

REPORTE DE PRÁCTICA NO. 1.3

ÁLGEBRA RELACIONAL Y SQL

ALUMNO: ISRAEL CAMPOS VÁZQUEZ
Dr. Eduardo Cornejo-Velázquez



1. Introducción

En esta práctica pondremos en práctica la teoría vista en clase sobre álgebra relacional, y cómo se representa en sentencias SQL.

Crearemos 2 tablas, le insertaremos registros, y haremos consultas en álgebra relacional y en sentencias SQL.

2. Marco teórico

Álgebra Relacional

El álgebra relacional es un conjunto de operaciones matemáticas utilizadas para manipular y consultar datos en bases de datos relacionales. Permite realizar operaciones como selección, proyección, unión, intersección, diferencia, producto cartesiano y distintas formas de combinación de relaciones (joins).

Según Elmasri y Navathe (2016) en su libro "Fundamentals of Database Systems" [1]:

"El álgebra relacional es un modelo formal basado en el uso de operadores sobre relaciones, proporcionando un mecanismo para especificar consultas de manera precisa y estructurada."

SQL

SQL (Structured Query Language) es un lenguaje estándar diseñado para gestionar y manipular bases de datos relacionales. Permite la definición de estructuras de datos, la inserción, actualización, eliminación y recuperación de información.

Según Elmasri y Navathe (2016) en su libro Fundamentals of Database Systems [1]:

"SQL es un lenguaje declarativo que permite a los usuarios especificar qué datos desean recuperar o manipular sin necesidad de definir cómo hacerlo, basado en el modelo relacional de bases de datos."

MySQL

MySQL es un sistema de gestión de bases de datos relacional (RDBMS) de código abierto basado en SQL. Es ampliamente utilizado debido a su eficiencia, rapidez y compatibilidad con múltiples plataformas. Desarrollado originalmente por MySQL AB y ahora mantenido por Oracle Corporation, MySQL es una opción popular para aplicaciones web, gestión de datos empresariales y sistemas de almacenamiento de información.

Creación de Tablas (CREATE TABLE)

Para definir una tabla en MySQL, se usa la instrucción CREATE TABLE. Esta instrucción permite especificar los nombres de las columnas, sus tipos de datos y restricciones.

Inserción de Datos (INSERT INTO VALUES)

Una vez creada la tabla, se pueden insertar registros en ella usando la sentencia INSERT INTO VALUES.

Consulta de Datos (SELECT)

La instrucción SELECT se usa para recuperar información de una o varias tablas en la base de datos.

3. Herramientas empleadas

1. DataGrip. DataGrip es un entorno de desarrollo integrado (IDE) especializado en bases de datos, desarrollado por JetBrains. Está diseñado para facilitar la administración, consulta y desarrollo de bases de datos SQL en diversos motores, como MySQL, PostgreSQL, SQL Server, Oracle, SQLite, MariaDB, y más. Será el software elegido para manipular nuestra base de datos.
2. MySQL Server. MySQL Server es un sistema de gestión de bases de datos relacional (RDBMS) desarrollado por Oracle Corporation. Está basado en el lenguaje SQL (Structured Query Language) y permite almacenar, administrar y recuperar datos de manera eficiente. Será nuestro sistema elegido para almacenar nuestra base de datos.

4. Desarrollo

EJERCICIOS

1. Escribe la sintaxis para crear la tabla "Employee".
2. Escribe la sintaxis para insertar 7 registros (de la imagen) a la tabla "Employee".
3. Escribe la sintaxis para crear la tabla "Reward".
4. Escribe la sintaxis para insertar 4 registros (en la imagen) a la tabla "Reward".
5. Obtener todos los empleados.
6. Obtener el primer nombre y apellido de todos los empleados.
7. Obtener todos los valores de la columna "First name" usando el alias "Nombre de empleado".
8. Obtener todos los valores de la columna "Last name" en minúsculas.
9. Obtener todos los valores de la columna "Last name" en mayúsculas.
10. Obtener los nombres únicos de la columna "Department".
11. Obtener los primeros 4 caracteres de todos los valores de la columna "First name".
12. Obtener la posición de la letra "h" en el nombre del empleado con First name="Jhon".
13. Obtener todos los valores de la columna "First name" después de remover los espacios en blanco de la derecha.
14. Obtener todos los valores de la columna "First name" después de remover los espacios en blanco de la izquierda.

Expresión Álgebra Relacional

5. Employee(Employee)
6. firstName,lastName(Employee)
7. firstName→nombreDeEmpleado(Employee)
8. LOWER(lastName)→lastNameLower(Employee)
9. UPPER(lastName)→lastNameUpper(Employee)
10. department(Employee)
11. SUBSTRING(firstName,1,4)(Employee)
12. firstName=John(LOCATE(h,firstName)(Employee))
13. RTRIM(firstName)(Employee)
14. LTRIM(FirstName)(Employee)

Sentencia SQL

Listing 1: Crear tabla Employee.

```
create table Employee(  
    employeeId int not null,
```

```

    fist_name varchar(80) not null,
    last_name varchar(80) not null,
    salary float not null,
    joining_date date not null,
    departement varchar(80) not null,
    primary key (employeeId)
);

```

Listing 2: Insertar registros en tabla Employee.

```

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, '2019-02-28', 'Finance'),
    (6, 'Alex', 'Chr keto', 4000000, '2019-05-10', 'IT'),
    (7, 'Yohan', 'Soso', 1230000, '2019-06-20', 'Banking');

```

Listing 3: Crear tabla Reward.

```

create table Reward(
    employeeRefId int not null,
    date_reward date not null,
    amount float not null,
    foreign key (employeeRefId) references employee(employeeId)
);

```

Listing 4: Insertar registros en tabla Reward.

```

insert into Reward values(1, '2019-05-11', 1000),
    (2, '2019-02-15', 5000),
    (3, '2019-04-22', 2000),
    (1, '2019-06-20', 8000);

```

Listing 5: Obtener todos los empleados.

```

select * from Employee;

```

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

```

select fist_name, last_name from Employee;

```

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

```

select fist_name as nombre_de_empleado from Employee;

```

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

```

select lower(last_name) from Employee;

```

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

```

select upper(last_name) from Employee;

```

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

```

select distinct departement from Employee;

```

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

```
select substring(fist_name , 1, 4) FROM Employee;
```

Listing 12: Obtener la posición de la letra “h” en el nombre del empleado con First name

```
select locate('h', fist_name) FROM Employee WHERE fist_name = 'John';
```

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

```
select rtrim(fist_name) FROM Employee;
```

Listing 14: Obtener todos los valores de la columna “First_name” después de remover los espacios en blanco de la izquierda.

```
select ltrim(fist_name) FROM Employee;
```

Capturas de Pantalla

```
practica_1_3> create table Employee(  
    employeeId int not null,  
    fist_name varchar(80) not null,  
    last_name varchar(80) not null,  
    salary float not null,  
    joining_date date not null,  
    departement varchar(80) not null,  
    primary key (employeeId)  
)  
[2025-02-10 14:24:44] completed in 11 ms
```

Figure 1: Create table Employee

```
practica_1_3> 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, '2019-02-28', 'Finance'),  
    (6, 'Alex', 'Chrketo', 4000000, '2019-05-10', 'IT'),  
    (7, 'Yohan', 'Soso', 1230000, '2019-06-20', 'Banking')  
[2025-02-10 14:41:25] 7 rows affected in 4 ms
```

Figure 2: Insert values into Employee

```

practica_1_3> create table Reward(
    employeeRefId int not null,
    date_reward date not null,
    amount float not null,
    foreign key (employeeRefId) references employee(employeeId)
)
[2025-02-10 14:24:47] completed in 10 ms

```

Figure 3: Create table Reward

```

practica_1_3> insert into Reward values(1, '2019-05-11',1000),
                                         (2, '2019-02-15',5000),
                                         (3, '2019-04-22',2000),
                                         (1, '2019-06-20',8000)
[2025-02-10 15:27:37] 4 rows affected in 6 ms

```

Figure 4: Insert values into Reward

	employeeId	first_name	last_name	salary	joining_date	departement
1	1	Bob	Kinto	1000000	2019-01-20	Finance
2	2	Jerry	Kansxo	6000000	2019-01-15	IT
3	3	Philip	Jose	8900000	2019-02-05	Banking
4	4	John	Abraham	2000000	2019-02-25	Insurance
5	5	Michael	Mathew	2200000	2019-02-28	Finance
6	6	Alex	Chrketo	4000000	2019-05-10	IT
7	7	Yohan	Soso	1230000	2019-06-20	Banking

Figure 5: Select all employees





	 fist_name 	 last_name 
1	Bob	Kinto
2	Jerry	Kansxo
3	Philip	Jose
4	John	Abraham
5	Michael	Mathew
6	Alex	Chrketo
7	Yohan	Soso

Figure 6: Select first name and last name



	 nombre_de_empleado 
1	Bob
2	Jerry
3	Philip
4	John
5	Michael
6	Alex
7	Yohan

Figure 7: Select first name as nombre del empleado

	<input type="checkbox"/> `lower(last_name)`  
1	kinto
2	kansxo
3	jose
4	abraham
5	mathew
6	chrketo
7	soso

Figure 8: Select last name in lowercase

	<input type="checkbox"/> `upper(last_name)`  
1	KINTO
2	KANSXO
3	JOSE
4	ABRAHAM
5	MATHEW
6	CHRKETO
7	SOSO

Figure 9: Select last name in uppercase




	 departement  
1	Finance
2	IT
3	Banking
4	Insurance

Figure 10: Select department just once




	 `substring(first_name, 1, 4)`  
1	Bob
2	Jerr
3	Phil
4	John
5	Mich
6	Alex
7	Yoha

Figure 11: Select the first 4 characters of the first name

	<input type="checkbox"/> `locate('h', first_name)`	▼	↕
1			3

Figure 12: Locate letter h in first name

	<input type="checkbox"/> `rtrim(first_name)`	▼	↕
1	Bob		
2	Jerry		
3	Philip		
4	John		
5	Michael		
6	Alex		
7	Yohan		

Figure 13: Select first name without spaces on the left

	<code>`ltrim(fist_name)`</code>
1	Bob
2	Jerry
3	Philip
4	John
5	Michael
6	Alex
7	Yohan

Figure 14: Select first name without spaces on the right

5. Conclusiones

Aprendí con esta práctica muchas funciones nuevas que no sabía que se podía hacer en SQL, como obtener una consulta en mayúsculas o minúsculas, quitarle espacios en cualquier lado o especificando un lado, buscar un caracter en específico, seleccionar cierto número de caracteres, etc.

Además, aprendí como representar dichas consultas en álgebra relacional, y que las sentencias create e insert no pueden representarse en álgebra relacional.

Referencias Bibliográficas

References

- [1] Elmasri, R., Navathe, S. B. (2016). Fundamentals of Database Systems (7th ed.). Pearson.