Introduction to SQL

Slides adapted from Prof. Heimann CMU-P

What is SQL?

What is SQL?

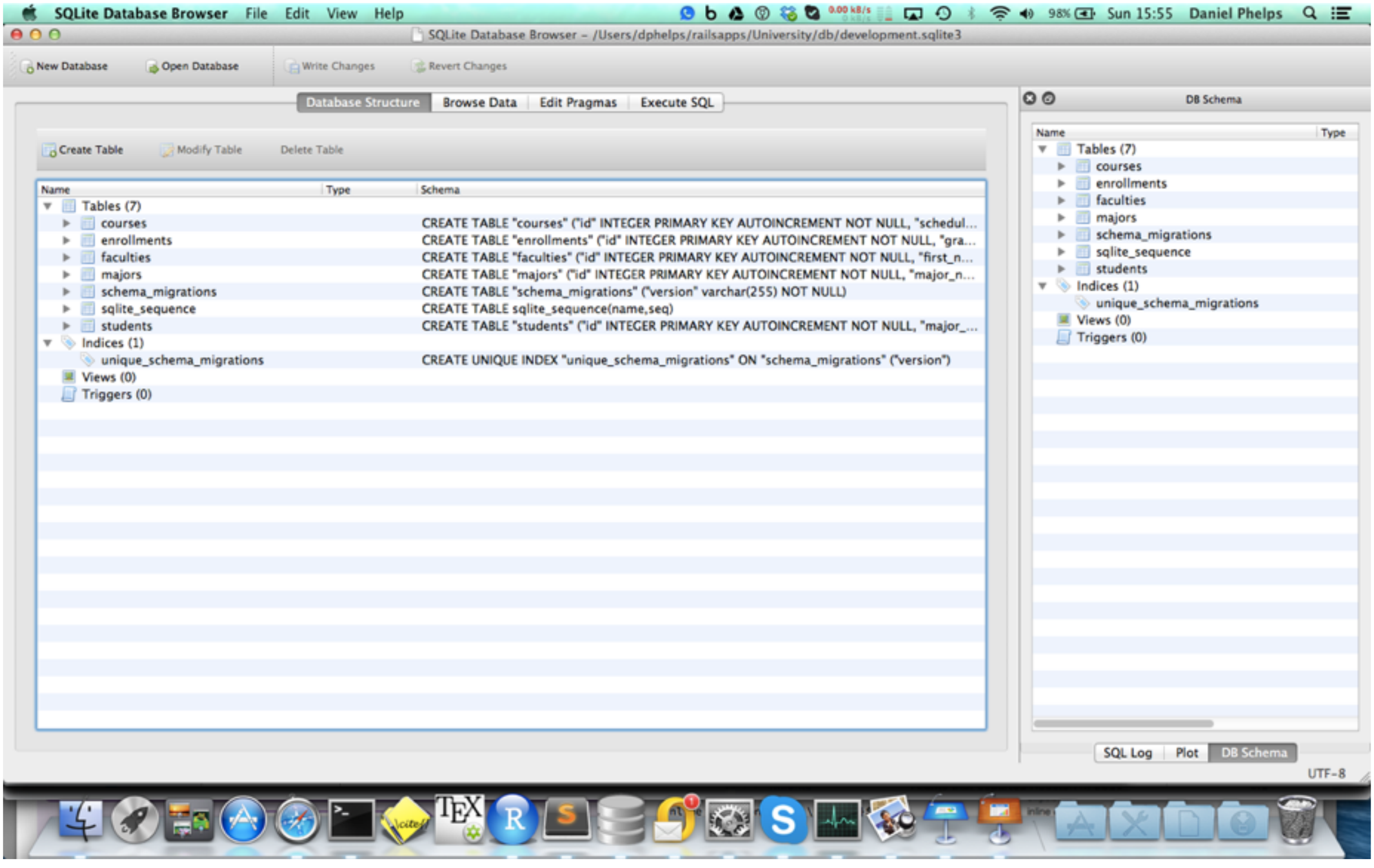
- “Structured Query Language (SQL), pronounced "sequel", is a language that provides an interface to relational database systems. It was developed by IBM in the 1970s for use in System R. SQL is a de facto standard, as well as an ISO and ANSI standard.” — *definition according to Oracle*

What is SQL?

- “Structured Query Language (SQL), pronounced "sequel", is a language that provides an interface to relational database systems. It was developed by IBM in the 1970s for use in System R. SQL is a de facto standard, as well as an ISO and ANSI standard.” — *definition according to Oracle*

- Allows for:
  - Data Extraction (`SELECT`)
  - Data Manipulation (`INSERT, UPDATE, DELETE`)
  - Data Definition (`CREATE, DROP, TRUNCATE`)
  - Data Control (`GRANT, REVOKE`)

Accessing SQLite



Queries: meet the clauses

Queries: meet the clauses

Order written

Queries: meet the clauses

Order written

- SELECT

Queries: meet the clauses

Order written

- SELECT
- FROM

Queries: meet the clauses

Order written

- SELECT
- FROM
- WHERE

Queries: meet the clauses

Order written

- SELECT
- FROM
- WHERE
- GROUP BY

Queries: meet the clauses

Order written

- SELECT
- FROM
- WHERE
- GROUP BY
- HAVING

Queries: meet the clauses

Order written

- SELECT
- FROM
- WHERE
- GROUP BY
- HAVING
- ORDER BY

Queries: meet the clauses

Order written

- SELECT
- FROM
- WHERE
- GROUP BY
- HAVING
- ORDER BY
- LIMIT

Queries: meet the clauses

Order written

- SELECT
- FROM
- WHERE
- GROUP BY
- HAVING
- ORDER BY
- LIMIT

Order processed

Queries: meet the clauses

Order written

- SELECT
- FROM
- WHERE
- GROUP BY
- HAVING
- ORDER BY
- LIMIT

Order processed

- FROM
- WHERE
- GROUP BY
- HAVING
- SELECT
- ORDER BY
- LIMIT

courses(id, faculty_id, schedule, code)

enrollments(id, student_id, course_id, grade)

faculties(id, first_name, last_name, department)

majors(id, major_name)

students(id, major_id, stu_lastname, stu_firstname, credits)

Working through some basic queries

Working through some basic queries

- List all records in the students table

Working through some basic queries

- List all records in the students table

```
SELECT * FROM students
```

More queries

More queries

- List all records in the students table for IS majors only

More queries

- List all records in the students table for IS majors only

```
SELECT * FROM students WHERE major_id = 2;
```

More queries

- List all records in the students table for IS majors only

```
SELECT * FROM students WHERE major_id = 2;
```

```
SELECT * FROM students
WHERE major_id =
 (SELECT id FROM majors WHERE major_name = 'IS');
```

More queries

- List all students, name only – in alphabetical order, together with their credits earned

```
SELECT stu_lastname, stu_firstname, credits
FROM students
ORDER BY stu_lastname;
```

```
SELECT CONCAT(stu_lastname, stu_firstname), credits
FROM students
ORDER BY stu_lastname;
```

More queries

- List all students, name only – in alphabetical order, together with their credits earned

```
SELECT stu_lastname, stu_firstname, credits
FROM students
ORDER BY stu_lastname;
```

More queries

- List all students, name only – in alphabetical order, together with their credits earned

```
SELECT stu_lastname, stu_firstname, credits
FROM students
ORDER BY stu_lastname;
```

```
SELECT (stu_lastname || stu_firstname), credits
FROM students
ORDER BY stu_lastname;
```

More queries

More queries

- List all students, other than IS majors (in alphabetical order) having more than 50 credits, together with their credits earned

More queries

- List all students, other than IS majors (in alphabetical order) having more than 50 credits, together with their credits earned

```
SELECT (stu_lastname || ", " || stu_firstname) AS Name,
credits
FROM students
WHERE credits > 50
AND major_id <>
 (SELECT id FROM majors WHERE major_name = 'IS')
ORDER BY stu_lastname;
```

More queries

More queries

- List all students (in alphabetical order) having between 25 and 50 credits, together with their credits earned

More queries

- List all students (in alphabetical order) having between 25 and 50 credits, together with their credits earned

```
SELECT (stu_lastname || " " || stu_firstname) AS Name,
credits
FROM students
WHERE credits >= 25 AND credits <= 50
ORDER BY stu_lastname;
```

More queries

- List all students (in alphabetical order) having between 25 and 50 credits, together with their credits earned

```
SELECT (stu_lastname || " " || stu_firstname) AS Name,
credits
FROM students
WHERE credits >= 25 AND credits <= 50
ORDER BY stu_lastname;
```

```
SELECT (stu_lastname || " " || stu_firstname) AS Name,
credits
FROM students
WHERE credits BETWEEN 25 AND 50
ORDER BY stu_lastname;
```

More queries

- List all students (in alphabetical order) having between 25 and 50 credits, together with their credits earned

```
SELECT (stu_lastname || " " || stu_firstname) AS Name,
credits
FROM students
WHERE credits >= 25 AND credits <= 50
ORDER BY stu_lastname;
```

```
SELECT (stu_lastname || " " || stu_firstname) AS Name,
credits
FROM students
WHERE credits BETWEEN 25 AND 50
ORDER BY stu_lastname;
```

More queries

More queries

- List all courses that meet on Thursdays

More queries

- List all courses that meet on Thursdays

```
SELECT *
FROM courses
WHERE schedule LIKE '%Th%' OR schedule LIKE 'Th%'
```

More queries

More queries

- List all courses and corresponding grades taken by Kaylin Rice

More queries

- List all courses and corresponding grades taken by Kaylin Rice

```
SELECT course_id, grade
FROM enrollments
WHERE student_id =
   (SELECT id
FROM students
WHERE stu_firstname = 'Kaylin'
AND stu_lastname = 'Rice');
```

More queries

- List all courses and corresponding grades taken by Kaylin Rice

```
SELECT course_id, grade
FROM enrollments
WHERE student_id =
   (SELECT id
FROM students
WHERE stu_firstname = 'Kaylin'
AND stu_lastname = 'Rice');
```

More queries

More queries

- List all courses taught by Enos Stokes

More queries

- List all courses taught by Enos Stokes

```
SELECT cl.code
FROM courses AS cl, faculties AS fa
WHERE fa.id = cl.faculty_id
  AND fa.first name = 'Enos'
  AND fa.last_name = 'Stokes'
ORDER BY cl.code;
```

More queries

- List all courses taught by Enos Stokes

```
SELECT cl.code
FROM courses AS cl, faculties AS fa
WHERE fa.id = cl.faculty_id
  AND fa.first name = 'Enos'
  AND fa.last_name = 'Stokes'
ORDER BY cl.code;
```

More queries

More queries

- List all instructors, by name and corresponding department, who have to teach a class on Tuesdays

More queries

- List all instructors, by name and corresponding department, who have to teach a class on Tuesdays

```
SELECT (fa.first_name || " " || fa.last_name) AS "Faculty
Name", fa.department AS Department
FROM courses AS cl, faculties AS fa
WHERE fa.id = cl.faculty_id
  AND cl.schedule LIKE '%Tu%'
  OR cl.schedule LIKE 'Tu%'
ORDER BY fa.last_name;
```

More queries

- List all instructors, by name and corresponding department, who have to teach a class on Tuesdays

```
SELECT (fa.first_name || " " || fa.last_name) AS "Faculty
Name", fa.department AS Department
FROM courses AS cl, faculties AS fa
WHERE fa.id = cl.faculty_id
  AND cl.schedule LIKE '%Tu%'
  OR cl.schedule LIKE 'Tu%'
ORDER BY fa.last_name;
```

More queries

More queries

- List all instructors, by name and corresponding department, who have to teach a class on Tuesdays, but only list them once.

More queries

- List all instructors, by name and corresponding department, who have to teach a class on Tuesdays, but only list them once.

```
SELECT DISTINCT (fa.first_name || " " || fa.last_name) AS
"Faculty Name", fa.department AS Department
FROM courses AS cl, faculties AS fa
WHERE fa.id = cl.faculty_id
  AND cl.schedule LIKE '%Tu%'
  OR cl.schedule LIKE 'Tu%'
ORDER BY fa.last_name;
```

More queries

- List all instructors, by name and corresponding department, who have to teach a class on Tuesdays, but only list them once.

```
SELECT DISTINCT (fa.first_name || " " || fa.last_name) AS
"Faculty Name", fa.department AS Department
FROM courses AS cl, faculties AS fa
WHERE fa.id = cl.faculty_id
  AND cl.schedule LIKE '%Tu%'
  OR cl.schedule LIKE 'Tu%'
ORDER BY fa.last_name;
```

More queries

- List all instructors, by name and corresponding department, who have to teach a class on Tuesdays, but only list them once.

```
SELECT DISTINCT (fa.first_name || " " || fa.last_name) AS
"Faculty Name", fa.department AS Department
FROM courses AS cl, faculties AS fa
WHERE fa.id = cl.faculty_id
  AND cl.schedule LIKE '%Tu%'
  OR cl.schedule LIKE 'Tu%'
ORDER BY fa.last_name;
```

More queries

More queries

- List all teachers, in alphabetical order, and their respective departments for the student Malika Dare

More queries

- List all teachers, in alphabetical order, and their respective departments for the student Malika Dare

```
SELECT ( fa.first_name || " " || fa.last_name ) AS
"Faculty Name", fa.department AS Department
FROM courses AS cl, faculties AS fa, students AS st,
enrollments AS en
WHERE fa.id = cl.faculty_id
  AND st.id = en.student_id
  AND cl.id = en.course_id
  AND st.stu_firstname = 'Malika'
  AND st.stu_lastname = 'Dare'
ORDER BY fa.last_name;
```

More queries

- List all teachers, in alphabetical order, and their respective departments for the student Malika Dare

```
SELECT ( fa.first_name || " " || fa.last_name ) AS
"Faculty Name", fa.department AS Department
FROM courses AS cl, faculties AS fa, students AS st,
enrollments AS en
WHERE fa.id = cl.faculty_id
  AND st.id = en.student_id
  AND cl.id = en.course_id
  AND st.stu_firstname = 'Malika'
  AND st.stu_lastname = 'Dare'
ORDER BY fa.last_name;
```

More queries

More queries

- List all students, by name (no duplicates) for which there is an enrollment record

More queries

- List all students, by name (no duplicates) for which there is an enrollment record

```
SELECT( st.stu_lastname || ", " || st.stu_firstname ) AS
"Student Name"
FROM students AS st
WHERE st.id IN
  (SELECT DISTINCT student_id FROM enrollments)
ORDER BY st.stu_lastname;
```

More queries

- List all students, by name (no duplicates) for which there is an enrollment record

```
SELECT( st.stu_lastname || ", " || st.stu_firstname ) AS
"Student Name"
FROM students AS st
WHERE st.id IN
  (SELECT DISTINCT student_id FROM enrollments)
ORDER BY st.stu_lastname;
```

More queries

More queries

- List all faculty, by name and ID, who have no courses (those slackers!)

More queries

- List all faculty, by name and ID, who have no courses (those slackers!)

```
QUERY 1:
```

More queries

- List all faculty, by name and ID, who have no courses (those slackers!)

```
QUERY 1:

SELECT DISTINCT fa.id AS "Faculty ID",
  ( fa.first_name || " " || fa.last_name ) AS
  "Faculty Member"
FROM faculties AS fa
WHERE fa.id NOT IN
  (SELECT DISTINCT faculty_id FROM courses)
ORDER BY fa.last_name;
```

More queries

- List all faculty, by name and ID, who have no courses (those slackers!)

```
QUERY 1:

SELECT DISTINCT fa.id AS "Faculty ID",
  ( fa.first_name || " " || fa.last_name ) AS
  "Faculty Member"
FROM faculties AS fa
WHERE fa.id NOT IN
  (SELECT DISTINCT faculty_id FROM courses)
ORDER BY fa.last_name;
```

More queries

More queries

- List all faculty, by name and ID, who have no courses (... another way to find those lazy professors)

More queries

- List all faculty, by name and ID, who have no courses (... another way to find those lazy professors)

```
QUERY 2:
```

More queries

- List all faculty, by name and ID, who have no courses (... another way to find those lazy professors)

```
QUERY 2:

SELECT DISTINCT fa.id AS "Faculty ID",
  ( fa.first_name || " " || fa.last_name ) AS
  "Faculty Member"
FROM faculties AS fa
  LEFT JOIN courses AS cl
  ON fa.id = cl.faculty_id
WHERE cl.id IS NULL
ORDER BY fa.last_name;
```

More queries

- List all faculty, by name and ID, who have no courses (... another way to find those lazy professors)

```
QUERY 2:

SELECT DISTINCT fa.id AS "Faculty ID",
  ( fa.first_name || " " || fa.last_name ) AS
  "Faculty Member"
FROM faculties AS fa
  LEFT JOIN courses AS cl
  ON fa.id = cl.faculty_id
WHERE cl.id IS NULL
ORDER BY fa.last_name;
```

More queries

More queries

- List all students, by name (no duplicates) for which there is an enrollment record

More queries

- List all students, by name (no duplicates) for which there is an enrollment record

```
SELECT DISTINCT ( st.stu_lastname || ", " ||
st.stu_firstname ) AS "Student Name"
FROM students AS st
WHERE st.id IN
  (SELECT DISTINCT student_id FROM enrollments)
ORDER BY st.stu_lastname, st.stu_firstname
```

More queries

- List all students, by name (no duplicates) for which there is an enrollment record

```
SELECT DISTINCT ( st.stu_lastname || ", " ||
st.stu_firstname ) AS "Student Name"
FROM students AS st
WHERE st.id IN
  (SELECT DISTINCT student_id FROM enrollments)
ORDER BY st.stu_lastname, st.stu_firstname
```

More queries

More queries

- List all students, in alphabetical order, not taking a political science (code POL) class

More queries

- List all students, in alphabetical order, not taking a political science (code POL) class

```
SELECT (st.stu_lastname || ", " || st.stu_firstname) AS
"Student Name"
FROM students AS st
WHERE st.id NOT IN
   (SELECT student_id FROM enrollments
    WHERE course_id LIKE 'POL%')
ORDER BY st.stu_lastname, st.stu_firstname;
```

More queries

- List all students, in alphabetical order, not taking a political science (code POL) class

```
SELECT (st.stu_lastname || ", " || st.stu_firstname) AS
"Student Name"
FROM students AS st
WHERE st.id NOT IN
   (SELECT student_id FROM enrollments
    WHERE course_id LIKE 'POL%')
ORDER BY st.stu_lastname, st.stu_firstname;
```

More queries

More queries

- Show the number of courses taken by Kavon Grimes

More queries

- Show the number of courses taken by Kavon Grimes

```
SELECT COUNT(*) AS "Number of Courses"
FROM enrollments
WHERE student_id =
   (SELECT id FROM students
    WHERE stu_firstname = 'Kavon' AND stu_lastname =
'Grimes');
```

More queries

- Show the number of courses taken by Kavon Grimes

```
SELECT COUNT(*) AS "Number of Courses"
FROM enrollments
WHERE student_id =
   (SELECT id FROM students
    WHERE stu_firstname = 'Kavon' AND stu_lastname =
'Grimes');
```

More queries

- Show the number of courses taken by Kavon Grimes

```
SELECT COUNT(*) AS "Number of Courses"
FROM enrollments
WHERE student_id =
   (SELECT id FROM students
    WHERE stu_firstname = 'Kavon' AND stu_lastname =
'Grimes');
```

More queries

More queries

- List all courses for which two or more students are enrolled together with the number of students

More queries

- List all courses for which two or more students are enrolled together with the number of students

```
SELECT en.course_id, COUNT(en. course_id) AS "# Enrolled"
FROM enrollments AS en
GROUP BY en.course_id
HAVING COUNT(en.course_id) >= 2
ORDER BY en.course_id;
```

More queries

- List all courses for which two or more students are enrolled together with the number of students

```
SELECT en.course_id, COUNT(en. course_id) AS "# Enrolled"
FROM enrollments AS en
GROUP BY en.course_id
HAVING COUNT(en.course_id) >= 2
ORDER BY en.course_id;
```

More queries

- List all courses for which two or more students are enrolled together with the number of students

```
SELECT en.course_id, COUNT(en. course_id) AS "# Enrolled"
FROM enrollments AS en
GROUP BY en.course_id
HAVING COUNT(en.course_id) >= 2
ORDER BY en.course_id;
```

More queries

More queries

- List all math courses (code MSF) for which students are enrolled together with the number of students

More queries

- List all math courses (code MSF) for which students are enrolled together with the number of students

```
SELECT en.course_id, COUNT(en.course_id) AS "# Enrolled"
FROM enrollments AS en, courses as cl
WHERE en.course_id=cl.id AND cl.code LIKE 'MSF%'
GROUP BY en.course_id
HAVING COUNT(en. course_id) > 0
ORDER BY en.course_id;
```

More queries

- List all math courses (code MSF) for which students are enrolled together with the number of students

```
SELECT en.course_id, COUNT(en.course_id) AS "# Enrolled"
FROM enrollments AS en, courses as cl
WHERE en.course_id=cl.id AND cl.code LIKE 'MSF%'
GROUP BY en.course_id
HAVING COUNT(en. course_id) > 0
ORDER BY en.course_id;
```

More queries

More queries

- Give the average number of credits for IS majors

More queries

- Give the average number of credits for IS majors

```
SELECT ROUND(AVG(st.Credits),2) AS "Avg IS student
credits"
FROM students AS st
WHERE major_id = (SELECT id FROM majors WHERE major_name =
'IS');
```

More queries

- Give the average number of credits for IS majors

```
SELECT ROUND(AVG(st.Credits),2) AS "Avg IS student
credits"
FROM students AS st
WHERE major_id = (SELECT id FROM majors WHERE major_name =
'IS');
```

More queries

- Give the average number of credits for IS majors

```
SELECT ROUND(AVG(st.Credits),2) AS "Avg IS student
credits"
FROM students AS st
WHERE major_id = (SELECT id FROM majors WHERE major_name =
'IS');
```

More queries

More queries

- Name the student (there may be more than one) with the most number of credits

More queries

- Name the student (there may be more than one) with the most number of credits

```
SELECT (st.stu_lastname || ", " || st.stu_firstname) AS
Student, st.credits AS Credits
FROM students AS st
WHERE st.credits =
   (SELECT MAX(credits) FROM students);
```

More queries

- Name the student (there may be more than one) with the most number of credits

```
SELECT (st.stu_lastname || ", " || st.stu_firstname) AS
Student, st.credits AS Credits
FROM students AS st
WHERE st.credits =
   (SELECT MAX(credits) FROM students);
```

More queries

- Name the student (there may be more than one) with the most number of credits

```
SELECT (st.stu_lastname || ", " || st.stu_firstname) AS
Student, st.credits AS Credits
FROM students AS st
WHERE st.credits =
   (SELECT MAX(credits) FROM students);
```

More queries

More queries

- List all courses (even those with zero enrollment) together with the enrollment

More queries

- List all courses (even those with zero enrollment) together with the enrollment

```
SELECT cl.id AS Course, COUNT(en.course_id) AS Enrollment
FROM courses AS cl
  LEFT JOIN enrollments AS en
  ON cl.id = en.course_id
GROUP BY en.course_id
ORDER BY cl.id;
```

More queries

- List all courses (even those with zero enrollment) together with the enrollment

```
SELECT cl.id AS Course, COUNT(en.course_id) AS Enrollment
FROM courses AS cl
  LEFT JOIN enrollments AS en
  ON cl.id = en.course_id
GROUP BY en.course_id
ORDER BY cl.id;
```

More queries

More queries

- List all students by name, in alphabetical order, together with their student IDs and the number of courses they have taken

More queries

- List all students by name, in alphabetical order, together with their student IDs and the number of courses they have taken

```
SELECT (st.stu_lastname || ", " || st.stu_firstname)
  AS Student, st.id AS ID,
  COUNT(en.student_id) AS "# Courses Taken"
FROM students AS st
  LEFT JOIN enrollments AS en
  ON st.id = en.student_id
GROUP BY st.id
ORDER BY st.stu_lastname;
```

More queries

- List all students by name, in alphabetical order, together with their student IDs and the number of courses they have taken

```
SELECT (st.stu_lastname || ", " || st.stu_firstname)
  AS Student, st.id AS ID,
  COUNT(en.student_id) AS "# Courses Taken"
FROM students AS st
  LEFT JOIN enrollments AS en
  ON st.id = en.student_id
GROUP BY st.id
ORDER BY st.stu_lastname;
```

More queries

More queries

- List all students by name who have not taken a class

More queries

- List all students by name who have not taken a class

```
SELECT DISTINCT (st.stu_lastname || ", " ||
st.stu_firstname) AS 'Students with no courses'
FROM students AS st
  LEFT JOIN enrollments AS en
  ON st.id = en.student_id
WHERE en.course_id IS NULL
ORDER BY st.stu_lastname;
```

More queries

- List all students by name who have not taken a class

```
SELECT DISTINCT (st.stu_lastname || ", " ||
st.stu_firstname) AS 'Students with no courses'
FROM students AS st
  LEFT JOIN enrollments AS en
  ON st.id = en.student_id
WHERE en.course_id IS NULL
ORDER BY st.stu_lastname;
```