Lab 3 Cross table query

United International College

Motivation

- Sometimes a single table does not contain all desired attributes.
- For example, one wants to find all cities in China.
  - City information is in the table `city=(city_id, city, country_id, last_update)`.
  - But “China” is a value for attribute `country` in the table `country`.
  - And `city` only has `country_id`.

- It is doable to find the country id for China first, then write another query to find the cities.
- But how can this be done in one query?

Cross table

- Recall that predicates can only compare the values of attributes or constants (so far at least).
- Here we want to compare `country_id` from `city` with `country_id` from `country`.

```
SELECT city
FROM city, country
WHERE city.country_id = country.country_id AND country='China'
```

- The system executes the query as

```
for each tuple t in city do
    for each tuple s in country do
        if t.country_id = s.country_id and s.country = "China" then
            print t.city
        end if
    end for
end for
```

Cross table

- Equivalent to querying from a (temporary) table `city×country`, the cartesian product of `city` and `country`.
- For example,

city_id	city	country_id
1	Beijing	1
2	Tokyo	2

×

country_id	country
1	China
2	Japan

=

city.			country.	
city_id	city	country_id	country_id	country
1	Beijing	1	1	China
2	Tokyo	2	1	China
1	Beijing	1	2	Japan
2	Tokyo	2	2	Japan

```
SELECT city FROM temp WHERE city.country_id=country.country_id AND country='China'
```

- To distinguish the two `country_id`, we use `city.country_id` and `country.country_id`.
- The attribute `city` does not need `city.city`, because there is no other attribute with the same name. (Same for `country`.)

Examples

- Find the phone number of the customer Lisa Anderson.
- Find the language of the film “Angels Life”.

Keys

- Find the films played by Angela (actor's first name).

```
SELECT actor.actor_id, first_name, last_name, film.title
FROM actor, film_actor, film
WHERE first_name = 'Angela' AND
      actor.actor_id = film_actor.actor_id AND
      film_actor.film_id = film.film_id
```

- Why we need to select the id, first name, and last name of the actor?

Keys

- There are two actors whose first name is “Angela”, Angela Hudson and Angela Witherspoon.
- To distinguish the films played by which Angela, we need actor id, which is the **key** of the schema.

- The **key** of a schema is a set of one or multiple attributes, which can uniquely define the tuples in the table.

- E.g. `actor_id` is the key of `actor`.

- Again, the meaning of keys in schemas is similar but not the same as keys in ER diagrams.

A table times itself

- Sometimes a table can cross itself.
- Find the id of cities which have more than one address.
- The table `address` contains `address_id` and `city_id`.
- The logic of this search is

```
for each tuple t in address do
    for each tuple s in address do
        if t.city_id = s.city_id
        and t.address_id != s.address_id then
            print t.city_id
        end if
    end for
end for
```

The nested loop is implemented by the table `address` crossing with itself.

A table times itself

```
for each tuple t in address do
    for each tuple s in address do
        flag = 0
        if t.city_id = s.city_id
        and t.address_id ! = s.address_id then
            print t.city_id
        end if
    end for
end for
```

```
SELECT a1.city_id
FROM address AS a1, address AS a2
WHERE a1.city_id = a2.city_id AND
    a1.address_id <> a2.address_id
```

- The two tables are renamed by "**AS**", to avoid ambiguous.
- If two tuples agree on `city_id` but have different `address_id`, this `city_id` has multiple addresses.
- `<>` means not equal.

Example

- Find the name (first and last) of actors who have the same first name with another actor.

Exercises

Write SQLs for the following questions.

1. Find the films (name) played by Zero Cage.
2. Find the films (name) rented by George Linton.
3. Find the customers (name) who have rented some action (category) films.
4. Find the customers who live in China and have rented some Japanese films.
5. Find all pairs of customers (name) who have rented a same film.
6. Find the actors who have played a same film with Bolger (the last name of an actor)

Save your queries in a txt file. Rename it as “COMP3013 Lab3 ###.txt”, where “###” is your student ID. And submit it on iSpace. The DDL is 24 hours after the lab.

End of Lab 3