

REPORTE DE PRÁCTICA NO. 2

NOMBRE DE LA PRÁCTICA —1.4. Práctica. Álgebra relacional y SQL (2)

Dr. Eduardo Cornejo-Velázquez

ALUMNO: Cesar Martinez Andrade



1. Introducción

En esta práctica, exploraremos el fascinante mundo de las funciones en el contexto de MySQL. Las funciones son herramientas poderosas que nos permiten realizar operaciones específicas dentro de nuestras consultas SQL. Al utilizarlas correctamente, podemos simplificar nuestro código, mejorar la eficiencia y obtener resultados más precisos.

Para llevar a cabo esta práctica, utilizaremos el software MySQL Command Client, una herramienta que nos permite interactuar con bases de datos MySQL mediante líneas de código. Aquí, escribiremos nuestras consultas y las compilaremos para obtener los resultados deseados.

Además, aprovecharemos los conceptos del álgebra relacional que hemos estudiado previamente en clase. El álgebra relacional es una teoría matemática que nos proporciona un conjunto de operaciones para manipular relaciones (tablas) en bases de datos. Al combinar estas operaciones con las funciones de MySQL, podremos resolver problemas complejos de manera elegante y estructurada.

Objetivos

1. Comprender el propósito y la sintaxis de las funciones en MySQL.
2. Familiarizarnos con el uso del software MySQL Command Client.
3. Aplicar el álgebra relacional para resolver problemas específicos.

2. Marco teórico

Álgebra Relacional

El álgebra relacional de manera breve es un conjunto de operaciones matemáticas que se aplican a relaciones (tablas) en una base de datos. Fue propuesto por Edgar F. Codd[4], el padre de las bases de datos relacionales. Algunas de las operaciones más importantes del álgebra relacional son:

- Selección[3]: Permite filtrar filas de una tabla según una condición específica. Por ejemplo, podemos seleccionar todos los estudiantes que tienen una calificación superior a 8.
- Proyección: Nos permite elegir columnas específicas de una tabla. Por ejemplo, podemos proyectar solo los nombres y apellidos de los profesores.
- Unión: Combina dos tablas para obtener un conjunto de filas sin duplicados. Por ejemplo, podemos unir las tablas de estudiantes y profesores para obtener una lista completa de personas relacionadas con la institución.

Funciones de SQL

Las funciones en SQL son operaciones predefinidas que se utilizan para procesar o transformar los datos de una base de datos. Estas funciones se combinan con la sentencia `SELECT` y otras sentencias para realizar diversas tareas. Aquí tienes una breve descripción de algunas funciones comunes:

1. Funciones de Agregado: Estas funciones realizan cálculos sobre grupos de filas y devuelven un solo valor. Ejemplos incluyen `SUM`, `AVG`, `COUNT`, `MIN` y `MAX`.
2. Funciones de Cadena: Manipulan cadenas de texto. Algunas funciones útiles son `CONCAT` (para concatenar cadenas), `SUBSTRING` (para extraer una parte de una cadena) y `UPPER/LOWER` (para convertir a mayúsculas o minúsculas).
3. Funciones de Fecha y Hora: Trabajan con fechas y horas. Ejemplos son `GETDATE()` (para obtener la fecha y hora actual), `DATEADD` (para agregar o restar días/meses/años) y `DATEDIFF` (para calcular la diferencia entre dos fechas).

4. **Funciones Matemáticas:** Realizan operaciones matemáticas. Por ejemplo, **ROUND** (redondeo), **ABS** (valor absoluto) y **POWER** (potencia).
5. **Funciones de Conversión de Datos:** Cambian el tipo de datos. Por ejemplo, **CAST** o **CONVERT** para convertir una cadena en un número.

3. Herramientas empleadas

Describir qué herramientas se han utilizado...

1. **MySQL Command Client:** Esta herramienta nos permitirá interactuar con la base de datos MySQL mediante comandos SQL.
2. **Álgebra Relacional:** Aplicaremos los conceptos aprendidos en clase para modelar y resolver problemas relacionados con bases de datos.
3. **Bloc de notas o Notepad++:** Para agrupar y generar más rápidamente las tuplas necesarias usando las tabulaciones y comandos de copiado y pegado

4. Desarrollo

IMPORTANTE

En esta ocasión, la generación de la base de datos sigue un enfoque similar al de la práctica anterior. Para optimizar el uso del espacio en disco, hemos decidido reutilizar la misma base de datos. Esto implica que tanto las tablas como los datos son idénticos a los que manejamos previamente. Sin embargo, hay una diferencia clave: el bloque de actividades solicitado será diferente. Así que, aunque la base de datos permanece constante, el desafío que enfrentamos en esta práctica tiene su propia singularidad.

Bloque de Actividades solicitado

1. Obtener el tamaño del texto en todos los valores de la columna "First_name".
2. Obtener el nombre de todos los empleados después de reemplazar 'o' con '#'.
3. Obtener el nombre y apellido de todos los empleados en una sola columna separados por "-".
4. Obtener el año, mes y día de la columna "Joining_date".
5. Obtener todos los empleados en orden ascendente por nombre.
6. Obtener todos los empleados en orden descendente por nombre.
7. Obtener todos los empleados en orden ascendente por nombre y en orden descendente por salario.
8. Obtener todos los empleados con el nombre "Bob".
9. Obtener todos los empleados con el nombre "Bob" o "Alex".
10. Obtener todos los empleados que no tengan el nombre "Bob" o "Alex".
11. ¿Qué es una inyección SQL?

Sentencias SQL

En el Listado 1 se presentaran todas las lineas de código para resolver el bloque de actividades solicitado anteriormente.

Listing 1: Crear base de datos competencia.

```
CREATE DATABASE PracticaAlgebraRelacionalySQL1;
USE PracticaAlgebraRelacionalySQL1;

CREATE TABLE Employee(Employee_id INT PRIMARY KEY,
First_name VARCHAR(20),
Last_name VARCHAR(20),
Salary INT(15),
Joining_date DATE,
Departament VARCHAR(20));

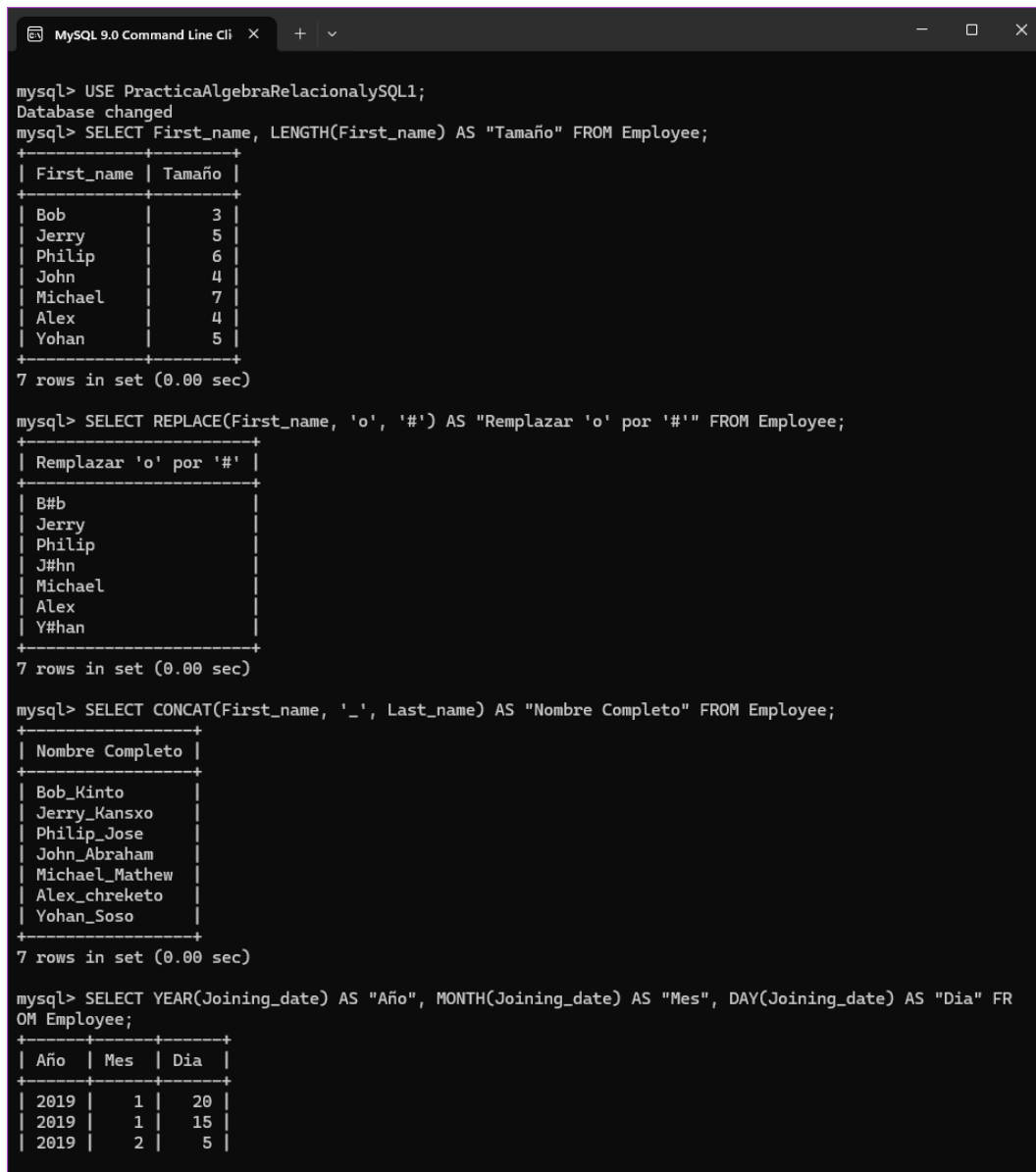
CREATE TABLE Reward(Employee_ref_id INT PRIMARY KEY,
date_reward date,
amount int(5));

INSERT INTO Employee VALUES
(1,"Bob","Kinto",1000000,"2019-01-20","Finance"),
(2,"Jerry","Kansxo",6000000,"2019-01-15","IT"),
(3,"Philip","Jose",8900000,"2019-02-05","Banking"),
(4,"John","Abraham",2000000,"2019-02-25","Insurance"),
(5,"Michael","Mathew",2200000,"2019-02-18","Finance"),
(6,"Alex","chreketo",4000000,"2019-05-10","IT"),
(7,"Yohan","Soso",1230000,"2019-06-20","Banking");

INSERT INTO Reward VALUES
(1,"2019-05-11",1000),
(2,"2019-02-15",5000),
(3,"2019-04-22",2000),
(4,"2019-06-20",8000);

SELECT First_name, LENGTH(First_name) AS "Tamano" FROM Employee;
SELECT REPLACE(First_name, 'o', '#') AS "Reemplazar - 'o' - por - '#'" FROM Employee;
SELECT CONCAT(First_name, '_', Last_name) AS "Nombre-Completo" FROM Employee;
SELECT YEAR(Joining_date) AS "Anio",
MONTH(Joining_date) AS "Mes",
DAY(Joining_date) AS "Dia" FROM Employee;
SELECT * FROM Employee ORDER BY First_name ASC;
SELECT * FROM Employee ORDER BY First_name DESC;
SELECT * FROM Employee ORDER BY First_name ASC, Salary DESC;
SELECT * FROM Employee WHERE First_name = 'Bob';
SELECT * FROM Employee WHERE First_name IN ( 'Bob', 'Alex' );
SELECT * FROM Employee WHERE First_name NOT IN ( 'Bob', 'Alex' );
```

Capturas de Pantalla



```
mysql> USE PracticaAlgebraRelacionalySQL1;
Database changed
mysql> SELECT First_name, LENGTH(First_name) AS "Tamaño" FROM Employee;
+-----+-----+
| First_name | Tamaño |
+-----+-----+
| Bob        | 3      |
| Jerry      | 5      |
| Philip     | 6      |
| John       | 4      |
| Michael    | 7      |
| Alex       | 4      |
| Yohan      | 5      |
+-----+-----+
7 rows in set (0.00 sec)

mysql> SELECT REPLACE(First_name, 'o', '#') AS "Reemplazar 'o' por '#'" FROM Employee;
+-----+-----+
| Reemplazar 'o' por '#' |
+-----+-----+
| B#b                   |
| Jerry                 |
| Philip                |
| J#hn                  |
| Michael               |
| Alex                  |
| Y#han                 |
+-----+-----+
7 rows in set (0.00 sec)

mysql> SELECT CONCAT(First_name, '_', Last_name) AS "Nombre Completo" FROM Employee;
+-----+-----+
| Nombre Completo |
+-----+-----+
| Bob_Kinto       |
| Jerry_Kansxo    |
| Philip_Jose     |
| John_Abraham    |
| Michael_Mathew  |
| Alex_chreketo   |
| Yohan_Soso      |
+-----+-----+
7 rows in set (0.00 sec)

mysql> SELECT YEAR(Joining_date) AS "Año", MONTH(Joining_date) AS "Mes", DAY(Joining_date) AS "Dia" FROM Employee;
+-----+-----+
| Año | Mes | Dia |
+-----+-----+
| 2019 | 1  | 20  |
| 2019 | 1  | 15  |
| 2019 | 2  | 5   |
+-----+-----+
```

Figure 1: Consultas solicitadas

```

MySQL 9.0 Command Line Cli  X  +  -  X
+-----+-----+-----+-----+-----+-----+
| 7 | Yohan | Soso | 1230000 | 2019-06-20 | Banking |
| 3 | Philip | Jose | 8900000 | 2019-02-05 | Banking |
| 5 | Michael | Mathew | 2200000 | 2019-02-18 | Finance |
| 4 | John | Abraham | 2000000 | 2019-02-25 | Insurance |
| 2 | Jerry | Kansxo | 6000000 | 2019-01-15 | IT |
| 1 | Bob | Kinto | 1000000 | 2019-01-20 | Finance |
| 6 | Alex | chreketo | 4000000 | 2019-05-10 | IT |
+-----+-----+-----+-----+-----+
7 rows in set (0.00 sec)

mysql> SELECT * FROM Employee ORDER BY First_name ASC, Salary DESC;
+-----+-----+-----+-----+-----+-----+
| Employee_id | First_name | Last_name | Salary | Joining_date | Departament |
+-----+-----+-----+-----+-----+-----+
| 6 | Alex | chreketo | 4000000 | 2019-05-10 | IT |
| 1 | Bob | Kinto | 1000000 | 2019-01-20 | Finance |
| 2 | Jerry | Kansxo | 6000000 | 2019-01-15 | IT |
| 4 | John | Abraham | 2000000 | 2019-02-25 | Insurance |
| 5 | Michael | Mathew | 2200000 | 2019-02-18 | Finance |
| 3 | Philip | Jose | 8900000 | 2019-02-05 | Banking |
| 7 | Yohan | Soso | 1230000 | 2019-06-20 | Banking |
+-----+-----+-----+-----+-----+
7 rows in set (0.00 sec)

mysql> SELECT * FROM Employee WHERE First_name = 'Bob';
+-----+-----+-----+-----+-----+-----+
| Employee_id | First_name | Last_name | Salary | Joining_date | Departament |
+-----+-----+-----+-----+-----+-----+
| 1 | Bob | Kinto | 1000000 | 2019-01-20 | Finance |
+-----+-----+-----+-----+-----+
1 row in set (0.00 sec)

mysql> SELECT * FROM Employee WHERE First_name IN ('Bob','Alex');
+-----+-----+-----+-----+-----+-----+
| Employee_id | First_name | Last_name | Salary | Joining_date | Departament |
+-----+-----+-----+-----+-----+-----+
| 1 | Bob | Kinto | 1000000 | 2019-01-20 | Finance |
| 6 | Alex | chreketo | 4000000 | 2019-05-10 | IT |
+-----+-----+-----+-----+-----+
2 rows in set (0.00 sec)

mysql> SELECT * FROM Employee WHERE First_name NOT IN ('Bob','Alex');
+-----+-----+-----+-----+-----+-----+
| Employee_id | First_name | Last_name | Salary | Joining_date | Departament |
+-----+-----+-----+-----+-----+-----+
| 2 | Jerry | Kansxo | 6000000 | 2019-01-15 | IT |
| 3 | Philip | Jose | 8900000 | 2019-02-05 | Banking |
| 4 | John | Abraham | 2000000 | 2019-02-25 | Insurance |
| 5 | Michael | Mathew | 2200000 | 2019-02-18 | Finance |
| 7 | Yohan | Soso | 1230000 | 2019-06-20 | Banking |
+-----+-----+-----+-----+-----+
5 rows in set (0.00 sec)

```

Figure 2: Consultas solicitadas.

Algebra Relacional

4.5.1 Sintaxis para insertar registros

$$Employee \leftarrow Employee \cup \{1, "Bob", "Kinto", 1000000, "2019-01-20", "Finance"\} \quad (1)$$

$$Employee \leftarrow Employee \cup \{2, "Jerry", "Kansxo", 6000000, "2019-01-15", "IT"\} \quad (2)$$

$$Employee \leftarrow Employee \cup \{3, "Philip", "Jose", 8900000, "2019-02-05", "Banking"\} \quad (3)$$

$$Employee \leftarrow Employee \cup \{4, "John", "Abraham", 2000000, "2019-02-25", "Insurance"\} \quad (4)$$

$$Employee \leftarrow Employee \cup \{5, "Michael", "Mathew", 2200000, "2019-02-18", "Finance"\} \quad (5)$$

$$Employee \leftarrow Employee \cup \{6, "Alex", "chreketo", 4000000, "2019 - 05 - 10", "IT"\} \quad (6)$$

$$Employee \leftarrow Employee \cup \{7, "Yohan", "Soso", 1230000, "2019 - 06 - 20", "Banking"\} \quad (7)$$

$$Reward \leftarrow Reward \cup \{1, "2019 - 05 - 11", 1000\} \quad (8)$$

$$Reward \leftarrow Reward \cup \{2, "2019 - 02 - 15", 5000\} \quad (9)$$

$$Reward \leftarrow Reward \cup \{3, "2019 - 04 - 22", 2000\} \quad (10)$$

$$Reward \leftarrow Reward \cup \{4, "2019 - 06 - 20", 8000\} \quad (11)$$

4.5.2 Consultas solicitadas

Obtener el tamaño del texto en todos los valores de la columna "Firstname".

$$\pi Firstname, LEN(Firstname) Employee \quad (12)$$

Obtener el nombre de todos los empleados después de reemplazar 'o' con 'hashtag'.

$$\pi REPLACE(Firstname, 'o', '#') Employee \quad (13)$$

Obtener el nombre y apellido de todos los empleados en una sola columna separados por "guion bajo"

$$\pi CONTACT(Firstname, '_', Lastname) Employee \quad (14)$$

Obtener el año, mes y día de la columna "Joiningdate"

$$\pi (YEAR(joiningdate), MONTH(Joiningdate), DAY(Joiningdate)) Employee \quad (15)$$

Obtener todos los empleados en orden ascendente por nombre.

$$\sigma Employee_{Firstname} SORT(Firstname) ASC \quad (16)$$

Obtener todos los empleados en orden descendente por nombre.

$$\sigma Employee_{Firstname} SORT(Firstname), DESC \quad (17)$$

Obtener todos los empleados en orden ascendente por nombre y en orden descendente por salario:

$$\sigma Employee_{Firstname} (SORT(Firstname) ASC, SORT(Salary) DESC) \quad (18)$$

Obtener todos los empleados con el nombre "Bob".

$$\sigma Firstname='BOB' Employee \quad (19)$$

Obtener todos los empleados con el nombre "Bob" o "Alex".

$$\sigma Firstname='BOB' \vee Firstname='Alex' Employee \quad (20)$$

Obtener todos los empleados que no tengan el nombre "Bob" o "Alex".

$$\sigma Firstname='BOB' \neq Firstname='Alex' Employee \quad (21)$$

4.5.3 ¿Que es una inyeccion SQL?

La inyección SQL es una vulnerabilidad común en aplicaciones web y sistemas que utilizan bases de datos. Permíteme explicarte:

- ¿Qué es la inyección SQL?
 - La inyección SQL[5] es una técnica de ataque en la que un ciberdelincuente inserta código malicioso en un sitio web a través de formularios o parámetros de consulta. Este código se mezcla con las sentencias SQL que el sitio utiliza para interactuar con su base de datos.
 - Básicamente, el atacante “inyecta” su propio código SQL en los campos de entrada del sitio web (como formularios de búsqueda o inicio de sesión). Si el sitio no está protegido adecuadamente, este código malicioso se ejecuta en la base de datos, permitiendo al atacante acceder, modificar o eliminar datos confidenciales.
 - Por ejemplo, si un sitio web tiene una consulta SQL como “SELECT * FROM usuarios WHERE nombre = ‘usuario’ AND contraseña = ‘contraseña’”, un atacante podría manipular el campo de entrada de “nombre” para introducir algo como “usuario’ OR ‘1’=‘1’ –”, lo que haría que la consulta devolviera todos los registros de usuarios sin necesidad de una contraseña válida.
- ¿Cómo se produce un ataque de inyección SQL?
 - El ataque de inyección SQL ocurre cuando un sitio web no valida o filtra adecuadamente los datos ingresados por los usuarios antes de ejecutar consultas SQL.
 - El atacante puede aprovechar esto para:
 - * Acceder a información confidencial (como contraseñas o datos personales).
 - * Modificar registros en la base de datos.
 - * Borrar datos.
 - * Incluso tomar el control total del sistema si la vulnerabilidad es grave.
- ¿Cómo protegerse contra la inyección SQL?
 - Sanitizar y validar todas las entradas de usuario antes de usarlas en consultas SQL.
 - Utilizar consultas parametrizadas o preparadas en lugar de concatenar directamente valores en las consultas.
 - Limitar los privilegios de acceso de la cuenta de base de datos utilizada por la aplicación.
 - Mantener el software actualizado y aplicar parches de seguridad.

En resumen, la inyección SQL es un riesgo importante, pero siguiendo buenas prácticas de seguridad, podemos proteger nuestros sistemas contra este tipo de ataques.

5. Conclusiones

En el contexto académico de las bases de datos, el álgebra relacional se presenta como una herramienta esencial. Durante esta práctica, he aprendido su relevancia al momento de insertar datos y he explorado algunas de las funciones que ofrece el lenguaje SQL. Entre estas funciones, encontramos elementos cruciales como WHILE, ASC, REPLACE y ORDER BY. Estas palabras reservadas me permitirán afinar y optimizar mis consultas en el futuro. Además de tener conocimiento del tema de inyeccion sql que hace mas interesante el tema de seguridad en base de datos.

6. Referencias

References

- [1] Algebra relacional. (n.d). <https://basesdatosdistribuidas.blogspot.com/2012/11/algebra-relacional.html>
- [2] Platzi. (n.d.). *Álgebra relacional y Bases de Datos*. <https://platzi.com/tutoriales/1566-bd/5823-algebra-relacional-y-bases-de-datos/>
- [3] VanMSFT. (n.d.). *Ejemplos de SELECT (Transact-SQL) - SQL Server*. Microsoft Learn. <https://learn.microsoft.com/es-es/sql/t-sql/queries/select-examples-transact-sql?view=sql-server-ver16>
- [4] colaboradores de Wikipedia. (2024, July 14). *Edgar Frank Codd*. Wikipedia, La Enciclopedia Libre. https://es.wikipedia.org/wiki/Edgar_Frank_Codd
- [5] Belcic, I. (2023, February 23). *¿Qué es la inyección de SQL y cómo funciona?* ¿Qué Es La Inyección De SQL Y Cómo Funciona? <https://www.avast.com/es-es/c-sql-injection>