CURSO DESARROLLO DE APLICACIONES MÓVILES ANDROID TRAINEE

Módulo 2.

SQL

Datos

En informática se conoce como dato a cualquier elemento informativo que tenga relevancia para un usuario. Desde su nacimiento, la informática se ha encargado de proporcionar herramientas que faciliten la manipulación de los datos.

Información

Información consiste en un conjunto de datos que poseen un significado, de modo tal que reducen la incertidumbre y aumentan el conocimiento de quien se acerca a contemplarlos. Estos datos se encuentran disponibles para su uso inmediato y sirven para clarificar incertidumbres sobre determinados temas.

Bases de Datos

Una base de datos es una colección organizada de información estructurada, o datos, típicamente almacenados electrónicamente en un sistema de computadora. Una base de datos es usualmente controlada por un sistema de gestión de base de datos (DBMS). En conjunto, los datos y el DBMS, junto con las aplicaciones que están asociados con ellos, se conocen como un sistema de base de datos, que a menudo se reducen a solo base de datos.

Tipos de Campos

- **Alfanuméricos: Contienen cifras y letras. Presentan una longitud limitada (255 caracteres).**
- **Numéricos: Existen de varios tipos, principalmente, enteros y reales.**
- **Booleanos: Poseen dos formas: Verdadero y falso (Sí o No).**
- **Fechas: Almacenan fechas.**
- **Autoincrementables: Son campos numéricos enteros que incrementan en una unidad su valor para cada registro incorporado.**

Tipos de Datos SQL

- **INT O INTEGER: Números enteros. Existen otros tipos de mayor o menor longitud específicos de cada base de datos.**
- **DOUBLE O REAL: Números reales (grandes y con decimales). Permiten almacenar todo tipo de número no entero.**
- **CHAR: Alfanuméricos de longitud fija predefinida.**
- **VARCHAR: Alfanuméricos de longitud variable.**
- **DATE: Fechas, existen múltiples formatos específicos de cada base de datos.**
- **BIT O BOOLEAN: Almacenan un bit de información (verdadero o falso).**

Sentencias SQL

DDL: lenguaje de definición de datos, son sentencias que nos permite definir, alterar, modificar objetos dentro de mi base de datos.

DML: lenguaje de manipulación de datos, permite manipular, consultar, eliminar los datos o registros de las tablas.

Sentencias SQL - DML

SELECT: Recupera datos de la base de datos.

INSERT: Añade nuevas filas de datos a la base de datos.

DELETE: Suprime filas de datos de la base de datos.

UPDATE: Modifica datos existentes en la base de datos.

Sentencias SQL - DDL

CREATE: nos crear distintos elementos de nuestra base de datos.

DROP: nos permite eliminar elementos de la base de datos.

ALTER:Modifica la estructura de una base de datos.

Creación de una Base de datos

Para la creación de una base de datos debemos usar el comando CREATE DATABASE seguido del nombre que tendrá nuestra base de datos, de la siguiente manera.

Por ejemplo:

```
create database nombreBD character set utf8;
```

Creación de una Tabla

Para la creación de un tabla debemos especificar alguno datos tales como: el nombre que le queremos asignar, los nombres de los campos y sus características.

Por ejemplo:

```
create table nombre_t(
    id_tabla int auto_increment,
    nombre varchar(250),
    descripcion text,
    primary key(id_tabla)
);
```

Modificar una Tabla

Podemos agregar nuevos campos a nuestra tabla, de la siguiente manera:

```
alter table nombre_tabla add column campo tipocampo
```

```
alter table libro add column ejemplares integer;
```

Modificar una Tabla

Podemos modificar el tipo de dato de un campo de nuestra tabla, de la siguiente manera:

```
alter table nombre_tabla modify campo tipocampo
```

```
alter table libro modify precioVenta double;
```

Modificar una Tabla - DROP

Podemos eliminar un campo de nuestra tabla, de la siguiente manera:

```
alter table nombre_tabla drop column campo
```

```
alter table libro drop column precioVenta;
```

Modificar una Tabla - DROP

Podemos eliminar una tabla, de la siguiente manera:

```
drop table nombre_tabla
```

```
drop table nombre_t;
```

Ejercicio

Según lo visto en clase crear y realizar lo solicitado:

- **Crear una base de datos con 2 tablas.**
- **Agregar dos nuevos campos a las tablas ya creadas.**
- **Cambiar el tipo de dato a un campo de la tabla.**
- **Eliminar un campo de cada tabla creada.**

Añadir registros

Los registros pueden ser introducidos con sentencias que utilizan la instrucción insert.

La sintaxis es la siguiente:

```
inser into nombre_tabla (campos)
values (valores);
```

```
insert into nombre_t (nombre,descripcion)
values ('Alejandro','Gerente general de Eca');
```

Ejercicio

Crear una base de datos, con las siguientes tablas:

Empleado: idEmpleado, nombre, apellidoP, apellidoM, edad, departamento, cargo, sueldo, fechaIngreso.

Libro: idLibro, nombreLibro, autor, editorial, edicion, añoPublicacion, categoria, ejemplares

Crear una tercera tabla con sus respectivos campos.

Ejercicio

Crear al menos 20 registros en cada tabla de la base datos creada.

Actualizar registros - UPDATE

Podemos actualizar registros de nuestras tablas con la sentencia UPDATE, se expresa de la siguiente manera:

```
update nombre_tabla set campo=valor where campo=valor
```

```
update libro set autorLibro='Andres Cortes' where id_libro='6';
```

Eliminar registros - DELETE

Podemos eliminar registros de nuestras tablas con la sentencia DELETE, se expresa de la siguiente manera:

```
delete from nombre_tabla where campo=valor
```

```
delete from libro where id_libro=8;
```

Sentencia TRUNCATE

Podemos eliminar todos los registros de nuestra tabla (formatear la tabla), se expresa de la siguiente manera:

```
truncate table nombre_tabla
```

```
truncate table tabla_2;
```

Estructura de una sentencia en SQL

COMANDO + CLÁUSULA + OPERADOR + FUNCIÓN

Sentencia básica (COMANDO -CLÁUSULA)

COMANDO + FROM + WHERE + GROUP BY + HAVING + ORDER BY

Ejemplo:

COMANDO + CAMPOS + CLÁUSULA + TABLA + OPERADOR

select nombre, apellido from alumno

select nombre, apellido from alumno where curso=Android

Comando SELECT - Cláusula FROM

Podemos seleccionar registros de nuestras tablas utilizando el comando select y la cláusula from, con la siguiente sintaxis:

```
select campo from nombre_tabla

select campo1, campo2 from nombre_tabla

select * from nombre_tabla
```

Cláusula where

Podemos seleccionar registros de nuestra tabla que cumplan ciertas condiciones con la cláusula where, de la siguiente manera:

```
select * from nombre_tabla where campo=valor
```

```
select * from nombre_t where id_tabla=1;
```

```
select * from empleado where departamento='ventas';
```

Cláusula where con operador lógico AND

Podemos seleccionar registros de nuestra tabla que cumplan ciertas condiciones con la cláusula where y además utilizar el operador lógico Y (and), de la siguiente manera:

```
select * from nombre_tabla where campo1=valor and campo2=valor
```

```
select * from empleado where cargo='secretaria' and departamento='RRHH';
```

Cláusula where con operador lógico OR

Podemos seleccionar registros de nuestra tabla que cumplan ciertas condiciones con la cláusula where y además utilizar el operador lógico O, de la siguiente manera:

```
select * from nombre_tabla where campo1=valor or campo2=valor
```

```
select * from empleado where cargo='secretaria' or cargo='psicologo';
```

```
select * from empleado where cargo='secretaria' or apellidoP='Segura';
```

Cláusula where con IN

Podemos seleccionar registros de nuestra tabla que cumplan ciertas condiciones con la cláusula where y además utilizando la palabra clave IN, que solo tomara en cuenta los valores coinciden con la lista, se expresa de la siguiente manera:

```
select * from nombre_tabla where campo in (lista de valores)
```

```
select * from nombre_t where id_tabla in (1,4);
```

```
select * from empleado where apellidoP in ('Segura','Castillo','Pinto');
```

Ejercicio

Utilizando las tablas libro y empleado cree las consultas para responder lo siguiente:

- **Determinar los datos de los libros con más 100 ejemplares.**
- **Determinar el nombre, apellidoP, apellidoM y cargo de los empleados que tienen un sueldo sobre los $500.000.- y que pertenecen al departamento (ustedes definen el departamento).**
- **Determinar los datos de los libros que son de la primera o segunda edición del autor (definido por ustedes).**
- **Determinar los Autores que tienen algunas de las siguiente ediciones 1ra, 2da o 3ra.**
- **Determinar los empleados que pertenecen al departamento (definido por ustedes).**

Cláusula where con NOT IN

Podemos seleccionar registros de nuestra tabla que cumplan ciertas condiciones con la cláusula where y además utilizando la palabra clave NOT IN, que solo tomara en cuenta los valores que no coinciden con la lista, se expresa de la siguiente manera:

```
select * from nombre_tabla where campo not in (lista de valores)
```

```
select * from nombre_t where id_tabla not in (1, 5);
```

```
select * from empleado where nombre not in ('Felipe','Andrea','Marcelo','Macarena');
```

Cláusula where con <> (Distinto)

Podemos seleccionar registros de nuestra tabla que cumplan ciertas condiciones con la cláusula where y además utilizando el operador <> (distinto), se expresa de la siguiente manera:

```
select * from nombre_tabla where campo <> valor
```

```
select * from persona where nombre <> 'Alejandra';
```

Cláusula between

Podemos seleccionar registros de nuestra tabla que cumplan ciertas condiciones con la cláusula where y between, con lo cual podemos crear un rango de valores (se utiliza para fechas o valores numéricos), se expresa de la siguiente manera:

```
select * from nombre_tabla where campo between valor and valor
```

```
select * from persona where fecha_nacimiento between '1985-01-01' and '2021-12-31';
```

Operador LIKE

Podemos seleccionar registros de nuestra tabla que cumplan ciertas condiciones con la cláusula where y el operador like, se utiliza para comparar una expresión con contenga ciertos caracteres:

```
select * from persona where nombre like 'a%';

select * from persona where nombre like '%a';

select * from persona where nombre like '%a%';
```

Ejercicio

Utilizando las tablas libro y empleado cree las consultas para responder lo siguiente:

- **Determinar los datos de los libros que no pertenecen a dos editoriales (definidas por ustedes).**
- **Determinar los datos nombreLibro, Editorial y ejemplares de los libros cuyos ejemplares se encuentren entre 100 y 330.**
- **Determinar el nombre apellidoP y cargo de los empleados que tienen un sueldo entre $550.000.- y $750.000.-**
- **Determinar nombre, apellidoP, apellidoM, cargo, departamento y edad de los empleados que no pertenecen al departamento (departamento definido por ustedes).**

Cláusula ORDER BY

Esta cláusula nos permite ordenar registros ya sean numéricamente, alfabéticamente o por fechas de manera descendente o ascendente, se expresa de la siguiente manera:

```
select * from nombre_tabla order by campo asc
select * from nombre_tabla order by campo desc
```

```
select * from persona order by nombre desc;
```

```
select * from persona order by nombre asc;
```

Ejercicio

Utilizando las tablas libro y empleado cree las consultas para responder lo siguiente:

- **Mostrar los cargos de los empleados ordenados por cargo.**
- **Mostrar el nombre y cargo de los empleados ordenados por cargo.**
- **Mostrar nombre, cargo y sueldo de los empleados ordenados por sueldos desde el más alto.**

Cláusula GROUP BY

Esta cláusula nos permite ordenar por grupos de datos, se expresa de la siguiente manera:

```
select * from nombre_tabla group by campo
```

```
select * from libro group by categoria;
```

Uso de Funciones

Función COUNT: podemos contar los registros de nuestra tabla, se expresa de la siguiente manera:

```
select count(campo) from nombr_tabla
```

```
select count(id_libro) as cantidad from libro;
```

Uso de Funciones

Función MAX: podemos determinar cual es el mayor valor de un campo en nuestros registros, se expresa de la siguiente manera:

```
select max(campo) from nombre_tabla
```

```
select max(edad) from tabla_2;
```

Uso de Funciones

Función MIN: podemos determinar cual es el mínimo valor de un campo en nuestros registros, se expresa de la siguiente manera:

```
select min(campo) from nombre_tabla
```

```
select min(edad) from tabla_2;
```

Uso de Funciones

Función SUM: podemos determinar la suma de los valores de un campo en nuestras tablas, se expresa de la siguiente manera:

```
select sum(campo) from nombre_tabla
```

```
select sum(sueldo) from empleado;
```

Uso de Funciones

Función AVG: podemos determinar el promedio de los valores de un campo en nuestras tablas, se expresa de la siguiente manera:

```
select avg(campo) from nombre_tabla
```

```
select avg(sueldo) from empleado;
```

Ejercicio

Utilizando las tablas libro y empleado cree las consultas para responder lo siguiente:

- **Determinar el mayor sueldo de cada departamento.**
- **Determinar el menor sueldo de cada departamento.**
- **Determinar cuántos empleados hay por departamento dentro de la empresa.**
- **Determinar cuánto es el gasto total mensual en sueldos dentro de la empresa.**
- **Agrupar por editorial y determinar cuántos libros pertenecen a cada una.**
- **Determinar la cantidad de libros en existencia dentro de la biblioteca (ejemplares).**

Creación de Vistas

Una vista es una muestra de datos de nuestras tablas, pero que no se pueden eliminar ni editar, podemos trabajar con ella como si de una tabla normal se tratase se crea de la siguiente manera:

```
create view nombreVista as
select campos from tabla

create view datosPersonales as
select nombre, apellidoP, apellidoM from empleado;
```

Subconsulta

Una subconsulta es una consulta dentro de otra.

Ejemplos:

Determinar los empleados cuyo sueldo es mayor al promedio.

```
select nombre, cargo, sueldo from empleado where sueldo >(select avg(sueldo) from empleado);
```

Determinar los empleados cuyos sueldo son mayores a todo los sueldo de RRHH.

```
select * from empleado where sueldo > all (select sueldo from empleado where departamento='RRHH');
```

Joins

Se utilizan para combinar filas de dos o más tablas basándose en un campo común entre ellas, devolviendo por tanto datos de diferentes tablas.

```
select campos
from tabla1
inner join tabla2
on tabla1.campo=tabla2.campo;
```

Joins - Ejemplo

Tenemos las siguientes tablas:

idcliente	nombre	apellido	telefono
1	Rodolfo	Tapia	222222
2	Marcelo	Rojas	222322
3	Jorge	Castillo	212121
4	Florencio	Valderrama	242424
NULL	NULL	NULL	NULL

idpedido	idcliente	total
1	2	16000
2	1	18000
3	2	12000
4	1	5000
5	3	6000
6	4	10000
NULL	NULL	NULL

Joins - Ejemplo

Determinar el nombre del cliente y los detalles de sus pedidos

```
select cliente.nombre, pedido.idpedido, pedido.total
from cliente
inner join pedido
on cliente.idcliente=pedido.idcliente;
```

	nombre	idpedido	total
▶	Rodolfo	2	18000
	Rodolfo	4	5000
	Marcelo	1	16000
	Marcelo	3	12000
	Jorge	5	6000
	Florencio	6	10000

Diseño de bases de datos

Nivel conceptual: entender el problema, identificar elementos, los conjuntos de datos y cómo están relacionados entre sí (entidad-relación).

Nivel lógico: es generar la estructura lógica que manipulara la base de datos por medio del DBMS, para esto utilizamos el modelo relacional, que nos permite establecer la estructura de campos y registros.

Entidad - Relación

Entidad: es un objeto real o abstracto del cual se desea información en la base de datos (personas, lugares, etc) que sean de interés para la empres o cliente.

La entidad se representa con un rectángulo y dentro de este lleva el nombre de la entidad.

Ejercicio identificar entidades

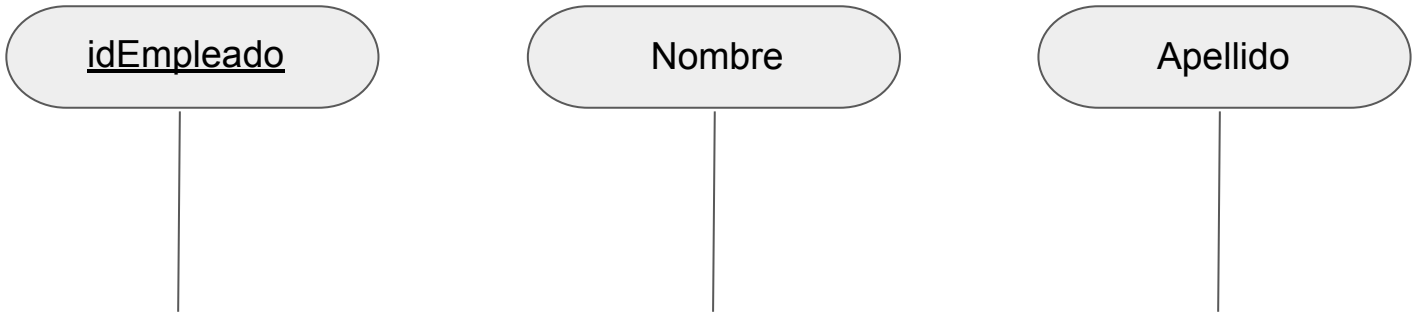
Una tienda que se dedica a la venta de videojuegos, necesita llevar el control de su inventario a través de un sistema que utilice una base de datos. El sistema debe manejar ventas realizadas e imprimir facturas con los datos de los clientes, así como también de deben almacenar las facturas de los proveedores para actualizar las existencias de los productos. El sistema de manera mensual debe generar un reporte sobre las existencias de los productos para realizar un inventario. El sistema debe manejar información global de ventas mensuales y por año, y la existencia general de productos en la tienda.

Entidad - Relación

Relación: es la relación que existe entre las entidades, se representa con un rombo, y dentro de este va un verbo o artículo para denotar la relación que existe.

Entidad - Relación

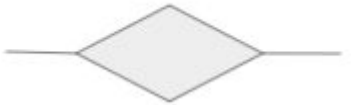
Campos o atributos: son todos los elementos que componen un entidad, se denotan de la siguiente manera, y van unidos a la entidad por líneas.



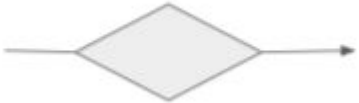
Entidad - Relación

Tipo de correspondencia: se refiere al número máximo de ocurrencias de cada entidad que pueden intervenir en una ocurrencia del vínculo o relación.

1:1 (uno a uno) se simboliza de la siguiente manera:



1:N (uno a muchos) se simboliza de la siguiente manera:



N:M (muchos a muchos) se simboliza de la siguiente manera:

Ejercicio - Realizar modelo entidad - relación

Una unidad educativa desea llevar el registro de sus alumnos a través de un sistema, dicha entidad cuenta con 6 carreras de 10 semestres cada una, cada semestre tiene 5 asignaturas, los profesores pueden ser asignados a una o más asignatura. El sistema debe llevar un registro curricular de los alumnos, los profesores realizan 4 evaluaciones por asignatura.

Transacciones en SQL

Son unidades o secuencias de trabajo realizadas de forma ordenada y separada en una base de datos. Normalmente representan cualquier cambio en la base de datos, y tienen dos objetivos principales:

- **Proporcionar secuencias de trabajo fiables que permitan poder recuperarse fácilmente ante errores y mantener una base de datos consistente incluso frente a fallos del sistema.**
- **Proporcionar aislamiento entre programas accediendo a la vez a la base de datos.**

Transacciones en SQL

Las transacciones siguen cuatro propiedades básicas, bajo el acrónimo ACID:

Atomicidad: si se tiene una operación o transacción en pasos o todos se ejecutan o no se ejecuta ninguna.

Consistencia o integridad: es una propiedad que asegura que se pueda ejecutar todo lo que se puede terminar, para toda transacción, esto permite asegurar que los datos son exactos y consistentes.

Transacciones en SQL

Aislamiento: esta propiedad asegura que una operación no afecta a otra, asegura que la realización de dos transacciones sobre la misma información sean independientes y no generen errores, por lo cual esta operación define como y cuando los cambios se hacen visibles por los demás procesos concurrentes. Esto es importante al momento de elegir un sistemas de gestión de bases de datos.

Durabilidad (persistencia): Esta propiedad asegura que una vez realizada la operación está persista y no se pueda deshacer aunque falle el sistema y que de esta forma los datos sobreviven de alguna manera.