SQL Refresher

Professor Houda Bouamor
Application Design & Development
Information Systems Program

Based on an example by Prof. Larry Heimann

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`)

Queries: meet the clauses

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

The University Database as an example

University Database Design in 3NF

faculties (id, firstname, lastname, department,rank)

courses (id, course_name, schedule, faculty_id)

majors (id, major_name)

students (id, firstname, lastname, credits, major_id)

enrollments (student_id, course_id, grade)

Working through some basic queries

- List all records in the students table

```
SELECT * FROM students;
```

More queries

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

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

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

More queries

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

```
SELECT lastname, firstname, credits
FROM students
ORDER BY lastname;
```

```
SELECT CONCAT(lastname, firstname), credits
FROM students
ORDER BY lastname;
```

```
SELECT CONCAT(lastname, ", ", firstname) AS Name, credits
FROM students
ORDER BY lastname;
```

More queries

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

```
SELECT CONCAT(lastname, ", ", firstname) AS Name, credits
FROM students
WHERE credits > 50
AND major_id <>
 (SELECT id FROM majors WHERE major_name = 'Math')
ORDER BY lastname;
```

More queries

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

```
SELECT CONCAT(firstname, " ", lastname) AS Name, credits
FROM students
WHERE credits >= 25 AND credits <= 50
ORDER BY lastname;
```

```
SELECT CONCAT(firstname, " ", lastname) AS Name, credits
FROM students
WHERE credits BETWEEN 25 AND 50
ORDER BY lastname;
```

More queries

- List all students that have a lastname that starts with “a”

```
SELECT *
FROM students
WHERE lastname LIKE 'a%'
```

- List all students that have a lastname that ends with “a”

```
SELECT *
FROM students
WHERE lastname LIKE '%a'
```

More queries

- List all students, by name, in the “Government and Politics” course (ID: POL103A), together with their grades

```
SELECT CONCAT(lastname, ", ", firstname) AS Student, grade
FROM enrollments, students
WHERE students.id = enrollments.student_id
  AND course_id = 'POL103A'
ORDER BY lastname;
```

More queries

- List all courses and corresponding grades taken by student Hasan Khan

```
SELECT course_id, grade
FROM enrollments
WHERE student_id =
   (SELECT id
FROM students
WHERE firstname = 'Hasan'
AND lastname = 'Khan');
```

More queries: Using Aliases

- List all courses taught by Chadi Aoun

```
SELECT co.course_id
FROM courses AS co, faculty AS fa
WHERE fa.id = co.faculty_id
  AND fa.firstname = 'Chadi'
  AND fa.lastname = 'Aound'
ORDER BY co.course_id;
```

More queries

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

```
SELECT CONCAT(fa.firstname, " ", fa.lastname) AS "Faculty
Name", fa.department AS Department
FROM courses AS co, faculty AS fa
WHERE fa.id = co.faculty_id
  AND co.schedule LIKE '%W%'
  OR co.schedule LIKE 'W%'
ORDER BY fa.lastname;
```

More queries

- List all teachers, in alphabetical order, and their respective departments for the student Richard Heath

```
SELECT CONCAT( fa.firstname, " ", fa.lastname ) AS
"Faculty Name", fa.department AS Department
FROM classes AS co, faculty AS fa, students AS st,
enrollments AS en
WHERE fa.id = co.faculty_id
  AND st.id = en.student_id
  AND co.id = en.course_id
  AND st.firstname = 'Richard'
  AND st.lastname = 'Heath'
ORDER BY fa.lastname;
```

More queries

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

```
SELECT CONCAT( st.lastname, ", ", st.firstname ) AS
"Student Name"
FROM students AS st
WHERE st.id IN
   (SELECT DISTINCT student_id FROM enrollments)
ORDER BY st.lastname;
```

More queries

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

```
QUERY 1:

SELECT DISTINCT fa.id AS "Faculty ID",
  CONCAT( fa.firstname, " ", fa.lastname ) AS
  "Faculty Member"
FROM faculty AS fa
WHERE fa.id NOT IN
  (SELECT DISTINCT faculty_id FROM courses)
ORDER BY fa.lastname;
```

More queries

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

QUERY 2:

```
SELECT DISTINCT fa.id AS "Faculty ID",
  CONCAT( fa.firstname, " ", fa.lastname ) AS
  "Faculty Member"
FROM faculty AS fa
  LEFT JOIN courses AS co
  ON fa.id = co.faculty_id
WHERE co.id IS NULL
ORDER BY fa.lastname;
```

More queries

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

```
SELECT DISTINCT CONCAT( st.lastname, ", ",
  st.firstname ) AS "Student Name"
FROM students AS st
WHERE st.id IN
  (SELECT DISTINCT student_id FROM enrollments)
ORDER BY st.lastname, st.firstname
```

More queries

- List all students, in alphabetical order, not taking a political science class

```
SELECT CONCAT(st.lastname, ", ",
   st.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.lastname, st.firstname;
```

More queries

- Show the number of courses taken by Ann Chin

```
SELECT COUNT(*) AS "Number of Courses"
FROM enrollments
WHERE student_id =
   (SELECT id FROM students
   WHERE firstname = 'Ann' AND lastname = 'Chin');
```

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 math courses 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
WHERE en.course_id LIKE 'MTH%'
GROUP BY en.course_id
HAVING COUNT(en.course_id) > 0
ORDER BY en.course_id;
```

More queries

- Give the average number of credits for math majors

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

More queries

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

```
SELECT CONCAT(st.lastname, ", ", st.firstname) AS Student,
st.credits AS Credits
FROM students AS st
WHERE st.credits =
   (SELECT MAX(credits) FROM students);
```

More queries

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

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

More queries

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

```
SELECT CONCAT(st.lastname, ", ", st.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 who have not taken a class

```
SELECT DISTINCT CONCAT(st.lastname, ", ",
  st.firstname) AS 'Students with no classes'
FROM students AS st
  LEFT JOIN enrollments AS en
  ON st.id = en.student_id
WHERE en.course_id IS NULL
ORDER BY st.lastname;
```

More queries

- List all students, name only – in alphabetical order, together with their credits earned. Replace any NULL values with ‘N/A’.

```
SELECT CONCAT(lastname, ", ", firstname) AS Name,
  CASE WHEN credits IS NULL THEN 'N/A'
  ELSE credits END AS Credits
FROM students
ORDER BY lastname;
```