UACJ

Alumno: Ivan Gerardo Lomeli Reyna
Matricula: 204168

Materia: BASES DE DATOS I
Titulo: Tarea 3
Docente: Jorge Isaac Arellano Flores
Fecha: 11/12/2023

Instrucciones:
Conteste claramente cada una de las siguientes preguntas

1.- Defina “lenguajes procedimentales”

Lenguaje de Manipulación de Datos (Data Manipulation Language, DML) es un idioma proporcionado por los sistemas gestores de bases de datos que permite a los usuarios de la misma llevar a cabo las tareas de consulta o modificación de los datos contenidos en las Bases de Datos del Sistema Gestor de Bases de Datos. El lenguaje de manipulación de datos más popular hoy día es SQL, usado para recuperar y manipular datos en una base de datos relacional. Otros ejemplos de DML son los usados por bases de datos IMS/DL1, CODASYL u otras.

Elementos del lenguaje de manipulación de datos
Select, Insert, Delete y Update

2.- Defina “lenguajes no-procedimentales”

Lenguajes de consulta no procedimentales
En los lenguajes no procedimentales el usuario describe la información deseada sin un procedimiento específico para obtener esa información.

CREATE | CREAR
Este comando permite crear objetos de datos, como nuevas bases de datos, tablas, vistas y procedimientos almacenados.

Ejemplo
(crear una tabla)
CREATE TABLE 'CUSTOMERS';

ALTER | MODIFICAR
Este comando permite modificar la estructura de un objeto. Se pueden agregar/quitar campos a una tabla, modificar el tipo de un campo, agregar/quitar índices a una tabla, modificar untrigger, etc.

Ejemplo
(agregar columna a una tabla)
ALTER TABLE 'ALUMNOS' ADD EDAD INT UNSIGNED;

DROP | ELIMINAR
Este comando elimina un objeto de la base de datos. Puede ser una tabla, vista, índice, trigger, función, procedimiento o cualquier objeto que el motor de la base de datos soporte. Se puede combinar con la sentencia ALTER.

Ejemplo
DROP TABLE 'ALUMNOS’;

TRUNCATE | BORRAR TABLA

Este comando trunca todo el contenido de una tabla. La ventaja sobre el comando DROP, es que, si se quiere borrar todo el contenido de la tabla, es mucho más rápido, especialmente si la tabla es muy grande. La desventaja es que TRUNCATE sólo sirve cuando se quiere eliminar absolutamente todos los registros, ya que no se permite la cláusula WHERE. Si bien, en un principio, esta sentencia parecería ser DML (Lenguaje de Manipulación de Datos), es en realidad una DDL, ya que internamente, el comando TRUNCATE borra la tabla y la vuelve a crear y no ejecuta ninguna transacción.

Ejemplo

```
TRUNCATE TABLE 'NOMBRE_TABLA';
```

Para utilizar operaciones de conjuntos debemos cumplir una serie de normas:
*Las consultas a unir deben tener el mismo número campos, y además los campos deben ser del mismo tipo.

3.- ¿Qué son los lenguajes formales de consulta?

Los lenguajes formales de consulta son ampliamente utilizados en el ámbito de la informática y la teoría de la computación para llevar a cabo búsquedas y consultas precisas en conjuntos de datos, especialmente en bases de datos. Estos lenguajes se crean con reglas y estructuras sintácticas bien definidas que permiten a los usuarios expresar consultas de manera formal y precisa. Dos ejemplos comunes de lenguajes formales de consulta son SQL (Structured Query Language) y XPath.

SQL (Structured Query Language): SQL es un lenguaje especializado utilizado para administrar y consultar bases de datos relacionales. Permite a los usuarios realizar diversas operaciones en los datos almacenados en una base de datos, como recuperar información específica, actualizar registros, insertar nuevos datos y eliminar registros. Las consultas SQL se escriben en un formato estructurado y siguen una sintaxis específica, lo que facilita la interacción con bases de datos.

XPath: XPath es un lenguaje utilizado para buscar y seleccionar partes específicas dentro de documentos XML (Extensible Markup Language). XML se utiliza para organizar datos en formato textual, mientras que XPath proporciona una forma precisa de navegar por la jerarquía de elementos y atributos presentes en los documentos XML. Los usuarios pueden utilizar expresiones XPath para especificar qué partes del documento desean recuperar o procesar.

4.- Defina DDL, DCL, DML, y de que se encarga cada uno.

El Lenguaje de Definición de Datos, conocido como DDL, permite crear y modificar la estructura de una base de datos. Las bases de datos son la clave de tener tantas aplicaciones robustas que puedan (en parte) abastecer nuestras necesidades, por ejemplo, en el caso de Facebook o Google.
¿Te imaginas un mundo sin bases de datos? No existirían las grandes empresas tech ni mucho menos Platzi, es por ello que las DB (Database) son muy importantes ya que en ella guardamos datos. Los datos por sí solos no son nada hasta que se contextualizan.
Para realizar operaciones en una base de datos relacional es importante conocer los siguientes conceptos:
DDL, DML, DCL y TCL
¿Qué es DDL? Lenguaje de Definición de Datos
DDL significa Data Definition Language o Lenguaje de Definición de Datos, en español. Este lenguaje permite definir las tareas de las estructuras que almacenarán los datos.
Sentencias de DDL (Data Definition Language)
CREATE: Utilizado para crear nuevas tablas, campos e índices.
ALTER: Utilizado para modificar las tablas agregando campos o cambiando la definición de los campos.
DROP: Empleado para eliminar tablas e índices.
TRUNCATE: Empleado para eliminar todos los registros de una tabla.
COMMENT: Utilizado para agregar comentarios al diccionario de datos.
RENAME: Tal como su nombre lo indica es utilizado para renombrar objetos.
¿Qué es DML? Lenguaje de Manipulación de Datos
DML significa Data Manipulation Language o Lenguaje de Manipulación de Datos, en español. Este lenguaje permite realizar diferentes acciones a los datos que se encuentran en una base de datos.

Permite recuperar, almacenar, modificar, eliminar, insertar y actualizar datos de una base de datos.

Elementos del DML (Data Manipulation Language)
SELECT: Utilizado para consultar registros de la base de datos que satisfagan un criterio determinado.
INSERT: Utilizado para cargar de datos en la base de datos en una única operación.
UPDATE: Utilizado para modificar los valores de los campos y registros especificados
DELETE: Utilizado para eliminar registros de una tabla de una base de datos.
¿Qué es DCL? Lenguaje de Control de Datos
Permite crear roles, permisos e integridad referencial, así como el control al acceso a la base de datos.

Elementos del DCL (Data Control Language)
GRANT: Usado para otorgar privilegios de acceso de usuario a la base de datos.
REVOKE: Utilizado para retirar privilegios de acceso otorgados con el comando GRANT.
¿Qué es TCL? Lenguaje de Control Transaccional
Permite administrar diferentes transacciones que ocurren dentro de una base de datos.
Elementos del TCL**(Transactional Control Language)**
COMMIT: Empleado para guardar el trabajo hecho.
ROLLBACK: Utilizado para deshacer la modificación que hice desde el último COMMIT.

5.- Del álgebra relacional defina: unión, intersección, diferencia, producto cartesiano, select, project, join, divide. De un ejemplo de cada uno de ellos.

Unión (Union):
Definición: La operación de unión combina dos conjuntos de tuplas y devuelve un nuevo conjunto de tuplas que contiene todas las tuplas de ambos conjuntos, sin duplicados.
Ejemplo:
Si tienes dos conjuntos A y B:
A = {1, 2, 3}
B = {3, 4, 5}
La unión de A y B sería {1, 2, 3, 4, 5}.
Intersección (Intersection):

Definición: La operación de intersección devuelve un conjunto de tuplas que están presentes en ambos conjuntos originales.
Ejemplo:
Utilizando los mismos conjuntos A y B del ejemplo anterior:
La intersección de A y B sería {3}.
Diferencia (Difference):

Definición: La operación de diferencia devuelve un conjunto de tuplas que están en un conjunto original pero no en el otro.
Ejemplo:
Con los mismos conjuntos A y B:
La diferencia entre A y B sería {1, 2} (tuplas presentes en A pero no en B).
Producto Cartesiano (Cartesian Product):

Definición: El producto cartesiano combina todas las tuplas de dos conjuntos, generando un nuevo conjunto en el que cada tupla del primer conjunto se combina con cada tupla del segundo conjunto.
Ejemplo:
Si tienes dos conjuntos C y D:
C = {A, B}
D = {1, 2}
El producto cartesiano de C y D sería {(A, 1), (A, 2), (B, 1), (B, 2)}.
Select (Selección):

Definición: La operación de selección filtra las tuplas de un conjunto según una condición específica.

Ejemplo:
Si tienes un conjunto E = {1, 2, 3, 4, 5} y deseas seleccionar solo los números pares, usarías la selección con la condición "número es par", lo que resultaría en {2, 4}.
Project (Proyección):

Definición: La operación de proyección elimina columnas específicas de un conjunto de tuplas, dejando solo las columnas deseadas.
Ejemplo:
Si tienes un conjunto de tuplas con atributos (nombre, edad, ciudad) y deseas proyectar solo el atributo "nombre," obtendrías un nuevo conjunto con solo los nombres.
Join (Unión):

Definición: La operación de unión combina dos conjuntos de tuplas relacionadas mediante un atributo común.
Ejemplo:
Si tienes dos conjuntos de empleados y departamentos, puedes realizar una unión basada en el atributo "departamento" para combinar información de empleados y sus respectivos departamentos.
Divide (División):

Definición: La operación de división se utiliza para encontrar tuplas en un conjunto que cumplan con ciertas condiciones y relacionarse con todas las tuplas en otro conjunto.
Ejemplo:
Imagina dos conjuntos, uno con información de clientes y otro con información de pedidos. La división podría utilizarse para encontrar clientes que han realizado todos los pedidos en la base de datos.

6.- Mencione 3 Lenguajes comerciales de consulta.

SQL (Structured Query Language):
Es uno de los lenguajes de consulta más ampliamente utilizados en la gestión de bases de datos comerciales. Se ha convertido en un estándar de facto para interactuar con bases de datos relacionales. Varias empresas ofrecen sistemas de gestión de bases de datos relacionales (RDBMS) que utilizan SQL como lenguaje principal, como Microsoft SQL Server, Oracle Database y IBM Db2.
PL/SQL (Procedural Language/Structured Query Language):
Es un lenguaje de programación de base de datos desarrollado por Oracle. Se utiliza en combinación con SQL para crear procedimientos almacenados, funciones y desencadenadores dentro del sistema de gestión de bases de datos Oracle Database. Permite a los desarrolladores crear aplicaciones y automatizar tareas dentro de la base de datos.
T-SQL (Transact-SQL):
Es una extensión del lenguaje SQL desarrollada por Microsoft y se utiliza en su sistema de gestión de bases de datos Microsoft SQL Server. Ofrece características adicionales y capacidades de programación para la manipulación de datos y la administración de bases de datos. T-SQL se usa para crear procedimientos almacenados, funciones y otros objetos dentro de SQL Server.
Estos lenguajes comerciales de consulta son ampliamente adoptados en la industria y son esenciales para interactuar con sistemas de gestión de bases de datos relacionales de uso común en entornos empresariales.

7.- Que es SQL y que significan sus iniciales

SQL:
Por sus siglas en inglés significa Lenguaje de Consulta Estructurada (Structured Query Language), es un lenguaje de programación diseñado para actualizar, obtener, y calcular información en bases de datos relacionales.
Es el lenguaje para administrar y consultar información de bases de datos relacionales. Sus siglas significan Structured Query Language (lenguaje de consulta estructurado) y entre sus funciones permite diseñar la estructura de la base de datos, poblar los datos hacer consultas y gestionar roles y permisos. A diferencia de lenguajes de programación de propósito general como JavaScript, Python o Java, SQL es de propósito específico, es decir, solo se usa con bases de datos y no puede crearse una aplicación completa solo con SQL.

Se divide en tres tipos de lenguajes: DDL, DCL Y DML.

DDL (Data Definition Language)
Con este lenguaje definimos la estructura de la base de datos: las tablas, atributos, índices y restricciones.

En este ejemplo agregamos la columna lastname a la tabla students para guardar texto con un límite de 50 caracteres:

ALTER TABLE students ADD COLUMN lastname VARCHAR(50);
DCL (Data Control Language)
Con este lenguaje creamos usuarios y asignamos permisos.

Por ejemplo, aquí otorgamos permiso al rol edteam para poder agregar registros en la tabla students:

GRANT INSERT ON TABLE students TO edteam;
DML (Data Manipulation Language)
Con este lenguaje manipulamos los datos: creamos nuevos registros, los consultamos, actualizamos y eliminamos. A este proceso se le conoce como CRUD (si aun no sabes qué es un CRUD, ingresa aquí).

En estos ejemplos agregamos un registro en la tabla students indicando las columnas y luego el valor de cada una:

INSERT INTO students (firstname) VALUES ('Pepito');

8.-Defina nuevamente la pregunta cuatro, pero ahora solo aplicándolo a SQL DDL (Data Definition Language - Lenguaje de Definición de Datos):

Descripción: DDL se encarga de definir y administrar la estructura y la organización de la base de datos. Se utiliza para crear, modificar y eliminar objetos de la base de datos, como tablas, índices, vistas y restricciones. Los comandos DDL no afectan los datos en sí, sino la forma en que se almacenan y se acceden.
Ejemplos de comandos DDL: CREATE TABLE, ALTER TABLE, DROP TABLE, CREATE INDEX, CREATE VIEW.
DCL (Data Control Language - Lenguaje de Control de Datos):

Descripción: DCL se ocupa de controlar los permisos y privilegios de acceso a la base de datos. Los comandos DCL permiten a los administradores de la base de datos otorgar o revocar permisos a los usuarios para realizar ciertas operaciones en la base de datos. Esto garantiza la seguridad y el control de la información.
Ejemplos de comandos DCL: GRANT, REVOKE.
DML (Data Manipulation Language - Lenguaje de Manipulación de Datos):

Descripción: DML se utiliza para manipular los datos dentro de la base de datos. Estos comandos permiten realizar operaciones de consulta, inserción, actualización y eliminación en los datos almacenados en las tablas. En esencia, DML se encarga de las operaciones de recuperación y modificación de registros en la base de datos.
Ejemplos de comandos DML: SELECT (para consultar datos), INSERT (para agregar nuevos registros), UPDATE (para modificar registros existentes) y DELETE (para eliminar registros).

9.- Dentro de SQL, defina las funciones básicas del DDL y de un ejemplo de cada una de ellas (al menos 7)

CREATE TABLE:

Función: Se utiliza para crear una nueva tabla en la base de datos, especificando los nombres de las columnas y sus tipos de datos.

Ejemplo:

```
CREATE TABLE Empleados (
   ID INT PRIMARY KEY,
   Nombre VARCHAR(50),
   Salario DECIMAL(10, 2)
);
```

ALTER TABLE:

Función: Permite modificar una tabla existente, agregando, modificando o eliminando columnas, restricciones u otras propiedades.

Ejemplo (agregar una columna):

```
ALTER TABLE Empleados
ADD FechaContrato DATE;
```

DROP TABLE:

Función: Se utiliza para eliminar una tabla y todos sus datos de la base de datos de manera permanente.

Ejemplo:

```
DROP TABLE Empleados;
```

CREATE INDEX:

Función: Permite crear índices en una tabla para mejorar el rendimiento de las consultas.

Ejemplo:

```
CREATE INDEX id_nombre ON Empleados (Nombre);
```

CREATE VIEW:

Función: Permite crear una vista, que es una consulta guardada que se puede utilizar como si fuera una tabla.

Ejemplo:

```
CREATE VIEW EmpleadosConSalarioAlto AS
SELECT Nombre, Salario
FROM Empleados
WHERE Salario > 50000;
```

CREATE DATABASE:

Función: Se utiliza para crear una nueva base de datos en el sistema de gestión de bases de datos (DBMS).

Ejemplo:

```
CREATE DATABASE Company;
```

CREATE SCHEMA:

Función: Permite crear un esquema que contiene objetos de la base de datos, como tablas, vistas y procedimientos almacenados.

Ejemplo:

```
CREATE SCHEMA RH;
```

10.- Dentro de SQL, defina las funciones básicas de DML y de un ejemplo de cada una de ellas (al menos 5).

SELECT:

Función: Se utiliza para recuperar datos de una o varias tablas. Es la operación más común en SQL y permite especificar las columnas y condiciones para filtrar los resultados.

Ejemplo:

```
SELECT Nombre, Salario
FROM Empleados
WHERE Departamento = 'Ventas';
```

INSERT INTO:

Función: Permite agregar nuevos registros a una tabla.

Ejemplo:

```
INSERT INTO Empleados (Nombre, Departamento, Salario)
VALUES ('Juan Pérez', 'Marketing', 55000);
```

UPDATE:

Función: Se utiliza para modificar registros existentes en una tabla. Puedes especificar las columnas a actualizar y las condiciones que determinan qué registros se modificarán.

Ejemplo:

```
UPDATE Empleados
SET Salario = 60000
WHERE Nombre = 'Juan Pérez';
```

DELETE:

Función: Permite eliminar registros de una tabla que cumplan con ciertas condiciones.

Ejemplo:

```
DELETE FROM Empleados
WHERE Departamento = 'Ventas';
```

MERGE (o UPSERT):

Función: La operación MERGE permite combinar las funcionalidades de INSERT, UPDATE y DELETE en una sola declaración. Se utiliza para sincronizar datos en una tabla de destino con datos de origen.

Ejemplo:

```
MERGE INTO Empleados AS Target
USING EmpleadosActualizados AS Source
ON Target.ID = Source.ID
WHEN MATCHED THEN
  UPDATE SET Target.Salario = Source.Salario
WHEN NOT MATCHED THEN
  INSERT (ID, Nombre, Salario)
  VALUES (Source.ID, Source.Nombre, Source.Salario);
```

11.- Enliste 15 palabras reservadas de SQL e indique para que sirven.

SELECT:

Se utiliza para recuperar datos de una o varias tablas.

FROM:

Indica la tabla o tablas de las que se extraerán los datos en una consulta SELECT.

WHERE:

Filtra los resultados de una consulta según una condición específica.

INSERT:

Se usa para agregar nuevos registros a una tabla.

UPDATE:

Permite modificar registros existentes en una tabla.

DELETE:

Elimina registros de una tabla que cumplan con ciertas condiciones.

CREATE:

Utilizado para crear objetos de base de datos, como tablas, vistas, índices, etc.

ALTER:

Modifica la estructura de objetos de base de datos existentes, como tablas o vistas.

DROP:

Elimina objetos de base de datos, como tablas o vistas, de manera permanente.

JOIN:

Combina datos de dos o más tablas relacionadas en una consulta SELECT.

GROUP BY:

Agrupa filas de datos en función de los valores de una o más columnas para realizar cálculos de agregación.

HAVING:

Filtra los resultados de una consulta GROUP BY según una condición que involucra cálculos de agregación.

ORDER BY:

Ordena los resultados de una consulta en función de una o más columnas, de manera ascendente o descendente.

INNER JOIN:

Realiza una unión de tablas en la que se recuperan solo las filas que tienen coincidencias en ambas tablas.

LEFT JOIN (o LEFT OUTER JOIN):

Realiza una unión de tablas que incluye todas las filas de la tabla de la izquierda y las filas coincidentes de la tabla de la derecha. Si no hay coincidencias en la tabla de la derecha, se muestran valores NULL.

12.- ¿Cómo se generan las tablas "virtuales"? y como se visualizan.

Generación de una vista:

Para crear una vista en SQL, se utiliza la sentencia CREATE VIEW. La vista se define mediante una consulta SQL que especifica las tablas reales y las columnas que se utilizarán en la vista.

Por ejemplo:

```
CREATE VIEW EmpleadosConSalarioAlto AS
SELECT Nombre, Salario
FROM Empleados
WHERE Salario > 50000;
```

Visualización de una vista:

Las vistas se visualizan de la misma manera que una tabla real en una consulta SQL. Puedes realizar consultas en una vista de la misma manera que lo harías en una tabla.

Por ejemplo:

```
SELECT * FROM EmpleadosConSalarioAlto;
```

13.- Investigar acerca de Normalización, que es y su definición.

La normalización es un proceso en el diseño de bases de datos relacionales que se utiliza para organizar datos de manera eficiente y minimizar la redundancia. El objetivo de la normalización es evitar problemas como la pérdida de datos, la inconsistencia y las actualizaciones redundantes al dividir las tablas en estructuras más pequeñas y relacionadas. Cada forma normal representa un nivel de organización de datos, y existen varias formas normales (generalmente se utilizan hasta la tercera forma normal, 3NF) que una base de datos puede cumplir.

Primera Forma Normal (1NF):

Una tabla está en 1NF si cada campo (columna) contiene un solo valor, y no hay valores repetidos en las filas.

Garantiza que los datos estén atómicos y no haya listas separadas por comas u otras estructuras de datos complejas en una sola celda.

Segunda Forma Normal (2NF):

Una tabla está en 2NF si cumple con 1NF y, además, todos los atributos no clave están completamente dependientes de la clave primaria.

Evita la redundancia de datos al asegurarse de que cada dato se almacene en un solo lugar.

Tercera Forma Normal (3NF):

Una tabla está en 3NF si cumple con 1NF y 2NF y, además, ningún atributo no clave depende de otro atributo no clave.

Ayuda a eliminar la dependencia transitiva de los datos, lo que significa que no hay cadenas de dependencia entre atributos no clave.

Estas son algunas de las formas normales más comunes, pero existen formas normales adicionales, como la Cuarta Forma Normal (4NF) y la Quinta Forma Normal (5NF), que se utilizan en situaciones más específicas.

14.- Mencione para qué sirve la Teoría de la Normalización y cuales son sus objetivos.

La Teoría de la Normalización se utiliza en el diseño de bases de datos relacionales para lograr ciertos objetivos clave relacionados con la organización y la eficiencia de los datos. Los principales objetivos de la normalización son:

Minimizar la redundancia de datos: La normalización busca evitar la duplicación innecesaria de información en la base de datos. Al eliminar la redundancia, se ahorra espacio de almacenamiento y se reduce la posibilidad de inconsistencias y errores en los datos.

Mantener la integridad de los datos: La normalización ayuda a garantizar la integridad de los datos al eliminar anomalías en la actualización, inserción y eliminación de registros. Esto asegura que los datos sean precisos y coherentes en todo el sistema.

Facilitar la modificación y la expansión de la base de datos: Una base de datos normalizada es más flexible y fácil de modificar o ampliar. Cuando se necesita realizar cambios en la estructura de la base de datos, la normalización minimiza el impacto en las otras partes del sistema.

Simplificar las consultas y mejorar el rendimiento: La normalización puede mejorar el rendimiento de las consultas al eliminar datos redundantes y permitir una búsqueda más eficiente de la información. También facilita la escritura de consultas más simples y comprensibles.

Promover la consistencia y la calidad de los datos: La normalización fomenta la consistencia en la forma en que se almacenan y se acceden a los datos. Esto conduce a una mayor calidad de los datos y reduce la probabilidad de errores.

Reducir el espacio de almacenamiento necesario: Al eliminar la duplicación de datos, la normalización permite un uso más eficiente del espacio de almacenamiento, lo que puede ser especialmente importante en bases de datos de gran tamaño.

15.- Defina las 5 formas normales que se usan en esa teoría.

La teoría de la normalización en bases de datos incluye cinco formas normales (1NF, 2NF, 3NF, 4NF y 5NF) que se utilizan para estructurar y organizar los datos en tablas de una base de datos relacional. Aquí te proporciono una breve definición de cada una de estas cinco formas normales:

Primera Forma Normal (1NF):

Una tabla está en 1NF si cada atributo (columna) contiene un solo valor, y no hay conjuntos ni listas de valores repetidos en las filas. En otras palabras, los datos son atómicos y no hay estructuras de datos complejas en una celda.
El objetivo de la 1NF es eliminar la redundancia de datos y asegurarse de que cada atributo contenga información única.
Segunda Forma Normal (2NF):

Una tabla está en 2NF si cumple con la 1NF y si todos los atributos no clave están completamente dependientes de la clave primaria. Esto significa que no debe haber dependencias parciales de los atributos no clave.

La 2NF elimina la redundancia y evita problemas de actualización, asegurando que cada dato esté almacenado en un solo lugar.
Tercera Forma Normal (3NF):

Una tabla está en 3NF si cumple con la 1NF y la 2NF y, además, ningún atributo no clave depende de otro atributo no clave (eliminando las dependencias transitivas).
La 3NF asegura que los datos estén organizados de manera eficiente y que no haya redundancia ni anomalías en la base de datos.
Cuarta Forma Normal (4NF):

La 4NF es una extensión de la 3NF que se utiliza en situaciones más complejas. Se aplica cuando una tabla tiene múltiples atributos multivaluados (es decir, atributos que pueden contener varios valores).
La 4NF se enfoca en eliminar dependencias multivaluadas entre los atributos y garantiza una mayor organización de datos.
Quinta Forma Normal (5NF):

La 5NF es una forma normal avanzada que se aplica a situaciones aún más complejas en las que una tabla tiene dependencias de unión entre atributos. Se utiliza para reducir la redundancia y garantizar que los datos estén altamente organizados.
La 5NF se centra en descomponer tablas complejas en estructuras más simples que minimizan la duplicación de datos.

16.- Que es el álgebra relacional y para que se aplica, que tipo de lenguaje es.

El álgebra relacional es un conjunto de operaciones matemáticas que se utilizan en la teoría de bases de datos relacionales y la gestión de bases de datos. Estas operaciones se aplican a las relaciones (tablas) en bases de datos relacionales y se utilizan para realizar consultas y manipulación de datos. El álgebra relacional se considera un lenguaje de consulta formal.

Características del álgebra relacional:

Basado en conjuntos: El álgebra relacional se basa en la teoría de conjuntos y opera en conjuntos de datos. Las operaciones se aplican a conjuntos de tuplas, y los resultados son conjuntos de tuplas.

Operaciones declarativas: Las operaciones en el álgebra relacional se expresan de manera declarativa, lo que significa que se describen qué datos se desean obtener o modificar, en lugar de como se deben realizar las operaciones.

Manipulación de datos: El álgebra relacional se utiliza para realizar operaciones como selección (filtrado de datos), proyección (selección de columnas específicas), unión, intersección, diferencia y otras operaciones que permiten realizar consultas y modificaciones en la base de datos.

El álgebra relacional se aplica en los siguientes contextos:

Consultas en bases de datos: Se utiliza para realizar consultas a bases de datos relacionales, lo que permite a los usuarios recuperar información específica de las tablas.

Diseño de bases de datos: Ayuda en el proceso de normalización y en la definición de la estructura de una base de datos relacional.

Optimización de consultas: En los sistemas de gestión de bases de datos, se utilizan técnicas basadas en el álgebra relacional para optimizar el rendimiento de las consultas, como la reorganización de consultas y la determinación del plan de ejecución más eficiente.

Referencias:

https://2019basededatos.blogspot.com/p/unidad-6.html
http://cidecame.uaeh.edu.mx/lcc/mapa/PROYECTO/libro14/53__lenguaje_de_manipulacin_de_datos_dml.html
https://platzi.com/tutoriales/50-sql-mysql-2016/1564-que-es-ddl-dml-dcl-y-tcl-integridad-referencial/
https://developer.mozilla.org/es/docs/Glossary/SQL
https://ed.team/blog/que-es-sql