

# LA BIBLIA PARA APROBAR FINAL BDD - MERLINO

🕒 Fecha	@11 de mayo de 2024 20:04
📖 Materia	Base de datos
☰ Area	

## 📖 Glosario

### 👤 Roles

#### 📁 Base de datos

- ▶ **Base de datos relacionales**
- ▶ **Base de datos NO-SQL**
- ▶ **Base de datos in memory**

#### 📁 Motor de base de datos

#### 📁 Sistema de gestion de base de datos (SGBS/DBMS)

#### 📁 SQL

#### 📁 Indices

#### 📁 Opeaciones

- ▶ Transaccion
  - 📁 Principio ACID
- ▶ Programacio Batch/Shell
- ▶ **BCP - Bulk copy program**
- ▶ ETL - Extraccion, transformacion y carga

▶	<a href="#">Store Procedures - Procedimientos almacenado</a>
▶	<a href="#">Triggers</a>
▶	<a href="#">BCP - Bulk copy program</a>
📁	<a href="#">Metadatos</a>
📁	<a href="#">Views - vistas</a>
📁	<a href="#">Paginacion</a>
📁	<a href="#">Frameworks</a>
📁	<a href="#">Bloque de bdd (Lock)</a>
📁	<a href="#">ODBC</a>
📈	<a href="#">Ganar performance en bdd</a>
★	<a href="#">MERLINO TIPS</a>
★	<a href="#">Buenas practicas de programacion al hacer consulta:</a>
📁	<a href="#">Plataforma de datos</a>
📁	<a href="#">Diagrama de entidad - relacion (DER)</a>
📁	<a href="#">Formas normales</a>
▶	<a href="#">Normalizacion</a>
▶▶	<a href="#">Primera Forma Normal (1FN)</a>
▶▶	<a href="#">Segunda Forma Normal (2FN)</a>
▶▶	<a href="#">Tercera Forma Normal (3FN)</a>
▶▶	<a href="#">Forma Normal de Boyce-Codd (BCNF)</a>
▶	<a href="#">Desnormalizacion</a>
👤	<a href="#">Preguntas que debe saber un administrador para elegir un motor de bdd:</a>
📁	<a href="#">Sincronizacion entre nodos distribuidos</a>
📁	<a href="#">El problema de JOIN</a>
📁	<a href="#">Servidor de Colas</a>
📁	<a href="#">Clase Arquitectura</a>

---

## **Glosario**

- **Performance:** eficiencia y velocidad con la que la bdd puede procesar consultas y manipular datos
- **Datos dummy:** datos ficticios o datos de ejemplo. conjunto creado especificamente para propósitos de demostración, entrenamiento o pruebas
- **Acceso programático:** hace referencia a la capacidad de un programa para interactuar y manipular datos almacenados en una bdd de manera automatizada y mediante código de programación  
Esto implica que el programa puede realizar consulta y/o manipulación de

datos, así como también ejecutar otros comandos y transacciones relacionadas con la bdd

- **Full scan:** Proceso por el cual el motor de bdd necesita recorrer toda la tabla obligatoriamente para saber si la respuesta a la consulta que ha recibido es completa, es decir, si es todo lo que tiene que enviar como respuesta a la consulta → Al haber índices, nos evitamos el full scan
  - **Dependencia transitiva:** Ocurre cuando una columna depende funcionalmente de otra columna que no es clave primaria
- 



## Roles

### Administrador

- Encargado de hacer los store procedures
- Maneja las sentencias que están orientadas a manipular las bdd y las tablas
- Debe entender sql, no se espera que sea un senior para el manejo de sql

### Programador

- No es obligatorio pero debería conocer los comandos para administrar una bdd
- 



## Base de datos

- Archivo de almacenamiento

uno al menos debería conocer estos tres modelos

### ► Base de datos relacionales

- guardan información tabular (en tablas)
- datos altamente estructurados
- Se debe definir previamente el esquema de datos.

### Ventajas de SQL VS NO-SQL

## ► Base de datos NO-SQL

- Todas aquellas bdd que no son tablas
- Por lo general son base de datos distribuidas
- Funcionan con un esquema dinámico
- guardan datos no estructurados → Esto significa que puede crear una aplicación sin tener que crear primero un esquema de la bdd.
- Tienen mejor performance para guardar archivos, imágenes.
- Los requisitos de datos pueden evolucionar según sea necesario, sin causar problemas con la bdd
- No está limitada a entradas solo de texto

## ► Base de datos in memory

- sistema de gestión de bdd que almacenan y manipulan datos en la RAM en lugar de discos físicos → más rápidas

Característica	Bases de datos relacionales: ventajas	Bases de datos NoSQL: ventajas
Esquema	El esquema fijo promueve la integridad y coherencia de los datos.	El esquema dinámico para datos no estructurados permite flexibilidad en la estructura y los tipos de datos.
Escalabilidad	Escalable verticalmente al aumentar la potencia informática.	Escalable horizontalmente, lo que permite agregar fácilmente más servidores.
Actas	El sólido soporte para las propiedades ACID garantiza un procesamiento de transacciones confiable.	Modelos de transacciones flexibles adaptados a casos de uso específicos, con algunas propiedades ACID compatibles.
Consultando	Potentes capacidades de consulta con un lenguaje estandarizado (SQL) para consultas complejas.	Capacidades de consulta flexibles adaptadas al tipo de base de datos (por ejemplo, clave-valor, documento, gráfico).
Casos de uso	Ideal para aplicaciones con estructuras y relaciones de datos bien definidas que requieren transacciones complejas.	Adecuado para manejar grandes volúmenes de datos no estructurados o semiestructurados y para aplicaciones escalables con modelos de datos en evolución.
Actuación	Optimizado para consultas y relaciones complejas, con rendimiento mantenido mediante indexación y optimización.	Alto rendimiento para operaciones de lectura/escritura, particularmente con grandes conjuntos de datos y funciones escalables para mantener el rendimiento bajo carga.

## Interfaz gráfica

Todas tienen una

Es un programa que puede venir adentro del paquete pero es independiente a la misma



## Motor de base de datos

- Programa que hace óptimo el acceso a los datos.
- Gestiona cómo se almacenan físicamente los datos en el disco, cómo se accede a ellos, cómo se optimizan las consultas para mejorar el

rendimiento y cómo se mantienen la integridad y la seguridad de la información.

- Administra datos en un tipo particular de archivo
- Programa que accede a la base de datos
- Un motor de base de datos es el núcleo de un sistema de gestión de bases de datos

#### **Que puede hacer el motor/gestor de base de datos:**

- gestiona
- seguridad
- resguardar
- planifica
- diseña
- mantenimiento

#### **Beneficios**

- hay motores que analizan el costo computacional y permiten optimizar sin que el usuario sepa. La sentencia del usuario capta no es la mejor, y la reestructuran o dan las herramientas para que así lo sea.

#### **Cosas que hacen decaer el rendimiento**

- lockeo

#### **Regla de escritorio para motores de bdd**

- como ya se dijo, preguntarle que diferencias tienen 2 motores de bdd, para saber cual me conviene mas



## **Sistema de gestión de base de datos (SGBS/DBMS)**

- Ejemplo de motor de bdd: MySQL, PostgreSQL, SQL Server, Oracle Database, entre otros
- Es un conjunto completo de software que incluye el motor de base de datos junto con otras herramientas y componentes que permiten a los usuarios interactuar con la base de datos de manera efectiva

#### **Diferencias entre SGBD y motor de base de datos**

Mientras que el motor se enfoca en la manipulación y gestión de los datos a nivel técnico, el DBMS abarca todo el ecosistema de herramientas y funcionalidades que permiten a los usuarios trabajar con la base de datos de manera completa y eficiente.

---

## SQL

### Analogía del traductor:

- **SQL** es como un traductor universal que **convierte un lenguaje natural en un lenguaje de consulta de base de datos**.
- El **lenguaje natural** es la forma en que los usuarios expresan sus necesidades de información.
- El **lenguaje de consulta de base de datos** es el lenguaje que la base de datos entiende.

### Que puede hacer SQL?

- Ejecutar consultas
- Recuperar, insertar, actualizar, eliminar datos/registros
- Crear bases de datos, tablas, vistas, procedimientos almacenados (store procedures)
- Establecer permisos sobre tablas, procedimientos y vistas.

### Beneficios de SQL:

- **Independencia del lenguaje de programación:** SQL no está ligado a un lenguaje de programación específico. Puedes usar SQL con diferentes lenguajes como Java, Python, C++, etc.
- **Independencia del motor de base de datos:** SQL es compatible con una amplia gama de motores de bases de datos, como MySQL, PostgreSQL, Oracle, SQL Server, etc.
- **Interfaz única para el usuario:** Los usuarios de la aplicación no necesitan conocer los detalles de la base de datos o el lenguaje de programación utilizado. Solo necesitan entender el lenguaje natural para realizar sus consultas.
- Se generaliza para abstraerse del motor de bdd,

## Estándares SQL:

- **ANSI C:** Es un estándar publicado para la programación en lenguaje C. Define las reglas y características del lenguaje para asegurar la compatibilidad entre diferentes implementaciones.
- **Cada motor de BDD tiene su propio dialecto SQL:** Aunque SQL es un lenguaje estándar, cada motor de base de datos tiene su propio dialecto con pequeñas variaciones en la sintaxis y las características disponibles

## Como ejecuta una consulta

1. SQL interpreta el analisis lexicografico (valida si lo que escribimos esta bien) analiza si los campos y las tablas estan bien escritas
  2. Luego analiza los permisos
  3. Luego ejecuta la consulta
- 



## Indices

- se utilizan para mejorar el rendimiento de consultas al acelerar la búsqueda / recuperación de datos
- Es muy importante porque generamos en memoria un árbol que es persistente (no se pierde) con la estructura y el posicionamiento de los datos. También evitamos el full scan (que está prohibido)

## Tipos:

- **Primarios:** No permiten valores duplicados y ordenan la tabla. El orden físico del archivo es el mismo que vamos a encontrar en el índice
- **Secundarios:** Permiten valores duplicados y no ordenan la tabla.
  - Nuevo índice para que la consulta acceda a través del mismo sea mucho más rápido y consuma menos recursos de CPU
  - permite tener campos repetidos
  - Estos no mueven datos, solo generan estructura

- **Foraneas:** Campos que son índices en otras tablas, que por lo general son claves primarias

### **Estructura de índices:**

- Al dar nombre a un índice, la base de datos crea un árbol B-tree.
- Este árbol optimiza las búsquedas, haciéndolas más eficientes.

### **Beneficios de los índices:**

- **Mejora del rendimiento:** Las consultas se ejecutan más rápido.
- **Acceso rápido a la información:** Se encuentran los datos específicos con mayor facilidad.
- **Eficiencia en la gestión de datos:** Se optimiza el uso de recursos.



### **Problema de “queso bruller”**

- En el índice pueden quedar agujeros en las ramas → por ser un árbol binario
- mucha de estas ramas hasta llegar a los nodos ojas quedan desactualizadas (no existen o se crearon nuevos)
- esto tira abajo la performance de los equipos
- Solo ocurren al borrar o crear nuevos nodos (índices), en la actualización no pasa
- Los administradores de bdd borran y crean nuevos índices para que no haya agujeros

### **► Borrado y regeneración de índices**

- Mejorar el rendimiento y optimización de recursos
- Acelerar la búsqueda

Regla de escritorio:




- Si accedemos a un campo que no es índice, hay que pedir enseguida que se cree un índice
- Cuando se hace una modificación en los datos, y esa modificación afecta a alguno de los índices que tenemos en los datos se hace en tiempo real

#### En resumen:

- Las bases de datos se componen de tablas, y las tablas se pueden optimizar con índices.
- Los índices permiten búsquedas más rápidas y eficientes, mejorando el rendimiento general de la base de datos.

## Opeaciones

### ► Transaccion

- Conjunto de operaciones que quieren que se tomen de manera unívoca, o se cumple todo, o no se cumple nada
- Cuando se hacen modificaciones a más de una tabla, se necesitan tomar una transacción para que no haya inconsistencias
- Utilizadas cuando se hacen actualizaciones masivas
- Se basa en el principio  **ACID**
- Para decirle al gestor que haga las modificaciones:
  - begin transaction (las hace pero no las ejecuta)
  - end transaction (si se cumplen, da el paso a ejecutar)

#### Optimizacion:

- Marcar la tabla como read only. Las consultas van a ser más performantes y rápidas  
No bloquean los recursos ni afectan

## Principio ACID

#### Atomicidad:

Unidad atomica de trabajo, implica que todas las operaciones de la transaccion se ejecutan con exito o ninguna se ejecuta en lo absoluto

### **Consistencia:**

Luego de que la transaccion se completo con exito, la bdd debe mantenerse en un estado de consistencia

es decir

todas las restricciones de integridad deben cumplirse en todo momento

### **Aislamiento:**

Cada transaccion debe ejecutarse como si fuera la unica transaccion en el sistema

es decir

Las operaciones de una transaccion no deben interferir con las operaciones de otras transacciones que se estan ejecutando simultaneamente

### **Durabilidad:**

Despues de que la transaccion se confirma en la base de datos, sus cambios deben ser permanentes y resistir a fallos del sistema

## **► Programacio Batch/Shell**

- Se refiere a un script que contiene instrucciones o comandos SQL que se ejecutan en lotes para realizar operaciones en la base de datos de manera automatizada.
- Comandos dependen del sistema operativo, ej: windows (batch) y shell (linux)
- Estos scripts pueden incluir comandos para crear, modificar, eliminar o consultar datos en la base de datos.
- Tenemos dos formas diferentes de hacer esta actualizacion masiva
  1. Se hace una pequeña aplicacion que solamente hace la modificacion que se le pide a la tabla.  
Si hubo un problema al hacer una accion (se hizo hasta la mitad) → puedo hacer un begin transaction y end transaction: se toma todo como una unica transaccion → me ayuda a seguir con la integridad de datos

→

**al terminar se especifica un commit o un rollback**

2. Se usa cuando la bdd es muy grande, si tabla es muy grande y la modificacion tmb, y sobre todo si la modificacion lleva consigo borrar y/o actualizar registros que son campos de claves. Entonces se hace un ►  
**ETL**

## ► **BCP - Bulk copy program**

- Herramienta utilizada para importar y exportar grandes volúmenes de datos en bases de datos
- Se conectan por la puerta de atras del motor de bdd e inserta/elimina masivamente registros
- Permite realizar operaciones de carga masiva (bulk) de datos desde o hacia archivos

## ► **ETL - Extraccion, transformacion y carga**

- Este es un proceso utilizado en la integración de datos donde se extraen datos de múltiples fuentes (de donde sea, ej: pagina web, un archivo, etc), se transforman según las necesidades y se cargan en una base de datos
- La idea es sacarle el esfuerzo al motor de base datos procesandolo por afuera (ej: de forma local) y con un programa batch hacer la carga y descarga
- Los programas de ETL a menudo se utilizan en flujos de trabajo (workflow) para automatizar la extracción, transformación y carga de datos de manera eficiente.
- ejemplo: al procesar los datos podria ordenarlos de forma fisica en cierto orden que me interese y ademas puedo mantener un indice que tenga una forma optimizada los mismos datos
  - Esto incluye la eliminación de duplicados, la validación de datos, la normalización y otros procesos que aseguran la calidad y consistencia de los datos antes de su uso.

### Proceso:

1. vuelca (dump) los datos a modificar (en un archivo o lo que queiras)
2. se hace una modificacion de los registros de forma batch
3. se hace un delete de esos registros, drop table, se borran los indices
4. por fuera hago un algoritmo de sorting para dejarlo de la misma estructura para la clave principal
5. se carga en el orden en el que esta dado la clave primario

## ► Store Procedures - Procedimientos almacenado

- Conjunto de instrucciones SQL que se guardan en la bdd y se pueden ejecutar manera repetida cuando sea necesario
- Utilizados para realizar tareas repetitivas o complejas en una base de datos de una manera mas eficiente y segura
- Se ejecutan manualmente al llamarlos.
- Usos comunes:
  - Procesamiento de datos
  - Validaciones y seguridad
  - Transacciones
  - Mejora el rendimiento
  - Reutilizacion de codigo

### Beneficiones

- Mejora el rendimiento → el analisis lexicografico que hace SQL solo lo hace una vez
- Aporta seguridad, ya que son instrucciones diseñadas especificamente por un administrador, pensadas para que los usuarios las ejecuten.
- Abstraen el trabajo a los programadores: **Lo unico que programamos son los stored procedures y desde programa accedemos a ellos**
- Hace que los programadores no tengan que ser expertos en sql

- Ordenar todo el trabajo de diseño y desarrollo / gestion paralela del desarrollo
  - Elimina tiempos muertos → el administrador puede crear un store procedure dammy/maqueta

## Como funciona

el sql para ejecutar una consulta hace lo que se ve al costado.

Si tenemos consultas en stored procedura → nos ahorramos de hacer siempre el analisis lexicografico

1. SQL interpreta el analisis lexicografico, analiza si la consulta que escribimos esta bien escrita
2. Luego analiza los permisos
3. Ejecuta la consulta

## Solo analiza los permisos

Todas las consultas deben pasar por el filtro de los administradores de bdd / todas las consultas estan guardadas en un stored procedures

1. Tiene ventajas de ir haciendo mas performante la base de datos
2. Tienen un inventario de todas las consultas

## Buenas practicas

- Se optimizan las operaciones de bdd
- Se validan el tipo y el rango de datos para c/uno de ellas
- Integridad de mi bdd
- Acceder a los datos y validar a los mismos, en el resultado de la consulta poder hacer una normalizacion dentro de mi bdd
- No esta pensado apra formatear datos, workflow o validar usuarios que pueden acceder o no a un proceso
- Meter todo en un SP es un error, pues estan hechas para acceder de manera optima a la bdd

## ► Triggers

- Acciones que se van a realizar en una bdd en funcion de algo

- Se ejecutan antes o despues de una consulta / transaccion
- Se usa principalmente cuando se va a insertar, borrar o modificar un registro
- Especie de procedimiento almacenado que se ejecuta automaticamente cuando ocurre un evento especifico en la bdd
- Se utilizan principalmente para mantener la integridad de los datos, realizar validaciones o ejecutar acciones adicionales en respuesta de ciertos cambios en la bdd
- se ejecutan automaticamente.

### **Ejemplos:**

1. Podria utilizarse para actualizar automaticamente un campo en una tabla cuando se inserta una nueva fila o para validar ciertos datos antes de permitir una operacion
2. Un email de alerta

### **Contras**

- el uso excesivo de triggers impacta en la performance

## ► **BCP - Bulk copy program**

- copia masiva que accede por una



## **Metadatos**

- datos de los datos
- Hacen referencia al conocimiento de cual es la estructura de los datos con los que vamos a trabajar



## **Views - vistas**

- Abstraccion de una o varias tablas → tabla virtual
- Solo existe a nivel logico

- Permite trabajar con los usuarios, las personas que acceden a la bdd y mostrarles un subconjunto de campos
- se hacen para simplicidad y seguridad.

#### **Ejemplo:**

- Si solo le quiero mostrar dos campos y esa consulta la hago de forma repetitiva y constante → voy a consumir muchas consultas
- Puedo crear esa consulta como una vista → el usuario la consulta como si fuera una tabla
- le hago creer al usuario que es una tabla (Pero esta fue controlada por el administrador)



## **Paginacion**

- Tecnica utilizada para dividir y presentar grandes conjuntos de datos en partes mas pequeñas y manejables, lo que beneficia tanto a los usuarios como a los sistemas que gestionan esa informacion
- En otras palabras: division del contenido en paginas pequeña
- buscar como pagina ese framework (habilidad de poder saltar entre frameworks)



## **Frameworks**

- proporcionan herramientas y funcionalidades para implementar paginacion de manera efectiva
- Saber como se conecta el framework a una bdd
  - buscar como pagina ese framework (habilidad de poder pasar de un frameworks a otro)



## Bloque de bdd (Lock)

- mecanismo utilizado para controlar el acceso concurrente a los datos donde la idea es garantizar la consistencia y la integridad de la información.
- impedir el acceso al mismo tiempo de otros usuarios.
- El motor define como va a tomar de manera única un registro para que esa particita no sea accedida al mismo tiempo por otros usuarios mientras se modifica el dato en ese momento.
- Por defecto la bdd hace lock automáticamente. Se genera un procedimiento por el cual según la cantidad de registros decide si es a nivel de tabla, registro, o celda.

### Tipos de bloqueo

- acción: de lectura, de escritura, de actualización
- nivel: de tabla, de fila/registro, de celda/columna de un registro

### Bloqueo de tablas

- El bloqueo de tablas en una base de datos ocurre cuando un proceso o transacción adquiere un bloqueo exclusivo en una tabla, impidiendo que otros procesos o transacciones realicen operaciones de lectura o escritura en la misma tabla hasta que el bloqueo sea liberado.
- Hay que entender como trabaja el proceso de update/borrado en el motor de bdd que estoy trabajando

**no es un conocimiento trivial, tengo que informarme**

### Beneficios

- Impide inconcesistencias
- integridad de la información

### Contras

- La bdd van a intentar lockear lo menos posible
- Impacta mucho en el motor
  - La obligación del motor es recibir las consultas y ver si estas se encuentran en lo que está marcado como lock.
  - Pero no puede verificar todas las consultas al mismo tiempo.



- Tiene que esperar a que se haga con realice a la primera consulta para atender la segunda y así sucesivamente
- Durante este proceso, el usuario lo que vio fue decaer el tiempo de performance en las respuestas

Ejemplo:

- cuando hago select → consulto → no hay bloqueos (tomo una foto)
- 



## ODBC

Maneras de acceso a una bdd:

- librería propia de la bdd
  - no existe una capa de abstracción en el medio
  - con esta manera gano performance pero pierdo movilidad
- **ODBC (Option data base connection)**

## No se suelen usar ahora

que son

- forma estándar de acceder a una bdd (sobre todo usando windows)
- capa de abstracción

**Ventajas**

- Podemos configurar un acceso estándar

**Contras**

- Agrega capa de abstracción → pierde performance
- 



## Ganar performance en bdd

- usar librería propia de la bdd → pero pierdo movilidad
- store procedures
- borrado y regeneración de índices

- base de datos no-sql
- eliminar locks → pero pierdo seguridad
- no usar triggers → pero pierdo
- proceso de desnormalizar tablas
- Tener muchas claves secundarias → genera nuevas estructuras que hacen mas eficiente la busqueda
- Hacer operaciones masivas a la noche → donde la cantidad de consultas disminuyen
- Reducir los joins → estos son costosos
  - ★ los 2 consejos de oro para hacer mas performante un join:
    1. los campos que por los que voy a acceder (donde se coloca la igualdad → ON) en la tabla donde se busca la informacion (no la que se recorre) sea indice. Asi evitamos full scan
    2. Inferir sobre como optimizar cada consulta dependiendo del motor de bdd. Pues cada motor de bdd hacen optimizaciones, entonces cada uno tiene formas diferentes de resolver cada operacion

---

## ★ MERLINO TIPS

- Frameworks:
    - Saber como se conecta el framework a una bdd
    - buscar como pagina ese framework (habilidad de poder pasar de un frameworks a otro)
  - es mucho mejor entender muy bien un lenguaje, los conceptos luego se trasladan
  - Cuando nos piden que tenemos que hacer una consulta o alguna aplicacion → tenemos que pedir que nos pasen el DER (diagrama de entidad relacion) de la bdd → nos permite visualizar la secuencia
  - Regla para el JOIN: si no sabemos cual usar, usamos LEFT JOIN (?)
-

# ★ Buenas practicas de programacion al hacer consulta:

debemos comentar encabezados:

1. Colocar autor (quien hizo la consulta)
2. Fecha
3. Breve descripcion de lo que hace la consulta
4. Indicar los parametros y tipos de entrada
5. Indicar la salida que obtiene le usuario

## Plataforma de datos

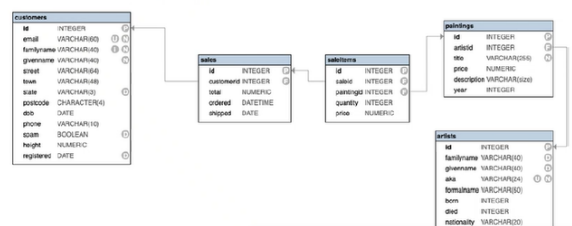
- Lugar donde vamos a consultar conjuntos de datos de manera segura
- Combinaciones y secuencias de datos, que pueden estar integrador por varios motores de bdd
- conjunto de herramientas y recursos que se utilizan para recompilar, almacenar, procesar, analizar y compartir datos de manera eficiente y segura

ejemplo:

- Mercado libre fue una plataforma de datos → las cestionones transaccionales estan separada de los datos
- En un supermercado cuand ollegamos a la caja → pasamos la tarjeta (esto tarda) → despues de unos segunods nos da la aprovacion → esto se debe a la infraestructura de datos

## Diagrama de entidad - relacion (DER)

- Cuando nos piden que tenemos que hacer una consulta o alguna aplicacion → tenemos que pedir que nos pasen el DER (diagrama de entidad relacion) de la bdd →



nos permite visualizar la secuencia

## Estructura

- La bdd puede tener 0 o N tablas
- Un motor de bdd administra 0 o N base de datos
- Los datos físicamente están guardados dentro de las tablas

## Indices

**Primary keys (P)**: son como físicamente se almacenan los datos en la base de datos

**clave foránea / Foreign (F)**: me dicen (de donde salen las flechas) que es clave en otra tabla



## Formas normales

- Se tiene que considerar que cada campo debe tener un valor por defecto → validación de datos

## ► Normalización

- Es el proceso de organizar los datos en una base de datos para reducir la redundancia y la dependencia de datos.
- Proteger la integridad de datos → el dato que veo, es el que necesito ver (bajar la probabilidad de que haya errores)
- Disminuir problemas de actualización
- Se busca dividir las tablas en entidades más pequeñas y relacionadas entre sí de manera lógica.

## Ejemplo:

- Si tienes una tabla de clientes con información como nombre, dirección y teléfono, y otra tabla de pedidos con información sobre productos y fechas
- al vincular los pedidos a los clientes a través de un identificador único, evitando así la repetición de datos de clientes en cada registro de pedido.

## Clave foranias

- Clave que es clave en otra tabla
- se relacionan generalmente con al clave primaria pero no necesariamente tiene que ser

## ▶▶ Primera Forma Normal (1FN)

- Todos los atributos son individuales / cada columna tiene un solo valor
- No existen valores repetidos
- No hay grupos repetidos de columnas

## ▶▶ Segunda Forma Normal (2FN)

- Todo registro depende de la clave principal
- Existen columnas que pueden depender de otras tablas
- Cumple con 1NF
- Cada columna no clave esta completamente dependiendo de la primaria

## ▶▶ Tercera Forma Normal (3FN)

- No hay columnas que dependan de otras tablas/columnas que no sean la clave principal
- Cumple con 2NF
- No existen dependencias transitivas entre las columnas no clave

## ▶▶ Forma Normal de Boyce-Codd (BCNF)

- Cumple con 3NF
- No tiene dependencias funcionales no triviales (no tiene campos repetidos)

- Se aplica para eliminar ciertas anomalías de actualización que pueden surgir en bdd normalizadas hasta 3NF
- Se espera que este entre 3NF y 4NF (se le llama 3.5NF)
- Dependencia funcional: Una o varias columnas determina otra columna, y esa determinación no es evidente o trivial

## ► Desnormalización

- Es el proceso contrario a la normalización, donde se combina información de varias tablas relacionadas para mejorar el rendimiento de consultas específicas.
- Se utiliza cuando se necesita optimizar el rendimiento de consultas complejas a costa de aumentar la redundancia de datos.
- Puede implicar la incorporación de datos derivados, la duplicación de información en tablas o la eliminación de algunas relaciones para simplificar las consultas.

### Ejemplo

- Si tienes una base de datos normalizada con múltiples tablas, pero necesitas realizar consultas complejas que requieren unir varias tablas repetidamente, puedes desnormalizar algunas tablas combinando datos relacionados en una sola tabla para mejorar el rendimiento de esas consultas específicas.



## Preguntas que debe saber un administrador para elegir un motor de bdd:

1. Que tipo de datos se va a usar?
2. Cual es la operatorio a la cual le quiero dar prioridad en cuestion de performance? (no necesariamente es la que mas se repite)  
es decir,

Cual/es o son las acciones / tipo de acceso que voy a querer que se destaque?

- Capas puedo hacer actualizaciones masivas de noche de forma batch u online por la noche, no me importa

---

## Sincronizacion entre nodos distribuidos

**Nodos de base de datos:** Son las instancias individuales de la base de datos que participan en la replicación. Pueden estar ubicados en servidores diferentes o en la misma máquina, dependiendo de la arquitectura y la configuración de la base de datos.

En el contexto de la sincronización entre nodos distribuidos en bases de datos, hay algunas diferencias notables entre las bases de datos SQL (estructuradas) y las NoSQL (no estructuradas) en términos de cómo se gestionan los nodos distribuidos y cómo los administradores de bases de datos interactúan con ellos.

es mas probable que un admin de nosql tenga que administrar esto que un admin de sql se encuentre con la necesidad de administrar esto

en nosql se suele resolver con un servidor mejor

---

## El problema de JOIN



En una operación de join en SQL, la cláusula `ON` se utiliza para especificar la condición de unión entre las tablas involucradas. La tabla que se menciona primero en la cláusula `FROM` (antes del `INNER JOIN` o cualquier tipo de join) se considera la tabla principal o tabla de búsqueda, mientras que la tabla mencionada después del `INNER JOIN` (o cualquier otro tipo de join) se considera la tabla que se recorre para encontrar las filas que coinciden con la condición de unión especificada en la cláusula `ON`.

```
SELECT c.nombre, p.id_pedido
FROM Clientes c
INNER JOIN Pedidos p ON c.id_cliente = p.id_cliente;
```

- Un problema es que la columna `p.id_cliente` no está indexada, esta va a hacer un full scan..
- Es que si no se ve que tipo de JOIN usar, puedo hacer consultas de forma incorrecta y que no me devuelva los campos que necesito. Hay que tener cuidado y revisar si necesito hacer inner, left o right.



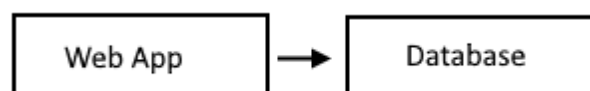
## Servidor de Colas

Digamos que tiene una aplicación web que escribe en una base de datos.

Quizás haya picos periódicos en los que quizás haya millones de solicitudes de escritura que causen una latencia significativa.

En el peor de los casos, esto quizás podría hacer que la aplicación web se cargue

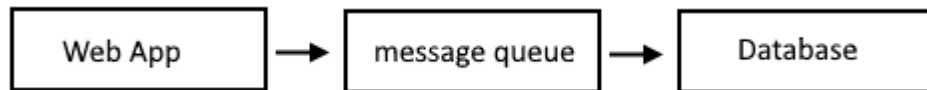
interminablemente, no se cargue en absoluto, devuelva errores de tiempo de espera, muestre errores inestables. No es bueno.



para eso se pone una cola



luego la base de datos podría consumir los mensajes en la cola. Esta es básicamente la arquitectura de publicación/suscripción (pub/sub), donde la aplicación web publica el mensaje en la cola y la base de datos se suscribe a la cola.



alguien me lo resume porfa

<https://www.freekb.net/Article?id=2042>



## Clase Arquitectura