Q: “How many entries do you have in your database who have applied for Fall 2025?”
A: Applicant count: 6634

SQL:

```
SELECT COUNT(*) as count FROM admissions_info
WHERE year=2025 AND season='fall';
```

Very simple count operation across rows with year 2025 and fall season.

Q: “What percentage of entries are from international students?”
A: Percent international: 47.83%

SQL:

```
SELECT
  intl_student_count * 100.0 / total as pct
FROM (
  SELECT
    COUNT(*) AS total,
    COUNT(*) FILTER (WHERE us_or_international='international') as intl_student_count
  FROM admissions_info
) AS intl_students;
```

Uses a subquery to fetch both the total and the filtered count of international students, then select the average.

Q: “What is the average GPA, GRE, GRE V, GRE AW of applicants who provide these metrics?”
A: GPA: 3.79, GRE: 178.64, GRE Verbal: 159.66, GRE AW: 6.68

SQL:

```
SELECT
 AVG(gpa) as avg_gpa,
 AVG(gre) as avg_gre,
 AVG(gre_v) as avg_gre_v,
 AVG(gre_aw) as avg_gre_aw
FROM admissions_info;
```

Straightforward AVG function used to aggregate all of the data in the columns we are interested in. Postgres will ignore rows that are NULL so they won’t pollute our average.

Q: “What is the average GPA of American students in Fall 2025?”
A: Average GPA: 3.77

SQL:

```
SELECT
 AVG(gpa) as avg_gpa
FROM admissions_info
WHERE year=2025
 AND season='fall'
 AND us_or_international != 'international';
```

Uses an AVG across the GPA column, primarily filtered on rows *not equal* to “international”. This feels more robust as TheGradCafe is a US-based website.

Q: “What percent of entries for Fall 2025 are Acceptances?”
A: Percent accepted: 35.92%

SQL:

```
SELECT
 accepted * 100.0 / total as pct
FROM (
 SELECT
   COUNT(*) AS total,
   COUNT(*) FILTER (WHERE status='accepted') as accepted
 FROM admissions_info
 WHERE year=2025 AND season='fall'
) AS fall_2025_students;
```

Very similar to the previous percent calculation. This SQL uses a subquery to fetch both the total and a filtered count. This time the subquery is also filtered on the term, giving us a narrow lens on the rows we are interested in.

Q: “What is the average GPA of applicants who applied for Fall 2025 who are Acceptances?”
A: Average GPA: 3.76

SQL:

```
SELECT
 AVG(gpa) as avg_gpa
FROM admissions_info
WHERE status='accepted' AND year=2025 AND season='fall';
```

Simple AVG function over a query with a WHERE clause set up with the restrictions we want.

Q: “How many entries are from applicants who applied to JHU for a masters degrees in Computer Science?”
A: Applicant count: 3

SQL:

```
SELECT
 COUNT(*)
FROM admissions_info
WHERE
 degree='masters'
 AND llm_generated_university='Johns Hopkins University'
 AND llm_generated_program='Computer Science';
```

Basic count operation, but we are using the LLM-generated columns in order to have more accurate results.

Q: “How many entries from 2025 are acceptances from applicants who applied to Georgetown University for a PhD in Computer Science?”
A: Applicant count: 0

SQL:

```
SELECT
 COUNT(*)
FROM admissions_info
WHERE
 degree='phd'
 AND llm_generated_university='George Town University'
 AND llm_generated_program='Computer Science'
 AND year=2025
 AND status='accepted';
```

Same as above, but with a whole bunch of filters. This query is very specific so we get no results. Also note the LLM interpreted the school name as “George Town”.

Q: “What is the average GPA for students accepted to UCLA vs USC?”
A: UCLA average GPA: 3.93, USC average GPA: 3.96

SQL:

```
SELECT
 AVG(gpa) FILTER (WHERE llm_generated_university=%s) as avg_gpa_ucla,
 AVG(gpa) FILTER (WHERE llm_generated_university=%s) as avg_gpa_usc
FROM admissions_info
WHERE
 status='accepted';
```

These two schools have a famous rivalry. This query asks for two averages in the SELECT, each one filtered on a different school name.

Q: “What is the average GRE for students in the past 4 years?”
A: 2021 average GRE: 166.10, 2022 average GRE: 167.88, 2023 average GRE: 166.89, 2024 average GRE: 168.06

SQL:

```
SELECT
 AVG(gre) FILTER (WHERE year=2021) as avg_gre_2021,
 AVG(gre) FILTER (WHERE year=2022) as avg_gre_2022,
 AVG(gre) FILTER (WHERE year=2023) as avg_gre_2023,
 AVG(gre) FILTER (WHERE year=2024) as avg_gre_2024
FROM admissions_info;
```

Looking for trends in our dataset, this query calculates many averages across a range of years.