Database Lab 10
Correlated Subquery and Division
Addendum

Dr. Jefferson Fong

Reminder

- Project ER Diagrams are due on Sunday 24 November, right before midnight.
- Static (HTML and CSS) web pages are due on 1 December.
    - You can use other software tools, but no extra credit will be given on how nice you web pages look.
    - Our focus is on the database. See the Project Requirement documentation for details.

- Assignment 2 is due on Sunday 1 December, right before midnight.

Slide 9-10

- To help you to remember this correlated subquery:
- In the example catalogue ÷ programme,
  - Catalogue is the **top dividend**, and programme is the **bottom divisor**.
  - Do the subquery first

    WHERE NOT EXISTS (***bottom*** EXCEPT ***top***)

    - The **top dividend** (inner most part) **C1** correlates to the **C1** in the **main query** for **top**.

Slide 10-11

- If your version of MySQL does not support EXCEPT, use NOT IN instead.

Example from Lab 10 PPT, Slides 9-11:

Query: Find course names that are taught by all programmes (or every programme).

Solution: Catelogue(c_name, p_name) ÷ Programme(p_name)

- Note the schema result is c_name.

Example in Slide 12

Query: Find customer_id who rented film from every staff.

Solution: rental(customer_id, staff_id) ÷ staff(staff_id)

So in Slide 11's query, change

Catalogue to Rental,
Programme to Staff,
c_name to customer_id,
p_name to staff_id.

```
SELECT DISTINCT customer_id
FROM rental AS r1
WHERE NOT EXISTS(
   SELECT *
   FROM staff
   WHERE staff_id NOT IN (
      SELECT staff_id
      FROM rental AS r2
      WHERE r2.customer_id=r1.customer_id
   )
);
```

Exercise 1: Find customers who rented films from all stores.

- The **apparent solution** seems to be

  Customer(customer_id, store_id) ÷ Store(store_id)
- But this query **returns empty rows!**

- The table Customer(customer_id, store_id, first_name, last_name, …) is misleading.
- The store_id refers to the store where the customer registered, not where he or she rented films.
  - Check the contents of the customer table in Sakila.
  - The apparent solution is for “find customers who **registered** in all stores”, not what we want.

Apparent solution:

```
SELECT c1.customer_id
FROM customer AS c1
WHERE NOT EXISTS(
   SELECT *
   FROM store
   WHERE store_id NOT IN (
      SELECT c2.store_id
      FROM customer AS c2
      WHERE
         c1.customer_id=c2.customer_id
   )
);
```

Change:
Catalogue to Customer,
Programme to Store,
c_name to customer_id,
p_name to store_id.

Exercise 1: Find customers who rented films from all stores.

Correct solution:

To find out which film the customer rented, we must go to the rental table.

Rental(rental_id, inventory_id, customer_id).

Walk to table Inventory(inventory_id, film_id, **store_id**). This store_id refers to the location of the film, from where the customer rented.

So join tables rental and inventory using inventory_id to get the correct store_id.

The innermost subquery is then (with customer AS c1)

```
SELECT store_id
FROM (rental AS r1 JOIN inventory USING (inventory_id))
WHERE c1.customer_id=r1.customer_id
```

Replace the **innermost subquery** marked in **red** in the **apparent solution** in the last slide.

"That's all Folks!"