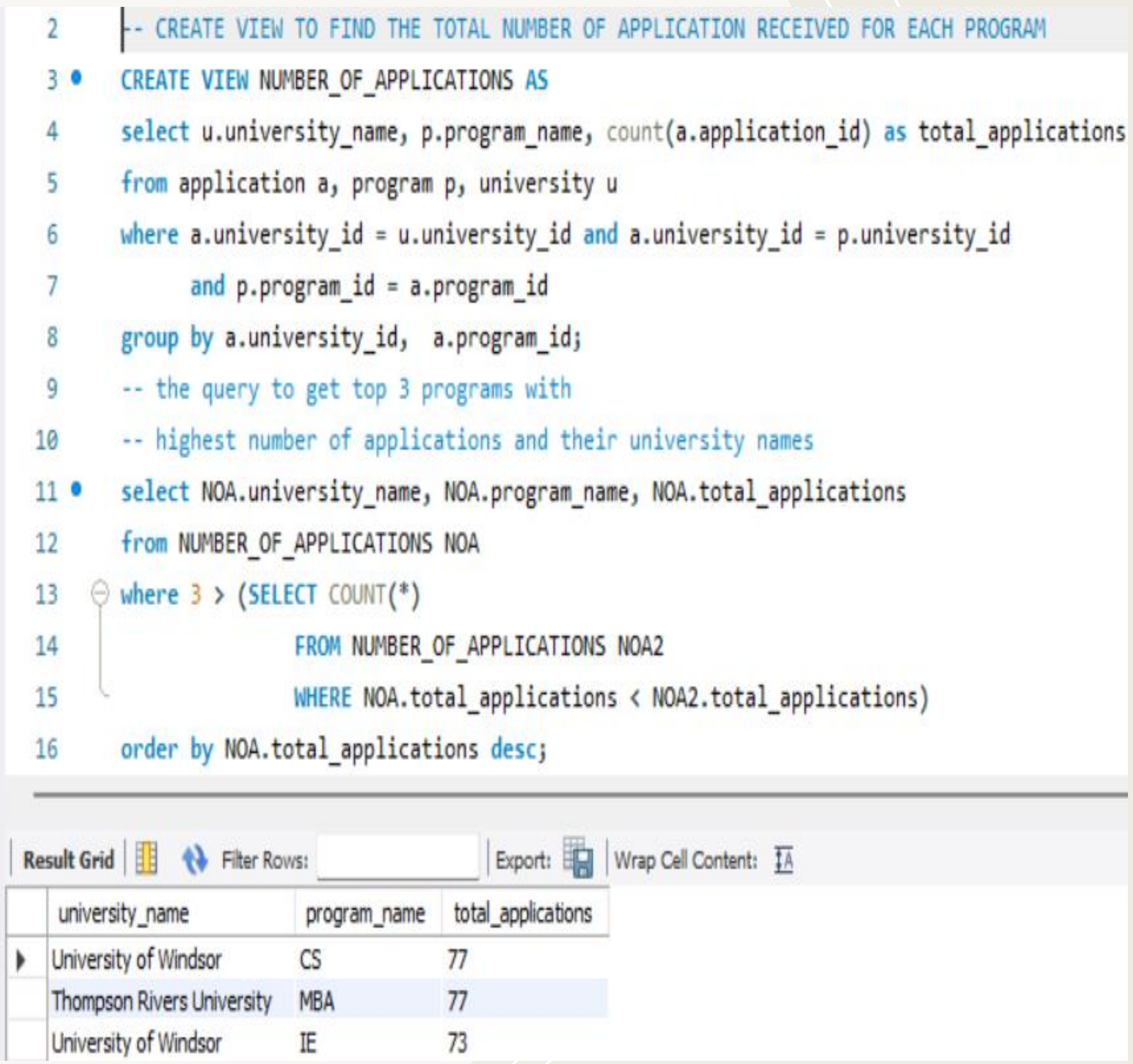
- AVERAGE SCORE OF ADMITTED STUDENTS
- THE ACCEPTANCE RATE OF EACH PROGRAM AND UNIVERSITY
- TOTAL NUMBER OF APPLICATIONS
- REVENUE GENERATED BY THE FIRM
- NUMBER OF EMPLOYEES

MYSQL QUERIES



```sql
-- CREATING A VIEW TO CALCULATE AVERAGE GRE SCORE OF ADMITTED STUDENTS FOR ALL PROGRAMS
CREATE VIEW AVG_GRE AS
SELECT U.UNIVERSITY_NAME, P.PROGRAM_NAME, AVG(US.GRE) AS AVERAGE_GRE
FROM APPLICATION A, UNIVERSITY U, USER_STUDENT US, PROGRAM P
WHERE A.UNIVERSITY_ID = U.UNIVERSITY_ID AND A.STUDENT_USER_ID = US.USER_ID
      AND A.PROGRAM_ID = P.PROGRAM_ID AND A.UNIVERSITY_ID = P.UNIVERSITY_ID
      AND A.authorize_decision = "admitted"
GROUP BY A.UNIVERSITY_ID, A.PROGRAM_ID;
-- THe query to get the top 3 programs and their university names
-- which have the highest average GRE SCORE of admitted students
select AG.university_name, AG.PROGRAM_NAME, AG.AVERAGE_GRE
from AVG_GRE AG
where 3 > (SELECT COUNT(*)
              FROM AVG_GRE AG2
              WHERE AG.AVERAGE_GRE < AG2.AVERAGE_GRE)
order by AG.AVERAGE_GRE desc;
```

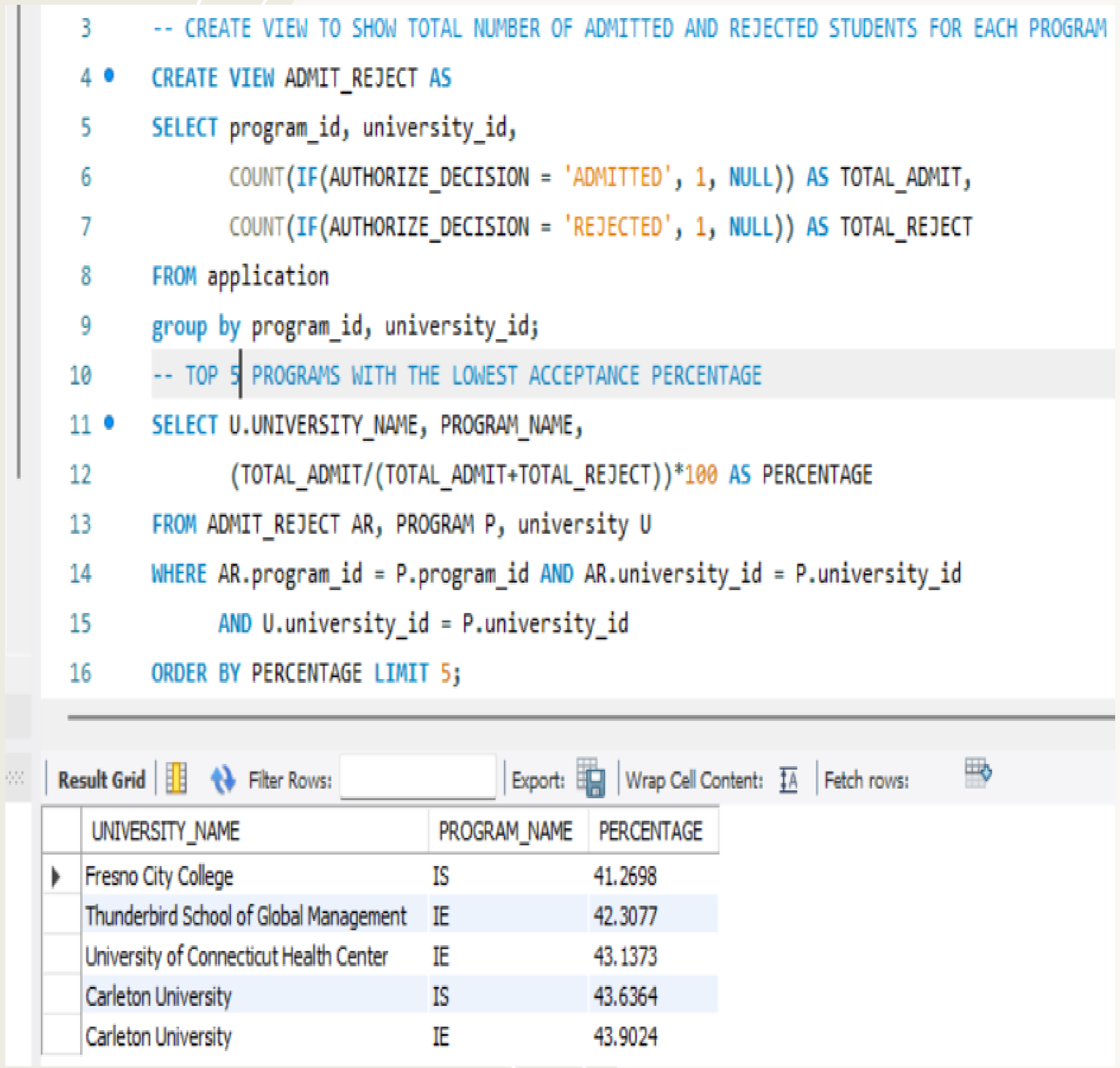
university_name	PROGRAM_NAME	AVERAGE_GRE
University of Miami	CS	319.1154
Carleton University	IS	318.9167
Thunderbird School of Global Management	CS	318.8077



```sql
-- CREATE VIEW TO FIND THE TOTAL NUMBER OF APPLICATION RECEIVED FOR EACH PROGRAM
CREATE VIEW NUMBER_OF_APPLICATIONS AS
select u.university_name, p.program_name, count(a.application_id) as total_applications
from application a, program p, university u
where a.university_id = u.university_id and a.university_id = p.university_id
      and p.program_id = a.program_id
group by a.university_id,  a.program_id;
-- the query to get top 3 programs with
-- highest number of applications and their university names
select NOA.university_name, NOA.program_name, NOA.total_applications
from NUMBER_OF_APPLICATIONS NOA
where 3 > (SELECT COUNT(*)
              FROM NUMBER_OF_APPLICATIONS NOA2
              WHERE NOA.total_applications < NOA2.total_applications)
order by NOA.total_applications desc;
```

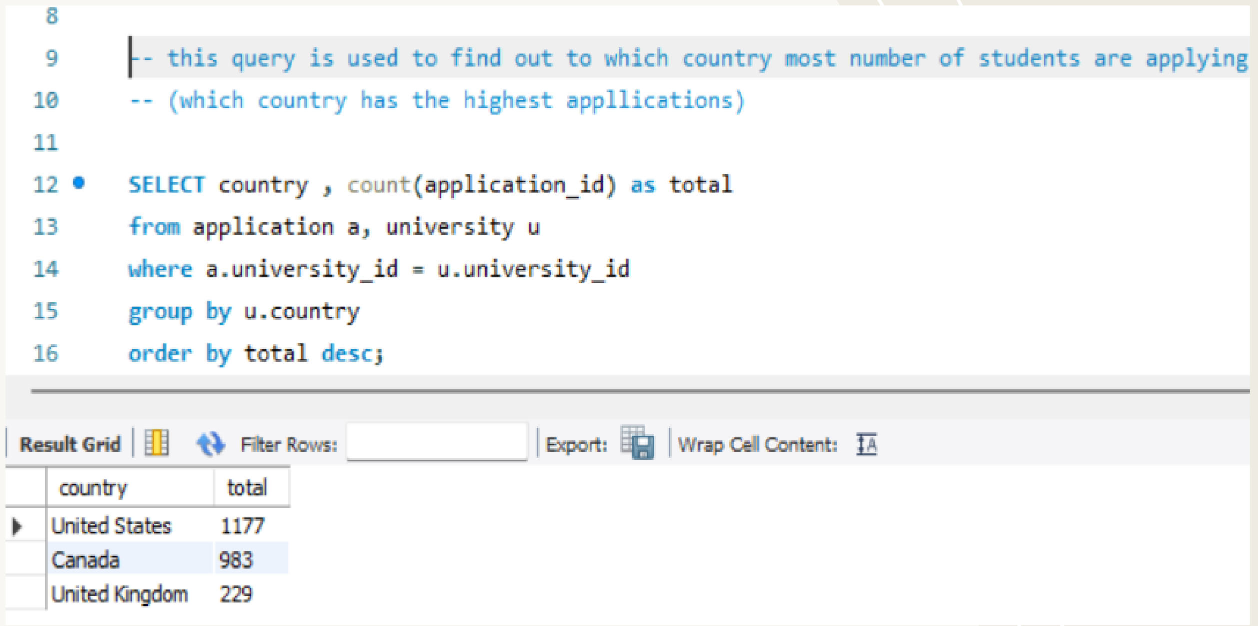
university_name	program_name	total_applications
University of Windsor	CS	77
Thompson Rivers University	MBA	77
University of Windsor	IE	73

MYSQL QUERIES CONTD.



```sql
-- CREATE VIEW TO SHOW TOTAL NUMBER OF ADMITTED AND REJECTED STUDENTS FOR EACH PROGRAM
CREATE VIEW ADMIT_REJECT AS
SELECT program_id, university_id,
       COUNT(IF(AUTHORIZE_DECISION = 'ADMITTED', 1, NULL)) AS TOTAL_ADMIT,
       COUNT(IF(AUTHORIZE_DECISION = 'REJECTED', 1, NULL)) AS TOTAL_REJECT
FROM application
group by program_id, university_id;
-- TOP 5 PROGRAMS WITH THE LOWEST ACCEPTANCE PERCENTAGE
SELECT U.UNIVERSITY_NAME, PROGRAM_NAME,
       (TOTAL_ADMIT/(TOTAL_ADMIT+TOTAL_REJECT))*100 AS PERCENTAGE
FROM ADMIT_REJECT AR, PROGRAM P, university U
WHERE AR.program_id = P.program_id AND AR.university_id = P.university_id
      AND U.university_id = P.university_id
ORDER BY PERCENTAGE LIMIT 5;
```

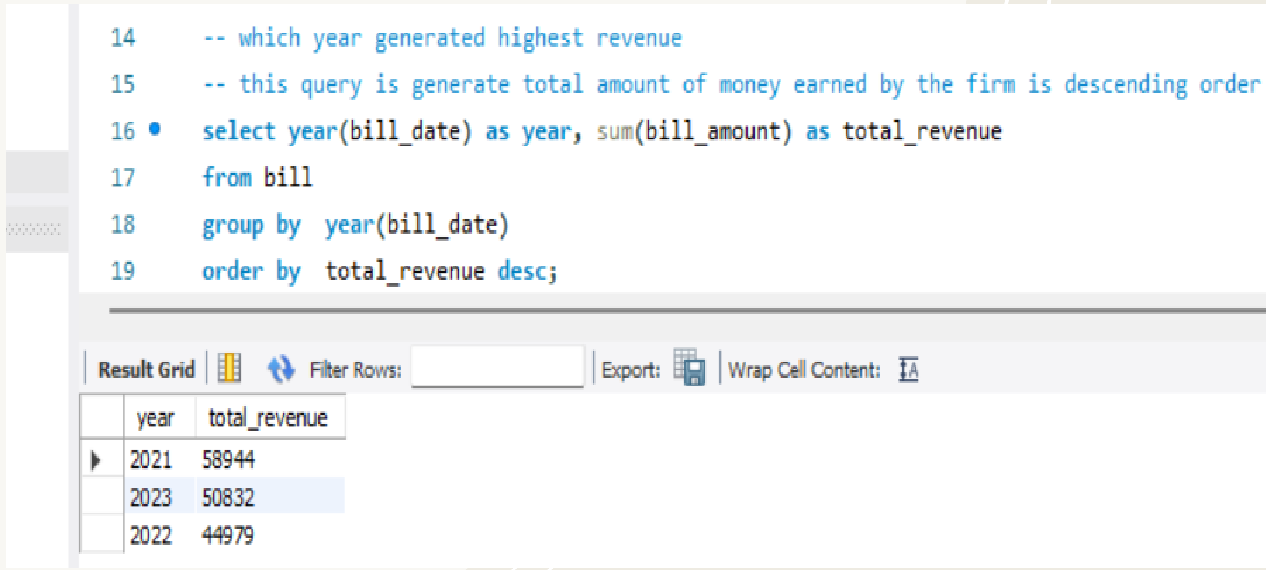
UNIVERSITY_NAME	PROGRAM_NAME	PERCENTAGE
Fresno City College	IS	41.2698
Thunderbird School of Global Management	IE	42.3077
University of Connecticut Health Center	IE	43.1373
Carleton University	IS	43.6364
Carleton University	IE	43.9024



```sql
-- this query is used to find out to which country most number of students are applying
-- (which country has the highest appllications)

SELECT country , count(application_id) as total
from application a, university u
where a.university_id = u.university_id
group by u.country
order by total desc;
```

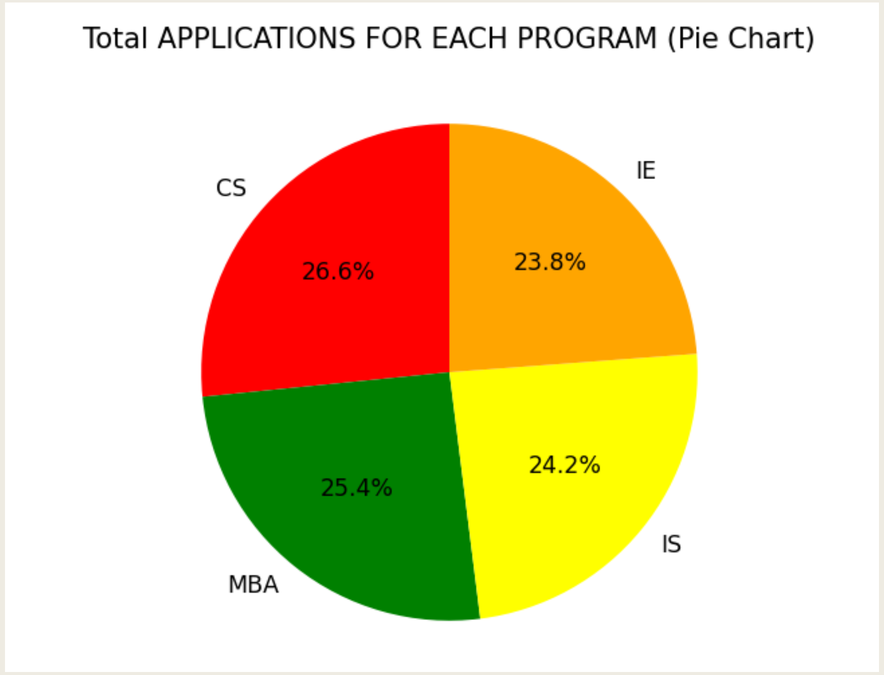
country	total
United States	1177
Canada	983
United Kingdom	229

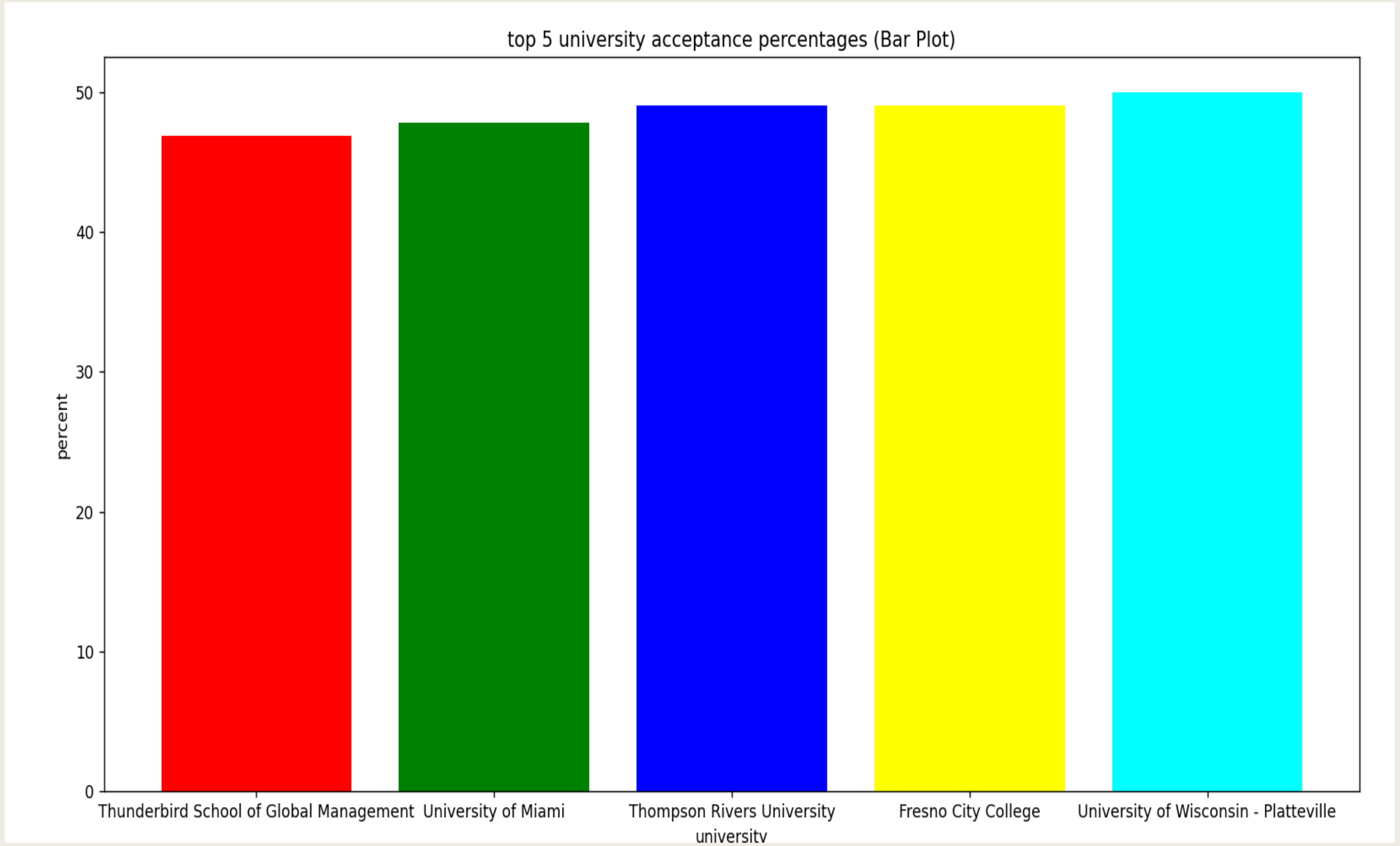


```sql
-- which year generated highest revenue
-- this query is generate total amount of money earned by the firm is descending order
select year(bill_date) as year, sum(bill_amount) as total_revenue
from bill
group by  year(bill_date)
order by  total_revenue desc;
```

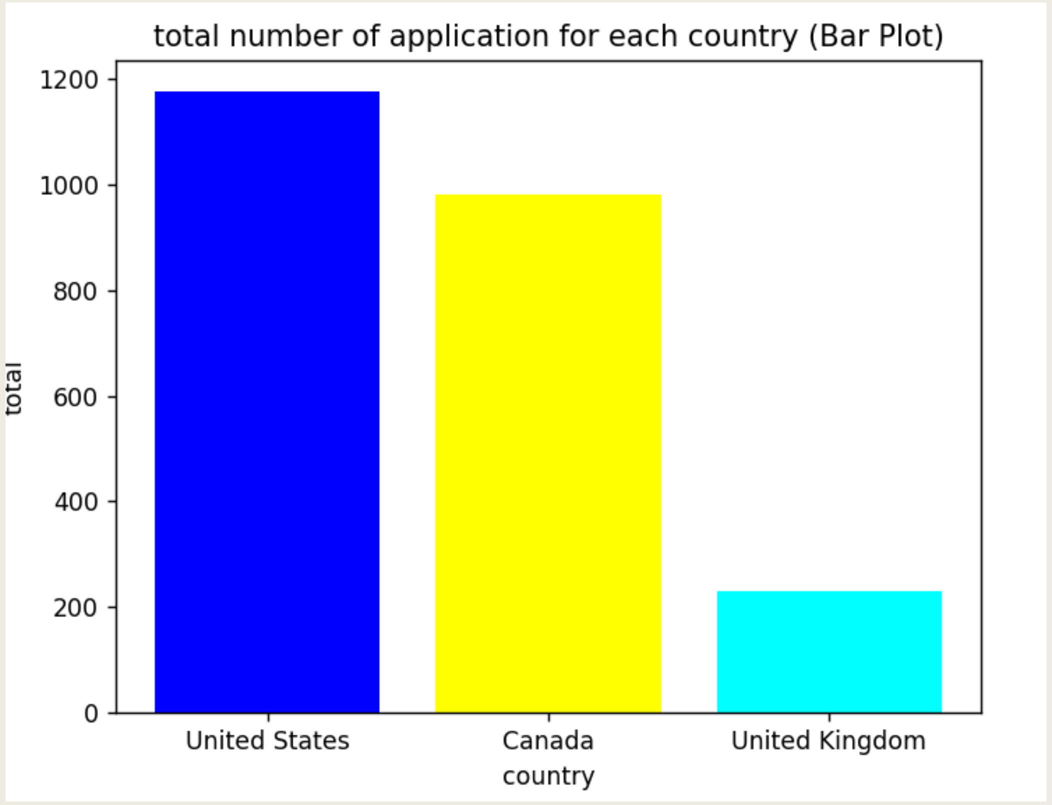
year	total_revenue
2021	58944
2023	50832
2022	44979

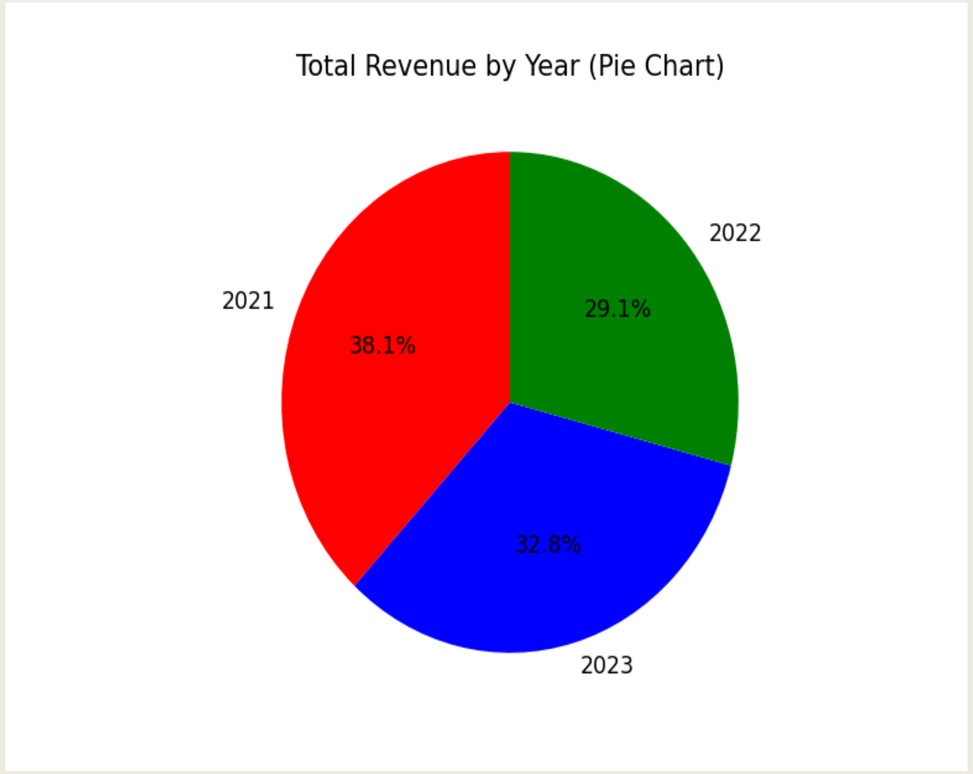
DATABASE ACCESS USING PYTHON



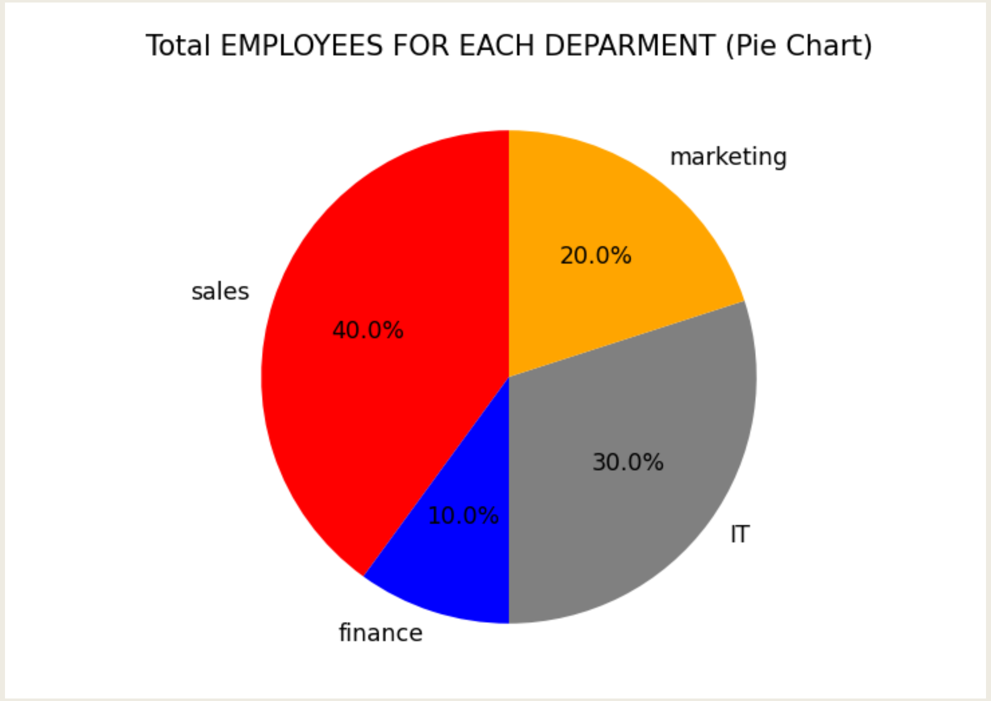


DATABASE ACCESS USING PYTHON



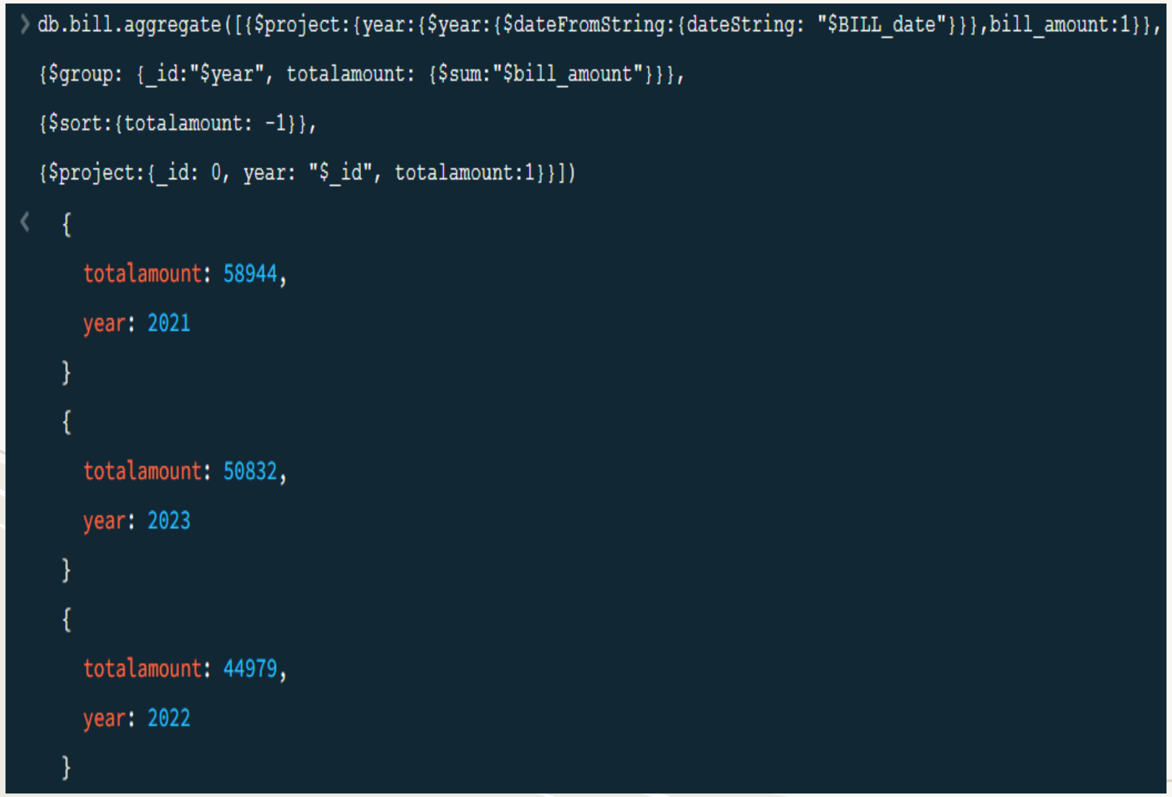


DATABASE ACCESS USING PYTHON



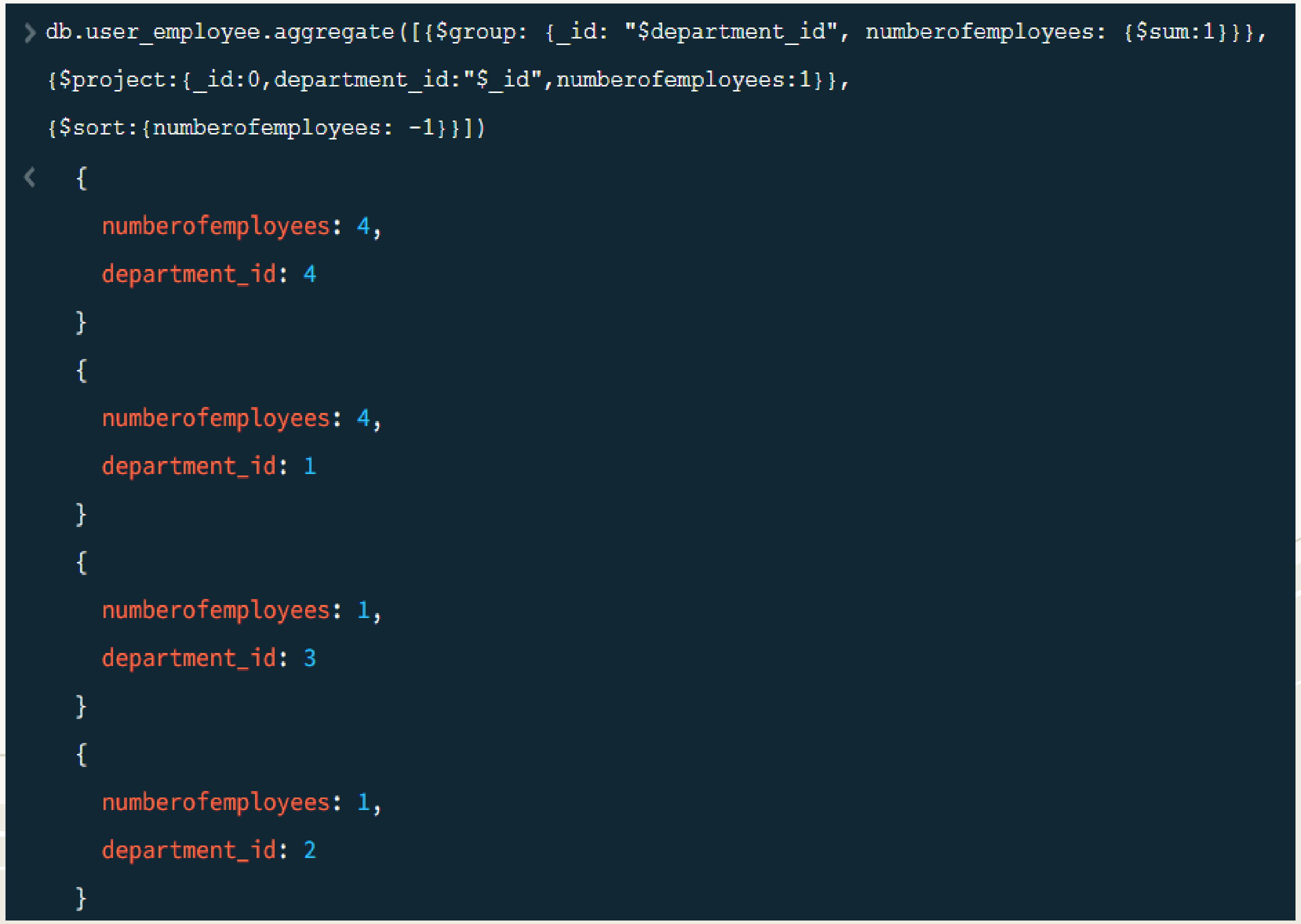
NOSQL(MONGODB) QUERIES

YEAR WITH THE HIGHEST REVENUE



```
db.bill.aggregate([{$project:{year:{$year:{$dateFromString:{dateString: "$BILL_date"}}},bill_amount:1}},
 {$group: {_id:"$year", totalamount: {$sum:"$bill_amount"}}},
 {$sort:{totalamount: -1}},
 {$project:{_id: 0, year: "$_id", totalamount:1}}])
  {
    totalamount: 58944,
    year: 2021
  }
  {
    totalamount: 50832,
    year: 2023
  }
  {
    totalamount: 44979,
    year: 2022
  }
```

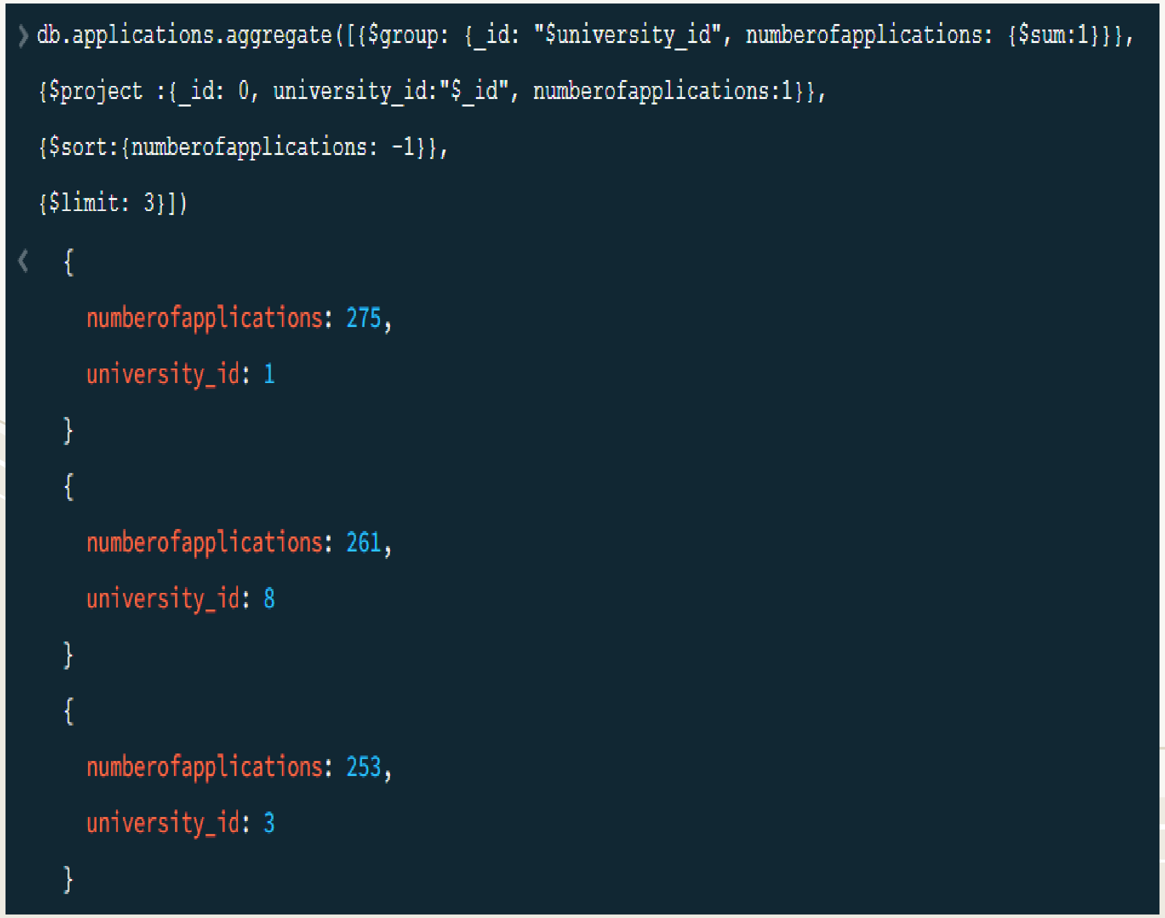
WHICH DEPARTMENT HAS THE HIGHEST EMPLOYEES



```
db.user_employee.aggregate([{$group: {_id: "$department_id", numberofemployees: {$sum:1}}},
 {$project:{_id:0,department_id:"$_id",numberofemployees:1}},
 {$sort:{numberofemployees: -1}}])
  {
    numberofemployees: 4,
    department_id: 4
  }
  {
    numberofemployees: 4,
    department_id: 1
  }
  {
    numberofemployees: 1,
    department_id: 3
  }
  {
    numberofemployees: 1,
    department_id: 2
  }
```

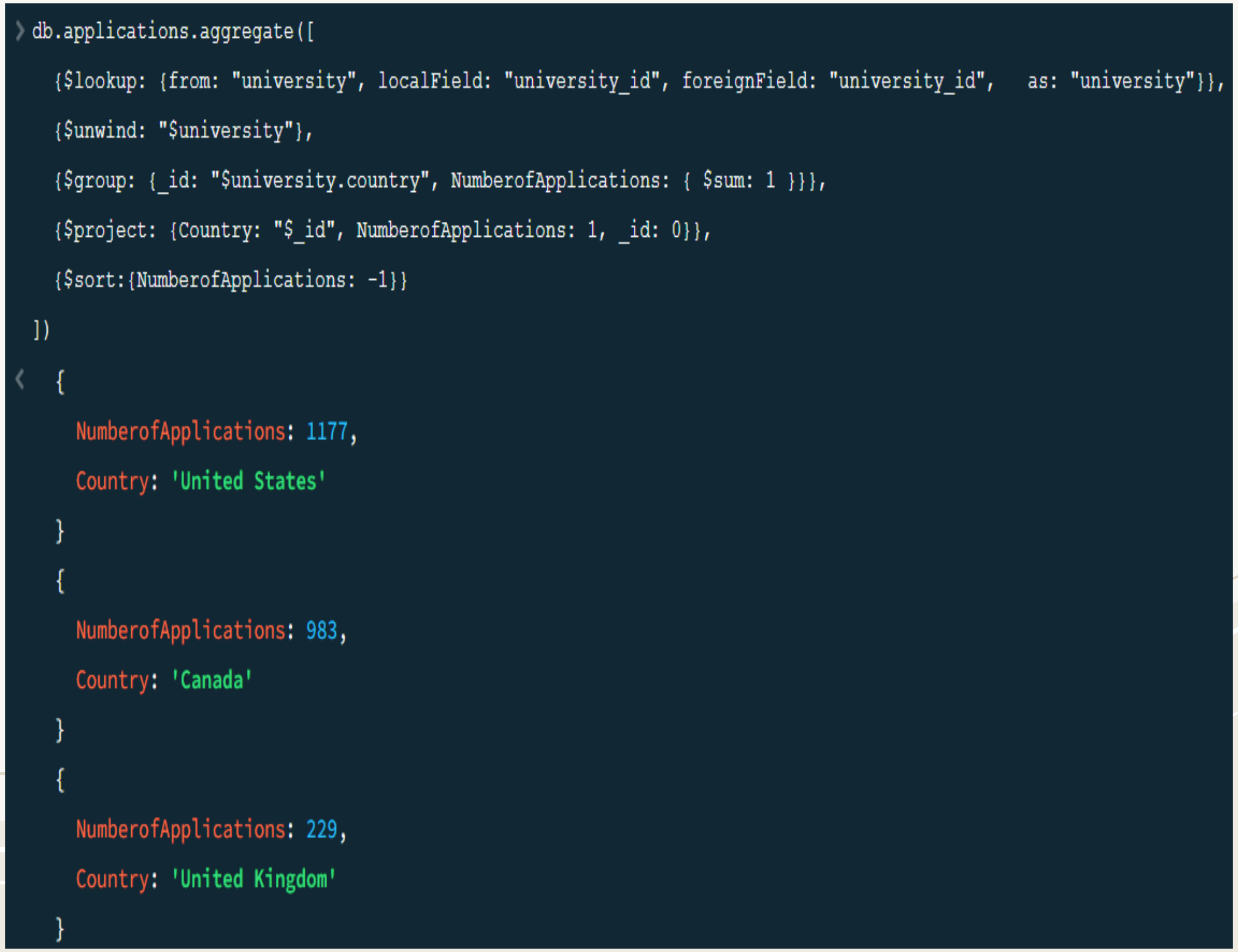
NOSQL QUERIES CONTD.

TOP 3 UNIVERSITIES WITH THE HIGHEST NUMBER OF APPLICATIONS



```
> db.applications.aggregate([{$group: {_id: "$university_id", numberofapplications: {$sum:1}}},
  {$project :{_id: 0, university_id:"$_id", numberofapplications:1}},
  {$sort:{numberofapplications: -1}},
  {$limit: 3}])
<   {
      numberofapplications: 275,
      university_id: 1
    }
    {
      numberofapplications: 261,
      university_id: 8
    }
    {
      numberofapplications: 253,
      university_id: 3
    }
```

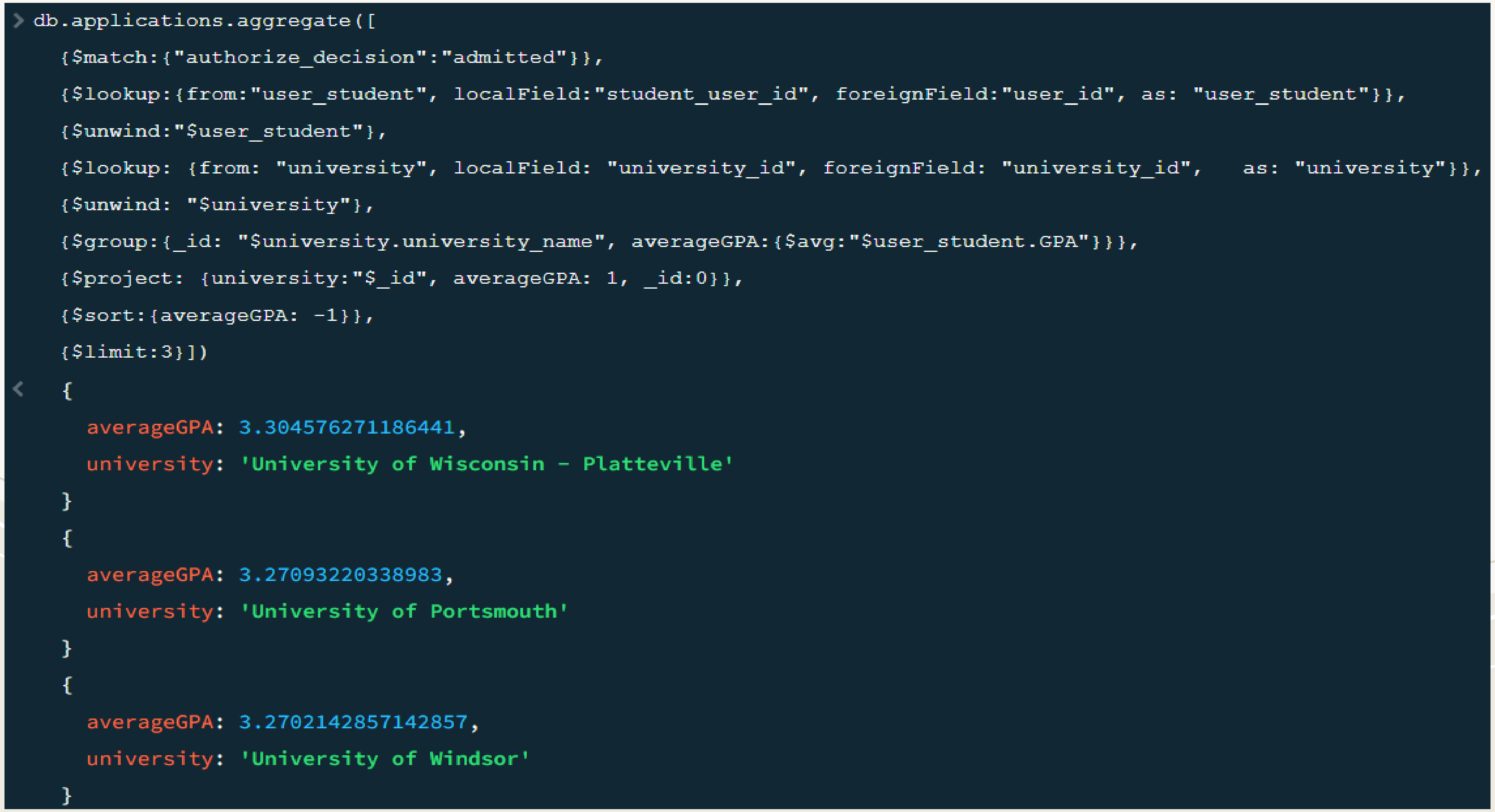
MOST POPULAR COUNTRY



```
> db.applications.aggregate([
    {$lookup: {from: "university", localField: "university_id", foreignField: "university_id",   as: "university"}},
    {$unwind: "$university"},
    {$group: {_id: "$university.country", NumberofApplications: { $sum: 1 }}},
    {$project: {Country: "$_id", NumberofApplications: 1, _id: 0}},
    {$sort:{NumberofApplications: -1}}
  ])
<   {
      NumberofApplications: 1177,
      Country: 'United States'
    }
    {
      NumberofApplications: 983,
      Country: 'Canada'
    }
    {
      NumberofApplications: 229,
      Country: 'United Kingdom'
    }
```

NOSQL CONTD.

TOP 3 UNIVERSITIES WITH THE HIGHEST AVERAGE SCORE OF ADMITTED STUDENTS



```
> db.applications.aggregate([
    {$match:{"authorize_decision":"admitted"}},
    {$lookup:{from:"user_student", localField:"student_user_id", foreignField:"user_id", as: "user_student"}},
    {$unwind:"$user_student"},
    {$lookup: {from: "university", localField: "university_id", foreignField: "university_id",   as: "university"}},
    {$unwind: "$university"},
    {$group:{_id: "$university.university_name", averageGPA:{$avg:"$user_student.GPA"}}},
    {$project: {university:"$_id", averageGPA: 1, _id:0}},
    {$sort:{averageGPA: -1}},
    {$limit:3}])
< {
    averageGPA: 3.304576271186441,
    university: 'University of Wisconsin - Platteville'
  }
  {
    averageGPA: 3.27093220338983,
    university: 'University of Portsmouth'
  }
  {
    averageGPA: 3.2702142857142857,
    university: 'University of Windsor'
  }
```

THANK YOU