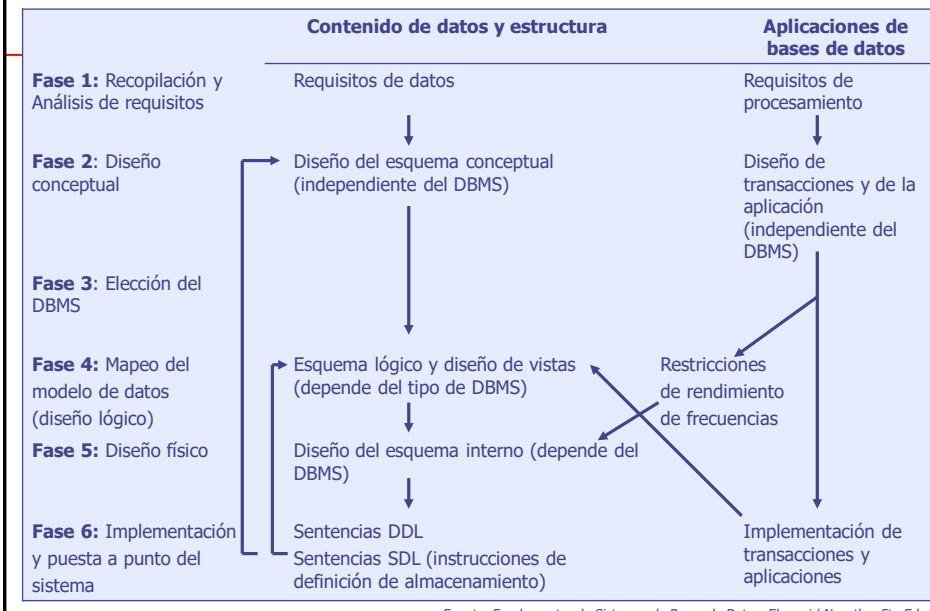
Objetivos del Curso

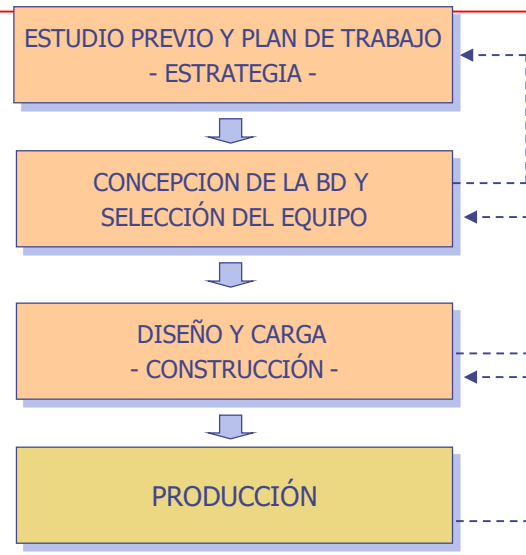
- Comprender la arquitectura y funcionamiento de los manejadores de bases de datos.
- Desarrollar la capacidad para gestionar la implementación de una base de datos, considerando los procesos de administración, optimización y mantenimiento, así como la seguridad de la información.

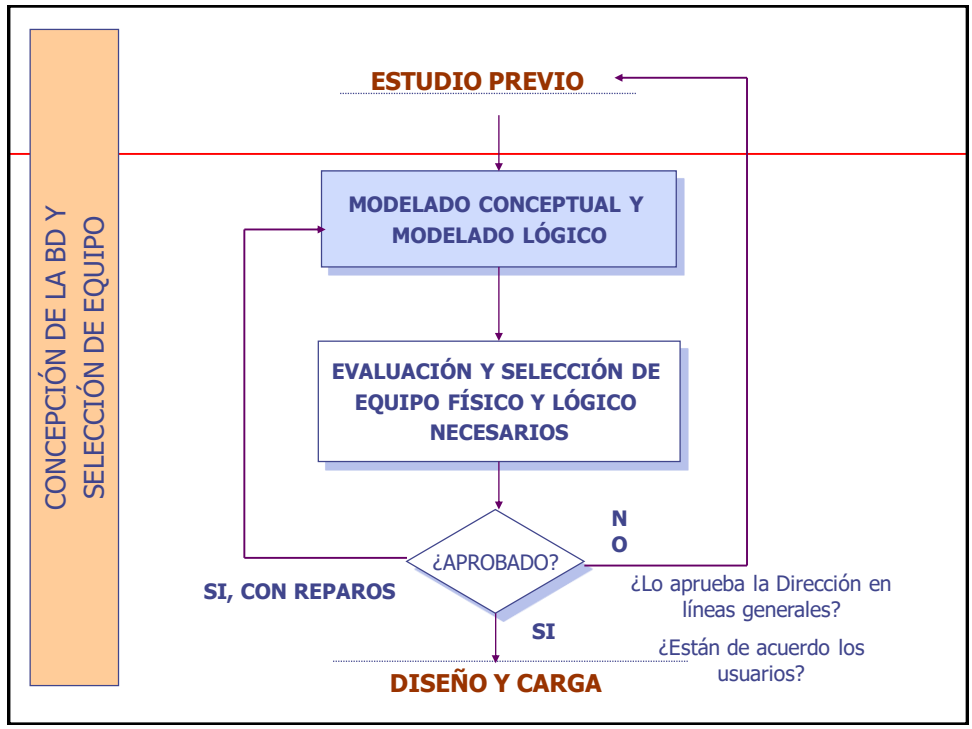
Fases en el diseño e implementación de una base de datos grande

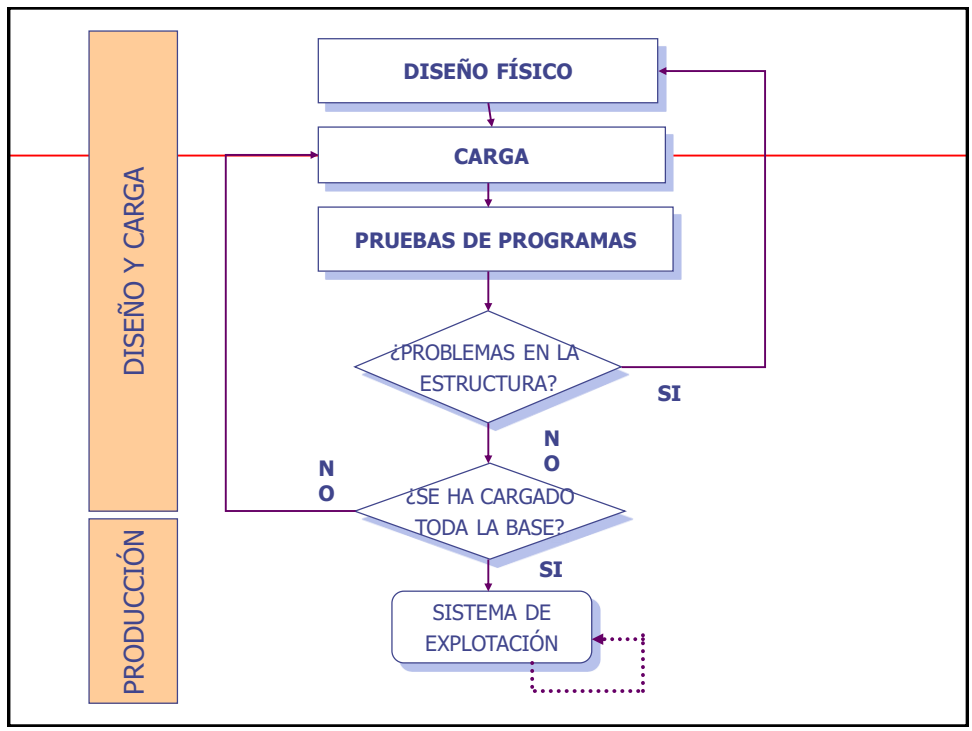


Fuente: Fundamentos de Sistemas de Bases de Datos. Elmasri / Navathe. 5ta Ed.

El Ciclo de Vida de una BD

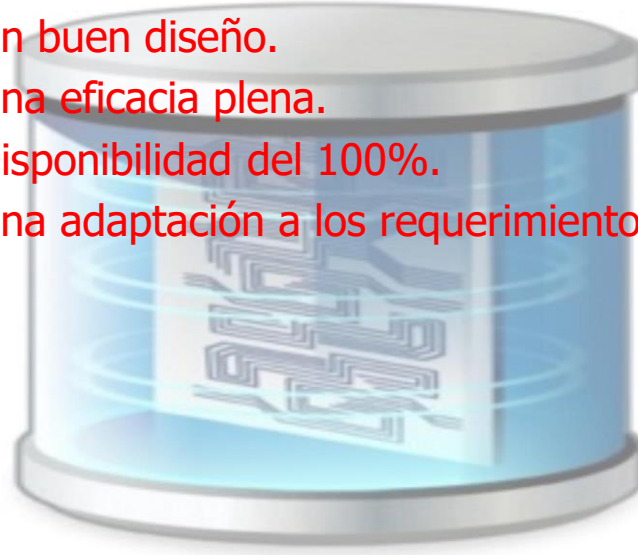



CONCEPCIÓN DE LA BD Y SELECCIÓN DE EQUIPO
ESTUDIO PREVIO
MODELADO CONCEPTUAL Y MODELADO LÓGICO
EVALUACIÓN Y SELECCIÓN DE EQUIPO FÍSICO Y LÓGICO NECESARIOS
¿APROBADO?
NO
¿Lo aprueba la Dirección en líneas generales?
SI, CON REPAROS
SI
¿Están de acuerdo los usuarios?
DISEÑO Y CARGA


DISEÑO Y CARGA
DISEÑO FÍSICO
CARGA
PRUEBAS DE PROGRAMAS
¿PROBLEMAS EN LA ESTRUCTURA?
SI
NO
NO
¿SE HA CARGADO TODA LA BASE?
SI
PRODUCCIÓN
SISTEMA DE EXPLOTACIÓN

Características de una BD

- Un buen diseño.
- Una eficacia plena.
- Disponibilidad del 100%.
- Una adaptación a los requerimientos.



Motivación

- **¿ Como podemos implementar exitosamente una Base de Datos ?**

Lenguaje

SQL

SQL

- El **Lenguaje de consulta estructurado** (**S**tructured **Q**uery **L**anguage) es un lenguaje declarativo de acceso a bases de datos relacionales que permite especificar diversos tipos de operaciones en éstas
- Una de sus características es el manejo del álgebra relacional permitiendo efectuar consultas con el fin de recuperar -de una forma sencilla- información de interés de una base de datos, así como también hacer cambios sobre ella

SQL

- **El SQL es un lenguaje de acceso a bases de datos que explota la flexibilidad y potencia de los sistemas relacionales permitiendo gran variedad de operaciones**
- **Es un lenguaje declarativo de "alto nivel" o "de no procedimiento", que gracias a su fuerte base teórica y su orientación al manejo de conjuntos de registros, permite una alta productividad en codificación**

SQL

- Este curso pretende brindar lo básico y esencial de SQL
- No se pretende realizar un estudio exhaustivo de todas las opciones, comandos y aspectos de almacenamiento y administración que se pueden considerar en SQL
- Sólo se pretende presentar y explicar los comandos más utilizados con sus opciones más útiles, dejando los detalles más específicos a los manuales de referencia

Introducción al SQL

- **Sentencias DDL:**
  - CREATE
  - ALTER
  - DROP
  - TRUNCATE
  - COMMENT
  - RENAME

Introducción al Lenguaje SQL

- **Sentencias DCL:**
  - **GRANT**
  - **REVOKE**

Introducción al Lenguaje SQL

- **Sentencia DML:**
  - **SELECT**
  - **INSERT**
  - **UPDATE**
  - **DELETE**
  - SELECT
    - Cláusulas SELECT, FROM, INTO, WHERE
    - Cláusulas GROUP BY, HAVING
    - Cláusula ORDER BY
    - Tipos de JOIN
    - Subqueries

Introducción al Lenguaje SQL

- **Sentencias TCL:**
  - **COMMIT**
  - **SAVEPOINT**
  - **ROLLBACK**
  - **SET TRANSACTION**

SEGURIDAD
EN BASE DE DATOS

Objetivo

- Conocer los términos y conceptos para gestionar la seguridad de una base de datos en una organización identificando los riesgos y estableciendo los controles necesarios para mitigarlos

Objetivo

- En las bases de datos se plantean problemas de seguridad como la compartición de datos, acceso a estos, protección contra fallos, contra accesos no permitidos, etc.
- El DBMS facilita mecanismos para prevenir los fallos (subsistema de control), para detectarlos (subsistema de detección) y para corregirlos (subsistema de recuperación).
- Aspectos fundamentales de la seguridad:
  - Confidencialidad. No desvelar datos a usuarios no autorizados. Comprende también la privacidad (protección de datos personales).
  - Accesibilidad o disponibilidad. La información debe estar disponible y también el acceso a los servicios.
  - Integridad. Permite asegurar que los datos no han sido falseados o modificados de forma indebida.

Seguridad y Auditoria

- Las bases de datos son el activo más importante para las organizaciones.
- Los datos confidenciales en manos ajenas puede ser muy riesgoso.
- Por ello se deben controlar aspectos cruciales en la seguridad de la misma.
- Con la auditoría de bases de datos se busca monitorear y garantizar que la información está segura, además de brindar ayuda a la organización para detectar posibles puntos débiles y así tomar precauciones para resguardar aún más los datos.

Ejercicio 1: Identificación de Activos

Identifique 3 Activos de Información:

- ✓ _____
- ✓ _____
- ✓ _____

Ejercicio 2: Identificación de Amenazas

Identifique 3 Amenazas:

- ✓ _____
- ✓ _____
- ✓ _____

Ejercicio 3: Identificación de Vulnerabilidades

Identifique 3 Vulnerabilidades:

- ✓ _____
- ✓ _____
- ✓ _____

¿Qué es un Activo de información?

Es todo aquello que tiene valor para la organización y por lo tanto requiere protección:

Documentos en papel: contratos, guías
Software: aplicativos y software de sistemas
Dispositivos físicos: computadoras, medios removibles
Personas: clientes, personal, etc.
Imagen y reputación de la Institución: marca
Servicios: comunicaciones, internet, energía.

Tipos de Activos que requieren protegerse




Activo de Información es todo lo que es o contiene información. Si la información no está dentro, no es un activo de información. No confundir con activos fijos.

LA CONTRALORÍA GENERAL DE LA REPÚBLICA

Seguridad vía SQL

Privilegios:

- **Nivel de cuenta. Privilegios de usuario. CREATE SCHEMA, CREATE TABLE, CREATE VIEW, ALTER, DROP, MODIFY, SELECT**
- **Nivel de relación. Se aplican a las relaciones individuales: SELECT, MODIFY, REFERENCES.**
- **Los privilegios se dan o quitan con GRANT y REVOKE.**
- **Además se pueden crear y eliminar roles de usuario con CREATE ROLE… y DROP ROLE…**

Encriptación

- **La encriptación es básicamente transformar datos en alguna forma que no sea legible sin el conocimiento de la clave o algoritmo adecuado. El propósito de esta es mantener oculta la información que consideramos privada a cualquier persona o sistema que no tenga permitido verla.**

Encriptación

- **La encriptación es el proceso de volver ilegible información considerada importante. La información una vez encriptada sólo puede leerse aplicándole una clave.**
- **Se trata de una medida de seguridad que es usada para almacenar o transferir información delicada que no debería ser accesible a terceros. Pueden ser contraseñas, números de tarjetas de crédito, conversaciones privadas, etc.**

Proceso de Encriptación

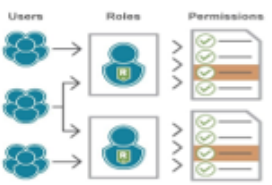
- **Texto a codificar:** **ENCRYPTION**
- **Caracteres del Texto:** **E N C R Y P T I O N**
- **Códigos ASCII:** **69 78 67 82 89 80 84 73 79 78**
- **Texto codificado:** **“œ?©Ÿ?¨**

Usuario

- ✓ **Un usuario es un nombre definido en la base de datos que puede conectarse a ella y acceder a determinada información según los permisos que tenga asignados por el administrador**
- ✓ **El objetivo de la creación de usuarios es establecer una cuenta segura y útil, que tenga los privilegios adecuados y los valores por defecto apropiados**

Rol

- **Un rol es el conjunto de permisos. Los roles permiten agrupar los derechos y gestionar más fácilmente los diferentes usuarios.**
- **Siempre es preferible asignar los derechos a los roles y posteriormente asignar los roles a los usuarios que tenga asignados por el administrador**



Rol en MS SQL Server

- **Cada rol agrupa un conjunto de permisos**
- **Facilitan la administración de la seguridad**
- **Se definen a nivel de servidor, independiente, de las bases de datos**
- **Un inicio de sesión puede pertenecer a cero o más roles de servidor**
- **Un inicio de sesión que pertenezca a un rol de servidor adquiere los permisos de ese rol**

Microsoft SQL Server

Privilegios

- **Los privilegios son atributos que permiten a un usuario realizar determinadas operaciones dentro de una BD o acceder a objetos de otros usuarios**
- **Existen dos tipos de privilegios:**
  - **Privilegios sobre los objetos**
  - **Privilegios del sistema**

Creación de Usuarios, Permisos y Roles

FACULTAD de INGENIERÍA de
SISTEMAS E INFORMÁTICA

Creación de Login

- Un **login** es la capacidad de poder utilizar una instancia del **Servidor SQL**, está asociado con un usuario de Windows o con un usuario de **SQL**.

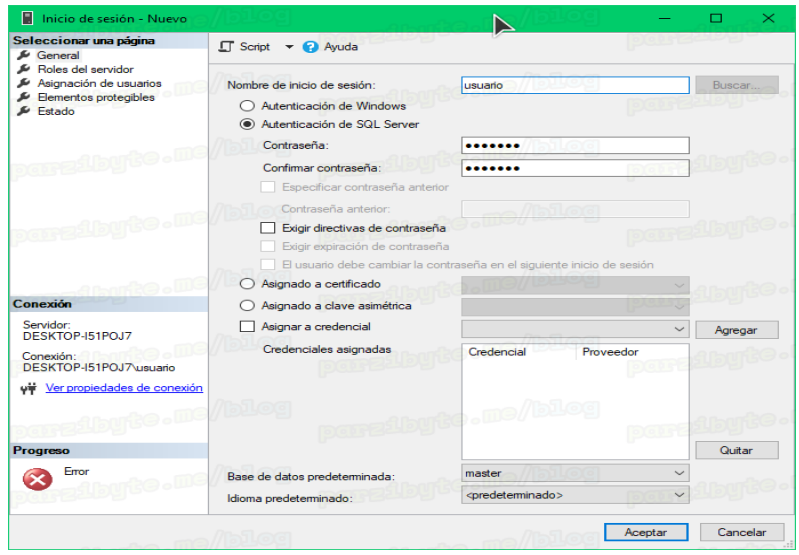
Creación de Login

CREATE LOGIN <login_name> WITH PASSWORD = '<enterStrongPasswordHere>';

CREATE LOGIN <login_name> WITH PASSWORD = '<enterStrongPasswordHere>' MUST_CHANGE, CHECK_EXPIRATION = ON;

```
CREATE LOGIN [MyUser]
WITH PASSWORD = 'MyPassword',
DEFAULT_DATABASE = MyDatabase,
CHECK_POLICY = OFF,
CHECK_EXPIRATION = OFF ;
```

Creación de un login



Roles

- Los roles son los conjuntos de permisos.
- Los roles permiten agrupar los derechos y gestionar más fácilmente los diferentes usuarios y las conexiones.
- Siempre es preferible asignar los derechos a los roles y posteriormente asignar los roles a los usuarios.
- Con una estructura como esta, la adición y la modificación de permisos o de usuarios son más sencillas.

Roles

- **Nivel de servidor**

Para controlar el acceso a los recursos del servidor

- **Nivel de BD**

Acciones administrativas de la BD

Roles predefinidos

Rol fijo de base de datos	Descripción
db_owner	Puede realizar todas las actividades de configuración y mantenimiento en la base de datos y también pueden eliminar la base de datos en SQL Server.
db_securityadmin	Puede modificar la pertenencia a un rol y administrar permisos. La adición de principiantes a este rol podría permitir la escalada de privilegios no deseados.
db_accessadmin	Puede agregar o quitar acceso a la base de datos para los inicios de sesión de Windows, los grupos de Windows y los inicios de sesión de SQL Server.
db_backupoperator	Puede hacer un backup de la base de datos.
db_ddladmin	Puede ejecutar cualquier comando de lenguaje de definición de datos (DDL) en una base de datos.
db_datawriter	Puede agregar, eliminar o cambiar datos en todas las tablas de usuario.
db_datareader	Puede leer todos los datos de todas las tablas de usuario.
db_denydatawriter	No puede agregar, modificar o eliminar datos de las tablas de usuario de una base de datos.
db_denydatareader	No puede leer ningún dato en las tablas de usuario de una base de datos.

Ejemplo de creación de login

```
create login Miguel with password 'Xyz%1234#'
go
use Ciclismo
go
create user Miguel for login Miguel
```

GRANT y REVOKE

- **Sentencias DCL:**
  - **GRANT**
  - **REVOKE**

GRANT

GRANT lista_privilegios ON tabla TO lista_usuarios [WITH GRANT OPTION]

- **Lista_privilegios: privilegios: SELECT, INSERT, DELETE, UPDATE**
- **Tabla: nombre de la tabla**
- **Lista_usuarios: usuarios a los que se les dara los privilegios**

Ejemplos de GRANT

- **Ejemplo de como se asigna todos los privilegios en la tabla Persona a Juan:**

  GRANT ALL ON Persona TO Juan

  WITH GRANT OPTION
- **Ejemplo de como se da el privilegio de SELECT sobre la tabla Persona a Juan:**

  GRANT SELECT ON Persona TO Juan

Ejemplos de GRANT

- **Ejemplo de como se asigna algunos privilegios en la columna Nombre en la tabla Persona a Juan:**

  GRANT SELECT, INSERT,UPDATE (Nombre) ON Persona

  TO Juan

  WITH GRANT OPTION

REVOKE

REVOKE [GRANT OPTION FOR] lista_privilegios ON tabla FROM lista_usuarios { RESTRICT | CASCADE }

- **Lista_privilegios: privilegios: SELECT, INSERT, DELETE, UPDATE**
- **Tabla: nombre de la tabla**
- **Lista_usuarios: usuarios a los que se les quitara los privilegios**

Ejemplos de REVOKE

- **Ejemplo de como se desasigna el privilegio de consultar datos de la tabla Persona a Juan:**

  REVOKE SELECT ON Persona FROM Juan

Ejemplos de GRANT Y REVOKE

```
GRANT SELECT, INSERT on Cliente to Miguel;
GO
GRANT ALL on Cliente to Miguel;
GO
REVOKE INSERT on Cliente to Miguel
```

Ejemplos de ROLES

```
ALTER ROLE db_reader add member Miguel;
GO
ALTER ROLE db_reader drop member Miguel;
GO
ALTER ROLE db_writer add member Miguel;
```

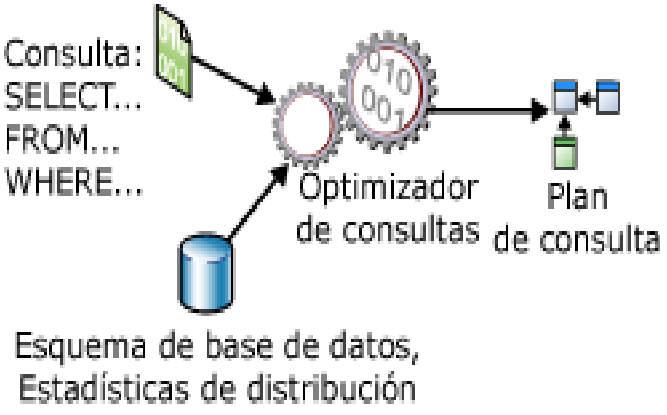
Ejemplos de DROP

```
DROP USER Miguel;
GO
DROP LOGIN Miguel
```

PROCESAMIENTO Y OPTIMIZACION DE CONSULTAS

FACULTAD de INGENIERÍA de
SISTEMAS E INFORMÁTICA

Procesamiento en SQL



Procesamiento en SQL

Una instrucción SELECT define únicamente los siguientes elementos:

- **El formato del conjunto de resultados, este elemento se especifica principalmente en la lista de selección**
- **Sin embargo, también afectan a la forma final del conjunto de resultados otras cláusulas como ORDER BY y GROUP BY**
- **Las tablas que contienen los datos de origen. Esto se especifica en la cláusula FROM**

Procesamiento en SQL

- **¿ Cómo se relacionan lógicamente las tablas para la instrucción SELECT ?**

Esto se define en las especificaciones de combinación, que pueden aparecer en la cláusula WHERE

- **Las condiciones que deben cumplir las filas de las tablas de origen para satisfacer los requisitos de la instrucción SELECT. Estas condiciones se especifican en las cláusulas WHERE y HAVING**

Plan de Ejecución

- Un plan de ejecución es el conjunto de pasos que tiene que realizar el DBMS para ejecutar una consulta
- Un plan de ejecución de una consulta SQL es una definición de:
  - La secuencia en la que se tiene acceso a las tablas de origen
  - Los métodos que se utilizan para extraer los datos de cada tabla

Optimización

- El proceso de selección de un plan de ejecución entre varios planes posibles se conoce como optimización
- El optimizador de consultas es uno de los componentes más importantes de un sistema de base de datos SQL

Análisis de Consultas y Transacciones

Para cada consulta establecer:

a. Las tablas a las que accederá
b. Los atributos sobre los que se especificarán condiciones de selección (WHERE)
c. Los atributos sobre los que se especificarán condiciones de reunión o de enlace de tablas
d. Los atributos cuyos valores se obtendrá en la consulta

*Los atributos de los incisos **b** y **c** son candidatos a constituir índices (estructuras de acceso)*

Análisis de Consultas y Transacciones

Para cada transacción de actualización establecer:

a. Las tablas que actualizará
b. El tipo de operación en cada tabla (insertar, modificar o eliminar)
c. Los campos sobre los que se especificarán condiciones de selección para operaciones de eliminación o modificación
d. Los campos cuyos valores alterará una operación de modificación

- ✓ *Los campos del inciso **c** son candidatos para índices*
- ✓ *Los campos del inciso **d** son candidatos a evitar en los índices, ya que su modificación requerirá la actualización de estas estructuras de acceso.*

AJUSTE DE INDICES

Objetivos:

- Evaluar dinámicamente los requerimientos, que pueden cambiar según época del año, día del mes o de la semana
- Reorganizar los índices para obtener mejor rendimiento

AJUSTE DE INDICES

- Ciertas consultas pueden tardar mucho en ejecutarse por falta de un índice apropiado
- Puede haber índices que no se utilicen
- Puede haber índices que originen trabajo adicional por estar definidos sobre atributos que sufren continuos cambios

AJUSTE DE CONSULTAS

Indicadores:

- Demasiados accesos al disco (por ejemplo una consulta de emparejamiento exacto que recorre una tabla completa)
- El plan de ejecución de consulta muestra que no se están usando los índices relevantes.

Ajuste de Consultas - Casos

1. Muchos optimizadores no usan índices en presencia de:
   - Expresiones aritméticas (SALARIO/365 > 10.50)
   - Comparaciones numéricas de campos de diferente tamaño y precisión (ACANT = BCANT donde ACANT es de tipo Integer y BCANT es Smallinteger)
   - Comparaciones con NULL (FECHA IS NULL)
   - Comparaciones de subcadenas (como (APELLIDO LIKE '%EZ')

Ajuste de Consultas - Casos

2. Los índices no suelen usarse en consultas anidadas que utilizan IN:

```
SELECT NSS FROM EMPLEADO
WHERE DNO IN      (SELECT DNUMERO
                   FROM DEPARTAMENTO
                   WHERE  NSS_JEFE = '3334444')
```

Puede no utilizar el índice definido sobre DNO en EMPLEADO, mientras que la utilización de DNO = DNUMERO en la cláusula WHERE con una consulta de un solo bloque puede ocasionar que el índice sí se utilice

Ajuste de Consultas - Casos

3. **Algunos *DISTINCT* pueden ser redundantes y podrían evitarse sin modificar el resultado**

   Un *DISTINCT* generalmente provoca una operación de clasificación y debe evitarse siempre que sea posible

Ajuste de Consultas - Casos

4. **El uso innecesario de tablas temporales puede evitarse juntando varias consultas en una sola, *a menos que* la relación temporal sea necesaria para algún resultado intermedio**

Ajuste de Consultas - Casos

5. **En algunas situaciones en las que se usa consultas correlacionadas son útiles las tablas temporales**

```
SELECT NSS
FROM EMPLEADO E
WHERE SALARIO = SELECT MAX(SALARIO)
                FROM EMPLEADO AS M
                WHERE M.DNO = E.DNO)
```

Esto tiene el peligro potencial de buscar en toda la tabla M EMPLEADO interna para cada tupla de E EMPLEADO externa.

Ajuste de Consultas - Casos

Para hacerlo más eficiente puede descomponerse en dos consultas, la primera de las cuales calcula el salario máximo de cada departamento:

```
SELECT MAX(SALARIO) AS SALARIO_MAYOR, DNO
INTO TEMP
FROM EMPLEADO
GROUP BY DNO;

SELECT NSS
FROM EMPLEADO, TEMP
WHERE SALARIO = SALARIO_MAYOR AND
EMPLEADO.DNO = TEMP.DNO
```

Ajuste de Consultas - Casos

6. **De haber varias opciones posibles para la condición de reunión, elegir una que use un índice de agrupación (CLUSTER), y evitar aquellas que contengan comparaciones de cadenas:**

   Si el campo NOMBRE es una clave candidata en EMPLEADO y en ALUMNO, es mejor usar

   EMPLEADO.NSS = ALUMNO.NSS

   como condición de reunión, en lugar de

   EMPLEADO.NOMBRE = ALUMNO.NOMBRE

   si NSS tiene un índice de agrupación en una o en ambas tablas.

Ajuste de Consultas - Casos

7. **En algunos optimizadores de consultas el orden en el que aparecen las tablas en el FROM puede afectar el procesamiento de la reunión**
   - En esos casos debe cambiarse el orden para que procese primero las tablas con menos datos, y la más grande se use con el índice correspondiente

Ajuste de Consultas - Casos

8. **Algunos optimizadores dan peores tiempos con consultas anidadas que con sus equivalentes no anidadas**

```
SELECT NSS FROM EMPLEADO
WHERE DNO IN (SELECT DNUMERO
              FROM DEPARTAMENTO
              WHERE NSS_JEFE = '3334444')
```

La transformación de subconsultas correlacionadas puede llevar a que se creen tablas temporales

Ajuste de Consultas - Casos

9. **Muchas aplicaciones se basan en vistas que definen los datos de interés para las aplicaciones**
   - A veces estas vistas pueden ser excesivas cuando la consulta puede realizarse directamente sobre la tabla base, en lugar de usar una vista que se ha definido sobre una reunión

Ajuste de Consultas - Casos

10. **Una consulta con varias condiciones OR puede hacer que no se empleen los índices que existen**

    Puede rehacerse expresándose como una unión de consultas, cada una con una condición sobre un atributo que hace que se emplee el índice

    SELECT *NOMBRE, APELLIDO, SALARIO, EDAD*
    FROM *EMPLEADO*
    WHERE EDAD > 45 ***OR*** *SALARIO < 5000*

Ajuste de Consultas - Casos

SELECT *NOMBRE, APELLIDO, SALARIO, EDAD*
FROM *EMPLEADO*
WHERE *EDAD > 45*
UNION
SELECT *NOMBRE, APELLIDO, SALARIO, EDAD*
FROM *EMPLEADO*
WHERE *SALARIO < 5000*

Puede usar los índices definidos sobre **SALARIO** y sobre **EDAD**

Ajuste de Consultas - Casos

11. **Las condiciones WHERE pueden reescribirse de modo que se utilicen índices para varias columnas:**

SELECT *REGION, TIPO_PROD, MES, VENTAS*
FROM *ESTADISTICA_VENTAS*
WHERE *REGION = 3* ***AND*** *((TIPO_PROD BETWEEN 1 AND 3)* ***OR*** *(TIPO_PROD BETWEEN 8 AND 10))*

Puede usar un índice únicamente sobre REGION y debe buscar a través de todas las páginas hoja del índice un emparejamiento con TIPO_PROD.

Ajuste de Consultas - Casos

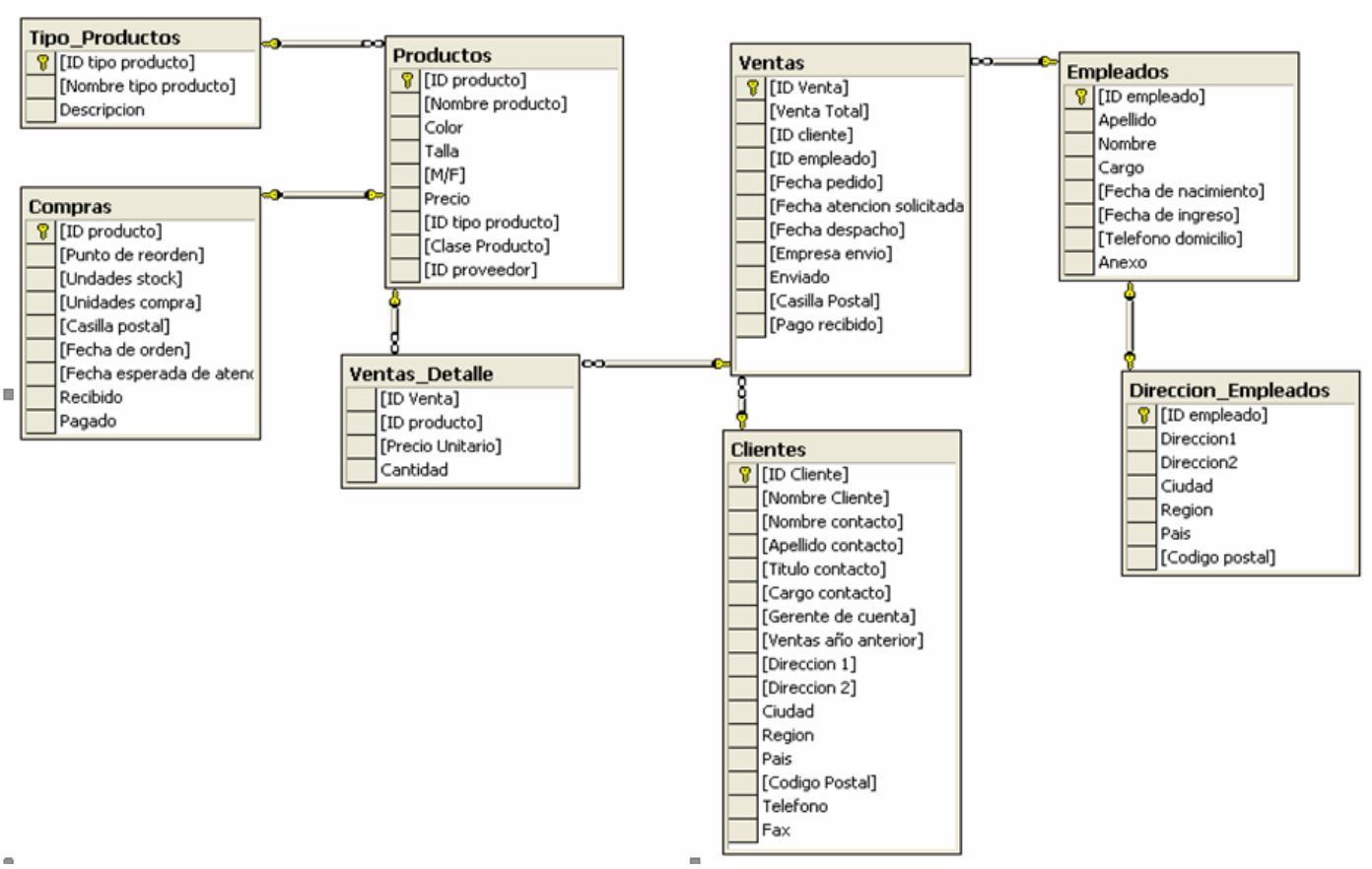
En cambio:

SELECT *REGION, TIPO_PROD, MES, VENTAS*
FROM *ESTADISTICA_VENTAS*
WHERE *(REGION = 3* ***AND*** *(TIPO_PROD BETWEEN 1 AND 3))* ***OR*** *(REGION = 3* ***AND*** *(TIPO_PROD BETWEEN 8 AND 10))*

Puede usar un índice compuesto sobre (REGION, TIPO_PROD) y trabajará mucho más eficientemente.

RESUMEN

- El diseño físico de la base de datos debe considerar la creación de estructuras de acceso adecuadas
- El mantenimiento y optimización de los índices es una labor permanente
- Se puede obtener mejoras dramáticas en el rendimiento de la BD optimizando el código de las consultas SQL, dejando la desnormalización como última alternativa, después aún de la mejora en hardware



FACULTAD de INGENIERÍA de
SISTEMAS E INFORMÁTICA

INDEXACION

Indices

- El índice de una base de datos es una estructura de datos que mejora la velocidad de las operaciones, permitiendo un rápido acceso a los registros de una tabla.
- Al aumentar la velocidad de acceso, se suelen usar sobre aquellos campos sobre los cuales se hagan frecuentes búsquedas.

Indices

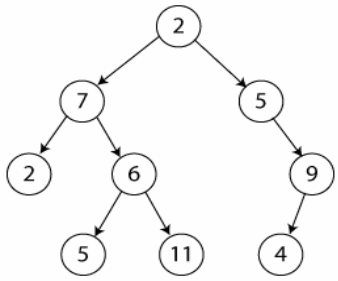
- Los índices guardan:
  - ✓ El elemento que se desea indexar.
  - ✓ La posición en la tabla.
- Para buscar un elemento que esté indexado, sólo hay que buscar en el índice dicho elemento para una vez encontrado, devolver el registro que se encuentre en la posición marcada por el índice.

Indices

- Los índices pueden ser creados usando una o más columnas, proporcionando la base tanto para búsquedas rápidas al azar como de un ordenado acceso a registros eficiente.
- Los índices son construidos sobre árboles B y B+.

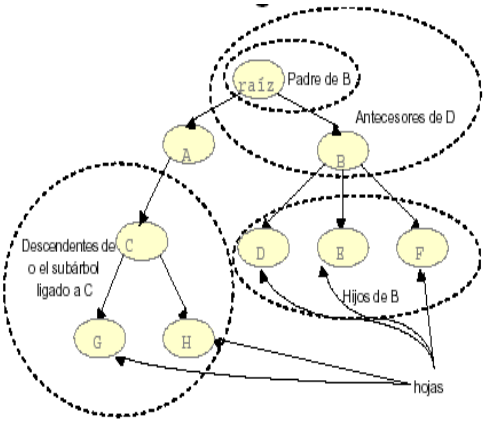
Arbol

- Un árbol es una estructura de datos que imita la forma de un árbol (un conjunto de nodos conectados)
- Un nodo es la unidad sobre la que se construye el árbol y puede tener cero o más nodos hijos conectados a él



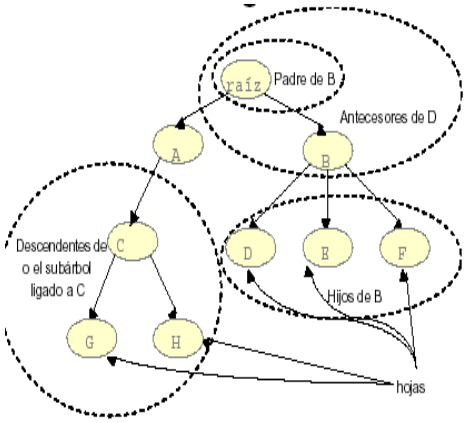
Terminología en Arboles

- Un árbol consta de *nodos* conectados.
- Cada árbol (salvo un *árbol vacío* degenerado) cuenta con un nodo distinguido llamado *raíz*.
- No puede haber rutas circulares en las conexiones de un árbol, de tal forma que sólo puede existir una ruta única desde cada nodo hasta la raíz.



Terminología de Árbol

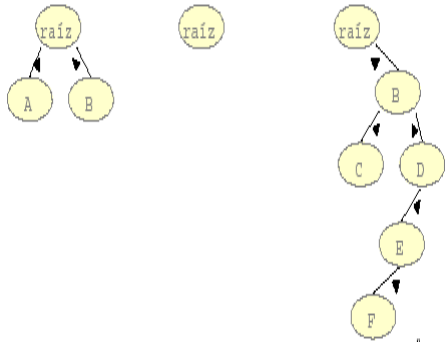
- Todos los nodos conectados a un nodo concreto son *hijos* o bien el *padre* de dicho nodo.
- Si el nodo conectado se encuentra en la única ruta a la raíz, dicho nodo recibe el nombre de padre. Todos los nodos, salvo la raíz, tienen un único padre.
- El resto de nodos conectados a un nodo concreto son los hijos del nodo.



Arbol Binario

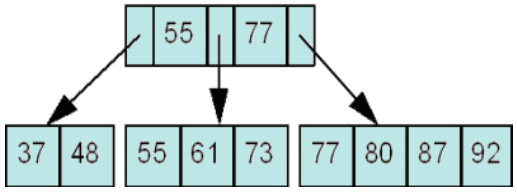
- Un tipo de árbol habitual y de gran utilidad es el llamado *árbol* binario, que permite que un nodo tenga, al menos, dos hijos.
- A continuación se incluye una definición formal del árbol binario que hace hincapié en el carácter recursivo del árbol:

  Un árbol binario es un árbol vacío, o bien un nodo raíz con subárboles formados por árboles binarios a la derecha y a la izquierda.



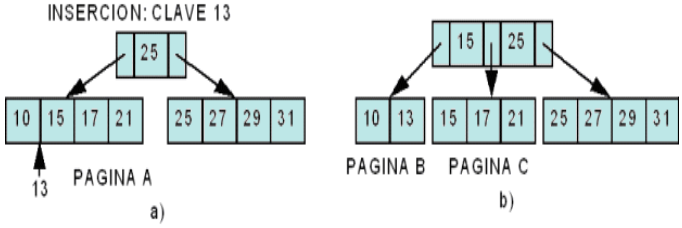
Arbol B+

- Los árboles B+ se han convertido en la técnica mas utilizada para la organización de archivos con índices. La principal característica de estos árboles es que todas las claves se encuentran en las hojas y por lo tanto cualquier camino desde la raíz hasta alguna de las claves tienen la misma longitud



Arbol B+

- **La dificultad se presenta cuando desea insertarse una clave en una pagina que se encuentra llena (m = 2d). En este caso, la pagina afectada se divide en 2, distribuyéndose las m + 1 claves de la siguiente forma:**
- **" las d primeras claves en la pagina de la izquierda y las d + 1 restantes claves en la pagina derecha ". Una copia de la clave del medio sube a la pagina antecesora**



Indices

- **Mejoran el acceso a los datos almacenados**
- **Empleados para clasificación y recuperación de datos**

Tabla Alumnos

#Alumno	Al_Apellido	Al_Nombre	Especialidad
100	Pérez	Juan	Ingeniería de Sistemas
200	López	María	Ingeniería de Sistemas
300	Sánchez	Elena	Ingeniería Electrónica
400	Pérez	Andrés	Ingeniería Electrónica
500	Valdivia	Carlos	Ingeniería de Sistemas
600	Zumaeta	José	Ingeniería Electrónica
700	Montes	Fernando	Ingeniería de Sistemas

Al_Apellido	#Alumno
López	200
Montes	700
Pérez	100, 400
Sánchez	300
Valdivia	500
Zumaeta	600

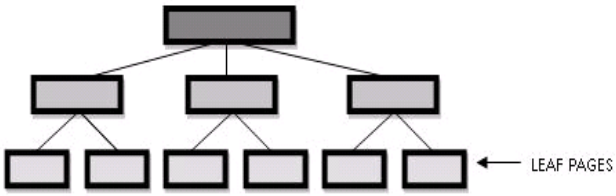
Indice por Apellido

Indice por Especialidad

Especialidad	#Alumno
Ingeniería de Sistemas	100, 200, 500, 700
Ingeniería Electrónica	300, 400, 600

Indices

- El diseño y creación de índices (indexes) es muy importante cuando se quiere mejorar la performance del Servidor de Base de datos



Indices

- Los índices no son considerados como parte del diseño lógico de la base de datos, por eso, pueden se adicionados, removidos y cambiados sin afectar el esquema de la BD

- Tipos de índices:
  - Clustered
  - Non Clustered

Indices - Tipos

Indice Clustered

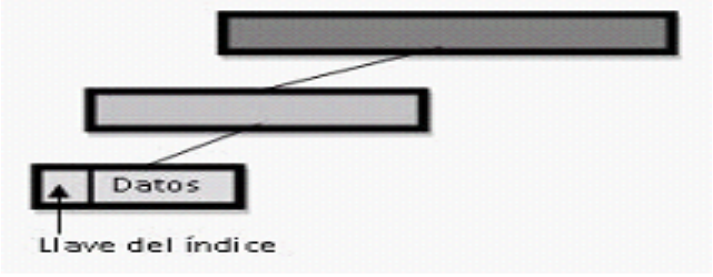
- Es un índice en el cual el orden físico de las filas corresponde al orden de las filas en el índice
- Sólo se puede tener un índice clustered por tabla
- Las operaciones de UPDATE y DELETE son mas rápidas con el uso de estos índices

Indices - Tipos

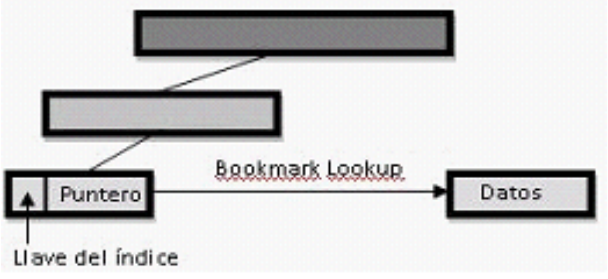
Indices Non-Clustered

- En este índice, se especifica el orden lógico de una tabla para poder acceder a los datos de forma eficiente
- El orden físico de una tabla no corresponde al orden de estos índices
- Se puede definir múltiples nonclustered indexes por tabla

Búsqueda en Indices Clustered



Búsqueda en Indices No Clustered



Creación de Indice

CREATE
[UNIQUE] [CLUSTERED | [NONCLUSTERED]
INDEX índice
ON tabla(campo [ASC|DESC][, campo [ASC|DESC], ...])

Ejemplos: CREATE INDEX

- En el ejemplo siguiente se crea un índice en la tabla Clientes con el campo IdCliente. Dos registros no pueden tener los mismos datos en el campo IdCliente y no se permiten los valores Nulos.

```
CREATE UNIQUE INDEX DNIClien
ON Clientes (DNICliente)
WITH DISALLOW NULL;
```

Ejemplos: CREATE INDEX

- En el ejemplo siguiente se crea un índice en la tabla Clientes con el campo IdCliente. Dos registros no pueden tener los mismos datos en el campo IdCliente y no se permiten los valores Nulos.

```
CREATE CLUSTERED INDEX IdClien
ON Clientes (IdCliente)
WITH DISALLOW NULL;
```

Ejemplos: CREATE INDEX

- En el ejemplo siguiente se crea un índice, que consta de los campos TeléfonoTrabajo y Extensión, en la tabla Empleados:

```
CREATE NONCLUSTERED
 INDEX ÍndiceNuevo
 ON Empleados (TelTrabajo,
  Extensión);
```

Operaciones de Indexación Online

- La opción de indexado online permite modificaciones concurrentes (actualizaciones, borrados e inserciones) en las tablas subyacentes o datos con índices cluster y de cualquier índice asociado durante la ejecución de DDL de indexación
- Por ejemplo, mientras se está reconstruyendo un índice cluster, se puede seguir haciendo actualizaciones a los datos y consultas sobre estos datos

OPTIMIZACION DE CONSULTAS

FACULTAD de INGENIERÍA de
SISTEMAS E INFORMÁTICA

Objetivos

- Revisar las consultas y resolver problemas de rendimiento
  - Revisar un plan de ejecución
  - Identificar consultas mal elaboradas
  - Revisar un plan de ejecución
  - Mantener y optimizar índices

Problema

- **En una institución internacional con muchos miembros a nivel mundial se realizan búsquedas utilizando las primeras letras de sus apellidos**
- **Los datos de la columna Apellidos de la tabla Miembros utilizan mayúsculas y minúsculas combinándolas y para encontrar a los miembros se utilizaba la función UPPER en la consulta por Apellidos**

Consulta Problema

```
SELECT Apellidos
FROM Miembros
WHERE UPPER(Apellidos) LIKE 'MAN%'
```

- Premisa, no se puede modificar el juego de caracteres del servidor que es sensitivo

- ¿ Porque esta consulta es notablemente lenta ?

Una Solución al Problema

```
ALTER TABLE Miembros
ADD ApellidoMayus AS UPPER(Apellidos)
GO
CREATE NONCLUSTERED INDEX
IXMiembrosApellidoMayus
ON Miembros(ApellidoMayus)
GO
```

Una Solución al Problema

```
SELECT Apellidos
FROM   Miembros
WHERE ApellidosMayus LIKE 'MAN%'
```

Otros Motivos de Consultas Lentas

- ✓ Problemas de red
- ✓ Memoria inadecuada en el equipo servidor o falta de memoria disponible para el servidor
- ✓ Falta arreglo de discos (RAID)
- ✓ Falta de estadísticas útiles
- ✓ Falta de índices útiles
- ✓ Falta de particiones útiles

Problemas de Red

- ❑ El problema de rendimiento de las consultas puede estar relacionado con un componente distinto a las mismas consultas
- ❑ Por ejemplo, el problema se puede deber al rendimiento lento de la red
- ❑ También se debe monitorear:
  - La actividad de los discos
  - La actividad de la CPU
  - El uso de la memoria

Estadísticas Utiles

- **Las estadísticas para la optimización de consulta son objetos que contienen información estadística acerca de la distribución de valores en una o más columnas de una tabla**
- **El optimizador de consultas utiliza estas estadísticas para estimar la cardinalidad, o número de filas en el resultado de la consulta**
- **Estas estimaciones de cardinalidad habilitan al optimizador de consultas para crear un plan de consulta de alta calidad**

Estadísticas Utiles

Para actualizar la información sobre la distribución de las claves en una tabla especifica se utiliza UPDATE STATISTICS

```
UPDATE STATISTICS SalesOrderDetail;
GO
UPDATE STATISTICS SalesOrderDetail
AK_SalesOrderDetail_rowguid;
GO
```

Índices Utiles

- Si se utiliza un gran número de índices en una tabla, el rendimiento de las instrucciones INSERT, UPDATE, DELETE y MERGE se verá afectado, ya que todos los índices deben ajustarse adecuadamente a medida que cambian los datos de la tabla
- Evite crear demasiados índices en tablas que se actualizan con mucha frecuencia y mantenga los índices razonables, es decir, defínalos con el menor número de columnas posible

Índices Utiles

- Utilice un número mayor de índices para mejorar el rendimiento de consultas en tablas con pocas necesidades de actualización, pero con grandes volúmenes de datos.
- Un gran número de índices contribuye a mejorar el rendimiento de las consultas que no modifican datos, ya que el optimizador de consultas dispone de más índices entre los que elegir para determinar el método de acceso más rápido

Índices Utiles

- La indexacion de tablas pequeñas puede no ser una solución óptima, porque se tarda más tiempo en realizar la búsqueda de los datos a través del índice que en realizar un simple recorrido de la tabla
- De este modo, es posible que los índices de tablas pequeñas no se utilicen nunca; sin embargo, sigue siendo necesario su mantenimiento a medida que cambian los datos de la tabla

Plan de Ejecución

- **Un plan de ejecución es el conjunto de pasos que tiene que realizar el DBMS para ejecutar una consulta**
- **Un plan de ejecución de una consulta SQL es una definición de:**
  - **La secuencia en la que se tiene acceso a las tablas de origen**
  - **La forma como utilizara los índices de las tablas**

Optimización

- El proceso de selección de un plan de ejecución entre varios planes posibles se conoce como optimización
- El optimizador de consultas es uno de los componentes más importantes de un sistema de base de datos SQL

Identificación de consultas mal elaboradas

- Las herramientas que se pueden utilizar son:
  - *SQL Server Management Studio*
  - *El analizador de SQL Server*
  - El asistente para la optimización del motor de base de datos
  - Las vistas de administración dinámicas

Identificación de consultas mal elaboradas

```
USE Ciclismo
GO
SELECT DISTINCT  *
FROM  Empleados
WHERE SUBSTRING(Apellido,1,1) = 'P'
```

Identificación de consultas mal elaboradas

- ✓ Ingresar al SQL Server Profiler
- ✓ Identificar un trace
- ✓ Seleccionar:
  - ExistingConnection
  - TSQL:SQL:BatchCompleted
- ✓ Escribir un query
- ✓ Revisar la traza

Identificación de consultas mal elaboradas

Realizar la revisión de la traza para:

```
USE Ciclismo
GO
SELECT  *
FROM  Empleados
WHERE
  Apellido  LIKE  'P%'
```

Identificación de consultas mal elaboradas

¿ Analizar a que pueden deberse las diferencias de rendimiento entre ambas consultas ?

Análisis de Planes de Ejecución

- Si se quiere determinar porque tarda tanto en ejecutarse y que esta causando el problema de una consulta se tiene que examinar su plan de ejecución
- Se puede utilizar las siguientes herramientas:
  - Las opciones de la instrucción SET
  - Las clases de eventos del analizador
  - El plan de ejecución grafico de SQL Server Management Studio

Plan de Ejecución Actual y Estimado

- Se puede ver los siguientes planes:
  - El plan de ejecución de un plan para una consulta que fue ejecutada (plan actual)
  - El plan de ejecución de cómo se ejecutara una consulta (el plan estimado)

Plan de Ejecución Actual y Estimado

```
USE AdventureWorks
GO
SELECT *
FROM  HumanResources.Employee
WHERE  SUBSTRING(NationalIDNumber,1,1) = '8'
UNION
SELECT  *
FROM  HumanResources.Employee
WHERE Title like 'P%'
```

Plan de Ejecución Actual y Estimado

```
USE AdventureWorks
GO
SELECT  *
FROM  HumanResources.Employee
WHERE  NationalIDNumber LIKE '8%' OR
          Title LIKE 'P%'
```

MANTENIMIENTO DE BASE DE DATOS

FACULTAD de INGENIERÍA de
SISTEMAS E INFORMÁTICA

Que es el mantenimiento de una BD ?

- El **mantenimiento de una base de datos** es el conjunto de tareas para supervisar el estado de la **base de datos** permitiendo garantizar un mejor rendimiento y disponibilidad de la misma.

Tareas de mantenimiento de una BD

- El **mantenimiento de una base de datos** implica tareas como:
  - Actualizar estadísticas.
  - Reorganización de índices.
  - Monitorear la utilización de la base de datos, servidores y espacio en disco.
  - Planificar estrategias de copia de seguridad y recuperación

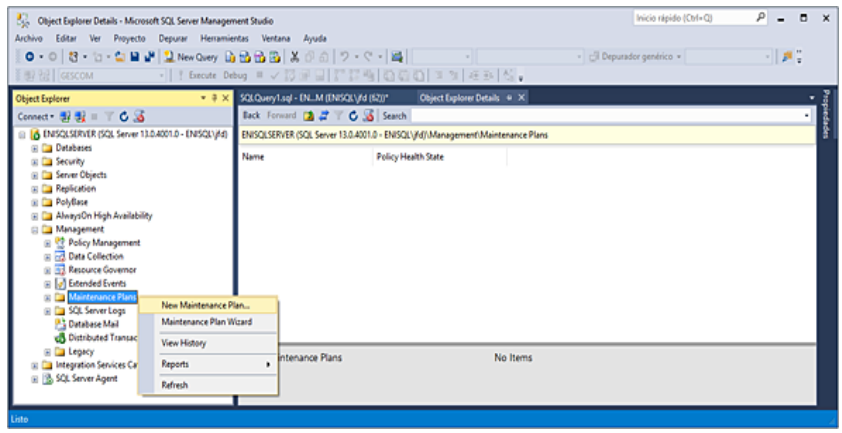
Plan de mantenimiento de una BD

- Para las operaciones rutinarias de mantenimiento de una BD como las copias de seguridad o de mantenimiento de los índices, es posible definir planes de mantenimiento.
- Los planes de mantenimiento se pueden definir con un asistente automatizado o se pueden definir en forma manual (ofrece más opciones de configuración).

Plan de mantenimiento de una BD

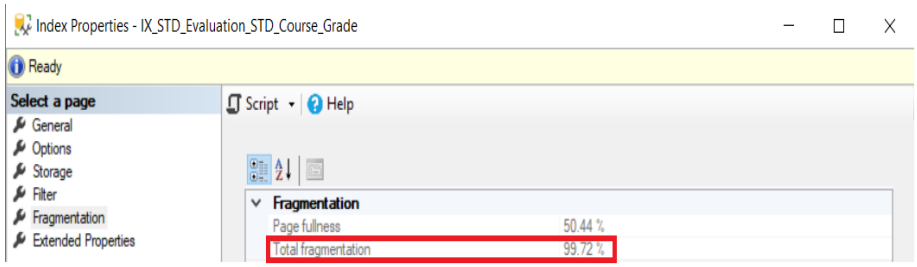
- Los planes de mantenimiento se definen como paquete SSIS (SQL Server Integration Services) y es SQL Server Agent quien se encarga de ejecutar el trabajo que lanza este paquete.
- La definición de un nuevo plan de mantenimiento se puede realizar con el asistente de definición de un nuevo plan, que se ejecuta desde el menú contextual asociado al nodo Management - Maintenance Plans, desde el explorador de objetos.

Plan de mantenimiento de una BD



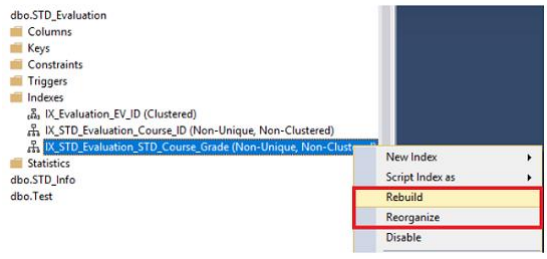
Revisión de fragmentación de Índices

La desfragmentación de un índice asegura que las páginas de índice sean contiguas, brindando formas más rápidas y eficientes para acceder a la información, en vez de leer páginas esparcidas a través de páginas múltiples separadas.



Reorganización de Indices

Un índice puede ser reconstruido o reorganizado usando el SQL Server Management Studio



Reorganización de Indices

ALTER INDEX REBUILD

```
ALTER INDEX [IX_STD_Evaluation_STD_Course_Grade]
 ON [dbo].[STD_Evaluation]
 REBUILD PARTITION = ALL
 WITH (PAD_INDEX = OFF,
 STATISTICS_NORECOMPUTE = OFF,
 SORT_IN_TEMPDB = OFF,
 ONLINE = OFF,
 ALLOW_ROW_LOCKS = ON,
 ALLOW_PAGE_LOCKS = ON)
```

Reorganización de Indices

ALTER INDEX REORGANIZE

```
ALTER INDEX [IX_STD_Evaluation_STD_Course_Grade]
 ON [dbo].[STD_Evaluation]
 REORGANIZE
  WITH ( LOB_COMPACTION = ON )
```

Reorganización de Indices

ALTER INDEX REORGANIZE con la opción ALL

```
ALTER INDEX ALL ON [dbo].[STD_Evaluation]
REORGANIZE ;
```

ALTER INDEX REBUILD con la opción ALL

```
ALTER INDEX ALL ON [dbo].[STD_Evaluation]
REBUILD
 WITH (PAD_INDEX = OFF,
 STATISTICS_NORECOMPUTE = OFF,
 SORT_IN_TEMPDB = OFF,
 ONLINE = OFF,
 ALLOW_ROW_LOCKS = ON,
 ALLOW_PAGE_LOCKS = ON)
```