Lab 4 Join

United International College

Motivation

- Are you tired of condition checking in cross table queries?
- The **`NATURAL JOIN`** operator associates two tables by the common attributes.
- After **`NATURAL JOIN`**, the duplicated attributes are omitted.
- For example,

```
SELECT * FROM city NATURAL JOIN country
```

is implemented as

```
SELECT city.country_id, city.last_update, city_id, city, country
FROM city, country
WHERE city.country_id = country.country_id AND
           city.last_update = country.last_update
```

Example

- Find the address for each customer, show the customer name, street, district, and postal code.

Join Condition

- Why does the following query not work?

```
SELECT first_name, last_name, address, district, postal_code
FROM customer NATURAL JOIN address
```

- It is equivalent to

```
SELECT first_name, last_name, address, district, postal_code
FROM customer, address
WHERE customer.address_id = address.address_id AND
      customer.last_update = address.last_update
```

- `last_update` is also checked because it is a common attribute.
- If customers and addresses are not updated at the same time, `last_update` is not equal. Then, the predicate is always false.

Join Condition

- Join conditions define in which condition the tuples are associated.
- Two tuples are associated if
  - **`NATURAL`**: all common attributes have the same value.
  - **`ON <predicate>`**: the predicate is evaluated to be true.
  - **`USING`** $(A_1, A_2, \cdots, A_n)$: the common attributes in list have the same value.
- For example, these queries are equivalent.

  -
    ```
    SELECT first_name, last_name, address, district, postal_code
    FROM customer NATURAL JOIN address
    ```

  -
    ```
    SELECT first_name, last_name, address, district, postal_code
    FROM customer JOIN address ON customer.address_id = address.address_id AND
    customer.last_update = address.last_update
    ```

  -
    ```
    SELECT first_name, last_name, address, district, postal_code
    FROM customer JOIN address USING (address_id, last_update)
    ```

Join Condition

- A JOIN without any condition is same as a cartesian product.
- Compare the outcome of these queries.

```
SELECT * FROM staff, store

SELECT * FROM staff JOIN store
```

- Sometimes more than two tables are joined together.

```
SELECT * FROM table1 NATURAL JOIN table2 NATURAL JOIN table3
```

- The query is understood as

```
SELECT * FROM (table1 NATURAL JOIN table2) NATURAL JOIN table3
```

Example

- Fix the problem caused by the common attribute `last_update` on page 4.
- Find the name of the manager of each store. You have to use `JOIN`.

Notes:

- The predicate in the `ON` clause is user defined, which is very flexible.
- `NATURAL` and `USING` combine the common attributes. But `ON` duplicates common attributes.

Join Type

- Sometimes users want to keep the unmatched tuples after joining two tables.
- **`OUTER JOIN`** can handle it.
  - `table1 NATURAL` **`LEFT OUTER`** `JOIN table2`
    All tuples in table1 are in the result. For the unmatched tuples, the values of the attributes from table2 are **NULL**, meaning "unknown". (NULL values will be introduced in following labs.)

  - `table1 NATURAL` **`RIGHT OUTER`** `JOIN table2`
    The unmatched tuples from table2 are kept.

  - `table1 NATURAL` **`FULL OUTER`** `JOIN table2`
    All tuples (from both table1 and table2) are kept.

  - `NATURAL` is the join condition.
  - On the opposite of **`OUTER`**, **`INNER JOIN`** does not keep the unmatched tuples.
  - Same as `JOIN`. "`INNER`" is usually omitted.

OUTER JOIN

- Suppose we try to join the two tables.

Name	A_id
Alice	1
Bob	3

Table: person

A_id	Address
1	2000 Jintong
2	300 Jinfeng

Table: address

-
  ```
  SELECT *
  FROM person NATURAL LEFT OUTER JOIN address
  ```

Name	A_id	Address
Alice	1	2000 Jintong
Bob	3	NULL

-
  ```
  SELECT *
  FROM person NATURAL RIGHT OUTER JOIN address
  ```

Name	A_id	Address
Alice	1	2000 Jintong
NULL	2	300 Jinfeng

-
  ```
  SELECT *
  FROM person NATURAL FULL OUTER JOIN address
  ```

Name	A_id	Address
Alice	1	2000 Jintong
Bob	3	NULL
NULL	2	300 Jinfeng

Example

- For each of the address, find the customer who lives there. Display all addresses, even for the address that nobody lives.

Exercises

Write SQLs for the following questions. **You have to use `JOIN` for each of the query**.

1. Find the films (title) played by Zero Cage.
2. Find the films (title) rented by George Linton. The join condition is `ON`.
3. Find the customers (name) who have rented some action (category) films. The join condition is `USING`.
4. Join the tables `film`, `film_category`, and `category`, using both conditions `ON` and `USING`.
5. Find all pairs of customers (name) who have rented a same film. Any join condition is fine.
6. Find the films rented by each customer. If a customer has not rented any film, give it a NULL value.

Save your queries in a txt file. Rename it as "COMP3013 Lab4 ###.txt", where "###" is your student ID. And submit it on iSpace. The DDL is 24 hours after the lab.

End of Lab 4