**Plan de Clases: Base de Datos II - MongoDB**

**Clase 1: Introducción y Repaso de SQL (Duración: 4 horas)**

**Objetivos:**

* Recordar conceptos clave de bases de datos relacionales.
* Revisar consultas avanzadas en SQL.
* Introducir bases de datos NoSQL y su importancia.

**Horario Detallado:**

**Bloque 1 (0:00 - 1:00 horas): Introducción y Repaso de SQL**

1. **Bienvenida y presentación del curso (15 min)**

#### ****Bloque 1 (0:00 - 1:00 horas): Introducción y Repaso de SQL****

##### **1. Bienvenida y Presentación del Curso**

**Propósito:** Iniciar la clase con una introducción amigable y una breve explicación sobre los temas que se tratarán en el curso.

* **Bienvenida:**
  + "¡Bienvenidos al curso de Base de Datos II! Durante este semestre,
  + **Presentación del docente y estudiantes:**
    - Breve introducción sobre tu experiencia en bases de datos y lo que los estudiantes pueden esperar aprender durante el curso.
    - Pregunta abierta a los estudiantes: "¿Quiénes han trabajado antes con bases de datos relacionales? ¿Alguien ha usado MongoDB antes?"
* **Explicación de la Metodología del Curso:**
  + Este curso combinará **teoría** y **práctica**. Habrá una parte teórica donde veremos conceptos y ejemplos, y una parte práctica donde trabajaremos en ejercicios y proyectos reales.
  + "Vamos a alternar entre explicaciones teóricas y actividades prácticas. Al final del curso, ustedes tendrán una comprensión sólida tanto de las bases de datos relacionales como NoSQL."
* **Importancia de las Bases de Datos:**
  + "Las bases de datos son fundamentales en la vida moderna. Casi todo lo que hacemos en línea, desde comprar en una tienda en línea hasta ver nuestras redes sociales, depende de bases de datos."
  + "Hoy en día, las bases de datos no solo deben manejar grandes volúmenes de datos, sino que también deben ser rápidas, escalables y adaptarse a necesidades cambiantes. Es por esto que existen bases de datos tanto relacionales como NoSQL."

1. **Repaso de conceptos básicos de SQL (45 min)**
   * "Las bases de datos relacionales organizan la información en tablas con claves primarias y foráneas."
   * **Diferencias entre SQL y MongoDB:**
     + **Estructura:** SQL usa tablas con filas y columnas; MongoDB usa documentos JSON dentro de colecciones.
     + **Relaciones:** En SQL, las relaciones son mediante claves foráneas; en MongoDB, los documentos pueden embeberse unos dentro de otros.
     + **Escalabilidad:** SQL escala verticalmente (añadir más potencia al servidor), mientras que MongoDB escala horizontalmente (añadir más servidores).
     + **Consultas:** SQL usa SELECT, JOIN, etc., mientras que MongoDB usa find(), aggregate(), etc.
     + **Transacciones:** SQL sigue estrictamente ACID; MongoDB permite transacciones pero está más optimizado para grandes volúmenes de datos distribuidos.
   * **Explicación detallada de claves primarias y foráneas:**
     + Clave primaria: "Es un identificador único para cada fila de una tabla."
     + Clave foránea: "Es una referencia a una clave primaria en otra tabla, lo que permite establecer relaciones."
   * **Consultas básicas y multi-tablas con JOIN:**
     + Explicación de INNER JOIN, LEFT JOIN, RIGHT JOIN y FULL JOIN.
     + Ejemplo de consulta con INNER JOIN:

SELECT empleados.nombre, departamentos.nombre

FROM empleados

JOIN departamentos ON empleados.id\_departamento = departamentos.id;

* Análisis de cómo optimizar estas consultas.

**Práctica:**

* Realizar ejercicios de JOIN con diferentes combinaciones de tablas.
* Optimizar consultas y comparar su rendimiento con EXPLAIN.

**Bloque 2 (1:00 - 2:00 horas): Consultas Avanzadas en SQL**

1. **Optimización de consultas (30 min)**
   * "Para mejorar el rendimiento de nuestras consultas, utilizamos índices. Veamos cómo crearlos y analizarlos."
   * Explicación del uso de EXPLAIN para analizar consultas.

EXPLAIN SELECT \* FROM empleados WHERE nombre = 'Juan';

* Creación y uso de índices:

CREATE INDEX idx\_nombre ON empleados(nombre);

1. **Subconsultas y funciones agregadas (30 min)**
   * "Las subconsultas permiten hacer consultas dentro de consultas."
   * Ejemplo de subconsulta anidada:

SELECT nombre FROM empleados WHERE id\_departamento =

(SELECT id FROM departamentos WHERE nombre = 'Ventas');

* Explicación y uso de GROUP BY, HAVING, COUNT(), AVG(), SUM().

SELECT id\_departamento, COUNT(\*)

FROM empleados

GROUP BY id\_departamento

HAVING COUNT(\*) > 5;

**Práctica:**

* Resolver ejercicios de subconsultas y funciones agregadas.

**Bloque 3 (2:00 - 3:00 horas): Introducción a NoSQL**

1. **Limitaciones de las bases de datos relacionales (20 min)**
   * "¿Qué pasa cuando necesitamos escalar una base de datos? SQL puede ser limitante."
   * Problemas de escalabilidad y rendimiento.
   * Limitaciones en el manejo de grandes volúmenes de datos y concurrencia.
2. **Introducción a bases de datos NoSQL (40 min)**
   * "Existen distintos tipos de bases de datos NoSQL: Documentos, Clave-Valor, Columnar y Grafos."
   * Comparación con bases de datos relacionales.
   * "MongoDB es una base de datos NoSQL orientada a documentos."
   * Explicación sobre JSON y BSON como formato de almacenamiento de MongoDB.

**Práctica:**

* Identificación de escenarios en los que NoSQL es preferible sobre SQL.
* Discusión sobre los casos de uso de MongoDB en proyectos reales.

**Bloque 4 (3:00 - 4:00 horas): Implementación Práctica y Conclusión**

1. **Introducción a MongoDB y su ecosistema (30 min)**
   * "Vamos a instalar MongoDB y explorar sus herramientas."
   * Explicación de MongoDB Compass, Shell y VS Code.
   * Pasos para instalar MongoDB en Windows y Linux.
2. **Primeros pasos con MongoDB (30 min)**
   * Creación de la primera base de datos y colección en MongoDB Compass.

use miBaseDeDatos

* Inserción de documentos en MongoDB:

db.miColeccion.insertOne({"nombre": "Juan", "edad": 30})

* Explicación de operaciones básicas: insertOne, find, updateOne, deleteOne.

**Práctica:**

* Instalación guiada de MongoDB.
* Creación de bases de datos y colecciones en MongoDB Compass.
* Inserción y consulta de documentos en MongoDB Shell.

**Cierre de la clase:**

* Recapitulación de los conceptos vistos.
* Espacio para preguntas y resolución de dudas.
* "En la siguiente clase, veremos en detalle la arquitectura de MongoDB."

Hostname: cluster0-shard-00-00-jxeqq.mongodb.net

Authentication: Username / Password

Username: m001-student

Password: m001-mongodb-basics

Replica Set Name: Cluster0-shard-0

Read Preference: Primary Preferred

mongodb+srv://m001-student:m001-mongodb-basics@cluster0-jxeqq.mongodb.net/test?retryWrites=true&w=majority

**Comandos Básicos de MongoDB por Consola**

**1. Conectarse a MongoDB**

Si MongoDB está ejecutándose localmente:

mongosh

Si necesitas conectarte a un servidor específico:

mongosh "mongodb://usuario:contraseña@host:puerto"

**2. Ver bases de datos disponibles**

show dbs

**3. Seleccionar una base de datos**

use miBaseDeDatos

(Si la base de datos no existe, se creará al insertar datos.)

**4. Ver colecciones dentro de la base de datos**

show collections

**Operaciones CRUD en MongoDB**

**5. Crear (Insertar datos)**

* Insertar un documento:

db.miColeccion.insertOne({ nombre: "Juan", edad: 25 })

* Insertar múltiples documentos:

db.miColeccion.insertMany([

{ nombre: "Ana", edad: 30 },

{ nombre: "Luis", edad: 28 }

])

**6. Leer (Consultar datos)**

* Mostrar todos los documentos de una colección:

db.miColeccion.find()

* Mostrar documentos con formato legible:

db.miColeccion.find().pretty()

* Filtrar por una condición:

db.miColeccion.find({ edad: 25 })

**7. Actualizar datos**

* Actualizar un solo documento:

db.miColeccion.updateOne(

{ nombre: "Juan" },

{ $set: { edad: 26 } }

)

* Actualizar múltiples documentos:

db.miColeccion.updateMany(

{ edad: { $lt: 30 } },

{ $set: { estado: "activo" } }

)

**8. Eliminar datos**

* Eliminar un solo documento:

db.miColeccion.deleteOne({ nombre: "Ana" })

* Eliminar múltiples documentos:

db.miColeccion.deleteMany({ edad: { $lt: 30 } })

**9. Salir de mongosh**

exit

**Prácticas para la Clase de Base de Datos II - SQL Avanzado y MongoDB**

**Script de Creación de Base de Datos y Carga de Datos**

CREATE DATABASE empresa;

USE empresa;

CREATE TABLE departamentos (

id INT AUTO\_INCREMENT PRIMARY KEY,

nombre VARCHAR(100) NOT NULL

);

CREATE TABLE empleados (

id INT AUTO\_INCREMENT PRIMARY KEY,

nombre VARCHAR(100) NOT NULL,

salario DECIMAL(10,2) NOT NULL,

fecha\_contratacion DATE NOT NULL,

id\_departamento INT,

FOREIGN KEY (id\_departamento) REFERENCES departamentos(id)

);

INSERT INTO departamentos (nombre) VALUES ('Ventas'), ('Recursos Humanos'), ('IT');

INSERT INTO empleados (nombre, salario, fecha\_contratacion, id\_departamento) VALUES

('Juan Perez', 55000, '2022-06-15', 1),

('Ana Gomez', 48000, '2021-09-20', 2),

('Carlos Lopez', 65000, '2020-03-10', 3),

('Maria Ruiz', 70000, '2019-08-01', 1),

('Pedro Sanchez', 72000, '2018-05-12', 2),

('Laura Fernandez', 68000, '2023-01-25', 3),

('Miguel Torres', 50000, '2022-10-10', 1),

('Lucia Ramirez', 47000, '2021-12-05', 2),

('Jose Martinez', 62000, '2020-07-19', 3),

('Elena Rios', 53000, '2023-02-14', 1);

**Práctica 1: Repaso de SQL Avanzado**

**Ejercicio 1: Creación de Tablas y Relaciones**

1. Crea una base de datos llamada empresa.
2. Crea dos tablas: empleados y departamentos.
3. La tabla departamentos debe contener los siguientes campos:
   * id (clave primaria, entero, autoincremental)
   * nombre (varchar, 100 caracteres)
4. La tabla empleados debe contener:
   * id (clave primaria, entero, autoincremental)
   * nombre (varchar, 100 caracteres)
   * salario (decimal, 10,2)
   * fecha\_contratacion (date)
   * id\_departamento (clave foránea referenciando departamentos.id)
5. Inserta datos de prueba en ambas tablas.

**Ejercicio 2: Consultas SQL Avanzadas**

1. Muestra todos los empleados y sus respectivos departamentos usando JOIN.

SELECT e.nombre, d.nombre AS departamento

FROM empleados e

JOIN departamentos d ON e.id\_departamento = d.id;

Cuenta cuántos empleados tiene cada departamento usando GROUP BY y COUNT().

SELECT d.nombre, COUNT(e.id) AS total\_empleados

FROM empleados e

JOIN departamentos d ON e.id\_departamento = d.id

GROUP BY d.nombre;

1. Encuentra los departamentos que tienen más de 5 empleados.

SELECT d.nombre

FROM empleados e

JOIN departamentos d ON e.id\_departamento = d.id

GROUP BY d.nombre

HAVING COUNT(e.id) > 5;

1. Usa EXPLAIN para analizar una consulta de selección.

EXPLAIN ANALYZE SELECT \* FROM empleados WHERE salario > 50000;

Obtén los empleados con los 3 salarios más altos.

SELECT nombre, salario

FROM empleados

ORDER BY salario DESC

LIMIT 3;

1. Encuentra empleados contratados en los últimos 2 años.

SELECT nombre, fecha\_contratacion

FROM empleados

WHERE fecha\_contratacion >= CURDATE() - INTERVAL 2 YEAR;

**Práctica 2: Introducción a NoSQL y MongoDB**

**Ejercicio 1: Instalación y Configuración**

1. Instala MongoDB en tu sistema.
2. Usa mongosh para conectarte a la base de datos.
3. Crea una base de datos llamada empresaNoSQL.
4. Crea una colección llamada empleados.

**Ejercicio 2: Inserción de Datos**

1. Inserta empleados en la colección empleados.
2. db.empleados.insertMany([
3. { "nombre": "Juan Perez", "edad": 30, "departamento": "Ventas", "salario": 55000, "fecha\_contratacion": "2022-06-15" },
4. { "nombre": "Ana Gomez", "edad": 28, "departamento": "Recursos Humanos", "salario": 48000, "fecha\_contratacion": "2021-09-20" },
5. { "nombre": "Carlos Lopez", "edad": 35, "departamento": "IT", "salario": 65000, "fecha\_contratacion": "2020-03-10" }
6. ]);

**Ejercicio 3: Consultas Avanzadas en MongoDB**

1. Muestra todos los empleados de la colección.
2. db.empleados.find({}).pretty();
3. Encuentra los empleados del departamento de "Ventas".
4. db.empleados.find({ departamento: "Ventas" }).pretty();

**Comparación SQL vs MongoDB**

| **Característica** | **SQL (Relacional)** | **MongoDB (NoSQL)** |
| --- | --- | --- |
| **Estructura de datos** | Tablas y relaciones | Documentos JSON |
| **Escalabilidad** | Vertical | Horizontal |
| **Flexibilidad** | Esquema fijo | Esquema dinámico |
| **Consultas complejas** | JOINs y normalización | Agregaciones y documentos anidados |

**Código para Generar la Base de Datos en Python**

from pymongo import MongoClient

# Conectar a MongoDB

client = MongoClient("mongodb://localhost:27017/")

db = client["empresaNoSQL"]

# Crear colección e insertar documentos

empleados = db["empleados"]

empleados.insert\_many([

{"nombre": "Juan Perez", "edad": 30, "departamento": "Ventas", "salario": 55000, "fecha\_contratacion": "2022-06-15"},

{"nombre": "Ana Gomez", "edad": 28, "departamento": "Recursos Humanos", "salario": 48000, "fecha\_contratacion": "2021-09-20"},

{"nombre": "Carlos Lopez", "edad": 35, "departamento": "IT", "salario": 65000, "fecha\_contratacion": "2020-03-10"}

])