

Introduction to SQL

Miao Qiao

The University of Auckland



Basic Query Structure

- A typical SQL query has the form:

select A_1, A_2, \dots, A_n
from r_1, r_2, \dots, r_m
where P

<i>department</i>	<i>instructor</i>	<i>teaches</i>	<i>section</i>	<i>takes</i>	<i>student</i>	<i>course</i>	<i>advisor</i>
<u><i>dept_name</i></u> <i>building</i> <i>budget</i>	<u><i>ID</i></u> <i>name</i> <i>dept_name</i> <i>salary</i>	<u><i>ID</i></u> <u><i>course_id</i></u> <u><i>sec_id</i></u> <u><i>semester</i></u> <u><i>year</i></u>	<u><i>course_id</i></u> <u><i>sec_id</i></u> <u><i>semester</i></u> <u><i>year</i></u> <i>building</i> <i>room_number</i> <i>time_slot_id</i>	<u><i>ID</i></u> <u><i>course_id</i></u> <u><i>sec_id</i></u> <u><i>semester</i></u> <u><i>year</i></u> <i>grade</i>	<u><i>ID</i></u> <i>name</i> <i>dept_name</i> <i>tot_cred</i>	<u><i>course_id</i></u> <i>title</i> <i>dept_name</i> <i>credits</i>	<u><i>s_id</i></u> <u><i>i_id</i></u>

Basic Query Structure (demo)

1. Find the names of all instructors
2. Find the department names of all instructors, and remove duplicates
3. Find the department names of all instructors, not removing duplicates
4. Find all attributes of instructors
5. Find a relation that is the same as the *instructor* relation, except that the value of the attribute *salary* is divided by 12
6. Find all instructors in Comp. Sci. dept with salary > 70000
7. Find the names of all instructors who have taught some course and the course_id
8. Find the names of all instructors in the Art department who have taught some course and the course_id
9. Find the names of all instructors who have a higher salary than some instructor in 'Comp. Sci'.

		<i>teaches</i>		<i>section</i>	<i>takes</i>		<i>course</i>	<i>advisor</i>
<i>department</i>	<i>instructor</i>	<u><i>ID</i></u>	<u><i>course_id</i></u>	<u><i>course_id</i></u>	<u><i>ID</i></u>	<u><i>course_id</i></u>	<u><i>course_id</i></u>	
<u><i>dept_name</i></u>	<u><i>ID</i></u>	<i>course_id</i>	<i>sec_id</i>	<i>semester</i>	<i>course_id</i>	<i>sec_id</i>	<i>title</i>	<u><i>s_id</i></u>
<i>building</i>	<i>name</i>	<i>sec_id</i>	<i>year</i>	<i>building</i>	<i>sec_id</i>	<i>semester</i>	<i>dept_name</i>	<i>i_id</i>
<i>budget</i>	<i>dept_name</i>	<i>semester</i>	<i>room_number</i>	<i>room_number</i>	<i>year</i>	<i>grade</i>	<i>credits</i>	
	<i>salary</i>	<i>year</i>	<i>time_slot_id</i>	<i>time_slot_id</i>				

Basic Query Structure (demo)

- Find the names of all instructors whose name includes the substring “in”.
- String Operations
 - The operator **like** uses patterns that are described using two special characters:
 - percent (%). The % character matches any substring.
 - underscore (_). The _ character matches any character
- Find the names of all instructors whose name has 4 characters.
- Find the names of all instructors whose name has at least 4 characters.

				section				
department	instructor	teaches			takes	student	course	
<u>dept_name</u>	<u>ID</u>	<u>ID</u>	<u>course_id</u>	<u>course_id</u>	<u>ID</u>	<u>ID</u>	<u>course_id</u>	
building	name	<u>sec_id</u>	<u>semester</u>	<u>sec_id</u>	<u>course_id</u>	name	title	advisor
	dept_name	<u>year</u>	<u>building</u>	<u>year</u>	<u>sec_id</u>	dept_name	dept_name	<u>s_id</u>
	salary	<u>year</u>	<u>room_number</u>	<u>time_slot_id</u>	<u>semester</u>	tot_cred	credits	<u>i_id</u>
					<u>year</u>			
					grade			

Basic Query Structure (demo)

1. List in alphabetic order the names of all instructors
2. List in descending alphabetic order the names of all instructors
3. List in order of the combination of the names and salary of all instructors
4. Find the names of all instructors with salary between \$90,000 and \$100,000
5. Find courses that ran in Fall 2017 or in Spring 2018
6. Find courses that ran in Fall 2017 and in Spring 2018
7. Find courses that ran in Fall 2017 but not in Spring 2018
8. Find courses that ran in Fall 2017 or in Spring 2018, retain all duplications
9. Find all instructors whose salary is null
10. Find all instructors whose salary is not null
11. Null under and, or, with true/false

		<i>teaches</i>		<i>section</i>	<i>takes</i>		<i>course</i>	<i>advisor</i>
<i>department</i>	<i>instructor</i>	<u><i>ID</i></u>	<u><i>course_id</i></u>	<u><i>course_id</i></u>	<u><i>ID</i></u>	<u><i>course_id</i></u>	<u><i>course_id</i></u>	
<u><i>dept_name</i></u>	<u><i>ID</i></u>	<i>sec_id</i>	<i>semester</i>	<i>sec_id</i>	<i>course_id</i>	<i>sec_id</i>	<i>title</i>	<u><i>s_id</i></u>
<i>building</i>	<i>name</i>	<i>year</i>	<i>building</i>	<i>semester</i>	<i>sec_id</i>	<i>year</i>	<i>dept_name</i>	<i>i_id</i>
<i>budget</i>	<i>dept_name</i>	<i>year</i>	<i>room_number</i>	<i>year</i>	<i>grade</i>	<i>tot_cred</i>	<i>credits</i>	

Basic Query Structure (demo)

- These functions operate on the multiset of values of a column of a relation, and return a value

avg: average value

min: minimum value

max: maximum value

sum: sum of values

count: number of values

- Find the average salary of instructors in the Computer Science department
- Find the total number of instructors who teach a course in the Spring 2018 semester
- Find the number of tuples in the *course* relation

		<i>teaches</i>		<i>section</i>	<i>takes</i>		<i>course</i>	<i>advisor</i>
<i>department</i>	<i>instructor</i>	<u><i>ID</i></u>	<u><i>course_id</i></u>	<u><i>course_id</i></u>	<u><i>ID</i></u>	<u><i>course_id</i></u>	<u><i>course_id</i></u>	
<u><i>dept_name</i></u>	<u><i>ID</i></u>	<u><i>sec_id</i></u>	<u><i>year</i></u>	<u><i>sec_id</i></u>	<u><i>sec_id</i></u>	<u><i>semester</i></u>	<u><i>title</i></u>	<u><i>s_id</i></u>
<i>building</i>	<i>name</i>	<i>semester</i>	<i>building</i>	<i>room_number</i>	<i>year</i>	<i>grade</i>	<i>dept_name</i>	<i>i_id</i>
<i>budget</i>	<i>dept_name</i>	<i>year</i>	<i>time_slot_id</i>				<i>credits</i>	
	<i>salary</i>							

Aggregate – Group By - Having

- Find the average salary of instructors in each department
- Find the names and average salaries of all departments whose average salary is greater than 42000
 - Note: predicates in the **having** clause are applied after the formation of groups whereas predicates in the **where** clause are applied before forming groups

<i>department</i>	<i>instructor</i>	<i>teaches</i>	<i>section</i>	<i>takes</i>	<i>student</i>	<i>course</i>	<i>advisor</i>
<u>dept_name</u> building budget	<u>ID</u> name dept_name salary	<u>ID</u> <u>course_id</u> <u>sec_id</u> <u>semester</u> <u>year</u>	<u>course_id</u> <u>sec_id</u> <u>semester</u> <u>year</u> building room_number time_slot_id	<u>ID</u> <u>course_id</u> <u>sec_id</u> <u>semester</u> <u>year</u> grade	<u>ID</u> name dept_name tot_cred	<u>course_id</u> title dept_name credits	<u>s_id</u> <u>i_id</u>

Nested Subqueries

- A **subquery** is a **select-from-where** expression that is nested within another query.
- The nesting can be done in the following SQL query

```
select A1, A2, ..., An
from r1, r2, ..., rm
where P
```

as follows:

- **From clause:** r_i can be replaced by any valid subquery
- **Where clause:** P can be replaced with an expression of the form:

$B <\text{operation}> (\text{subquery})$

B is an attribute and $<\text{operation}>$ to be defined later.

- **Select clause:**

A_i can be replaced by a subquery that generates a single value.

		teaches		section	takes		student	course	advisor	
department	instructor	<u>ID</u>	<u>course_id</u>	<u>course_id</u>	<u>sec_id</u>	<u>semester</u>	<u>ID</u>	<u>course_id</u>	<u>course_id</u>	
<u>dept_name</u>	<u>ID</u>	<u>course_id</u>	<u>sec_id</u>	<u>sec_id</u>	<u>course_id</u>	<u>sec_id</u>	<u>ID</u>	<u>course_id</u>	<u>title</u>	<u>s_id</u>
building	name	<u>semester</u>	<u>year</u>	<u>building</u>	<u>sec_id</u>	<u>semester</u>	name	<u>dept_name</u>	<u>dept_name</u>	<u>i_id</u>
budget	dept_name	<u>year</u>	<u>time_slot_id</u>	<u>room_number</u>	<u>year</u>	<u>grade</u>	tot_cred	credits		

FIN

Any questions?