Lab 4
Advanced Data Manipulation 2

CSE 4308
DATABASE MANAGEMENT SYSTEM LAB

September 29, 2024

Contents

1 Foreign key

Foreign keys are used to restrict the domain of columns of one table to the values of another table. The statement for declaring a foreign key in MySQL is as follows:

```
CREATE TABLE table_name
(
  attribute1 datatype [ NULL | NOT NULL ],
  attribute2 datatype [ NULL | NOT NULL ],
  ...,
  [CONSTRAINT constraint_name] FOREIGN KEY (foreign_attribute1,
   ...)
    REFERENCES reference_table_name (reference_attribute1, ...)
    ON DELETE CASCADE | SET NULL
    ON UPDATE CASCADE | SET NULL
)
```

For example, the DEPT_NAME column of the COURSE table can only have values of the departments that are available in the DEPARTMENT table.

```
CREATE TABLE DEPARTMENT
(
  DEPT_NAME VARCHAR(20),
  TITLE VARCHAR(30),
    EST_YEAR VARCHAR(4),
  CONSTRAINT PK_DEPARTMENT PRIMARY KEY (DEPT_NAME)
)
```

```
CREATE TABLE COURSE
(
  COURSE_ID VARCHAR(8),
  TITLE VARCHAR(30),
    PROGRAM VARCHAR(5),
  DEPT_NAME VARCHAR(20),
  CREDITS INT,
  CONSTRAINT PK_COURSE PRIMARY KEY (COURSE_ID, PROGRAM),
  CONSTRAINT FK_COURSE_DEPARTMENT FOREIGN KEY (DEPT_NAME)
    REFERENCES DEPARTMENT (DEPT_NAME)
    ON DELETE CASCADE
)
```

DEPT_NAME	TITLE	EST_YEAR
CS	Computer Science	1990
EE	Electrical Engineering	1985
ME	Mechanical Engineering	1992

Table 1: DEPARTMENT Table

COURSE_ID	TITLE	PROGRAM	DEPT_NAME	CREDITS
CS101	Intro to CS	CS	CS	3
EE201	Circuits	EE	EE	4
ME301	Thermodynamics	ME	ME	4

Table 2: COURSE Table

Here you need to ensure the referenced attribute of the referencing table is of the same data type as the referenced attribute of the referenced table. It is preferred to use the primary key (or composite primary key) of the referenced table as the foreign key. **Moreover, the referenced table must be created before the referencing table.**
In some cases, a referencing table may need to reference itself; this is known as self-referencing.

2 Distinct

The DISTINCT statement is used to return only distinct (unique) values. Inside a table, a column often contains many duplicate values; and sometimes you only want to list the different (distinct) values. The syntax in MySQL is the same as in Oracle:

```
SELECT DISTINCT attributename
    FROM tablename;
```

For example,

```
SELECT DISTINCT DEPT_NAME FROM EMPLOYEE;
```

3 Range Operator

3.1 BETWEEN Operator

The BETWEEN operator selects values within a given range. The values can be numbers, strings, or dates. The BETWEEN operator is inclusive: begin and end values will be included. However, you have to provide the lower value first then the upper value.

```
SELECT attribute_1, .....
FROM tablename
WHERE attribute_1 BETWEEN l_value AND u_value;
```

For example,

```
SELECT NAME, DEPT_NAME, SALARY
FROM EMPLOYEE
WHERE SALARY BETWEEN 35000 AND 80000;
```

For excluding a range, we can use the NOT operator before BETWEEN. For example,

```
SELECT NAME, DEPT_NAME, SALARY
FROM EMPLOYEE
WHERE SALARY NOT BETWEEN 35000 AND 80000;
```

3.2 IN Operator

The IN operator allows one to specify multiple values in the WHERE clause. It is a shorthand for multiple OR conditions.

```
SELECT attribute_1, .....
FROM tablename
WHERE attribute_1 IN (value1, value2, ...);
```

For example,

```
SELECT NAME, SALARY
FROM EMPLOYEE
WHERE DEPT_NAME IN ('DEV', 'TESTING');
```

Similar to BETWEEN, we can use NOT to exclude some values.

4 Aliases

SQL aliases are used to give a table or a column in a table a temporary name. We can also use it for renaming an expression or an entire query. The primary purpose of using aliases is to make a column or table more readable. For example:

```
SELECT ID, NAME, CONCAT('ANNUAL SALARY: ', SALARY * 12) AS
   ANNUAL_SALARY
FROM EMPLOYEE;
```

Here, using AS is optional for the expression, table, and any query.

5 String Operator

5.1 LIKE Operator

In simple terms, the LIKE operator is used to match substrings in a query. It is used in a WHERE clause to search for a specified pattern in a column. There are two wildcards often used in conjunction with the LIKE operator:

- % to represent any substring.
- _ to represent any single character.

```
SELECT attribute_1, attribute_2 .....
FROM tablename
WHERE attribute_1 LIKE 'pattern';
```

Here, patterns are case-sensitive. For example,

```
SELECT NAME
FROM EMPLOYEE
WHERE NAME LIKE 'A%';
```

This will show you all the names that start with a capital A. To restrict the name size of the resultant name to, let's say, 4 characters, we can use the following statement:

```
SELECT NAME
FROM EMPLOYEE
WHERE NAME LIKE 'A___';
```

5.2 Concatenation

To concatenate multiple columns or any additional string with any column at the time of retrieving data, SQL supports the CONCAT() function in MySQL. For instance,

```
SELECT CONCAT('Employee NAME: ', NAME) AS 'Employee Name'
FROM EMPLOYEE;
```

This will show the term 'Employee NAME: ' as the prefix of any name.

6 Data Sorting

The ORDER BY keyword is used to sort the result-set in ascending or descending order. If one does not specify the sorting order, the ORDER BY keyword sorts the records in ascending order by default. Otherwise, to specify ascending or descending order of sorting, we use the keywords ASC and DESC respectively.

```
SELECT attribute_1, attribute_2 .....
FROM tablename
ORDER BY attribute_1 [ASC|DESC], ....;
```

For example,

```
SELECT NAME, SALARY
FROM EMPLOYEE
ORDER BY DEPT, SALARY DESC;
```

7 Sub-query

SQL provides a mechanism for nesting sub-queries. A sub-query is a SELECT-FROM-WHERE expression that is nested within another query. It can be placed in two places:

- In the WHERE clause of a SELECT statement. Example,

```
SELECT DEPT
FROM EMPLOYEE
WHERE SALARY > (
    SELECT AVG(SALARY)
    FROM EMPLOYEE
);
```

8 Lab Task

Column	Data Type	Description
pokemon_id	INT	Primary key
name	VARCHAR(50)	Name of the Pokémon
type	VARCHAR(20)	Type of the Pokémon (e.g., Fire)
hp	INT	Hit Points
attack	INT	Attack stat
defense	INT	Defense stat
speed	INT	Speed stat

Table 3: pokemon Table Schema

Column	Data Type	Description
trainer_id	INT	Primary key
first_name	VARCHAR(30)	First name of the trainer
last_name	VARCHAR(30)	Last name of the trainer
city	VARCHAR(30)	City where the trainer resides

Table 4: trainer Table Schema

1. Create Tables and insert the data

pokemon_id	name	type	hp	attack	defense	speed
1	Bulbasaur	Grass	45	49	49	45
2	Ivysaur	Grass	60	62	63	60
3	Venusaur	Grass	80	82	83	80
4	Charmander	Fire	39	52	43	65
5	Charmeleon	Fire	58	64	58	80
6	Charizard	Fire	78	84	78	100
7	Squirtle	Water	44	48	65	43
8	Wartortle	Water	59	63	80	58
9	Blastoise	Water	79	83	100	78
10	Pikachu	Electric	35	55	40	90
11	Raichu	Electric	60	90	55	110

Table 5: pokemon Table Data

trainer_id	first_name	last_name	city
1	Ash	Ketchum	Pallet Town
2	Misty	Williams	Cerulean City
3	Brock	Harrison	Pewter City
4	Gary	Oak	Pallet Town
5	Erika	Green	Celadon City

Table 6: trainer Table Data

2. Write a query to display the different types of Pokémon available in the `pokemon` table. Ensure that each type is listed only once.
3. List all Pokémon whose `attack` stat is between 50 and 80, inclusive.
4. Find all Pokémon whose names start with the letter 'C'.
5. Find all Pokémon whose names contain 'saur' anywhere in their names.
6. Find all Pokémon whose names have exactly 9 characters and the fifth character is 'e'.
7. Create a query to display the full names (first name and last name concatenated) of all trainers, along with their city.
8. List all Pokémon sorted first by `type` in ascending order and then by `attack` stat in descending order.
9. Suppose you need to track which trainer owns which Pokémon. To do this, you need to create a new table called `trainer_pokemon` with the following columns:

   - `trainer_id INT`
   - `pokemon_id INT`

   Create the `trainer_pokemon` Table

   Write a SQL statement to create the `trainer_pokemon` table.
10. Add a foreign key constraint on `trainer_id` referencing `trainer(trainer_id)` and another on `pokemon_id` referencing `pokemon(pokemon_id)`.

Submission Guidelines

You are required to submit a report that includes your code, a detailed explanation, and the corresponding output. Rename your report as `<StudentID_Lab_1.pdf>`.

- Your lab report must be submitted as a PDF. Recommended tool: Overleaf. You can use some of the available templates in Overleaf.
- If you are using Google Docs instead, use the plugin 'CodeBlocks' for syntax highlighting. Do not give screenshots of your code.
- Include all code/pseudo code relevant to the lab tasks in the lab report.
- In cases of high similarities between two reports, both reports will be discarded.