REPORTE DE PRÁCTICA NO. 2

ÁLGEBRA RELACIONAL Y SQL (1)

ALUMNO:
Ariana García Melo

1. Introducción

En la actualidad, las bases de datos juegan un papel fundamental en la gestión y almacenamiento de la información dentro de cualquier empresa u organización. Gracias a ellas, se pueden manejar grandes volúmenes de datos de manera estructurada y eficiente. En este contexto, MySQL se presenta como una de las herramientas más populares y robustas para la gestión de bases de datos relacionales, debido a su facilidad de uso, su compatibilidad con múltiples plataformas y su gran rendimiento en el manejo de consultas y transacciones.

Esta práctica tiene como objetivo familiarizarnos con la creación, manipulación y consulta de bases de datos mediante MySQL. A través de una serie de ejercicios prácticos, aprenderemos a definir esquemas de datos, insertar registros y realizar diversas consultas utilizando sentencias SQL. Además, se establecerá la relación entre el álgebra relacional y SQL, comprendiendo cómo se traducen las operaciones matemáticas en instrucciones ejecutables dentro de un sistema de bases de datos.

Se trabajará con dos tablas: "Employee" y "Reward". La primera almacena información sobre los empleados de una empresa, incluyendo datos como el nombre, salario, fecha de ingreso y departamento al que pertenecen. La segunda tabla, "Reward", está relacionada con "Employee" mediante una clave foránea y registra las recompensas recibidas por los empleados en diferentes fechas. A partir de estas estructuras, se ejecutarán diversas consultas y operaciones que permitirán analizar los datos de manera eficiente.

2. Marco teórico

Álgebra Relacional

El álgebra relacional es el fundamento teórico sobre el que se construyen los sistemas de bases de datos relacionales. Es un lenguaje formal que permite expresar consultas y manipular datos mediante un conjunto de operaciones matemáticas. Estas operaciones incluyen:

- Selección: Permite extraer filas específicas de una tabla basándose en una condición.
- Proyección: Extrae columnas específicas de una tabla.
- Unión: Combina los resultados de dos tablas siempre que tengan la misma estructura.
- Intersección: Devuelve los registros comunes entre dos tablas.
- Diferencia: Muestra los registros que están en una tabla pero no en otra.
- Producto cartesiano: Combina todas las filas de una tabla con todas las filas de otra.
- Combinación o Join: Relaciona dos tablas basándose en una condición específica.

SQL

SQL (Structured Query Language) es el lenguaje de programación estándar para la gestión de bases de datos relacionales. Se utiliza para definir estructuras de datos, insertar, actualizar y eliminar registros, así como realizar consultas y generar reportes.

MySQL y Sentencias Utilizadas

MySQL es un sistema de gestión de bases de datos relacional que implementa SQL y es ampliamente utilizado en aplicaciones web y empresariales debido a su alto rendimiento, escalabilidad y confiabilidad. En esta práctica, utilizaremos las siguientes sentencias:

- CREATE TABLE: Para definir la estructura de las tablas.
- INSERT INTO: Para agregar registros a las tablas.
- SELECT: Para consultar información almacenada.
- Funciones de cadena y agregación: Como LOWER(), UPPER(), DISTINCT(), LEFT(), RTRIM(), LTRIM(), entre otras.

3. Herramientas empleadas

1. MySQL Server: Motor de bases de datos utilizado para ejecutar las consultas.
2. MySQL Workbench: Herramienta gráfica para diseñar y ejecutar consultas SQL.

4. Desarrollo

Ejercicios

El conjunto de ejercicios está basado en las tablas Employee y Reward.

Employee table

Employee_id	First_name	Last_name	Salary	Joining_date	Departement
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	2019-02-28	Finance
6	Alex	chreketo	4000000	2019-05-10	IT
7	Yohan	Soso	1230000	2019-06-20	Banking

Reward table

Employee_ref_id	date_reward	amount
1	2019-05-11	1000
2	2019-02-15	5000
3	2019-04-22	2000
1	2019-06-20	8000

1. Escribe la sintaxis para crear la tabla “Employee”.

Listing 1: Código.

```
CREATE TABLE Employee (
    Employee_id INT PRIMARY KEY,
    First_name VARCHAR(50),
    Last_name VARCHAR(50),
    Salary INT,
    Joining_date DATE,
```

```
    Departement VARCHAR(50)
);
```

2. Escribe la sintaxis para insertar 7 registros a la tabla “Employee”.

```
INSERT INTO Employee (Employee_id, First_name, Last_name, Salary, Joining_date, Departement) VALUES
(1, 'Bob', 'Kinto', 1000000, '2019-01-20', 'Finance'),
(2, 'Jerry', 'Kansxo', 600000, '2019-05-15', 'IT'),
(3, 'Philip', 'Jose', 800000, '2019-02-25', 'Banking'),
(4, 'John', 'Abraham', 900000, '2019-02-05', 'Insurance'),
(5, 'Michael', 'Matthew', 1200000, '2019-06-20', 'Finance'),
(6, 'Alex', 'chreketo', 4000000, '2019-05-10', 'IT'),
(7, 'Yohan', 'Soso', 1230000, '2019-06-20', 'Banking');
```

Figure 1: Código

Employee_id	First_name	Last_name	Salary	Joining_date	Departement
1	Bob	Kinto	1000000	2019-01-20	Finance
2	Jerry	Kansxo	600000	2019-05-15	IT
3	Philip	Jose	800000	2019-02-25	Banking
4	John	Abraham	900000	2019-02-05	Insurance
5	Michael	Matthew	1200000	2019-06-20	Finance
6	Alex	chreketo	4000000	2019-05-10	IT
7	Yohan	Soso	1230000	2019-06-20	Banking
NULL	NULL	NULL	NULL	NULL	NULL

Figure 2: Resultado

3. Escribe la sintaxis para crear la tabla “Reward”.

Listing 2: Código.

```
CREATE TABLE Reward (
    Employee_ref_id INT,
    date_reward DATE,
    amount INT,
    FOREIGN KEY (Employee_ref_id) REFERENCES Employee(Employee_id)
);
```

4. Escribe la sintaxis para insertar 4 registros (en la imagen) a la tabla “Reward”.

```
INSERT INTO Reward (Employee_ref_id, date_reward, amount) VALUES
(1, '2019-05-11', 1000),
(2, '2019-02-05', 5000),
(3, '2019-02-11', 2000),
(1, '2019-06-20', 8000);
```

Figure 3: Código

	Employee_ref_id	date_reward	amount
▶	1	2019-05-11	1000
	2	2019-02-05	5000
	3	2019-02-11	2000
	1	2019-06-20	8000

Figure 4: Resultado.

5. Obtener todos los empleados.

$$\pi_{*}(\text{Employee})$$

Listing 3: Código

```
SELECT * FROM Employee;
```

	Employee_id	First_name	Last_name	Salary	Joining_date	Departement
▶	1	Bob	Kinto	1000000	2019-01-20	Finance
	2	Jerry	Kansxo	600000	2019-05-15	IT
	3	Philip	Jose	800000	2019-02-25	Banking
	4	John	Abraham	900000	2019-02-05	Insurance
	5	Michael	Matthew	1200000	2019-06-20	Finance
	6	Alex	chreketo	4000000	2019-05-10	IT
	7	Yohan	Soso	1230000	2019-06-20	Banking
*	NULL	NULL	NULL	NULL	NULL	NULL

Figure 5: Resultado

6. Obtener el primer nombre y apellido de todos los empleados.

$$\pi_{\text{First\_name, Last\_name}}(\text{Employee})$$

Listing 4: Código

```
SELECT First_name, Last_name FROM Employee;
```

	First_name	Last_name
▶	Bob	Kinto
	Jerry	Kansxo
	Philip	Jose
	John	Abraham
	Michael	Matthew
	Alex	chreketo
	Yohan	Soso

Figure 6: Resultado

7. Obtener todos los valores de la columna “First-name” usando el alias “Nombre de empleado”.

$$\rho_{\text{"Nombre de empleado"}/\text{First\_name}}(\pi_{\text{First\_name}}(\text{Employee}))$$

Listing 5: Código

```
SELECT First_name AS 'Nombre de empleado' FROM Employee;
```

	Nombre de empleado
▶	Bob
	Jerry
	Philip
	John
	Michael
	Alex
	Yohan

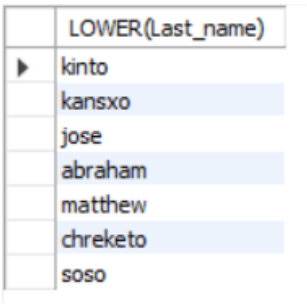
Figure 7: Resultado

8. Obtener todos los valores de la columna “Last-name” en minúsculas.

$$\pi_{\text{LOWER(Last\_name)}}(\text{Employee})$$

Listing 6: Código

```
SELECT LOWER(Last_name) FROM Employee;
```



	LOWER(Last_name)
▶	kinto
	kansxo
	jose
	abraham
	matthew
	chreketo
	soso

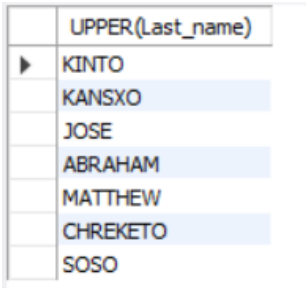
Figure 8: Resultado

9. Obtener todos los valores de la columna “Last-name” en mayúsculas.

$$\pi_{\text{UPPER(Last\_name)}}(\text{Employee})$$

Listing 7: Código

```
SELECT UPPER(Last_name) FROM Employee;
```



	UPPER(Last_name)
▶	KINTO
	KANSXO
	JOSE
	ABRAHAM
	MATTHEW
	CHREKETO
	SOSO

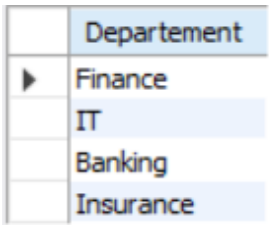
Figure 9: Resultado

10. . Obtener los nombre únicos de la columna “Departament”.

$$\delta(\pi_{\text{Departement}}(\text{Employee}))$$

Listing 8: Código

```
SELECT DISTINCT Departement FROM Employee;
```



	Departement
▶	Finance
	IT
	Banking
	Insurance

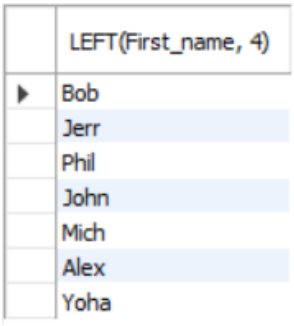
Figure 10: Resultado

11. Obtener los primeros 4 caracteres de todos los valores de la columna “First-name”

$$\pi_{\text{LEFT(First\_name, 4)}}(\text{Employee})$$

Listing 9: Código

```
SELECT LEFT(First_name, 4) FROM Employee;
```



	LEFT(First_name, 4)
▶	Bob
	Jerr
	Phil
	John
	Mich
	Alex
	Yoha

Figure 11: Resultado

12. Obtener la posición de la letra “h” en el nombre del empleado con First-name=“John”.

$$\pi_{\text{INSTR(First\_name, 'h')}}(\sigma_{\text{First\_name}='John'}(\text{Employee}))$$

Listing 10: Código

```
SELECT INSTR(First_name, 'h') AS 'Letra-H'
FROM Employee WHERE First_name = 'John';
```

	Letra H
▶	3

Figure 12: Resultado

13. Obtener todos los valores de la columna “First-name” después de remover los espacios en blanco de la derecha.

$$\pi_{\text{RTRIM(First\_name)}}(\text{Employee})$$

Listing 11: Código

```
SELECT RTRIM(First_name) FROM Employee;
```

	RTRIM(First_name)
▶	Bob
	Jerry
	Philip
	John
	Michael
	Alex
	Yohan

Figure 13: Resultado

14. Obtener todos los valores de la columna “First-name” después de remover los espacios en blanco de la izquierda.

$$\pi_{\text{LTRIM(First\_name)}}(\text{Employee})$$

Listing 12: Código

```
SELECT LTRIM(First_name) FROM Employee;
```

	LTRIM(First_name)
▶	Bob
	Jerry
	Philip
	John
	Michael
	Alex
	Yohan

Figure 14: Resultado

5. Conclusiones

A través de esta práctica, se logró una comprensión más profunda del uso de MySQL para la gestión de bases de datos. Se experimentó con la creación de tablas, la inserción de datos y la ejecución de consultas para recuperar información de manera eficiente. Además, se estableció la conexión entre el álgebra relacional y SQL, lo que permitió entender mejor cómo los conceptos teóricos se aplican en un sistema real de bases de datos.

Es fundamental comprender estas herramientas, ya que la gestión adecuada de la información es clave en cualquier sistema informático. Con este conocimiento, se pueden desarrollar aplicaciones más robustas, optimizar el rendimiento de consultas y garantizar la integridad de los datos almacenados.

Referencias Bibliográficas

References

[1] Silberschatz, A., Korth, H. F., Sudarshan, S. (**2011**). Database System Concepts. *McGraw-Hill*

[2] Documentación oficial de MySQL: `https://dev.mysql.com/doc/`