**Autoestudio #1**

**INVESTIGACION**

1. **SQL**
   1. **¿Qué es? ¿Para qué sirve?**

SQL (Structured Query Language) es un lenguaje declarativo de programación, basado en el cálculo y algebra relacional, que sirve para la administración de sistemas de gestión de bases de datos relacionales.

* 1. **¿Qué es DML, DLL, DCL, TCL?**

**DML (Data Manipulation Language)**

Es un lenguaje proporcionado por los sistemas gestores de bases de datos que permite a los usuarios de la misma llevar a cabo las tareas de consulta o modificación de los datos contenidos en las Bases de Datos del Sistema Gestor de Bases de Datos.

**DDL (Data Definition Language)**

Es un lenguaje proporcionado por el sistema de gestión de base de datos que permite a los programadores de la misma llevar a cabo las tareas de definición de las estructuras que almacenarán los datos así como de los procedimientos o funciones que permitan consultarlos.

**DCL (Data Control Language)**

Es un lenguaje proporcionado por el Sistema de Gestión de Base de Datos que incluye una serie de comandos SQL que permiten al administrador controlar el acceso a los datos contenidos en la Base de Datos.

**TCL (Transaction Control Language)**

Es un lenguaje de programación y un subconjunto de SQL, que se utiliza para controlar el procesamiento de transacciones en una base de datos.

* 1. **En este laboratorio, ¿en qué escribimos? ¿Por qué?**

En este laboratorio escribimos en algebra, calculo y en SQL, para así darnos la capacidad de entender el cómo se maneja el lenguaje y así poder tener unas bases sólidas sobre lo que manejaremos en el curso.

1. **Motor de bases de datos y bases de datos** 
   1. **¿Qué son?**

El Motor de base de datos es el servicio principal para almacenar, procesar y proteger los datos. El Motor de base de datos proporciona acceso controlado y procesamiento de transacciones rápido para cumplir con los requisitos de las aplicaciones consumidoras de datos más exigentes de su empresa.

* 1. **¿Qué motores ofrece sqlzoo.net [http://sqlzoo.net/]?**

SQL Server, Oracle, MySQL, DB2, and PostgreSQL.

* 1. **¿Qué bases de datos ofrece sqlzoo?**

Las bases de datos que ofrece sqlzoo son world y nobel.

**PRACTICA**

**Para cada una de las secciones indicadas:**

**a. Estudien la información de referencia de la sección**

**b. Escriban las consultas propuestas en cálculo y algebra.**

**c. Propongan una nueva consulta que use lo nuevo de la sección y escribanla en SQL**

1. **SQL INTRO**

SELECT \* FROM Customers;

**Calculo:**{X: Customers|x}

**Algebra:** Π (Customers)

1. **SQL SELECT**

**Calculo:** { x:Customers|x.CustomersName,x.City }

**Algebra:** Π CustomersName,City (Customers)

**SQL:** SELECT CustomerID, ContactName, Address FROM Customers;

1. **SQL SELECT DISTINCT**

**Calculo:** { x:Customers|x.CustomersName,x.City }

**Algebra:** Π Country (Customers)

**Calculo:** {+x:Customers| x.Country}

**Algebra:** No se puede sumar por algebra

**SQL:** SELECT DISTINCT City FROM Customers;

1. **SQL WHERE**

**Calculo:** { x:Customers |x.Country=’Mexico’: x}

**Algebra:** σ Country=’Mexico’(Customers)

**Calculo:** { x:Customers |x.CustomerID=1: x}

**Algebra:** σ CustomerID=1 (Customers)

**SQL:** SELECT \* FROM Customers

WHERE Country != 'Germany';

1. **SQL AND, OR, NOT**

**Calculo:** { x:Customers|x.Country='Germany' ∧ x.City='Berlin': x}

**Algebra:** σ Country='Germany' ∧ City='Berlin' (Customers)

**Calculo:** { x:Customers|x.City='Berlin' ∨ x.City='München': x}

**Algebra:** σ City='Berlin' ∨ City='München' (Customers)

**Calculo:** { x:Customers|¬Country='Germany': x}

**Algebra:** σ ¬Country='Germany' (Customers)

**Calculo:** { x:Customers|Country='Germany' ∧(City='Berlin' ∨ City='München'): x}

**Algebra:** σ Country='Germany' ∧ (City='Berlin' ∨ City='München') (Customers)

**Calculo:** { x:Customers|¬Country='Germany' ∧ ¬Country='USA': x}

**Algebra:** σ ¬Country='Germany' ∧ ¬Country='USA' (Customers)

**SQL:** SELECT \* FROM Customers

WHERE City != 'London' AND (Country = 'Mexico' or Country = 'France');

1. **SQL ORDER BY**

**Calculo:** No se puede calcular.

**Algebra:** No se puede calcular.

**SQL:** SELECT CustomerID,CustomerName, City, Country FROM Customers

WHERE Country = 'USA'

ORDER BY CustomerID DESC;

1. **SQL SELECT TOP, LIMIT, ROWNUN**

**Calculo:** {+ x:Customers|ROWNUM <= 3: x}

**Algebra:** σ ROWNUM <= 3(Customers)

**Calculo:** { x:Customers | Country='Germany' ∧ ROWNUM <= 3: x}

**Algebra:** σ Country='Germany' ∧ ROWNUM <= 3 (Customers)

**SQL:** SELECT \* FROM Customers

WHERE PostalCode>=05022

LIMIT 10;

1. **SQL MIN(), MAX()**

**Calculo:** {↓ x:Products|x.Price}

**Algebra:** No se puede calcular.

**Calculo:** {↑ x:Products|x.Price}

**Algebra:** No se puede calcular.

**SQL:** SELECT ProductName, MIN(SupplierID)

FROM Products;

1. **SQL COUNT, AVG, SUM**

**Calculo:** {+ x:Products|x.ProductID:1}

**Algebra:** No se puede hacer sumas en algebra

**Calculo:** {+ x:Products|x.ProductID:1}/2

**Algebra:** No se puede hacer sumas en algebra

**Calculo:** {+ x:OrderDetails|x.Quantity}

**Algebra:** No se puede hacer sumas en algebra

**SQL:** SELECT SUM(ProductID)

FROM Products

WHERE Price >= 23.5;

1. **SQL LIKE**

**SQL:** SELECT \* FROM Customers

WHERE CustomerID LIKE '1%';

1. **SQL WILCARDS**

**SQL:** SELECT \* FROM Customers

WHERE Country NOT LIKE '%[agscm]';

1. **SQL BETWEEN**

**Calculo:** { x:Products|Price>=10 ∧ Price<=20: x}

**Algebra:** σ Price>=10 ∧ Price<=20 (Products)

**Calculo:** { x:Products| Price<=10 ∧ Price>=20: x}

**Algebra:** σ Price>=10 ∧ Price<=20 (Products)

**SQL:** SELECT \* FROM Products

WHERE ProductName NOT BETWEEN 'Ikura' AND 'Tourtière'

ORDER BY Price ASC;

1. **SQL GROUP BY**

**SQL:** SELECT COUNT(CustomerID), City

FROM Customers

GROUP BY City;

1. **SQL HAVING**

**SQL:** SELECT AVG(CustomerID), City

FROM Customers

GROUP BY City

HAVING AVG(CustomerID) >30

ORDER BY AVG(CustomerID);

1. **SQL COMMENTS**

**SQL:** SELECT AVG(CustomerID), City

--DE LA TABLA CUSTOMERS

FROM Customers

--AGRUPA POR LAS CIUDADES

GROUP BY City

--EN LAS QUE EL PROMEDIO ES MAYOR A 30

HAVING AVG(CustomerID) >30

--ORDENADO RESPECTO AL PROMEDIO

ORDER BY AVG(CustomerID);

**Proponiendo w3scools**

1. **8 Consultas: una para cada uno de los tipos de operadores.**
2. **Restringir:**

SELECT CategoryName

FROM Categories

WHERE CategoryName > "H"

SELECT Description

FROM Categories

WHERE CategoryID IN (1,2,3,4)

SELECT CategoryName

FROM Categories

WHERE CategoryName LIKE "B%"

SELECT ProductName

FROM Products

WHERE SupplierID > 10 AND Price < 25

SELECT Unit

FROM Products

WHERE SupplierID IN (1,3,5,7,9,11,13) AND Price < 50

SELECT ProductID

FROM Products

WHERE 0< ProductID AND ProductID<15

SELECT ContactName

FROM Suppliers

WHERE Country in ("UK", "USA")

SELECT PostalCode

FROM Suppliers

WHERE Country LIKE ("U%")

ORDER BY PostalCode DESC

1. **Proyectar:**

SELECT CategoryID, Description

FROM Categories

SELECT CategoryName

FROM Categories

ORDER BY CategoryName ASC

SELECT ProductName

FROM Products

SELECT MAX(Price)

FROM Products

SELECT SUM(Price), SUM(ProductID)

FROM Products

SELECT ProductName, MAX(Price)

FROM Products

GROUP BY ProductName

SELECT City, Country

FROM Suppliers

ORDER BY City ASC, Country DESC

SELECT City, Country

FROM Suppliers

GROUP BY City, Country

1. **Multiplicar:**

SELECT City

FROM Suppliers, Categories

WHERE CategoryID = SupplierID

SELECT SUM(SupplierID), SUM(CategoryID)

FROM Suppliers, Categories

WHERE SupplierID != CategoryID

SELECT SupplierName

FROM Suppliers, Categories

WHERE SupplierName LIKE "E%" OR CategoryName LIKE "C%"

SELECT MAX(SupplierID,CategoryID)

FROM Suppliers, Categories

WHERE SupplierID<CategoryID

SELECT \*

FROM Suppliers, Products

WHERE Country = "USA" OR Price>=20 AND Price<=100

SELECT Description

FROM Categories, Products

WHERE Categories.CategoryID>4 AND ProductName LIKE "%s"

SELECT MIN(Price)

FROM Suppliers, Products

WHERE Country = "France"

SELECT \*

FROM Suppliers, Categories

WHERE SupplierID = CategoryID

ORDER BY Phone ASC

1. **3 Consultas anidadas que usen otra consulta (SELECT …) en FROM, en SELECT en WHERE**
2. SELECT (SELECT CategoryID FROM categories WHERE categoryName>'F' AND CategoryName<'K' )

FROM Categories

1. SELECT ProductID

FROM (SELECT ProductID, Price FROM Products WHERE Price>24)

WHERE ProductID <40caca

1. SELECT Address

FROM Suppliers

WHERE Country in (SELECT Country FROM Suppliers WHERE City in ('Londona', 'Berlin', 'New Orleands'))

**Bibliografía:**

<https://es.wikipedia.org/wiki/SQL>

<https://es.wikipedia.org/wiki/Lenguaje_de_manipulaci%C3%B3n_de_datos>

<https://es.wikipedia.org/wiki/Lenguaje_de_definici%C3%B3n_de_datos>

<https://es.wikipedia.org/wiki/Lenguaje_de_control_de_datos>

<http://robertodiazg.blogspot.com/2014/03/sentencias-sql-ddl-dml-dcl-y-tcl.html>

<https://prezi.com/ry9ckaivktcx/motores-de-base-de-datos/>