Consolidating Student Management Databases into a Unified Relational Database

John Chung, Daniela Alejandra Gonzalez, Noah Rae-Grant

GitHub Repository

1 Introduction

Our final project addresses a real-world challenge: creating a relational database for an educational institution that currently lacks a DBMS (database management system). The project is grounded in the specific needs and requests of an actual independent high school, referred to as "Academy X" in this report. Academy X aims to better understand trends in student outcomes by leveraging data from its Admissions Office and College Counseling Office.

This project explores the practical challenges of integrating siloed databases and demonstrates the use of SQL queries and visualizations to generate high-value insights for the institution.

2 Literature Review

Our project did not rely on existing research but instead utilized skills and methodologies taught directly in DS5110. Additionally, one group member, a subject matter expert with over a decade of experience at Academy X, provided critical insights. As a result, the SQL queries were designed to address the institution's actual needs and priorities.

3 Methodology

This final project incorporates many of the skills taught throughout the semester in DS5110:

- Entity Relation Diagrams (ERD)
- Data preprocessing and cleaning techniques
- Building a relational database (RDB) in SQLite
- Designing SQL queries
- Data visualization

The data used for this project is based on existing data at Academy X but was modified for confidentiality purposes. The data fields in our RDB mirror the actual existing data fields at *Academy X* as shown in the ERD below, but the actual values needed to be modeled to simulate realistic student data (see section *3.1 Data Collection* for more).

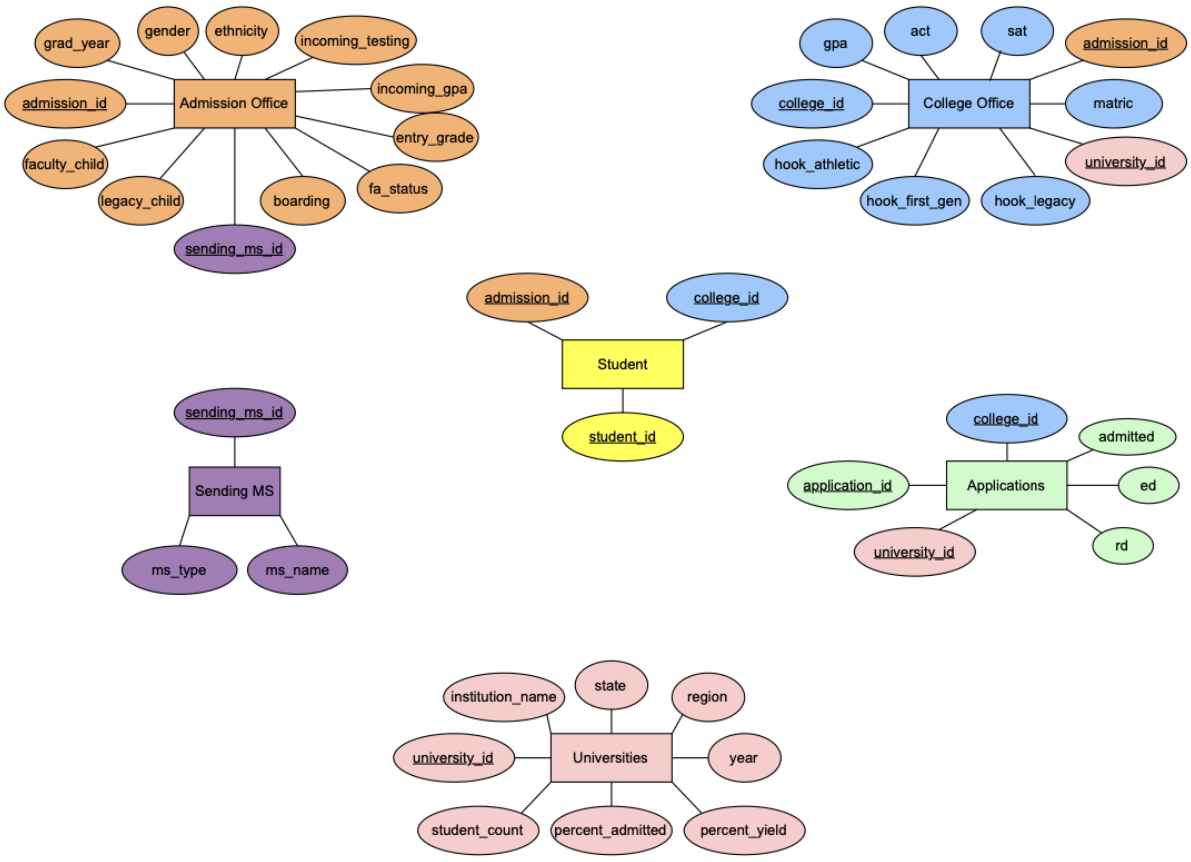


Fig 1. - ERD displaying entities and attributes. `STUDENT` provides the link between incoming (`ADMISSION`) and outgoing (`COLLEGE`) data.

3.1 Data Collection

The data for this project was derived from databases associated with Academy X's Admission and College Counseling offices. The attributes depicted in the ERD in Fig. 1 represent a subset of the actual data fields collected from these offices.

To comply with Academy X's data governance and privacy policies, real student data was not utilized. Instead, data was generated based on predetermined distribution levels, leveraging the expertise and insights of a group member with connections to Academy X.

For example, the `APPLICATIONS` table was designed according to the following specifications:

- Around 20% of the student body should be recruited athletes who only apply to one university.
- The remaining students submit applications to between 1 and 15 unique universities.

The number of applications submitted per student and universities applied to were structured to mimic realistic distributions. Code was developed to generate data satisfying these criteria, incorporating random seeding with preferential bias towards the most popular universities, which represent 78% of applications students submit from Academy X.

For example, the expected distribution of applications by `UNIVERSITIES.region` is outlined in the following table:

region	percent
New England (CT, ME, MA, NH, RI, VT)	39%
Mid East (DE, DC, MD, NJ, NY, PA)	22%
Southeast (AL, AR, FL, GA, KY, LA, MS, NC, SC, TN, VA, WV)	13%
Great Lakes (IL, IN, MI, OH, WI)	9%
Far West (AK, CA, HI, NV, OR, WA)	8%
Plains (IA, KS, MN, MO, NE, ND, SD)	3%
Rocky Mountains (CO, ID, MT, UT, WY)	3%
Southwest (AZ, NM, OK, TX)	3%

The screenshot below provides an excerpt of the corresponding code that adheres to the specified distribution:

```
regions = [
"New England (CT, ME, MA, NH, RI, VT)",
"Mid East (DE, DC, MD, NJ, NY, PA)",
"Southeast (AL, AR, FL, GA, KY, LA, MS, NC, SC, TN, VA, WV)",
"Great Lakes (IL, IN, MI, OH, WI)",
"Far West (AK, CA, HI, NV, OR, WA)",
"Plains (IA, KS, MN, MO, NE, ND, SD)",
"Rocky Mountains (CO, ID, MT, UT, WY)",
"Southwest (AZ, NM, OK, TX)",

]
region_distribution = [.39, .22, .13, .09, .08, .03, .03, .033]

all_the_rest_regions = random.choices(population=regions, weights = region_distribution, k=858)

for region in regions:
    print(f"{region}: {all_the_rest_regions.count(region)}")
```

3.2 Data Preprocessing

Even though our database is ultimately comprised of generated data, the data preprocessing stage of this project highlighted important best practices in database design.

3.2.1 Real-life Challenges: Using `student_name` as Primary Key

The initial phase of this project involved attempting to merge the real databases from Academy X's Admission and College Offices. A significant issue encountered during this process was that both offices used `student_name` as a primary key in their spreadsheets. Neither office followed consistent criteria for formatting names, resulting in mismatches between `ADMISSION.student_name` and `COLLEGE.student_name`.

ADMISSION.student_name	COLLEGE.student_name	match?
Chung, John	Chung, John	TRUE
Rae Grant, Noah	Rae-Grant, Noah	FALSE
Gonzalez, Daniela	Gonzalez, Dani	FALSE

In real-life, both offices retained additional data fields that could be used to validate and reconcile unmatched records, such as:

- Email address
- Graduation year
- Demographics (gender, race/ethnicity, financial aid status)

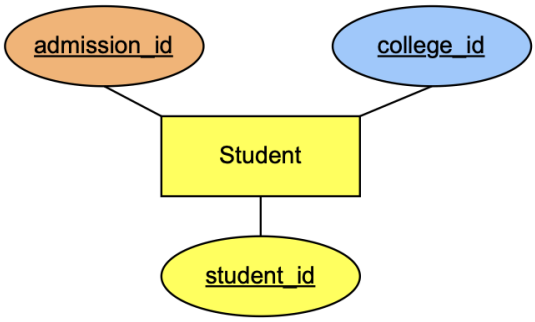
While we ultimately generated our data for privacy reasons, this issue underscores an important best practice in database design: the use of integer-based primary keys to uniquely identify records. To address similar real-life challenges, leveraging secondary fields for data validation (ie, email and graduation year) is a practical workaround to improve data integrity and ensure successful integration across systems.

ADMISSION.email	ADMISSION.student_name	COLLEGE.student_name	COLLEGE.email	name_match?	email_match?
chung.joh@northeastern.edu	Chung, John	Chung, John	chung.joh@northeastern.edu	TRUE	TRUE
raegrant.n@northeastern.edu	Rae Grant, Noah	Rae-Grant, Noah	raegrant.n@northeastern.edu	FALSE	TRUE
gonzalez.daniel@northeastern.edu	Gonzalez, Daniela	Gonzalez, Dani	gonzalez.daniel@northeastern.edu	FALSE	TRUE

3.2.2 Creation of STUDENTS as Junction Table

In order to integrate Academy X's data into a relational database, our group performed the following steps:

1. Added a column of unique integer primary keys into the `ADMISSION` and `COLLEGE` tables.
2. Created `STUDENTS` as a Junction Table that acts as the link between `ADMISSION` and `COLLEGE` tables
3. Validate identities of all students, ensuring that each `ADMISSION.admission_id` corresponds to the correct `COLLEGE.college_id`, removing any entries that do not have a match in both tables.
4. Generate `STUDENTS.student_id` once all student identities have been validated



	A	B	C
1	**student_id**	**college_id**	**admission_id**
2	1005	1	10528
3	1015	2	10532
4	1020	3	16650
5	1047	4	243176
6	1064	5	12785
7	1107	6	243164
8	1113	7	10434
9	1116	8	13429
10	1119	9	15913
11	1125	10	15964

3.2.1 Creation of Relational Database using SQLite

The project began with six raw CSV files named:

- Admission
- Applications
- College
- Sending MS
- Students
- Universities

Each file was processed in Google Colab using the Pandas library to examine data types, identify null values, and detect any discrepancies in the loaded data. Additionally, quality control checks were conducted for boolean values to ensure data integrity. When necessary, updates were made to the CSV files to prepare them for the creation of the relational database.

Below is an example of the code applied to the Applications file:

```
In _  1  applications = pd.read_csv('applications.csv')
      2  applications
```

Out 2

	application_id	student_id	university_id	admitted	ed	rd
0	1	2794	157100	1	1	0
1	2	9799	121257	0	0	1
2	3	2094	130794	0	1	0
3	4	1168	167598	1	0	1
4	5	9905	166027	0	0	1
...	...	...	...	...	..	..
3412	3413	2357	151306	1	0	1
3413	3414	4934	234076	0	0	1
3414	3415	9844	440828	1	0	1
3415	3416	1463	217156	0	0	1
3416	3417	3471	144050	1	0	1

[3417 rows x 6 columns]

```
In _  1  applications.isnull().sum()
```

Out 4

6 rows

	0
application_id	0
student_id	0
university_id	0
admitted	0
ed	0
rd	0

```
In _  1  applications.dtypes
```

Out 5

6 rows

	0
application_id	int64
student_id	int64
university_id	int64
admitted	int64
ed	int64
rd	int64

Once the quality control steps were applied to all the files, the relational Database `School.db` was created. For that we used SQLITE3 in the terminal and the submission of each .csv file was done with the employment of the .mode csv statement.

The code and schema of the RDB :

```
sqlite> .open School.db
sqlite> .schema
CREATE TABLE Admission (
admission_id INTEGER PRIMARY KEY AUTOINCREMENT,
    grad_year INTEGER,
    gender TEXT,
    ethnicity TEXT,
    entry_grade INTEGER,
    FA_status INTEGER,
    incoming_testing INTEGER,
    incoming_gpa FLOAT,
    boarding INTEGER,
    faculty_child INTEGER,
    legacy_child INTEGER,
    sending_ms_id INTEGER,
    FOREIGN KEY (sending_ms_id) REFERENCES Sending_MS(sending_ms_id)
);
CREATE TABLE sqlite_sequence(name,seq);
CREATE TABLE Sending_ms(
sending_ms_id INTEGER PRIMARY KEY AUTOINCREMENT,
ms_name TEXT,
ms_type TEXT
);
CREATE TABLE Universities(
university_id INTEGER PRIMARY KEY AUTOINCREMENT,
institution_name TEXT,
state TEXT,
region TEXT,
year INTEGER,
student_count INTEGER,
percent_admitted INTEGER,
percent_yield INTEGER
);
CREATE TABLE Students(
student_id INTEGER PRIMARY KEY AUTOINCREMENT,
college_id INTEGER,
admission_id INTEGER,
FOREIGN KEY (college_id) REFERENCES College(college_id),
FOREIGN KEY (admission_id) REFERENCES Admission(admission_id)
);
CREATE TABLE Applications(
application_id INTEGER PRIMARY KEY AUTOINCREMENT,
student_id INTEGER,
university_id INTEGER,
admitted INTEGER,
ed INTEGER,
rd INTEGER,
FOREIGN KEY (student_id) REFERENCES Students(student_id),
FOREIGN KEY (university_id) REFERENCES University(university_id)
);
CREATE TABLE College (
college_id INTEGER PRIMARY KEY AUTOINCREMENT,
admission_id INTEGER,
gpa REAL,
sat REAL,
act REAL,
matric TEXT NOT NULL,
university_id INTEGER,
hook_athlete INTEGER,
hook_first_gen INTEGER,
hook_legacy INTEGER,
FOREIGN KEY (admission_id) REFERENCES Admission(admission_id),
FOREIGN KEY (university_id) REFERENCES Universities(university_id)
);
```

3.3 Analysis Techniques

The project employed data preprocessing techniques and database management tools to create and analyze the `School.db` database. The approach is detailed as follows:

Data preprocessing was carried out using Python libraries Pandas and NumPy, which facilitated the cleaning and organization of data extracted from multiple CSV files. These tools enabled the handling of missing values, the standardization of column formats, and the validation of data integrity, ensuring that the data was prepared for database integration.

The database was created and managed using SQLite3, selected for its simplicity and flexibility. A database schema was designed to represent the relationships between entities such as students, courses, and grades. Using SQL, the structure of the database was defined with `CREATE TABLE` statements, preprocessed data was inserted into the tables, and relationships were established through `FOREIGN KEY` constraints. Additional techniques, including indexing and data validation, were applied to optimize query performance and ensure reliability.

Data analysis was performed through SQL queries designed to extract meaningful insights. These queries included aggregating metrics related to student performance, identifying patterns in grades and course enrollments, and executing complex joins across multiple tables to address analytical questions. For the visualization of the query results, Seaborn and Matplotlib were also employed.

This project effectively combined SQL for database management and Python for preprocessing and analysis, resulting in a robust and efficient data pipeline.

4 Results

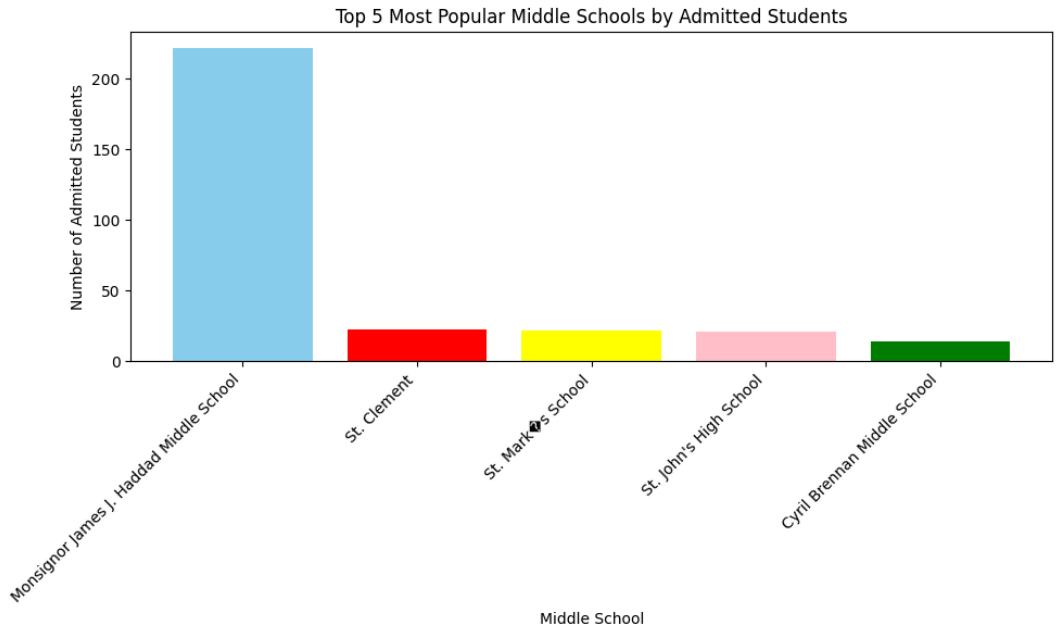
For certain queries that were multi-part, we've included only the output that resulted in a figure. The full results of those queries can be found in our GitHub repository.

Query 1: What are the top 5 most popular Middle Schools and how many admitted students did each send?

Query

```
SELECT Sending_ms.ms_name, COUNT(DISTINCT Admission.admission_id)
AS admitted_students_count
FROM Sending_ms
JOIN Admission ON Sending_ms.sending_ms_id =
Admission.sending_ms_id
JOIN College ON Admission.admission_id = College.admission_id
JOIN Students ON College.college_id = Students.college_id
JOIN Applications ON Students.student_id = Applications.student_id
WHERE Applications.admitted = 1
GROUP BY Sending_ms.ms_name
ORDER BY admitted_students_count DESC
LIMIT 5;
```

Result

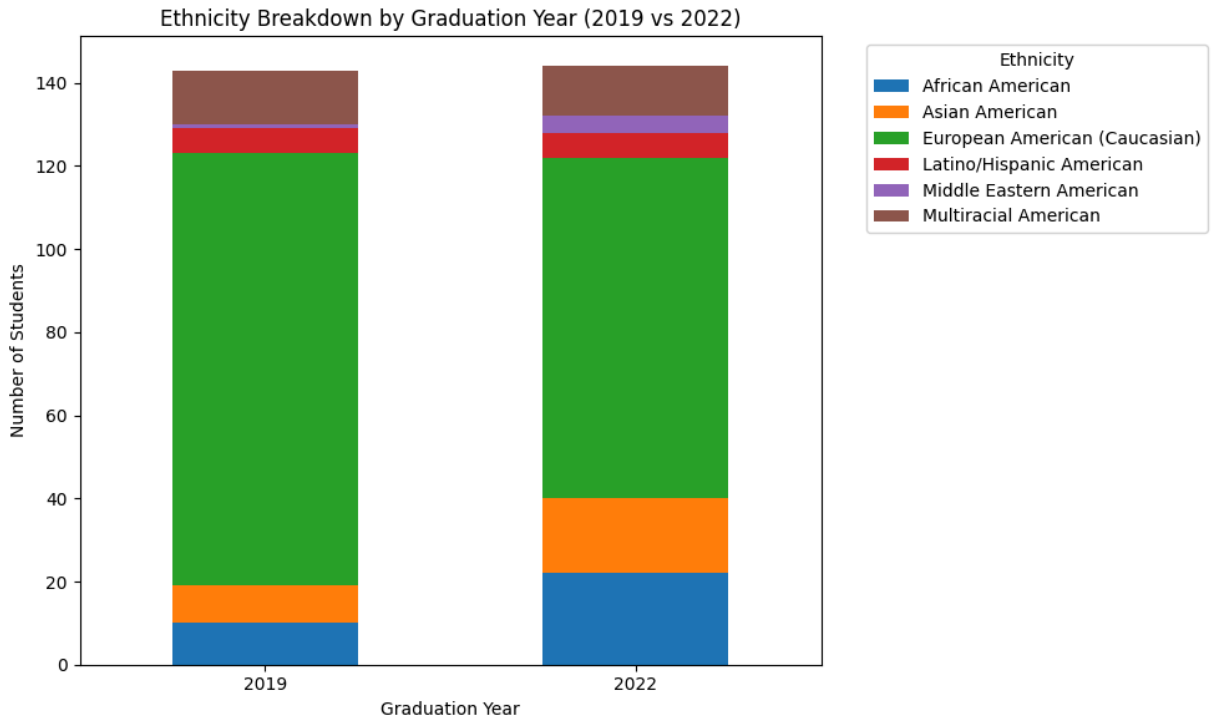


Query 2: What is the ethnicity breakdown of each graduating class? (Part of a multi-part question)

Query

```
SELECT
      Admission.grad_year,
      Admission.ethnicity,
      COUNT(Students.student_id) AS student_count
FROM
      Students
INNER JOIN
      Admission
ON
      Students.admission_id = Admission.admission_id
GROUP BY
      Admission.grad_year, Admission.ethnicity
ORDER BY
      Admission.grad_year, Admission.ethnicity;
```

Result (This query had multiple plots associated with it, but for brevity's sake we're only including the comparison chart between 2019 and 2022.)



Query 3: How does the ethnicity breakdown compare between students who are on 90%+ financial aid vs. students who are full pay? (Part of a multi-part question)

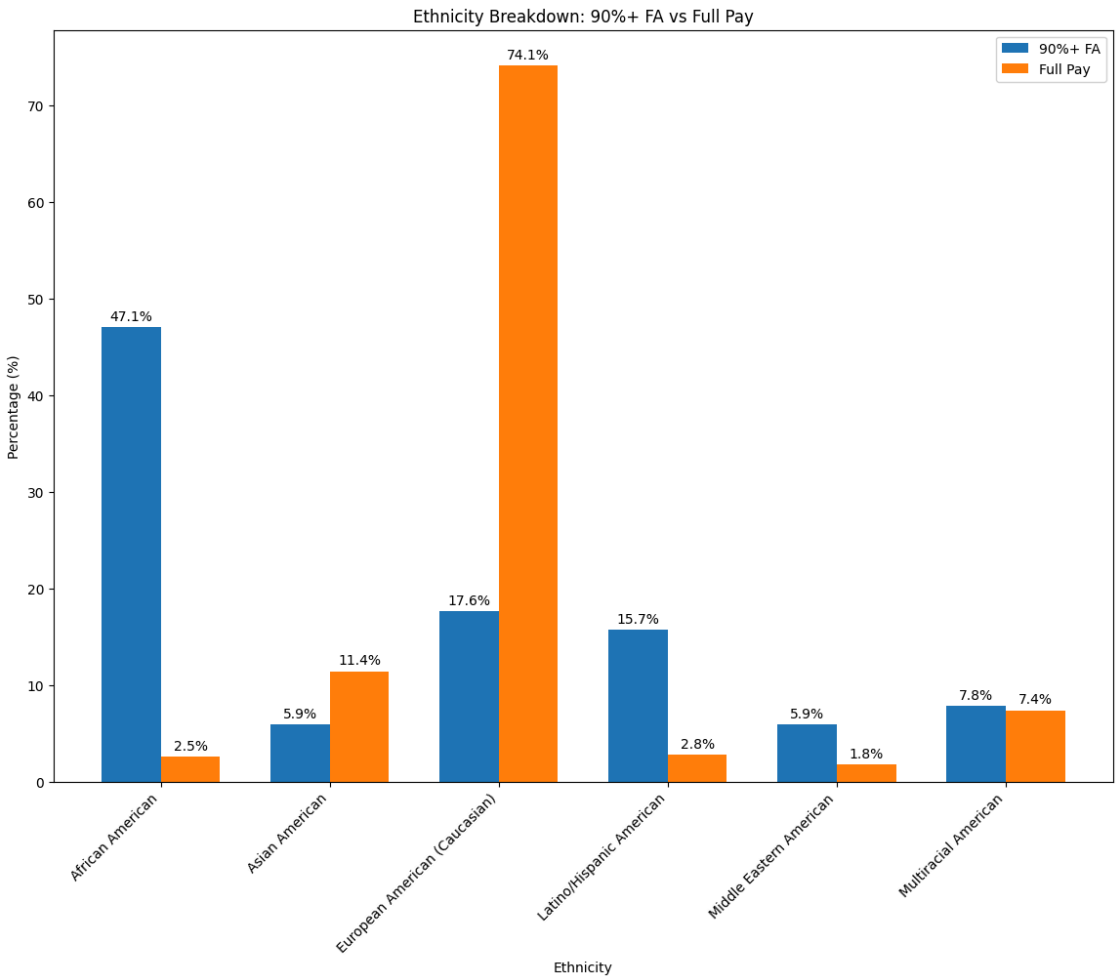
Query 1

```
SELECT Admission.ethnicity, COUNT(*) AS student_count
FROM Students
INNER JOIN Admission ON Students.admission_id = Admission.admission_id
WHERE Admission.FA_status = 2
GROUP BY Admission.ethnicity;
```

Query 2

```
SELECT Admission.ethnicity, COUNT(*) AS student_count
FROM Students
INNER JOIN Admission ON Students.admission_id = Admission.admission_id
WHERE Admission.FA_status = 0
GROUP BY Admission.ethnicity;
```

Result



Query 4: Return a list of all students who earned an A- GPA or higher
Note: Academy X uses an 11-point GPA scale; a GPA of 10 or greater corresponds to an A- or higher.

Query

```
SELECT *
FROM College
WHERE gpa >= 10.0;
```

Result (Head of the dataframe; full results in our GitHub repository)

	college_id	admission_id	gpa	sat	act	matric	university_id
0	1	10528	11.00	1580.0	35.0	University of Southern Maine	161554
1	2	10532	10.95	1520.0	0.0	American International College	164447
2	3	16650	10.93	1550.0	0.0	Saint Elizabeth School of Nursing	152497
3	4	243176	10.92	1580.0	0.0	Tufts University	168148
4	5	12785	10.90	1570.0	0.0	Boston College	164924

Query 5: Return a list of all students who were recruited athletes grouped by graduation year. Include the university name that the student ended up matriculating at and whether the student was an early admit or not.

Query

```
SELECT
      Admission.grad_year,
      Students.student_id,
      Universities.institution_name AS university_name,
      CASE
      WHEN College.hook_athlete = 'True' THEN 'Yes'
      ELSE 'No'
      END AS recruited_athlete,
      CASE
      WHEN Applications.ed = 1 THEN 'Yes'
      ELSE 'No'
      END AS early_admit
FROM Students
INNER JOIN
      Admission ON Students.admission_id = Admission.admission_id
INNER JOIN
      College ON Students.college_id = College.college_id
INNER JOIN
      Universities ON College.university_id = Universities.university_id
INNER JOIN
      Applications ON Students.student_id = Applications.student_id
WHERE
      College.hook_athlete = 'True'
GROUP BY
      Admission.grad_year, Students.student_id,
Universities.institution_name, Applications.ed;
```

Result (Head of the dataframe; full results in our GitHub repository)

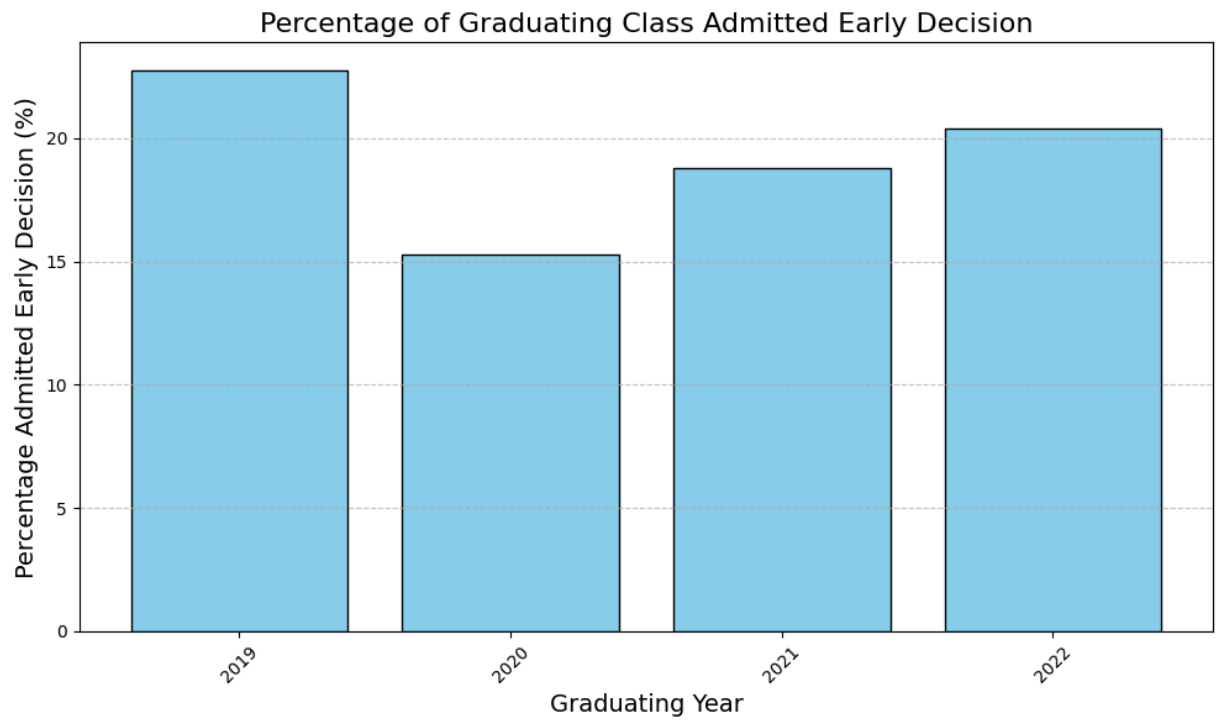
	grad_year	student_id	university_name	recruited_athlete	early_admit
0	2019	7447	Princeton University	Yes	No
1	2019	7447	Princeton University	Yes	Yes
2	2019	7518	Southeastern College-Charleston	Yes	No
3	2019	7518	Southeastern College-Charleston	Yes	Yes
4	2019	7554	Williams College	Yes	Yes

Query 6: What percentage of each graduating class was admitted early decision?

Query

```
SELECT
     Admission.grad_year AS graduating_year,
     COUNT(DISTINCT CASE
     WHEN Applications.ed = 1 AND Applications.admitted = 1 THEN
Applications.application_id
     END) * 1.0 /
     COUNT(DISTINCT CASE
     WHEN Applications.admitted = 1 THEN Applications.application_id
     END) * 100 AS percentage_early_decision_admitted
FROM
     Admission
JOIN
     Students ON Admission.admission_id = Students.admission_id
JOIN
     Applications ON Students.student_id = Applications.student_id
GROUP BY
     Admission.grad_year
ORDER BY
     Admission.grad_year
```

Result

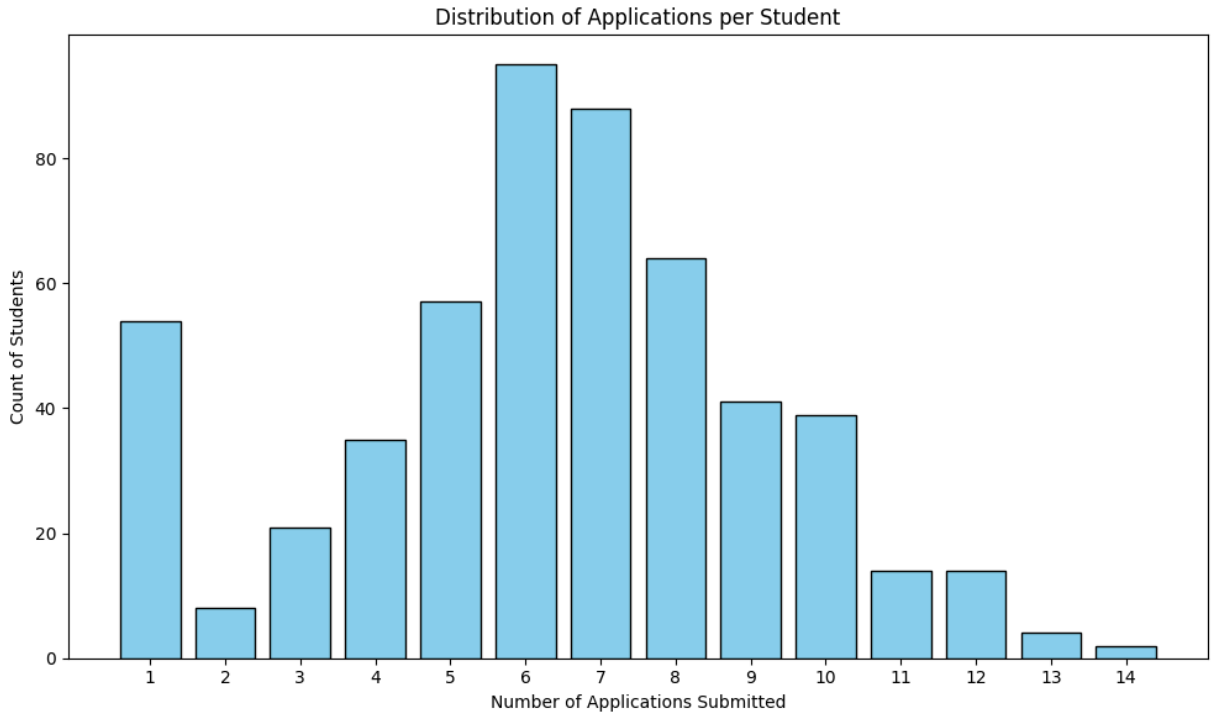


Query 7: Count the number of submitted applications per student. Create a bar graph of the distribution of the number of applications per student

Query

```
WITH temp_apps AS (
      SELECT
      DISTINCT Applications.student_id,
      COUNT(Applications.student_id) AS App_Count
      FROM Applications
      GROUP BY Applications.student_id)
SELECT DISTINCT App_Count, COUNT(App_Count) AS Num_Students
FROM temp_apps
GROUP BY App_Count;
```

Result

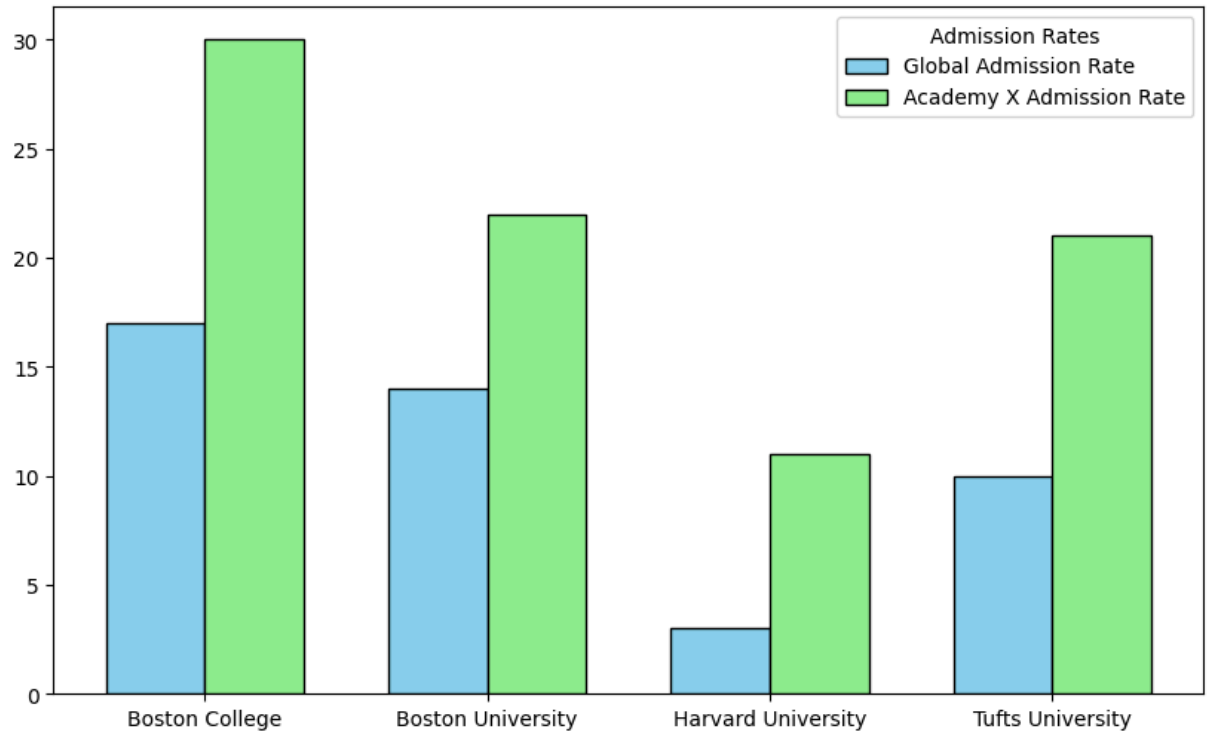


Query 8: Compare Academy X's admission rate into popular Boston Universities and compare this to each University's global admit rate

Query (Note: an "academy_percent_admitted" column was added with Pandas immediately after this query. This was calculated by dividing admitted by applications_count.)

```
SELECT
  Universities.institution_name,
  Universities.percent_admitted,
  COUNT(Applications.application_id) AS applications_count,
  COUNT(
      CASE WHEN Applications.admitted == '1' THEN 1
      END) AS admitted
FROM Applications
JOIN Universities ON Applications.university_id =
Universities.university_id
WHERE Universities.institution_name IN ('Harvard University', 'Boston
College', 'Tufts University', 'Boston University')
GROUP BY Universities.institution_name;
```

Result

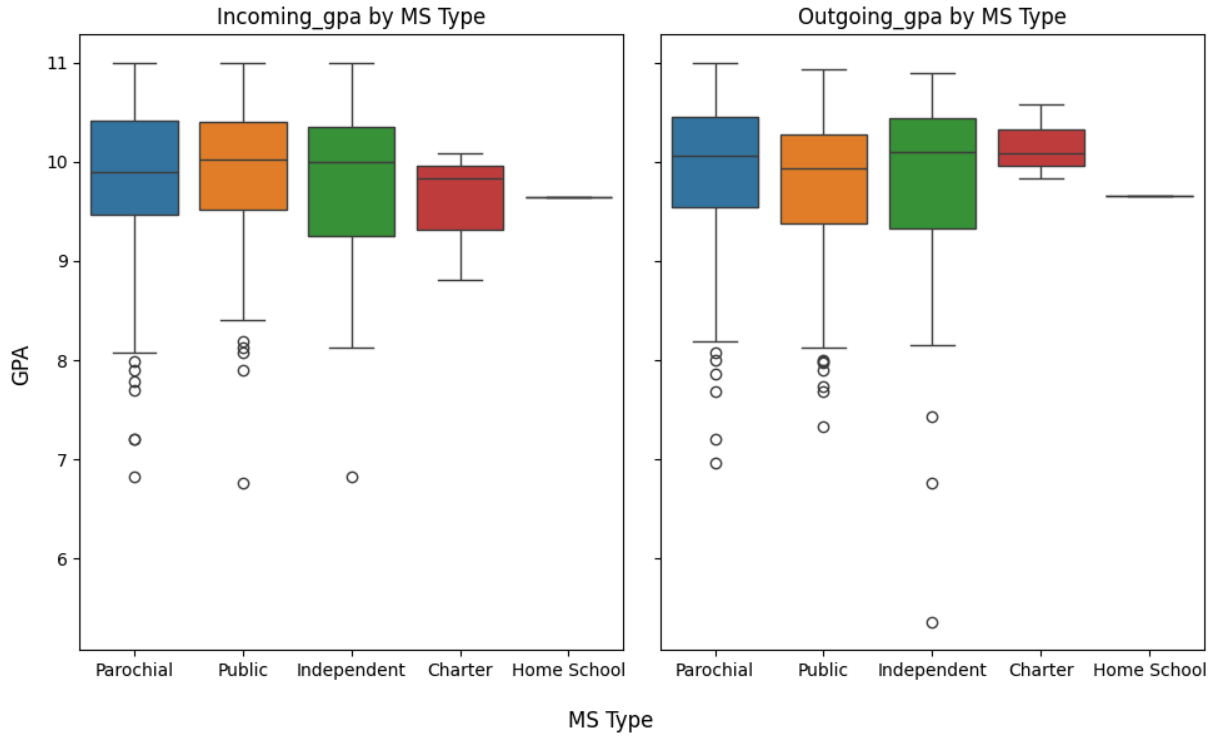


Query 9: Create a pair of boxplots comparing incoming and outgoing GPA grouped by MS type

Query

```
SELECT
  Admission.incoming_gpa,
  College.gpa AS outgoing_gpa,
  Sending_ms.ms_type
FROM Admission
JOIN Student ON Admission.admission_id
    = Student.admission_id
JOIN College ON Student.college_id
    = College.college_id
JOIN Sending_ms ON Admission.sending_ms_id
    = Sending_ms.sending_ms_id;
```

Result



5 Discussion

According to Query 1, the analysis focuses on identifying the top five middle schools based on the number of admitted students. The objective is to determine which schools contribute the most to the admissions pool and examine the potential implications of these results. The SQL query counts unique admissions per middle school, sorts them in descending order, and selects the top five. The findings reveal that Monsignor James J. Haddad MS stands out significantly, with over 200 admitted students, far surpassing the other schools on the list. This suggests that the school might have a larger student body, a higher level of academic preparation, or perhaps a closer relationship with the admitting institution. In comparison, the other schools, such as St. Clement and St. Mary's School, show much lower numbers of admitted students, with a more uniform distribution among them. This disparity highlights potential inequalities in terms of educational resources or opportunities across middle schools. The disproportionate representation of Monsignor James J. Haddad MS may be driven by factors such as curriculum quality, school size, or even demographic or geographic biases in the admissions process. On the other hand, schools with lower representation might face challenges related to academic preparation or access to resources. These results call for further investigation to understand the reasons behind this imbalance. Future analysis could consider admission rates relative to the total student population of each school, as well as external factors such as socioeconomic context or specific admission policies.

In Query 2, the analysis examines the ethnicity breakdown of each graduating class, with a specific focus on the years 2019 and 2022 for comparison. The SQL query groups students by graduation year and ethnicity, counting the number of students in each category. The chart reveals a consistent distribution of ethnic groups between the two years, with some minor variations. European American (Caucasian) students represent the largest group in both 2019 and 2022, followed by Latino/Hispanic American and Asian American students. Other ethnic groups, such as African American and Multiracial American, maintain smaller but stable proportions. The minimal changes between these years suggest that the ethnic composition of the graduating classes has remained relatively unchanged over time. This stability could indicate a consistent admissions policy and student demographic, but it also raises questions about whether the institution is actively pursuing greater diversity or if barriers to representation exist for certain ethnic groups. Understanding these aspects could provide valuable insights into the institution's efforts toward equity and inclusion.

According to Query 3, the analysis compares the ethnicity breakdown of students receiving 90%+ financial aid with those who are full pay. The chart reveals disparities between the two groups. Among students receiving 90%+ financial aid, African American (47.1%) and Latino/Hispanic American (32.8%) students dominate, collectively comprising the majority of this category. In contrast, the full pay category is overwhelmingly represented by European American (Caucasian) students, who account for 74.1% of this group. Ethnic groups such as Asian American, Middle Eastern American, and Multiracial American maintain relatively low and consistent representation across both categories. These findings suggest a strong correlation between ethnicity and financial aid status, highlighting significant socioeconomic disparities among the student population. The overrepresentation of African American and Latino/Hispanic students in the high financial aid category may reflect systemic inequalities in income and wealth distribution, while the dominance of European American students in the full pay category underscores their relative economic privilege. These results emphasize the

importance of financial aid programs in supporting diversity and equity within the student body. Without such programs, many African American and Latino/Hispanic students might face barriers to accessing higher education.

According to Query 4, the objective is to retrieve a list of students who achieved a GPA of 10.0 or higher, which corresponds to an A- or better on Academy X's 11-point GPA scale. The results provide key details such as the GPA, standardized test scores (SAT and ACT), and the universities where these students were admitted or enrolled. The results demonstrate a strong correlation between academic excellence, as reflected by GPA, and access to prestigious universities. Students with GPAs above 10.0 are more likely to gain admission to highly selective institutions, underscoring the importance of strong academic performance in the admissions process. Additionally, the presence of high-achieving students at a range of institutions, including less selective ones such as the University of Southern Maine and American International College, suggests that opportunities for high performers are not limited to elite universities. These findings emphasize the critical role of academic achievement in shaping educational opportunities. Further investigation could examine these additional variables to provide a more comprehensive understanding of the admissions process and the pathways available for high-achieving students.

Query 5 identifies students who were recruited athletes, grouped by graduation year, and provides details on the universities they matriculated at and whether they were early admitted. The data highlights that recruited athletes often matriculate at prestigious institutions like Princeton University and Williams College. Notably, athletic recruitment appears to give students a significant advantage in the admissions process, even for those not admitted through early decision. These findings suggest that athletic recruitment strongly influences admissions decisions, often favoring athletic ability over other qualifications. This raises questions about equity in the admissions process. Further analysis could explore trends in athletic recruitment and its impact on diversity and fairness within higher education.

Query 6 analyzes the percentage of each graduating class admitted through early decision. The results show a gradual increase in the percentage of early decision admits from 2019 to 2022, suggesting a growing reliance on early decision admissions over time. This trend indicates that early decision pathways may be becoming a more common method of admission, potentially benefiting students who can commit early but possibly disadvantaging those who need more time to compare financial aid offers. Further analysis could explore the demographics and academic profiles of early decision admits to assess the equity and accessibility of this admissions option.

Query 7 calculates the distribution of the number of college applications submitted per student. The resulting bar chart shows that most students submitted between 4 and 7 applications, with a noticeable peak at 6 applications. A smaller proportion of students submitted fewer than 4 or more than 10 applications, with very few submitting as many as 12 or more. This distribution suggests that submitting around 6 applications is common, likely reflecting a balance between maximizing options and managing application efforts and costs. The lower numbers at both extremes indicate that while some students may focus on a small number of specific schools or cast a wider net, these strategies are less common. This analysis could inform counseling strategies by emphasizing the benefits and challenges of different application volumes, helping students optimize their approach to college admissions.

Query 8 compares Academy X's admission rate to the global admission rates of four popular Boston universities: Harvard University, Boston College, Boston University, and Tufts University. The results indicate that Academy X students generally have higher admission rates compared to the global rates at these universities. For instance, Academy X's admission rate for Boston College is significantly higher than its global rate, and the same trend is observed for Boston University and Tufts University. However, for Harvard University, both the Academy X and global admission rates remain very low, reflecting the highly competitive nature of admissions at this institution. This analysis highlights the potential advantage Academy X students have in gaining admission to these universities, possibly due to strong academic preparation, targeted application strategies, or existing relationships between Academy X and these institutions. The disparity between Academy X and global rates at less selective universities, such as Boston College and Boston University, suggests that Academy X students may be particularly competitive in these applicant pools. Conversely, the limited difference at Harvard underscores the challenge of gaining admission even for well-prepared students. These findings could guide Academy X's future efforts in counseling students on university selection and application strategies. Further exploration could assess whether similar trends exist with other institutions or examine the factors contributing to the elevated admission rates for Academy X students.

Query 9 compares incoming and outgoing GPAs of students grouped by their middle school (MS) type using boxplots. The boxplots show that incoming GPAs are generally higher and have less variability compared to outgoing GPAs across most MS types. For instance, students from independent and parochial middle schools tend to have higher incoming GPAs, suggesting strong academic preparation. However, the outgoing GPA distributions reveal a broader range, with some students experiencing significant declines during high school, while others perform exceptionally well. Notably, home school students display the most consistent GPAs between admission and graduation, albeit with fewer data points. These results suggest that middle school type may influence initial academic preparedness (incoming GPA) but is less predictive of high school performance (outgoing GPA), as all groups exhibit wide variability by graduation. This could highlight the impact of factors beyond middle school type, such as the college environment, resources, or personal circumstances, on students' academic trajectories. Further analysis could explore the correlation between incoming and outgoing GPAs within each MS type or investigate additional variables, such as support systems or extracurricular engagement, to better understand the drivers of high school success.

6 Conclusion

This project provided key insights into the academic and demographic trends of students at Academy X, leveraging SQL queries to analyze various aspects of the admissions process and academic performance. The findings revealed significant patterns, including the influence of middle school type on incoming GPA, the advantage of early decision and athletic recruitment in admissions, and disparities in financial aid distribution by ethnicity. Additionally, Academy X students demonstrated higher admission rates at selective universities compared to global averages, highlighting their strong academic preparation. The analysis also showed that while middle school type impacts initial preparedness, college performance is influenced by a wider range of factors, emphasizing the importance of support systems during higher education.

However, the project faced some limitations. The analysis relied on the available datasets, which may not fully capture external factors like socioeconomic background, extracurricular activities, or geographic disparities that influence academic outcomes and admissions decisions. Furthermore, the focus on aggregate data may obscure individual-level variations and nuanced trends within subgroups.

Future research could address these limitations by integrating additional data sources, such as student essays, demographic profiles, and regional contexts. Longitudinal studies tracking students' progress beyond college could provide deeper insights into the factors driving long-term success. Additionally, expanding the analysis to include a broader range of institutions and admission policies could offer a more comprehensive view of systemic trends and opportunities for reform in higher education. This would enable a more holistic understanding of promoting equity and success for all students.