Currícula de Bases de Datos

Temario

Módulo 1: Introducción a los Datos y Bases de Datos

Unidad 1: Fundamentos Conceptuales

- Definición de datos e información
- Conceptos básicos de almacenamiento de información
- Evolución de los sistemas de almacenamiento de datos
- Introducción a las bases de datos: qué son y por qué son importantes

Unidad 2: Estructuras de Datos Básicas

- Tipos de datos fundamentales
- Estructuras de datos simples
- Introducción a la organización y representación de datos
- Ejercicios prácticos de modelado de información simple

Módulo 2: Modelo Relacional y Diseño de Bases de Datos

Unidad 3: Modelo Relacional

- Conceptos de tablas, registros y campos
- Claves primarias y foráneas
- Relaciones entre tablas
- Principios de normalización básica

Unidad 4: Diseño Conceptual de Bases de Datos

- Diagramas entidad-relación
- Identificación de entidades y atributos
- Modelado de relaciones entre entidades
- Taller de diseño de base de datos para un caso de estudio simple

Módulo 3: Lenguaje de Consulta SQL

Unidad 5: Consultas Básicas

- Introducción a SQL

- Sintaxis fundamental
- Comandos SELECT, WHERE, ORDER BY
- Filtrado y ordenamiento de datos
- Consultas simples sobre una tabla

Unidad 6: Consultas Avanzadas

- JOIN entre tablas
- Funciones de agregación
- Subconsultas
- Agrupamiento de datos
- Ejercicios de consultas complejas

Módulo 4: Administración y Gestión de Bases de Datos

Unidad 7: Fundamentos de Administración

- Introducción a los sistemas de gestión de bases de datos
- Conceptos de instalación y configuración
- Usuarios y permisos
- Conceptos básicos de seguridad

Unidad 8: Gestión y Mantenimiento

- Estrategias de respaldo y recuperación
- Optimización de consultas
- Introducción a la integridad de datos
- Manejo de transacciones básicas

Módulo 5: Proyecto Integrador

- Diseño de una base de datos para un problema real
- Implementación completa
- Consultas y reportes
- Presentación final del proyecto

Metodología

- 40% Teoría
- 60% Práctica
- Uso de herramientas como MySQL, PostgreSQL o SQLite
- Evaluaciones continuas y proyecto final

Requisitos Previos

- Conocimientos básicos de computación
- Lógica de programación
- Manejo básico de computadora

Módulo 1: Introducción a los Datos y Bases de Datos

Unidad 1: Fundamentos Conceptuales

¿Qué es un dato?

Un dato es la representación de una variable que puede ser cuantitativa o cualitativa que indica un valor que se le asigna a las cosas y se representa a través de una secuencia de símbolos, números o letras.
Los datos son elementos básicos que describen la realidad, y son utilizados como la materia prima para generar información
Los datos pueden ser de diferentes tipos, como numéricos, alfanuméricos, de fecha, entre otros, y pueden ser almacenados en sistemas informáticos para su posterior procesamiento y análisis.
Es importante recordar que los datos deben ser precisos, completos, relevantes y actualizados para poder generar información confiable y útil.

¿Qué es información?

La información puede ser de diferentes tipos y formatos, y es utilizada para describir y representar datos de la vida real

La información puede ser almacenada y procesada en sistemas informáticos para su posterior análisis y utilización.
Es importante recordar que la calidad de la información depende de la precisión, integridad, relevancia y actualidad de los datos que la conforman.

¿Qué es una base de datos?

Una base de datos es una recopilación organizada de información o datos estructurados, que normalmente se almacena de forma electrónica en un sistema informático.
Normalmente, una base de datos está controlada por un sistema de gestión de bases de datos (DBMS -> DataBase Management System).
Contribuyen a la creación de bases de datos más eficaces y consistentes.
Determinan las estructuras de almacenamiento del sistema.
Facilitan las búsquedas de datos de cualquier tipo y procedencia a los usuarios de negocio.
Ayudan a mantener la integridad de los activos informacionales de la empresa.
Introducen cambios en la información, si es requerido.
Simplifican los procesos de consulta.

Controlan los movimientos que se observan en la base de datos.

Database Management System

Conjunto de programas invisibles para el usuario final con el que se administra y gestiona la información que incluye una base de datos.

Tipos de base de datos

Base de datos jerárquicas
Base de datos de red
Base de datos transaccionales
Base de datos relacionales/no relacionales
Base de datos deductivas o lógicas
Base de datos multidimensionales
Base de datos orientadas a objetos
Base de datos documentales

Unidad 2: Estructuras de Datos Básicas

Tipos de datos

Numéricos (Int, BigInt, TinyInt)
Caracteres (Char, Varchar, Text, BLOB)
Fechas y horas (Date, Datetime,Time)
Booleanos

¿Qué son las Estructuras de Datos?

Imagina que tienes una habitación llena de libros, herramientas o cualquier otro objeto. Si los dejas tirados sin ningún orden, encontrar algo específico será un verdadero desafío. Las estructuras de datos son como sistemas de organización que nos ayudan a guardar, ordenar y encontrar información de manera eficiente.

Tipos de Datos Fundamentales

1. Números

- **Enteros**: Números sin decimales (0, -5, 42)
- **Decimales**: Números con punto decimal (3.14, -0.5, 2.0)
- **Booleanos**: Valores que solo pueden ser Verdadero (True) o Falso (False)

2. Texto

- **Cadenas (Strings)**: Secuencia de caracteres ("Hola", "Base de Datos", "2024")

3. Fechas y Horas

- Representación de momentos específicos en el tiempo

Estructuras de Datos Simples

Arreglos (Arrays)

- Como una fila de casilleros
- Pueden contener múltiples elementos del mismo tipo
- Ejemplo: [1, 2, 3, 4, 5]
- Cada elemento tiene una posición (índice)

Listas

- Similares a los arreglos, pero más flexibles
- Pueden cambiar de tamaño
- Pueden contener diferentes tipos de datos
- Ejemplo: ["manzana", 42, True]

Registros o Estructuras

- Agrupan diferentes tipos de datos relacionados
- Como una ficha de información

Ejemplo:
Estudiante = { nombre: "Juan", edad: 20, promedio: 8.5}

Organización y Representación de Datos

¿Por qué es importante?

- Facilita el acceso rápido a la información
- Permite realizar operaciones eficientes
- Ayuda a resolver problemas complejos de manera sistemática

Principios Básicos

- **Consistencia**: Mantener un formato uniforme
- **Claridad**: Datos fáciles de entender
- **Eficiencia**: Minimizar el espacio y tiempo de acceso

Ejemplo Práctico: Modelando Información de Estudiantes

Supongamos que queremos guardar información de estudiantes de un curso:

```
[
  {
```

```
    id: 1,
    nombre: "María",
    edad: 22,
    materias: ["Matemáticas", "Programación"],
    activo: True
  },
  {
    id: 2,
    nombre: "Carlos",
    edad: 25,
    materias: ["Base de Datos", "Redes"],
    activo: False
  }
]
```

Ejercicio Mental

Piensa en tu biblioteca personal:

- ¿Cómo organizarías tus libros?
- ¿Por autor? ¿Por género? ¿Por fecha de compra?

Cada forma de organizar es una estructura de datos diferente.

Consejos Finales

- No existe una estructura "perfecta"
- La elección depende del problema que quieras resolver
- Practicar es la mejor forma de aprender

Módulo 2: Modelo Relacional y Diseño de Bases de Datos

Unidad 3: Modelo Relacional

¿Qué es una base de datos relacional (RDBMS)?

Una base de datos relacional es una recopilación de elementos con relaciones predefinidas entre ellos.

AWS

Una base de datos relacional es un tipo de base de datos que almacena y proporciona acceso a puntos de datos relacionados entre sí.

Oracle

Almacena datos en filas y columnas formando tablas conectadas por claves comunes.
Tipo de BBDD más utilizados.
Pueden ser utilizadas por cualquier usuario.
Fácil gestión.
Se puede acceder rápidamente a los datos.

Conceptos importantes de BBDD

Una base de datos relacional es una base de datos que almacena datos en TABLAS.
Cada TABLA contiene filas (registros, también conocidos como TUPLAS) y columnas (campos).
Cada TABLA tiene una CLAVE PRINCIPAL (también conocida como PRIMARY_KEY o PK, su abreviación).
La TABLA puede contener o no una CLAVE EXTERNA (también conocida como FOREIGN KEY o FK).
Una RELACIÓN es una asociación entre tablas que se crean utilizando sentencias de unión para recuperar datos.

Unidad 4: Diseño Conceptual de Bases de Datos

Estándar ACID

Todas las transacciones de base de datos deben ser conformes a ACID (atómicas, coherentes, aisladas y duraderas) para garantizar la integridad de los datos.
La atomicidad requiere que la transacción se ejecute correctamente como un todo o, si una parte de la transacción falla, que toda ella quede invalidada.
La consistencia exige que los datos escritos en la base de datos como parte de la transacción cumplan todas las reglas definidas, así como las restricciones, incluidos los desencadenadores, las limitaciones y las cascadas.
El aislamiento es fundamental para lograr el control de concurrencia y asegurarse de que cada transacción sea independiente por sí misma.
La durabilidad requiere que todos los cambios realizados en la base de datos sean permanentes luego de que la transacción se haya completado de forma correcta.

Normalización de datos

Es un proceso que se utiliza en el diseño de bases de datos para eliminar la redundancia de datos y mejorar la eficiencia a la hora de gestionar los datos. A su vez, también preservan la integridad de los mismos.
Desarrollado por Codd en 1972.
La normalización se lleva a cabo mediante la creación de tablas separadas para cada entidad o concepto único que se necesita almacenar en la base de datos.

Formas normales
A la hora de normalizar una base de datos, tenemos diferentes “niveles”, los cuales son dependientes uno de los otros. Estos “niveles” antes mencionados se denominan formas normales y escalan de la 1FN a la 5FN.

En resumen, la normalización de datos es importante en el diseño de bases de datos para garantizar la eficiencia de la gestión de datos y evitar la redundancia de datos.
Hay diferentes niveles de normalización, cada uno de los cuales se representa por una forma normal que tiene ciertas reglas que deben cumplirse.

Primer Forma Normal (1FN): Requiere que cada columna de una tabla contengan valores atómicos y únicos.
Eliminar los grupos repetitivos de las tablas individuales
Crear una tabla separada para cada grupo de datos relacionados
Identificar cada grupo de datos relacionados con una clave primaria (PK / Primary Key)

Segunda Forma Normal (2FN): Requiere que cada tabla tenga una clave primaria única y que cada columna de la tabla esté directamente relacionada con la clave primaria.
Se introduce el concepto “Dependencia Funcional”
Esto significa que cada registro en la tabla debe estar identificado de manera única por la clave primaria y que cada columna de la tabla debe describir una propiedad única del registro.

Tercera Forma Normal (3FN): Requiere que cada columna de una tabla dependa solo de la clave primaria y no de otras columnas de la misma tabla. Esto significa que no debe haber dependencias transitivas en la tabla.

Diagrama Entidad de Relación

El modelo entidad-relación es el modelo conceptual más utilizado para el diseño conceptual de bases de datos. Fue introducido por Peter Chan en 1976. El modelo entidad-relación está formado por un conjunto de conceptos que permiten describir la realidad mediante un conjunto de representaciones gráficas y lingüísticas

¿Qué es una relación en bbdd?

Es una correspondencia o asociación entre dos o más entidades. Cada relación tiene un nombre que describe su función. Las relaciones se representan gráficamente mediante rombos y su nombre aparece en el interior.
Una relación es, básicamente, la forma en la cual se relacionan dos tablas.

¿Qué es una entidad?

Cualquier tipo de objeto o concepto sobre el que se recoge información: cosa, persona, concepto abstracto o suceso. Por ejemplo: coches, casas,empleados, clientes, empresas, oficios, diseños de productos, conciertos,excursiones, etc.

Las entidades se representan gráficamente mediante rectángulos y su nombre aparece en el interior.

Directrices

Es una instrucción o norma que se debe seguir para la ejecución de algo. En nuestro caso para la construcción de DER.

1. La semántica de los atributos.
2. La reducción de información redundante en las tuplas.
3. La reducción de los valores NULL en las tuplas.
4. Prohibición de la posibilidad de generar tuplas falsas

Semántica

Ser claros a la hora de definir toda relación o atributos de una tupla.
Diseñar un esquema de relación para que sea fácil explicar su significado
No combinar atributos de varios tipos de entidad y de relación en una única relación.

Redundancias y anomalías

Uno de los objetivos más claros de un diseño de base de datos (DER) es reducir el espacio de almacenamiento utilizado.
El agrupamiento de atributos en esquemas de relación tiene un efecto significativo sobre el espacio de almacenamiento.

Valores NULL en las tuplas

Si muchos atributos no se aplican a todas las tuplas de la relación, nos encontraremos con muchos NULL en esas tuplas, lo que puede desperdiciar espacio de almacenamiento.
Hasta donde sea posible, evite situar en una relación base atributos cuyos valores sean frecuentemente NULL.
En caso de no poderse evitar, asegúrese de que se aplican sólo en casos excepcionales y no los aplique a la mayor parte de las tuplas de la relación.

Tuplas falsas

Debido a un mal diseño de las relaciones pueden generarse problemas a la hora de obtener los datos desde varias de ellas.
Evite las relaciones que contienen atributos coincidentes que no son combinaciones de foreign key y clave principal porque la concatenación de estos atributos puede producir tuplas falsas.

Módulo 3: Lenguaje de Consulta SQL

Unidad 5: Consultas Básicas

SQL

SQL es el acrónimo de Structured Query Language o, en español, lenguaje de consulta estructurada.
Está diseñado para obtener, calcular, actualizar o eliminar datos de una base de datos.
Utiliza un lenguaje que es de fac
Principales motores: MySQL, SQLite, MariaDB
Lenguaje fácil de entender ya que utiliza sentencias adaptadas del idioma inglés (Create, Select, Insert, Delete, Group By, etc).
Si bien evolucionó a lo largo de los años, la base sigue siendo la misma.

Sentencias y sus tipos

A la hora de trabajar con SQL, este tiene varias sentencias, las cuales se clasifican en 3 principales grupos (DML, DDL y DCL)

DDL: Las sentencias DDL (Data Definition Language) son utilizadas para crear estructuras de datos (CREATE, ALTER, DROP, TRUNCATE)
DML: Las sentencias DML (Data Manipulation Language) son utilizadas para la manipulación de datos (SELECT, UPDATE, DELETE, INSERT)
DCL: Las sentencias DCL (Data Control Language) son utilizadas para dar o remover acceso a los usuarios (GRANT, REVOKE)

Sentencias DDL, Data Definition Language

CREATE: Para crear objetos en la BD (Tablas, Campos, Bases de Datos, etc)
ALTER: Alterar la estructura de la base de datos (Tablas, Campos, etc)
DROP: Eliminar todos los objetos de la base de datos (Tablas, Campos, Bases de Datos, etc)
TRUNCATE: Elimina todos los datos dentro de mi Base de Datos o Tablas)

Sentencias DML, Data Manipulation Language

SELECT: Recuperar datos de una ta
INSERT: Insertar datos en una tabla
UPDATE: Actualizar datos en una tabla
DELETE: Eliminar datos en una tabla

Sentencias DCL, Data Control Language

GRANT: Permite dar permisos a uno o varios usuarios o roles para realizar tareas determinadas.
REVOKE: Permite eliminar permisos que previamente se han concedido con GRANT.

¿Qué es MySQL?

Es un sistema de gestión de base de datos RELACIONALES (DBMS)
Es de código abierto con el respaldo de Oracle y basado en SQL.
Lanzado en 1995
Fundada por Axmark, Larsson y Widenius), paso por las manos de Sun Microsystems hasta pasar a manos de Oracle

Unidad 6: Consultas Avanzadas

Cláusulas

Cláusula INNER JOIN

Es una cláusula que busca coincidencias entre dos tablas en función a una columna que tienen en común.

Importante: SOLO SE MOSTRARÁ LA INTERSECCIÓN EN LOS RESULTADOS

Cláusula LEFT JOIN

A diferencia del INNER JOIN, donde se busca una intersección respetada por ambas tablas, aquí se le da prioridad a la tabla izquierda.
En caso de no existir una coincidencia con alguna de las filas de la tabla izquierda, igualmente se muestran TODOS los resultados de la primer tabla

Cláusula RIGHT JOIN

A diferencia del LEFT JOIN, donde se da prioridad a la tabla izquierda, en este caso se da prioridad a la tabla DERECHA.
En caso de no existir una coincidencia con alguna de las filas de la tabla derecha, igualmente se muestran TODOS los resultados de la segunda tabla

FULL OUTER JOIN

Con todos los casos antes mencionados, “perdíamos” información al realizar las consultas. El FULL OUTER JOIN o FULL JOIN se encarga de mostrar la filas de ambas tablas sin importar que existan coincidencias.

IMPORTANTE: FULL JOIN NO FUNCIONA EN MYSQL. Pero tiene solución

Módulo 4: Administración y Gestión de Bases de Datos

Unidad 7: Fundamentos de Administración

Introducción a los Sistemas de Gestión de Bases de Datos (SGBD)

¿Qué es un SGBD?

Un Sistema de Gestión de Bases de Datos (SGBD) es un software diseñado específicamente para definir, manipular, recuperar y administrar datos en una base de datos. Actúa como intermediario entre los usuarios y las bases de datos, permitiendo almacenar, organizar y acceder a la información de manera eficiente y controlada.

Funciones principales de un SGBD:

1. **Almacenamiento de datos**: Proporciona mecanismos para guardar grandes volúmenes de información de forma persistente.
2. **Acceso a datos**: Ofrece lenguajes y herramientas para consultar y manipular la información almacenada.
3. **Integridad de datos**: Garantiza que los datos cumplan con reglas predefinidas para mantener su consistencia.
4. **Concurrencia**: Permite que múltiples usuarios accedan simultáneamente a la base de datos sin generar conflictos.
5. **Seguridad**: Controla quién puede acceder a los datos y qué operaciones puede realizar.
6. **Recuperación**: Proporciona mecanismos para restaurar la base de datos en caso de fallos.

SGBD populares:

- **MySQL/MariaDB**: Ideal para aplicaciones web, software libre.
- **PostgreSQL**: Robusto, compatible con SQL, altamente extensible, software libre.
- **Oracle Database**: Utilizado en grandes empresas, solución comercial.
- **Microsoft SQL Server**: Integrado con productos Microsoft, solución comercial.
- **SQLite**: Base de datos embebida, ideal para aplicaciones móviles.
- **MongoDB**: Base de datos NoSQL orientada a documentos.

Arquitectura general de un SGBD:

El SGBD típicamente está compuesto por varias capas:

1. **Motor de almacenamiento**: Responsable del almacenamiento físico de los datos.
2. **Motor de consultas**: Procesa y optimiza las consultas para acceder a los datos.
3. **Gestor de transacciones**: Controla las operaciones que modifican la base de datos.
4. **Gestor de concurrencia**: Coordina el acceso simultáneo de múltiples usuarios.
5. **Gestor de recuperación**: Mantiene la integridad en caso de fallos.
6. **Catálogo del sistema**: Almacena metadatos sobre la estructura de la base de datos.

Conceptos de Instalación y Configuración

Proceso de instalación:

1. **Selección del SGBD**: Evaluar requisitos del proyecto para elegir el sistema adecuado.
2. **Requisitos del sistema**: Verificar hardware y software necesarios.
3. **Métodos de instalación**:
    - Instalación local (desarrollo)
    - Instalación en servidor (producción)
    - Instalación en la nube (servicios gestionados)
4. **Pasos generales**:
    - Descarga del software
    - Ejecución del instalador
    - Configuración inicial
    - Verificación de la instalación

Configuración básica:

1. **Archivos de configuración**:
    - Ubicación de los archivos de datos
    - Tamaño de memoria asignada
    - Puertos de comunicación
    - Conjunto de caracteres
2. **Parámetros críticos**:
    - **Memoria**: Buffer pools, cache de consultas
    - **Almacenamiento**: Tamaño de tablespace, logs de transacciones
    - **Conexiones**: Número máximo de conexiones simultáneas
    - **Localización**: Zona horaria, formatos de fecha
3. **Servicios y procesos**:
    - Inicio automático
    - Monitoreo
    - Servicios auxiliares (replicación, backups)

Usuarios y Permisos

Gestión de usuarios:

1 - Creación de usuarios:
CREATE USER 'username'@'host' IDENTIFIED BY 'password';

2 - Modificación de usuarios:
CREATE USER 'username'@'host' IDENTIFIED BY 'password';

3 - Eliminación de usuarios:
```
DROP USER 'username'@'host';
```

Sistema de permisos:

1 - Niveles de permisos:

- A nivel de servidor
- A nivel de base de datos
- A nivel de tabla
- A nivel de columna
- A nivel de procedimiento almacenado

2 - Tipos de privilegios:

- SELECT: Permite leer datos
- INSERT: Permite añadir datos
- UPDATE: Permite modificar datos
- DELETE: Permite eliminar datos
- CREATE: Permite crear objetos
- DROP: Permite eliminar objetos
- ALTER: Permite modificar objetos
- EXECUTE: Permite ejecutar procedimientos

3 - Asignación de permisos:

```
GRANT permiso ON objeto TO 'username'@'host';
```

4 - Revocación de permisos:
```
REVOKE permiso ON objeto FROM 'username'@'host';
```

5 - Roles (en SGBD que los soportan):

- Agrupación de permisos
- Asignación a múltiples usuarios
- Facilita la administración

Conceptos Básicos de Seguridad

Niveles de seguridad:

1. **Seguridad física**:
    - Protección de los servidores
    - Control de acceso físico
    - Redundancia de hardware
2. **Seguridad a nivel de red**:
    - Firewalls
    - Conexiones cifradas (SSL/TLS)
    - Redes privadas virtuales (VPN)
3. **Seguridad a nivel de SGBD**:
    - Autenticación
    - Control de acceso
    - Cifrado de datos
4. **Seguridad a nivel de aplicación**:
    - Prevención de inyección SQL
    - Validación de entradas
    - Principio de mínimo privilegio

Buenas prácticas de seguridad:

Contraseñas robustas:

- ➢ Longitud mínima
- ➢ Complejidad
- ➢ Rotación periódica

Auditoría:

- Registro de accesos
- Registro de operaciones críticas
- Revisión periódica de logs

Actualizaciones y parches:

- Mantener el SGBD actualizado
- Aplicar parches de seguridad
- Pruebas antes de actualizar

Cifrado:

- Datos en reposo
- Datos en tránsito
- Backups

Principio de mínimo privilegio:

- Otorgar solo los permisos necesarios
- Revisar permisos periódicamente
- Revocar accesos innecesarios

Unidad 8: Gestión y Mantenimiento

Estrategias de Respaldo y Recuperación

Tipos de backup:

1. **Backup completo (Full backup)**:
    - Copia de todos los datos
    - Mayor tiempo de backup
    - Restauración más rápida
2. **Backup incremental**:
    - Solo copia los cambios desde el último backup
    - Más rápido
    - Restauración más compleja (necesita backup completo + incrementales)
3. **Backup diferencial**:
    - Copia los cambios desde el último backup completo
    - Intermedio en tiempo
    - Restauración intermedia (necesita backup completo + último diferencial)
4. **Backup lógico vs. físico**:
    - **Lógico**: Scripts SQL (mysqldump, pg_dump)
    - **Físico**: Copias de archivos de datos

Planificación de backups:

1. **Frecuencia**:
    - Según criticidad de los datos
    - Según volumen de cambios
    - Según ventanas de mantenimiento
2. **Retención**:
    - Cuánto tiempo conservar los backups
    - Política de rotación
    - Requisitos legales
3. **Almacenamiento**:
    - Discos externos
    - Almacenamiento en la nube
    - Cintas magnéticas
    - Sitios externos (para protección contra desastres)

Proceso de recuperación:

1. **Tipos de recuperación**:
    - A un punto en el tiempo (Point-in-Time Recovery)

    - Recuperación completa
    - Recuperación a nivel de tabla
2. **Pasos generales**:
    - Detener servicios (si es necesario)
    - Restaurar backup
    - Aplicar logs de transacciones (si es necesario)
    - Verificar integridad
    - Reiniciar servicios
3. **Pruebas de recuperación**:
    - Probar periódicamente los backups
    - Documentar procedimientos
    - Medir tiempos de recuperación

Optimización de Consultas

Índices:

1 - Tipos de índices:

- ➢ Índices B-Tree (estándar)
- ➢ Índices Hash
- ➢ Índices de texto completo
- ➢ Índices espaciales

2- Creación de índices:
sql

```
CREATE INDEX idx_nombre ON tabla(columna);
```

3 - **Estrategias de indexación**:

- Indexar columnas en cláusulas WHERE
- Indexar columnas de JOIN
- Indexar columnas de ORDER BY
- Evitar sobre-indexación

Análisis de consultas:

1. **Plan de ejecución**:
    - EXPLAIN en MySQL/PostgreSQL
    - Execution Plan en SQL Server
    - Identificar operaciones costosas
2. **Herramientas de monitoreo**:
    - Slow query log
    - Performance Schema
    - Herramientas específicas del SGBD

Técnicas de optimización:

1. **Reescritura de consultas**:
    - Evitar funciones en columnas indexadas
    - Usar JOIN en lugar de subconsultas cuando sea posible
    - Limitar resultados con LIMIT/TOP
2. **Optimización de esquema**:
    - Normalización/desnormalización según caso
    - Tipos de datos apropiados
    - Particionamiento de tablas grandes
3. **Configuración del SGBD**:
    - Ajustar memoria cache
    - Configurar paralelismo
    - Optimizar escritura en disco

Introducción a la Integridad de Datos

Tipos de integridad:

1. **Integridad de entidad**:
    - Claves primarias
    - Valores únicos
    - Valores no nulos
2. **Integridad referencial**:
    - Claves foráneas
    - Acciones referenciales (CASCADE, SET NULL, etc.)
3. **Integridad de dominio**:
    - Tipos de datos
    - Restricciones CHECK
    - Valores por defecto

Implementación de integridad:

1. **Restricciones declarativas**:

    CREATE TABLE empleados (id INT PRIMARY KEY, nombre VARCHAR(100) NOT NULL, salario DECIMAL(10,2) CHECK (salario > 0), id_departamento INT, FOREIGN KEY (id_departamento) REFERENCES departamentos(id));

Disparadores (triggers):

- Para reglas complejas
- Se ejecutan automáticamente ante eventos
- Pueden validar o modificar datos

Procedimientos almacenados:

- Implementan lógica de negocio
- Validan datos complejos

- Mantienen consistencia

Manejo de Transacciones Básicas

Concepto de transacción:

Una transacción es una secuencia de operaciones que se ejecutan como una única unidad de trabajo. Debe cumplir con las propiedades ACID:

- **Atomicidad**: Todas las operaciones se completan o ninguna.
- **Consistencia**: La BD pasa de un estado válido a otro válido.
- **Aislamiento**: Las transacciones no interfieren entre sí.
- **Durabilidad**: Los cambios persisten incluso ante fallos.

Control de transacciones:

1. **Inicio y finalización**:

BEGIN TRANSACTION; -- Operaciones SQL COMMIT; -- Confirma cambios -- o ROLLBACK; -- Deshace cambios

Niveles de aislamiento:

- READ UNCOMMITTED
- READ COMMITTED
- REPEATABLE READ
- SERIALIZABLE

Problemas comunes:

- Lecturas sucias
- Lecturas no repetibles
- Lecturas fantasma
- Deadlocks (bloqueos mutuos)

Gestión de concurrencia:

1. **Bloqueos (locks)**:
    - Compartidos (lectura)
    - Exclusivos (escritura)
2. **Estrategias de gestión**:
    - Pesimista: bloquear antes de acceder
    - Optimista: verificar conflictos al confirmar
3. **Resolución de deadlocks**:
    - Detección automática
    - Tiempos de espera
    - Orden de acceso a recursos

Ejercicios Prácticos

Ejercicio 1: Instalación y Configuración

Objetivo: Instalar un SGBD y realizar la configuración básica.

Instrucciones:

1. Instalar MySQL, PostgreSQL o SQLite en tu sistema.
2. Configurar los parámetros básicos (memoria, conexiones, directorio de datos).
3. Crear una base de datos para pruebas.
4. Documentar el proceso y los parámetros configurados.

Ejercicio 2: Gestión de Usuarios y Permisos

Objetivo: Crear usuarios con diferentes niveles de acceso.

Instrucciones:

1. Crear la siguiente estructura de base de datos:

```
CREATE DATABASE tienda;
USE tienda;

CREATE TABLE productos (
  id INT PRIMARY KEY,
  nombre VARCHAR(100),
  precio DECIMAL(10,2),
  stock INT
);

CREATE TABLE ventas (
  id INT PRIMARY KEY,
  fecha DATE,
  id_producto INT,
  cantidad INT,
  FOREIGN KEY (id_producto) REFERENCES productos(id)
);
```

1. Crear los siguientes usuarios:
    - Administrador: acceso total
    - Vendedor: puede consultar todos los datos y modificar ventas
    - Inventario: puede consultar y modificar productos
    - Invitado: solo puede consultar productos (no precios)
2. Verificar que cada usuario puede realizar solo las operaciones permitidas.

Ejercicio 3: Backup y Recuperación

Objetivo: Implementar estrategias de backup y recuperación.

Instrucciones:

1. Con la base de datos del ejercicio anterior:
    - Realizar un backup completo
    - Insertar nuevos datos
    - Realizar un backup incremental o diferencial
    - Simular un fallo (eliminar algunos datos)
    - Recuperar la base de datos
2. Documentar:
    - Comandos utilizados
    - Tiempo requerido para cada operación
    - Problemas encontrados y soluciones

Ejercicio 4: Optimización de Consultas

Objetivo: Analizar y optimizar consultas SQL.

Instrucciones:

1. Crear una base de datos con suficientes datos para pruebas (mínimo 10,000 registros).
2. Escribir al menos tres consultas complejas que incluyan:
    - JOIN entre varias tablas
    - Funciones de agregación
    - Ordenamiento
    - Filtros complejos
3. Para cada consulta:
    - Analizar el plan de ejecución
    - Identificar cuellos de botella
    - Crear índices apropiados
    - Reescribir la consulta si es necesario
    - Comparar el rendimiento antes y después

Ejercicio 5: Integridad y Transacciones

Objetivo: Implementar mecanismos de integridad y manejo de transacciones.

Instrucciones:

1. Diseñar una base de datos para un banco con tablas para:
    - Clientes
    - Cuentas
    - Transacciones
2. Implementar restricciones de integridad:
    - El saldo no puede ser negativo
    - Cada transacción debe afectar dos cuentas (origen y destino)
    - El historial de transacciones debe mantenerse consistente

3. Escribir procedimientos almacenados para:
    - Transferencia entre cuentas (usando transacciones)
    - Depósito y retiro
    - Generación de extracto de cuenta
4. Probar los procedimientos en escenarios normales y de error.

Ejercicio 6: Proyecto Integrador

Objetivo: Aplicar todos los conocimientos en un proyecto completo.

Instrucciones:

1. Diseñar e implementar un sistema de bases de datos para un contexto a elección:
    - Sistema de gestión de biblioteca
    - Sistema de inventario y ventas
    - Sistema de reservas de hotel
    - Sistema de gestión académica
2. El proyecto debe incluir:
    - Diseño completo de la base de datos
    - Implementación con restricciones de integridad
    - Usuarios con diferentes permisos
    - Procedimientos almacenados para operaciones comunes
    - Estrategia de backup
    - Plan de optimización de consultas
    - Documentación completa
3. Presentar:
    - Scripts SQL
    - Diagrama de la base de datos
    - Manual de administración
    - Informe de pruebas