

Cuaderno de apuntes
Creado por Alejandra milena Orrego Higueta
Estudiante de Ingeniería de Sistemas

Creado en el año 2022, octubre
Bases de datos SQL server
Ultima actualización
Octubre 27 2022

¿Qué es una base de datos?

Es un conjunto o colección de Datos Inter Relacionados, que tienen un propósito o finalidad.

¿Qué conforma una base de datos?

- Archivo de datos
- Log transaccional

Para poder diseñar una base de datos hay diferentes modelos que nos permiten ver de manera abstracto las características que tendrá la Base de datos.

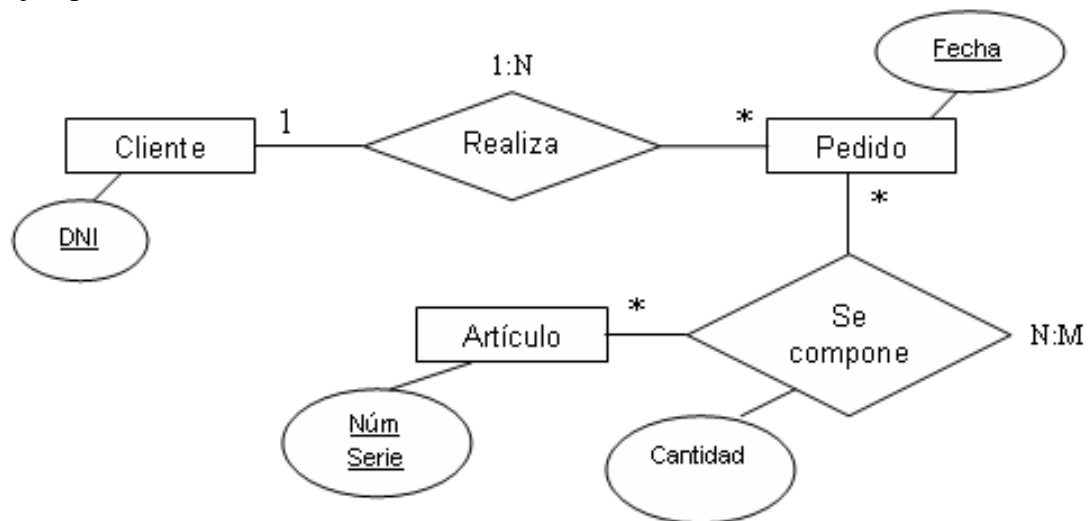
Pasos para la creación de una base de datos

1) ¿Qué son los Modelos de la BD?

Son una representación abstracta de los datos de una organización y las relaciones entre ellos. De esta manera podemos decir que un modelo de datos describe una organización.

Uno de los modelos mas usados es el modelo entidad relación MER, este consta de Entidades, Relaciones, atributos y cardinalidad.

Ejemplo:



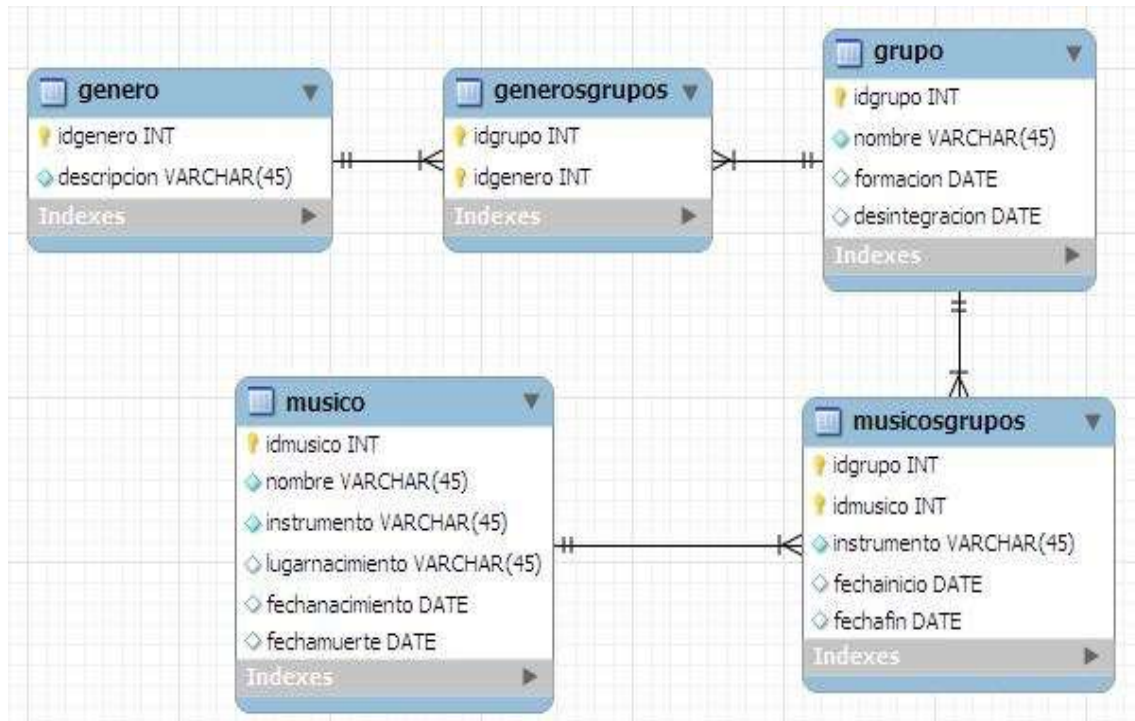
Las entidades siempre deben tener atributos: cliente es una entidad que tiene un atributo DNI.

Dos entidades se pueden relacionar y su cardinalidad puede ser de uno a muchos o de muchos a uno.

Cuando tenemos dos entidades en las que su relación es de muchos a muchos se crea una tabla intermedia que las relaciona, a esto se le denomina tabla de rompimiento.

- 2) Otro modelo que se usa para representar las bases de datos es el modelo relacional, este vendría hacer el segundo paso luego de haber modelado el diagrama entidad relación del ejemplo anterior.

Ejemplo Modelo relacional



Este modelo relacional debe contener sus diferentes claves primarias y foráneas, las tablas asociativas o de rompimiento e integridad referencial.

- 3) Luego de tener modelada la base de datos es importante empezar a identificar si hay datos que se repitan o inconsistencias en las tablas, para esto lo que se hace es normalizar la base de datos con 4 formas normales, estas nos sirven como guía para que nuestra base de datos quede refinada y optimizada para su posterior codificación.

1 FN- Primera Forma Normal, aspectos para tener en cuenta

- Los datos para los atributos deben tomar una realización de un dominio.
- Grupos de atributos repetidos en una tabla
- Las tablas deben tener atributo principal

<u>Cliente</u>				
<u>Id</u>	<u>nombreC</u>	<u>apellidoC</u>	<u>TelefonoC1</u>	<u>telefonoC2</u>

Telefonocliente		
nroR	idC	Telefono
1	90	315
2	90	456

Aquí se observa cómo no es ideal tener atributos del mismo tipo en una sola tabla.

2 FN- Segunda Forma Normal, aspectos para tener en cuenta

- Se debe cumplir 1FN
- Especificar rol para todos y cada uno de los atributos de las tablas.
- Identificar dependencia funcional entre los atributos no clave y la clave primaria.
Debe existir dependencia completa y no parcial.

Libro	Libro	Materia	Materia
codigoL(ISBN)	codigoL(ISBN) pk	codigoM	codigoM pk
tituloL	tituloL s	nombreM	nombreM s
autorL	autorL s	nivelM	nivelM s
editorialL	editorialL s	creditosM	creditosM s
fechaPrest	fechaPrest s	estudiante	estudiante
fechaDev	fechaDev s	calificacion(nota)	calificacion(nota)
clienteL(lector)	clienteL(lector) s	profesor	profesor

Aquí se puede ver lo que se hace es que se identifican esos atributos que no tienen una dependencia funcional con la clave primaria. Como ejemplo podemos ver que en la tabla materia la clave primaria codigoM no tiene dependencia con estudiante, calificación y profesor por lo cual no deben ir en esta tabla sino en tablas que se relacionen más con este tipo de atributos.

3 FN- Tercera Forma Normal, aspectos para tener en cuenta.

- Se debe cumplir 2FN
- Dar solución a las inconsistencias encontradas en 2FN; dependencias funcionales entre atributos no clave. Eliminar los atributos no clave que dependen de otros atributos no clave.

Estudiante		Materia	
codigoE	pk	codigoM	pk
nombreE	s	nombreM	s
apellidoE	s	nivelM	s
identificaciónE	c	creditosM	s
		estudiante	
		calificacion(nota)	
		profesor	
Calificacion		Profesor	
nroRegistro	pk	codigoP	pk
fechaR	s	nombreP	s
codigoE	fk	apellidoP	s
codigoM	fk	identificaciónP	c
codigoP	fk		

REGLA(FORMA NORMAL)	CRITERIO 1		AJUSTE	CRITERIO2		AJUSTE	CRITERIO3		AJUSTE
CUMPLE	SI	NO		SI	NO		SI	NO	
1 FN									
2 FN									
3 FN									
4 FN									
Observaciones generales:									

4) Conocer el lenguaje SQL transaccional para poder codificar la base de datos

Las bases de datos se componen de 4 lenguajes que tienen diferentes propósitos aquí explico cuáles son.

Lenguaje DDL: lenguaje de definición de datos

Este lenguaje se utiliza para la definición de tablas, vistas e índices entre otros en la base de datos.

Comandos:

CREATE para crear objetos

ALTER para modificar la estructura de objetos

DROP para eliminar objetos

TRUNCATE para eliminar todos los registros de una tabla.

COMMENT para agregar comentarios de un objeto al diccionario de datos

RENAME para cambiar el nombre de un objeto

Lenguaje DML: lenguaje de manipulación de datos

Como su nombre lo indica provee comandos para la manipulación de los datos, es decir, podemos seleccionar, insertar, eliminar y actualizar datos.

Comandos:

SELECT para consultar datos.

INSERT Insertar datos.

UPDATE actualizar datos.

DELETE eliminar algunos o varios registros.

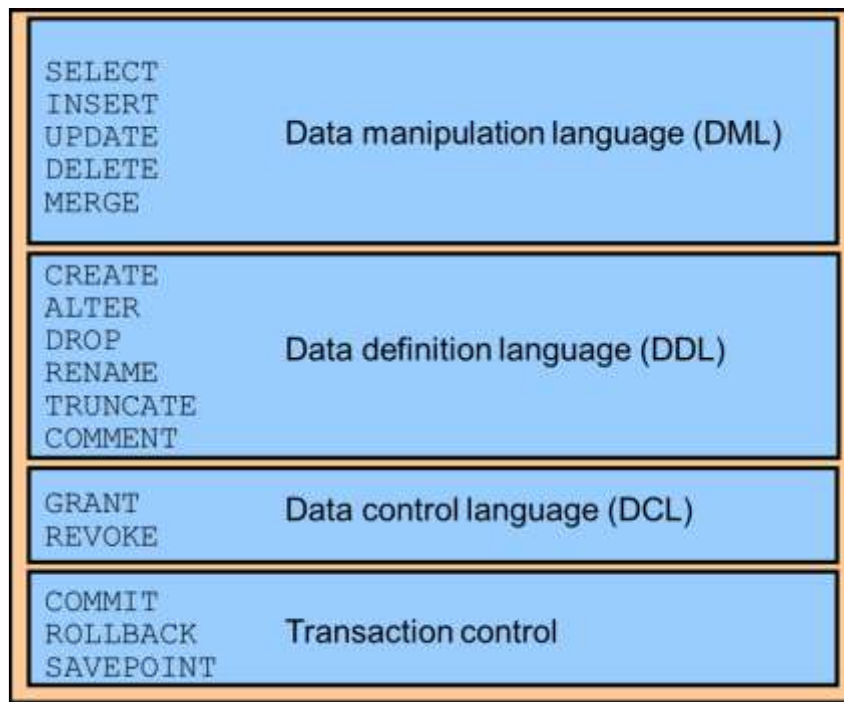
Lenguaje de Control de Datos DCL

Este lenguaje provee comandos para manipular la seguridad de la base de datos, respecto al control de accesos y privilegios entre los usuarios.

Comandos:

GRANT Para otorgar privilegios a un usuario sobre un objeto.

REVOKE Para quitar privilegios dados a un usuario sobre un objeto.



Comandos para crear restricciones

Las siguientes restricciones se usan comúnmente en SQL:

NOT NULL : garantiza que una columna no pueda tener un valor NULL

UNIQUE : garantiza que todos los valores de una columna sean diferentes

PRIMARY KEY : una combinación de NOT NULL y UNIQUE. Identifica de forma única cada fila en una tabla

FOREIGN KEY : evita acciones que destruyan enlaces entre tablas

CHECK : garantiza que los valores de una columna satisfagan una condición específica

DEFAULT : establece un valor predeterminado para una columna si no se especifica ningún valor.

Funciones en SQL server

Funciones que afectan 1 solo registro	
Funciones Matemáticas	Funciones que le permiten manipular datos numéricos de manera más eficaz.
Funciones de Cadena	Funciones que le permiten manipular datos de cadena de manera más eficaz.
Funciones de Fecha	Funciones que le permiten manipular datos de fecha y hora de manera más eficaz.
Funciones de Comparación	COALESCE, DECODE, y NULLIF.
Funciones que afectan a varios registros (grupos)	
Funciones de Agregación	AVG(), COUNT(), MIN(), MAX(), y SUM().

Otros comandos muy útiles son:

El SQL Like es un tipo de operador lógico que se usa para poder determinar si una cadena de caracteres específica coincide con un patrón específico. Se utiliza normalmente en una sentencia Where para buscar un patrón específico de una columna.

- ‘Carácter + %’: muestra patrones que se encuentran en las cadenas de caracteres cuya coincidencia la establece es, el primer carácter especificado en la cadena sin importar el resto de la cadena.
- ‘% + Carácter’: muestra patrones que se encuentran en las cadenas de caracteres cuya coincidencia la establece es, el último carácter especificado en la cadena sin importar el resto de la cadena.
- ‘[a-z]%’: muestra patrones que se encuentran en las cadenas de caracteres cuya coincidencia la establece es, el intervalo que se establece en el corchete; el cual indica que el primer carácter sin importar el resto de la cadena origina el filtro.
- ‘_+ carácter%’: permite especificar la posición o ubicación en una cadena, de un carácter especificado para la búsqueda sin importar el resto de los caracteres que la conforman.

La cláusula GROUP BY es un comando SQL que se usa para agrupar filas que tienen los mismos valores.

La cláusula GROUP BY se utiliza en la instrucción SELECT. Opcionalmente se usa junto con funciones agregadas para producir informes resumidos de la base de datos.

Función having: Se utiliza para incluir condiciones con alguna función de agregado SQL. El comando WHERE no se puede utilizar con funciones de agregado, entonces utilizamos en su lugar, HAVING

```
SELECT edadD, nombD FROM Deportista
group by nombD
HAVING avg(estaturaD) >170
```

Cláusula WITH: Usando la cláusula WITH, puedes reutilizar un bloque SELECT más de una vez en la misma consulta.

Los resultados obtenidos, son guardados temporalmente en memoria. Se puede utilizar para mejorar el rendimiento.


```

1 WITH
2 dept_costs AS (
3     SELECT d.department_name, SUM(e.salary) AS dept_total
4     FROM employees e
5     JOIN departments d ON e.department_id = d.department_id
6     GROUP BY d.department_name
7 ),
8 avg_cost AS (
9     SELECT SUM(dept_total)/COUNT(*) AS dept_avg
10    FROM dept_costs
11 )
12 SELECT *
13 FROM dept_costs
14 WHERE dept_total > (
15     SELECT dept_avg
16     FROM avg_cost
17 )
18 ORDER BY department_name;

```

Subconsultas

Es una declaración SQL que incorpora otra consulta en la cláusula **WHERE**. Consulta Anidada.

```

SELECT edadD, nombD FROM Deportista
WHERE estaturaD > (SELECT avg(EstaturaD) FROM Deportista)

```


Funciones multifila


```

SELECT AVG(salary), MAX(salary),
       MIN(salary), SUM(salary)
FROM   employees
WHERE  job_id LIKE '%REP%';

```

Uniones SQL





Uniendo tablas SQL

Natural joins

- NATURAL JOIN
- USING
- ON

Self-join

Nonequijoins

Outer join

- LEFT OUTER
- RIGHT OUTER
- FULL OUTER

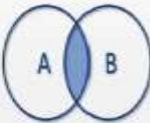
Cross join

```

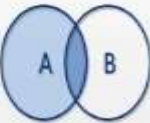
SELECT table1.column, table2.column
FROM table1
[NATURAL JOIN table2] |
[JOIN table2 USING (column_name)] |
[JOIN table2
 ON (table1.column_name = table2.column_name)] |
[LEFT|RIGHT|FULL OUTER JOIN table2
 ON (table1.column_name = table2.column_name)] |
[CROSS JOIN table2];

```

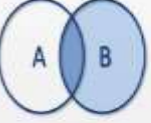
INNER JOIN



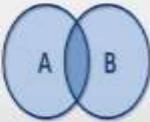
OUTER JOIN



LEFT



RIGHT



FULL

Programación en SQL server

Sintaxis para declarar variable

DECLARE@nombrevariable + <tipo dato><longitud>

Declare @idD int

Declare @nom varchar(50)

Declare @direccion varchar(50)

Declare @fecha date

Declare @promedio numeric(4,2)

Condicional if--else

Ejemplo

declare @promedioEdad int

select @promedioEdad=avg(edadD) from tblDeportista

if(@promedioEdad > 27)

print 'Cumple Criterio'

else

print 'No Cumple Criterio'

Ciclo while

Ejemplo

declare @k int

set @k=1

while @k<10

Begin

set @k=@k+1

print @k

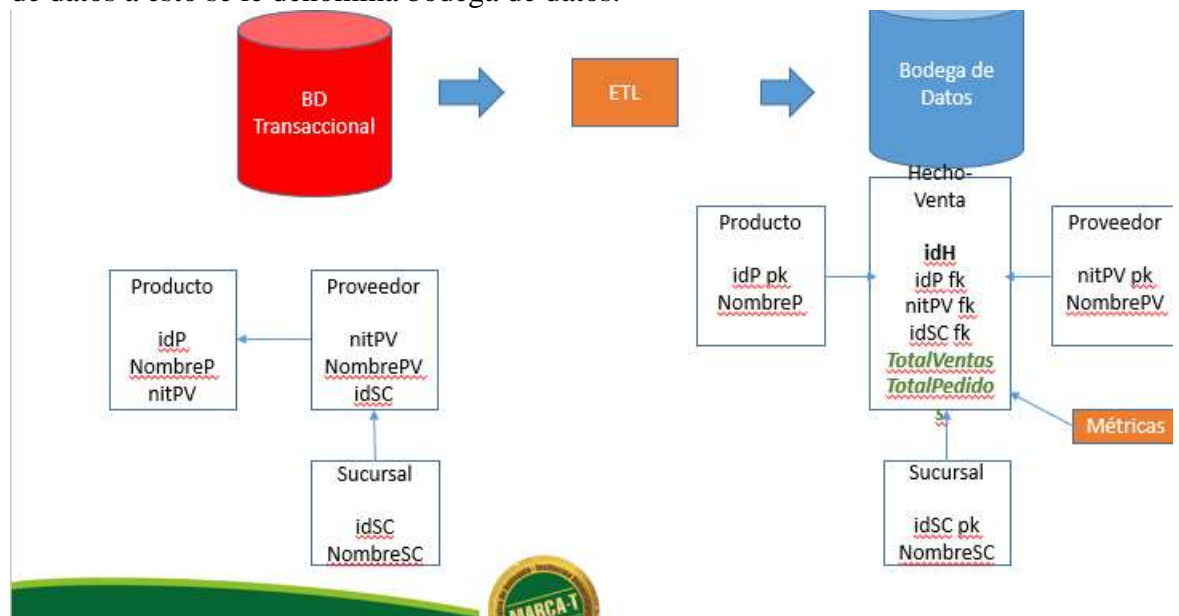
end

Estructura case

La sentencia "case" compara valores y devuelve un resultado.
La sintaxis es la siguiente:

```
SELECT OrderID, Quantity,  
       CASE  
         WHEN Quantity > 30 THEN 'The quantity is greater than  
         WHEN Quantity = 30 THEN 'The quantity is  
         ELSE 'The quantity is under  
         END AS QuantityText  
FROM OrderDetails;
```

- 5) Este paso sirve para mantener un histórico de los datos más relevantes de la base de datos a esto se le denomina bodega de datos.



Bodegas de datos:

- 1) Los datos se generan como un histórico
- 2) Los datos que alimentan la bodega de datos vienen de diferentes fuentes de datos (Excel, csv)

OLTP: procesamiento transaccional---bases de datos Operativas en donde se hacen CRUD

ETL: extracción, transformación y carga

OLAP: procesamiento analítico en línea--- bases de datos multidimensionales, orientadas al análisis (Bodega de datos) select -- insert (NO hay interfaces, algoritmo intermedio -- se denomina carga masiva de información)

Las bodegas permiten: ELT

planear la extracción y transformación

planear prototipos para las aplicaciones Finales

En la bodega se tiene en cuenta una representación de forma en estrella o copo de nieve.

La bodega consta de tres componentes una tabla de hechos, dimensiones y métricas para sacar estadísticas.

Hecho: eventualidad o situación que ocurre en el tiempo (Novedad)
dimensiones: entre más dimensiones más detalles se tienen del hecho

cada **dimensión** tiene una clave primaria y la tabla hecho tiene su clave primaria y las foráneas de las dimensiones-- y los atributos **métricas** o medidas

Métricas - medidas

valores numéricos medibles y evaluables son valores sumados o totalizados a partir de operaciones específicas son medidas relacionadas con los indicadores clave de proceso

PKI=variable que tiene un valor medible o evaluable el cual permite determinar el estado del proceso (ideal, Alerta, crítico).

Ejemplos

Hecho: mortalidad(cardiopatías)
dimensiones--Tipo paciente, causas, Zona

Hecho: deserción académica
dimensiones-- programa, tipo estudiante, causas

Hecho accidentalidad
dimensiones--tipo vehículo, tipo conductor, causas

Hecho ventas
dimensiones cliente, producto, Tienda
unidades precio, unidades, descuento
(el tiempo se puede manejar como un atributo dentro de la tabla de hecho o como una dimensión) el tiempo ayuda a hacer predicciones

Como se ve en la figura de la bodega de datos para pasar datos de una base de datos transaccional a una bodega de datos se debe crear un ETL para programar se debe actualizar primero las dimensiones y luego el paso de información a la tabla de hecho.

- 6) Para lograr un mejor nivel de seguridad en las bases de datos es importante crear **usuarios con diferentes permisos y roles** en donde se especifique que pueden o no hacer esos usuarios.

Modos de interacción: el log transaccional me permite llevar un seguimiento de quien entro a que horas y que hizo, por esto son importantes los usuarios

usuario administrador de toda la base de datos DBA (monitorea crecimiento de la BD) puede crear objetos --- consultar crear usuarios y asignar roles

usuario público: puede hacer consultas

Para crear usuarios y darles permisos se tienen diferentes formas una de ellas es la Estructura en bases de datos donde tendremos

usuario: user-tipo-clave

tipoUsua: admin-public-creador (administrador DDL)

tipousuario rol: que roles desempeña cada tipo de usuario

rol: insertar-eliminar-modificar-crear objeto.