

# REPORTE DE PRÁCTICA NO. 1

## Ejercicios 1

ALUMNO:

Jordan Gael Sosa De la Cruz  
Hassiel Camacho Meneses



## 1. Introducción

Escribir y ejecutar sentencias SQL sobre las tablas Employee y Reward, además de presentar sus correspondientes en álgebra relacional y su ejecución en MySQL.

## 2. Marco teórico

### Álgebra Relacional Extendida

El álgebra relacional[1] es un lenguaje formal de manipulación de datos que opera sobre relaciones (tablas) en una base de datos. Permite consultar, combinar y transformar los datos mediante un conjunto de operaciones básicas que producen nuevas relaciones como resultado. Entre sus operaciones fundamentales se incluyen:

1. Selección ( $\sigma$ ): extrae filas que cumplen una condición específica.
2. Proyección ( $\Pi$ ): extrae columnas específicas de una relación.
3. Unión ( $\cup$ ): combina todas las tuplas de dos relaciones compatibles.
4. Diferencia ( $-$ ): obtiene tuplas que están en una relación y no en otra.
5. Producto cartesiano ( $\times$ ): combina todas las tuplas de dos relaciones.
6. Renombramiento ( $\rho$ ): cambia el nombre de las relaciones o de sus atributos.
7. Join ( $\bowtie$ ): combina relaciones basándose en una condición de coincidencia.

El álgebra relacional sirve como base teórica para los sistemas de bases de datos relacionales y como modelo de consulta, ya que cualquier consulta SQL puede representarse mediante expresiones de álgebra relacional.

### SQL

SQL (Structured Query Language)[1] es el lenguaje estándar para interactuar con bases de datos relacionales. Permite definir, manipular y controlar los datos mediante tres tipos de sublenguajes:

1. DDL (Data Definition Language): para crear, modificar o eliminar estructuras de la base de datos (tablas, índices, etc.).
2. DML (Data Manipulation Language): para insertar, actualizar, eliminar y consultar datos.
3. DCL (Data Control Language): para controlar permisos y accesos a la base de datos.

SQL combina consultas declarativas (el usuario especifica qué datos quiere, no cómo obtenerlos) con operaciones sobre relaciones, y se basa en conceptos del modelo relacional, siendo el puente entre la teoría del álgebra relacional y la implementación práctica en sistemas como MySQL, Oracle o SQL Server.

### 3. Herramientas empleadas

1. MySQL Server. Es un sistema de gestión de bases de datos relacional (RDBMS) de código abierto desarrollado originalmente por MySQL AB y ahora mantenido por Oracle. Permite almacenar, organizar y recuperar datos de manera eficiente usando el lenguaje SQL (Structured Query Language). MySQL es conocido por su alta velocidad, fiabilidad y facilidad de uso, y es ampliamente utilizado en aplicaciones web, sistemas de software y entornos empresariales para manejar bases de datos de todo tamaño.

## 4. Desarrollo

### Análisis de requisitos

Realizar las sentencias SQL que satisfagan cada punto, adjuntando su resultado y su representación en álgebra relacional. Como base se utilizarán las tablas de la Figura 1.

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

Figure 1: Tablas del caso.

El siguiente listado describe las tareas a realizar con las tablas dadas.

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 nombre únicos de la columna “Departament”.
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 = “John”.
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.

## Álgebra relacional

El siguiente listado muestra las representaciones en álgebra relacional de las tareas dadas anteriormente.


1. No hay representación para la creación de tablas.
2.  $Employee \leftarrow Employee \cup E$
3. No hay representación para la creación de tablas.
4.  $Reward \leftarrow Reward \cup E$
5.  $\sigma(Employee)$
6.  $\Pi_{First\_name, Last\_name}(Employee)$
7.  $\Pi_{First\_name \rightarrow Nombre\_de\_empleado}(Employee)$
8.  $\Pi_{LOWER(Last\_name)}(Employee)$
9.  $\Pi_{UPPER(Last\_name)}(Employee)$
10.  $\Pi_{Departement}(Employee)$
11.  $\Pi_{SUBSTRING(First\_name, 1, 4)}(Employee)$
12.  $\Pi_{LOCATE('h', First\_name)}(\sigma_{First\_name = 'John'}(Employee))$
13.  $\Pi_{RTRIM(First\_name)}(Employee)$
14.  $\Pi_{LTRIM(First\_name)}(Employee)$

## Sentencias SQL

En el siguiente listado se presentan las sentencias SQL que satisfacen las tareas dadas anteriormente.

Listing 1: Crear tabla Employee en Figura 2

```
CREATE TABLE Employee (  
    Employee_id INT PRIMARY KEY,  
    First_name VARCHAR(255) ,  
    Last_name VARCHAR(255) ,  
    Salary FLOAT,  
    Joining_date DATE,  
    Departement VARCHAR(255));
```

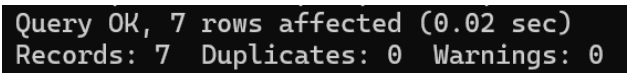


```
Query OK, 0 rows affected (0.04 sec)
```

Figure 2: Ejecución en MySQL.

Listing 2: Poblar tabla Employee en Figura 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', 'chreketo', 4000000, '2019-05-10', 'IT'),  
(7, 'Yohan', 'Soso', 1230000, '2019-06-20', 'Banking');
```



```
Query OK, 7 rows affected (0.02 sec)  
Records: 7 Duplicates: 0 Warnings: 0
```

Figure 3: Ejecución en MySQL.

Listing 3: Crear tabla Reward en Figura 4

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



```
Query OK, 0 rows affected (0.04 sec)
```

Figure 4: Ejecución en MySQL.

Listing 4: Poblar tabla Reward en Figura 5

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

```
Query OK, 4 rows affected (0.00 sec)
Records: 4  Duplicates: 0  Warnings: 0
```

Figure 5: Ejecución en MySQL.

Listing 5: Seleccionar todo de la tabla Employee en Figura 6

```
SELECT * FROM Employee;
```

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

7 rows in set (0.00 sec)

Figure 6: Ejecución en MySQL.

Listing 6: Seleccionar nombres y apellidos de la tabla Employee en Figura 7

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

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

7 rows in set (0.00 sec)

Figure 7: Ejecución en MySQL.

Listing 7: Seleccionar nombres y usando un alias en la tabla Employee en Figura 8  
**SELECT** First\_name **AS** "Nombre-de-empleado" **FROM** Employee;

```
+-----+  
| Nombre de empleado |  
+-----+  
| Bob                |  
| Jerry              |  
| Philip              |  
| John                |  
| Michael             |  
| Alex                |  
| Yohan               |  
+-----+  
7 rows in set (0.00 sec)
```

Figure 8: Ejecución en MySQL.

Listing 8: Seleccionar apellidos en minúsculas en la tabla Employee en Figura 9  
**SELECT LOWER**(Last\_name) **FROM** Employee;

```
+-----+  
| LOWER(Last_name) |  
+-----+  
| kinto             |  
| kansxo             |  
| jose              |  
| abraham            |  
| mathew             |  
| chreketo           |  
| soso               |  
+-----+  
7 rows in set (0.01 sec)
```

Figure 9: Ejecución en MySQL.



Listing 9: Seleccionar apellidos en mayúsculas en la tabla Employee en Figura 10  
**SELECT UPPER( Last\_name ) FROM Employee ;**

```
+-----+
| UPPER( Last_name ) |
+-----+
| KINTO               |
| KANSXO              |
| JOSE                |
| ABRAHAM             |
| MATHEW              |
| CHREKETO            |
| SOSO                |
+-----+
7 rows in set (0.00 sec)
```

Figure 10: Ejecución en MySQL.

Listing 10: Seleccionar departamentos únicos en la tabla Employee en Figura 11  
**SELECT DISTINCT Departement FROM Employee ;**

```
+-----+
| Departement         |
+-----+
| Finance             |
| IT                  |
| Banking             |
| Insurance           |
+-----+
4 rows in set (0.01 sec)
```

Figure 11: Ejecución en MySQL.

Listing 11: Seleccionar los primeros 4 caracteres de los nombres en la tabla Employee en Figura 12  
**SELECT SUBSTRING( First\_name , 1, 4) FROM Employee ;**

```
+-----+
| SUBSTRING( First_name, 1, 4) |
+-----+
| Bob                          |
| Jerr                         |
| Phil                        |
| John                        |
| Mich                       |
| Alex                       |
| Yoha                       |
+-----+
7 rows in set (0.01 sec)
```

Figure 12: Ejecución en MySQL.

Listing 12: Seleccionar la posición de la letra 'h' en los nombres en la tabla Employee donde el nombre es igual a 'John' en Figura 13

```
SELECT LOCATE( 'h' , First_name) FROM Employee WHERE First_name = 'John' ;
```

```
+-----+
| LOCATE('h', First_name) |
+-----+
|                        3 |
+-----+
1 row in set (0.01 sec)
```

Figure 13: Ejecución en MySQL.

Listing 13: Seleccionar los nombres en la tabla Employee eliminando los espacios a la derecha en Figura 14

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

```
+-----+
| RTRIM(First_name) |
+-----+
| Bob               |
| Jerry             |
| Philip            |
| John              |
| Michael           |
| Alex              |
| Yohan             |
+-----+
7 rows in set (0.01 sec)
```

Figure 14: Ejecución en MySQL.

Listing 14: Seleccionar los nombres en la tabla Employee eliminando los espacios a la izquierda en Figura 15

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

```
+-----+
| LTRIM(FIRST_name) |
+-----+
| Bob               |
| Jerry             |
| Philip            |
| John              |
| Michael           |
| Alex              |
| Yohan             |
+-----+
7 rows in set (0.01 sec)
```

Figure 15: Ejecución en MySQL.

## 5. Conclusiones

En esta actividad hemos descubierto nuevas funciones que se pueden utilizar para hacer consultas SQL, además de reforzar el uso del editor de texto LaTeX.

En relación con el álgebra relacional fue interesante ver que existía una extensión que incluye funciones que existen en SQL y que están reconocidos en su notación.

## Referencias Bibliográficas

## References

- [1] Silberschatz, A., Korth, H. F., & Sudarshan, S. (2006). Database system concepts (5th ed.). McGraw-Hill.