

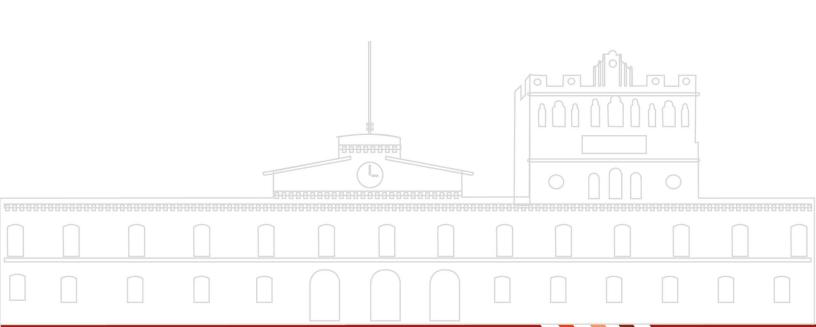


REPORTE DE PRÁCTICA NO. 1.3

Álgebra relacional y SQL 1

ALUMNO:

Daniel Monroy Garnica



1. Introducción

Esta práctica se trata de la aplicación del álgebra relacional al diseño y ejecución de consultas SQL. Con el uso de algunos ejercicios propuestos y aplicarlos con tablas SQL que creamos, se busca aprender a formular consultas complejas, optimizar la búsqueda de información y comprender el funcionamiento de diversas sentencias SQL y como representarlas con el álgebra relacional.

2. Marco teórico

Algrbra relacional

El álgebra relacional es un conjunto de operaciones y principios matemáticos utilizados en la manipulación y procesamiento de sistemas de bases de datos relacionales, utilizados principalmente para la consulta, gestión y recuperación eficiente de datos. El objetivo principal del Álgebra Relacional es proporcionar un marco formal y consistente para la ejecución de consultas, recuperación y manipulación de datos en sistemas de gestión de bases de datos relacionales.

SQL

El lenguaje de consulta estructurado (SQL) es un lenguaje de programación estandarizado y específico de dominio que sobresale en el manejo de relaciones de datos. Se emplea ampliamente para almacenar, manipular y recuperar datos en sistemas como MySQL, SQL Server y Oracle. SQL es utilizado por los administradores de bases de datos, desarrolladores y analistas de datos para tareas como definición de datos, control de acceso, uso compartido de datos, escritura de scripts de integración de datos y ejecución de consultas analíticas.

Sentencias de SQL

- CREATE DATABASE: Declaración se utiliza para crear una nueva base de datos SQL.
- CREATE TABLE: Se utiliza para crear una nueva tabla en una base de datos.
- INSERT: Agrega una o varias filas a una tabla.
- SELECT: Se utiliza para seleccionar datos de una base de datos.
- AS:Se utiliza para cambiar el nombre de una columna o tabla con un alias.
- LOWER: Devuelve una expresión de caracteres después de convertir en minúsculas los datos de caracteres en mayúsculas.
- UPPER: Devuelve una expresión de caracteres con datos de caracteres en minúsculas convertidos a mayúsculas.
- DISTINC: Solo devuelve valores distintos en el conjunto de resultados.
- LEFT: Devuelve la parte izquierda de una cadena de caracteres con el número de caracteres especificado.
- LOCATE: Devuelve la posición de la primera aparición de una subcadena en una cadena.
- RTRIM: Quita el carácter de espacio u otros caracteres especificados del final de una cadena.
- LTRIM: Quita el carácter de espacio u otros caracteres especificados del inicio de una cadena.

3. Herramientas empleadas

 MySQL Workbench es una herramienta visual unificada para arquitectos de bases de datos, desarrolladores y administradores de bases de datos. Esta proporciona modelado de datos, desarrollo de SQL y herramientas de administración integrales para configuración de servidores, administración de usuarios, copias de seguridad y mucho más.

4. Desarrollo

Álgebra relacional

```
    employee ← employee U {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','Baking'};
    reward ← reaward U {1,'2019-05-11',1000}, {2,'2019-02-15',5000}, {3,'2019-04-22',2000}, {4,'2019-06-20',8000};
    σ(employee)
    πfirst-name,last-name (employee)
    πfirst-name (ρ=Nombre-de-empleado(employee))
    δπdepartment (employee)
    δπdepartment (employee)
    πleft(first-name, 4) (employee)
    πleft(first-name) (σFirst-name=John(employee))
    πRTRIM (First-name) (employee)
    πLTRIM (First-name) (employee)
```

Sentencias SQL

Listing 1: Crear la base de datos las tablas y los registros para la practica.

create database EPracticos1;

create table employee
(
 employee_id int not null,
 First_name varchar(40) NOT NULL,
 Last_name varchar(40) NOT NULL,
 Salary int not null,
 Joining_date date not null,
 Department varchar(40) not null,
 PRIMARY KEY (employee_id)
);

insert into employee values (1,'Bob','Kinto',1000000,'2019-01-20','Finance'),
(2,'Jerry','Kansxo',60000000,'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', 'Baking');
create table reward
    employee_ref_id int not null,
    date_reward date not null,
    amount int not null,
   PRIMARY KEY (employee_ref_id)
);
insert into reward values (1, '2019-05-11', 1000), (2, '2019-02-15', 5000),
(3, 2019-04-22, 2000), (4, 2019-06-20, 8000);
             Listing 2: Consultas a la base de datos
select * from employee;
select first_name , last_name from employee;
select first_name as Nombre_de_empleado from employee;
select lower(last_name) from employee;
select upper(last_name) from employee;
select DISTINCT department from employee;
select left(first_name, 4) from employee;
select locate ('h', First_name) from employee where first_name='John';
select RTRIM (First_name) from employee;
select LTRIM (First_name) from employee;
```

Resultados de las consultas

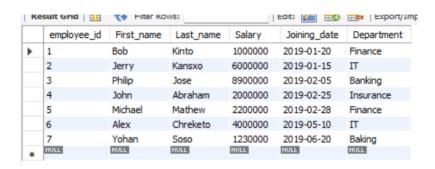


Figure 1: select * from employee;

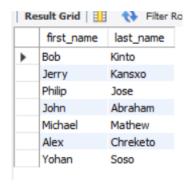
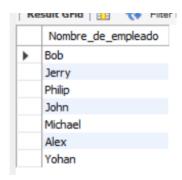


Figure 2: select first-name, last-name from employee;



 $Figure \ 3: \ select \ first-name \ as \ Nombre-de-empleado \ from \ employee;$

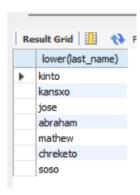


Figure 4: select lower(last-name) from employee;



Figure 5: select upper(last-name) from employee;

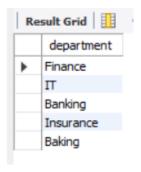


Figure 6: select DISTINCT department from employee;

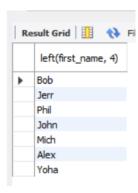


Figure 7: select left(first-name, 4) from employee;

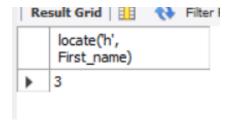


Figure 8: select locate('h', First-name) from employee where first-name='John';

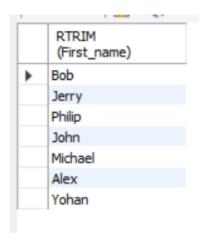


Figure 9: select RTRIM (First-name) from employee;

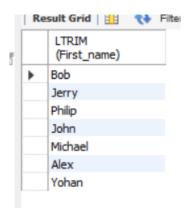


Figure 10: select LTRIM (First-name) from employee;

5. Conclusiones

Por lo que hecho en esta practica pude comprender de mejor manera el uso de sentencias en SQL y también como es que el álgebra relacional puede ayudar a dar una idea de como se van a realizar las consultas de una base de datos.

Referencias Bibliográficas

References

- [1] Algebra relacional AppMaster.($\mathbf{s.f.}$). AppMaster Ultimate All-in No-code Platform. https://appmaster.io/es/glossary/algebra-relacional-es
- [2] Ibm.(2025, 16 enero). SQL. ¿Qué es SQL? https://www.ibm.com/mx-es/think/topics/structured-query-language
- [3] MikeRayMSFT.($\mathbf{s.f.}$). Documentación de Microsoft SQL SQL Server. Microsoft Learn. https://learn.microsoft.com/es-es/sql/?view=sql-server-ver16
- [4] W3Schools.com.(s.f.).https://www.w3schools.com/sql/default.asp
- [5] MySQL :: MySQL Workbench. (s.f.). https://www.mysql.com/it/products/workbench/