

Database Lab 5

Aggregation

Addendum

Dr. Jefferson Fong



Tips for doing exercises (and assignments):

- Go through sakalia.sql and check the schema; make notes of the results; (do with a partner).
 - actor(actor_id, first_name, last_name)
 - ...
 - category(category_id, name)
 - ...
 - film(film_id, title, description, release_year, language_id)
 - film_category(film_id, category_id)
 - ...
 - language(language_id, name)
- Note:
 - In each table, the primary key is in red; it's difficult to see underline in PPT.
 - Attribute last_update is in every table; we will suppress that in our slides.

Exercise Solutions

- Lecture exercises solutions are in the following week's lecture notes.
- ~~Lab exercises solutions are in iSpace beneath the lab submission links, available after the submission deadline for that lab.~~

Aggregation

- You've learned aggregation functions in Excel: count(), average(), max(), min(), etc.
- In this lab, you will learn to use aggregation functions in SQL.

Slide 2

- In XAMPP, try

```
SELECT language_id, name FROM language      /* 6 languages */
```

```
SELECT COUNT(*) FROM language              /* 6 */
```

```
SELECT * FROM `film` WHERE language_id = 1  /* lots of English films */
```

```
SELECT * FROM `film` WHERE language_id <> 1 /* no non-English films?? */
```

Slide 3

1. From the description of query (top of Slide 3), check the schemas.
 - Figure out which tables to walk.

```
film(film_id, title, description, release_year, language_id, ...)  
language(language_id, name)
```

- If we **only** want to display the **language_id** and **count(film_id)**, **don't need JOIN** table **language** (which contains language **name**).

```
SELECT language_id, count(film_id)  
FROM film JOIN language USING(language_id)  
GROUP BY (language_id)
```

2. Group together all records (rows) with the same language_id.
3. Count the distinct film_id in each group.

Slide 3

- If we want to display the language **name**, we need **JOIN** tables.

```
SELECT name, count(film_id)
FROM film JOIN language USING(language_id)
GROUP BY (language_id)
```

(1) Check the tables in the schema; see previous slide.

- **JOIN** tables *film* and *language* USING their common attribute *language_id*.

(2) **GROUP** the result from (1) **BY** *language_id*.

(3) **Count** the distinct *film_id* in each group.

- Get 1000 films in English, no films in other languages.

Slide 7 Examples

Slide 8 – 10 Solution to Slide 7 examples

First Example in Slide 7, solution in Slide 8. (Follow Slide 3.)

- Consider the database schema to answer these questions.
 - (1a) What info do we want? [**category name** and the **count** of **film_id** in each category]
 - Which tables contain the information needed?
category(**category_id**, **name**) and film_category(**film_id**, **category_id**)
 - (1b) Which attributes do these tables have in common? [**category_id**]
 - (1c) Write the FROM part
 - (2a) How does intermediate result look? [group by **category_id**]
 - (2b) Write GROUP BY.
 - (3a) How to display the final result? [**name** of category, count(**film_id**)]
 - (3b) Write SELECT.

Second Example in Slide 7, solution in Slide 9

- Consider the database schema to answer these questions.
 1. Elizabeth Brown is in customer(customer_id, store_id, first_name, last_name, ...)
 2. Money spent is in payment(payment_id, customer_id, ... amount, ...)

Third Example in Slide 7, solution in Slide 10

- Combine first and second examples.
- Category join film_category gives (category_id, name, film_id)
- Customer join payment gives (customer_id, store_id, first_name, last_name, ..., payment_id, staff_id, rental_id, ...)
- We need to go from (category join film_category) to (customer join payment)
 - Maybe rental_id seems promising; so check table rental.

Exercises – Slide 13.

1. Which table? film(rental_duration).

```
SELECT rental_duration FROM film; /* Checking */
```

```
SELECT AVG(rental_duration) FROM film; /* Compute the AVG */
```

2. Which table?

customer(customer_id, first_name='Norman', last_name='Currier')

rental(customer_id, rental_id, inventory_id, ...)

Inventory(inventory_id, film_id) ← That's what we want!!

Then write the FROM part.

3. Same tables as in 2, so use the same FROM as in 2.

Rented by “each customer”, so use GROUP BY customer_id