

Plan de mejoramiento

Karen Natalia Devera Roperó

Análisis y desarrollo de sistemas de información, Servicio nacional de aprendizaje

2451627: Diseñar el sistema de acuerdo con los requisitos del cliente

Int: Heidy Lizbet Adarme Romero

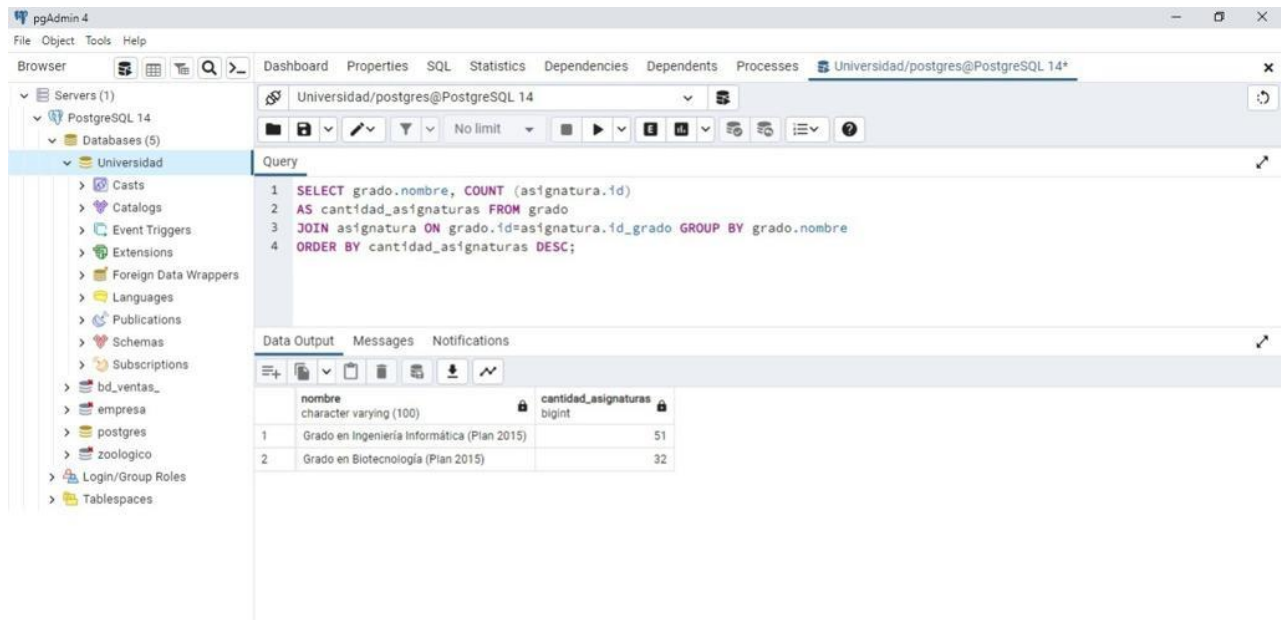
16 de julio del 2023

Introducción

Este plan de mejoramiento lo realizo con el propósito de alcanzar los logros que no pude obtener durante el periodo establecido en la formación, y darle una mayor comprensión y profundizar aún más los conocimientos en el tema de Bases de datos. En este trabajo realizare consultas en la herramienta de Postgres, una comparación de las bases de datos relacionales y no relaciones, Analizare porque son importantes las bases de datos, que papel juegan en el big data y cual motor de bases recomendaría como analista y finalmente un video tutorial especificando como subir la evidencia a GitHub.

Desarrollo

- De acuerdo al modelo lógico realizar las siguientes consultas en la herramienta de PostgreSQL, evidenciar la sintaxis realizada.
 - Obtener el nombre del grado y la cantidad de asignaturas que tiene cada grado, ordenados por la cantidad de asignaturas de forma descendente.



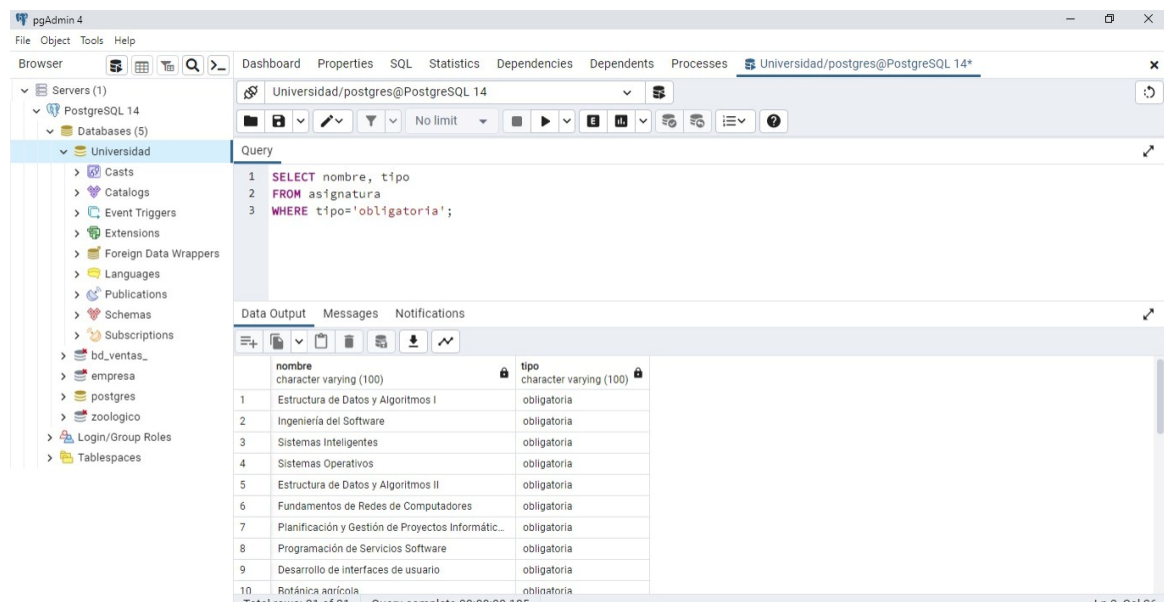
The screenshot shows the pgAdmin 4 interface. The left sidebar displays the database structure, including the 'Universidad' database. The main window shows a SQL query in the 'Query' tab:

```
1 SELECT grado.nombre, COUNT (asignatura.id)
2 AS cantidad_asignaturas FROM grado
3 JOIN asignatura ON grado.id=asignatura.id_grado GROUP BY grado.nombre
4 ORDER BY cantidad_asignaturas DESC;
```

The 'Data Output' tab shows the results of the query:

	nombre character varying (100)	cantidad_asignaturas bigint
1	Grado en Ingeniería Informática (Plan 2015)	51
2	Grado en Biotecnología (Plan 2015)	32

- Mostrar el nombre y el tipo de las asignaturas obligatorias.



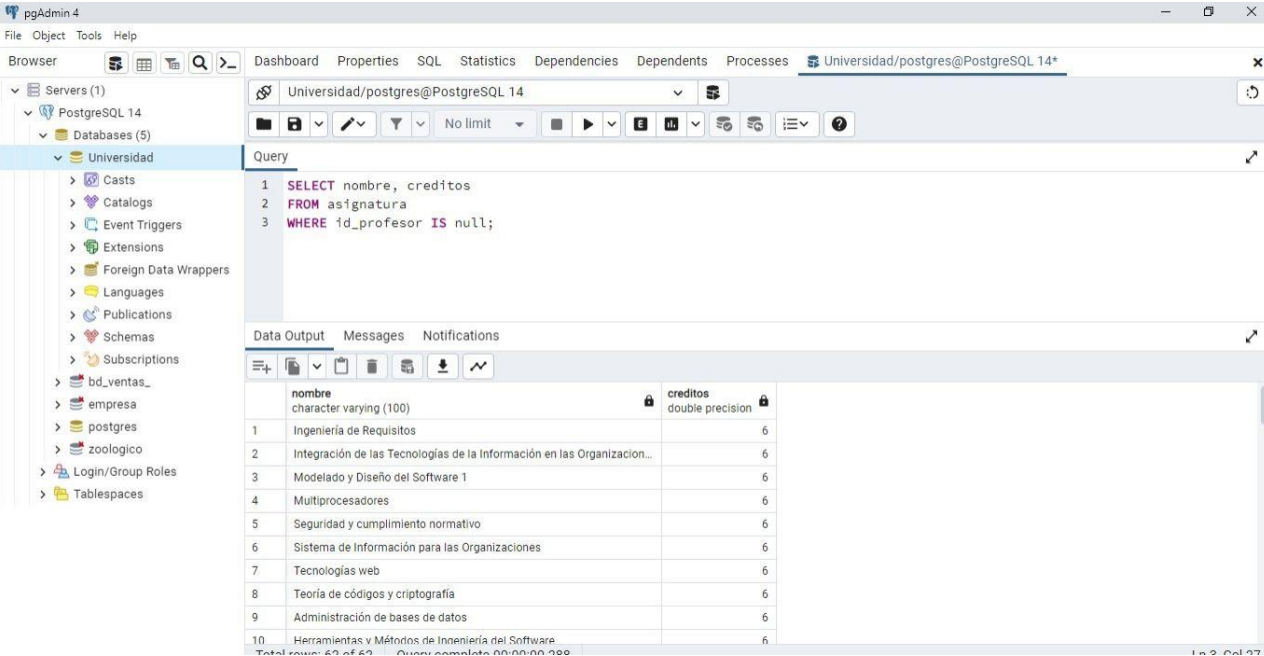
The screenshot shows the pgAdmin 4 interface. The left sidebar displays the database structure, including the 'Universidad' database. The main window shows a SQL query in the 'Query' tab:

```
1 SELECT nombre, tipo
2 FROM asignatura
3 WHERE tipo='obligatoria';
```

The 'Data Output' tab shows the results of the query:

	nombre character varying (100)	tipo character varying (100)
1	Estructura de Datos y Algoritmos I	obligatoria
2	Ingeniería del Software	obligatoria
3	Sistemas Inteligentes	obligatoria
4	Sistemas Operativos	obligatoria
5	Estructura de Datos y Algoritmos II	obligatoria
6	Fundamentos de Redes de Computadores	obligatoria
7	Planificación y Gestión de Proyectos Informáticos	obligatoria
8	Programación de Servicios Software	obligatoria
9	Desarrollo de Interfaces de usuario	obligatoria
10	Botánica agrícola	obligatoria

- Obtener el nombre y el número de créditos de las asignaturas sin profesor asignado.



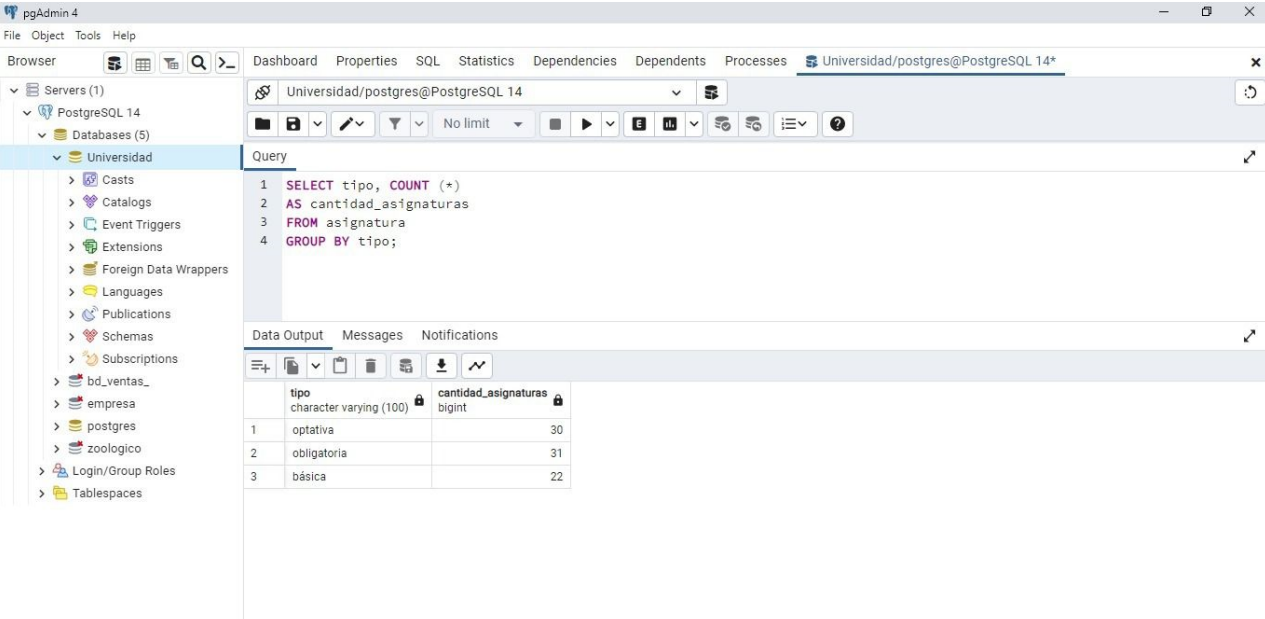
The screenshot shows the pgAdmin 4 interface. The left sidebar displays the database structure, including the 'Universidad' database. The main pane shows a SQL query:

```
1 SELECT nombre, credits
2 FROM asignatura
3 WHERE id_profesor IS null;
```

The 'Data Output' tab shows the results of the query in a table with two columns: 'nombre' (character varying (100)) and 'credits' (double precision). The results are as follows:

	nombre	credits
1	Ingeniería de Requisitos	6
2	Integración de las Tecnologías de la Información en las Organizaciones	6
3	Modelado y Diseño del Software 1	6
4	Multiprocesadores	6
5	Seguridad y cumplimiento normativo	6
6	Sistema de Información para las Organizaciones	6
7	Tecnologías web	6
8	Teoría de códigos y criptografía	6
9	Administración de bases de datos	6
10	Herramientas y Métodos de Ingeniería del Software	6

- Obtener el nombre y la cantidad de asignaturas de cada tipo (básica, obligatoria, optativa).



The screenshot shows the pgAdmin 4 interface. The left sidebar displays the database structure, including the 'Universidad' database. The main pane shows a SQL query:

```
1 SELECT tipo, COUNT (*)
2 AS cantidad_asignaturas
3 FROM asignatura
4 GROUP BY tipo;
```

The 'Data Output' tab shows the results of the query in a table with two columns: 'tipo' (character varying (100)) and 'cantidad_asignaturas' (bigint). The results are as follows:

	tipo	cantidad_asignaturas
1	optativa	30
2	obligatoria	31
3	básica	22

- Mostrar el nombre y la fecha de nacimiento de los profesores ordenados por edad, de mayor a menor.

The screenshot shows the pgAdmin 4 interface. The left sidebar displays the database structure, including the 'Universidad' database. The main window shows a SQL query in the 'Query' tab:

```
1 SELECT nombre, fecha_nacimiento
2 FROM persona
3 WHERE tipo = 'profesor'
4 ORDER BY fecha_nacimiento DESC;
5
```

The 'Data Output' tab shows the results of the query, ordered by birth date in descending order:

	nombre	fecha_nacimiento
1	Antonio	1982-03-18
2	Francesca	1980-10-31
3	Alejandro	1980-03-14
4	Alfredo	1980-02-01
5	Zoe	1979-08-19
6	David	1978-01-19
7	Cristina	1977-08-21
8	Esther	1977-05-19
9	Manolo	1977-01-02
10	Micaela	1976-02-25

Total rows: 12 of 12 Query complete 00:00:00.323 Ln 5, Col 1

- Obtener el nombre del profesor con la cantidad y nombre de asignaturas impartidas por cada profesor.

The screenshot shows the pgAdmin 4 interface. The left sidebar displays the database structure, including the 'Universidad' database. The main window shows a SQL query in the 'Query' tab:

```
1 SELECT persona.nombre, COUNT(*) AS cantidad_asignaturas, (asignatura.nombre) AS nombre_asignaturas FROM profesor
2 INNER JOIN persona ON profesor.id_profesor = persona.id JOIN asignatura ON profesor.id_profesor = asignatura.id_profesor
3 GROUP BY profesor.id_profesor, persona.nombre, asignatura.nombre;
```

The 'Data Output' tab shows the results of the query, grouped by professor:

	nombre	cantidad_asignaturas	nombre_asignaturas
1	Manolo	1	Sistemas Operativos
2	Manolo	1	Ingeniería del Software
3	Zoe	1	Álgebra lineal y matemática discreta
4	Manolo	1	Fundamentos de electrónica
5	Zoe	1	Sistemas Inteligentes
6	Manolo	1	Bases de Datos
7	Zoe	1	Arquitectura de Computadores
8	Manolo	1	Introducción a la programación
9	Manolo	1	Cálculo
10	Zoe	1	Estructura y tecnología de computadores

- Obtener el nombre y la cantidad de alumnos matriculados en cada asignatura.

The screenshot shows the pgAdmin 4 interface. The left sidebar displays the database structure for 'PostgreSQL 14' under the 'Universidad' database. The main pane shows a SQL query:

```
1 SELECT asignatura.nombre AS nombre_asignatura, COUNT(am.id_alumno) AS cantidad_alumnos FROM asignatura
2 LEFT JOIN alumno_se_matricula_asignatura am ON asignatura.id = am.id_asignatura GROUP BY asignatura.id, asignatura.nombre;
```

The 'Data Output' tab shows the results of the query:

	nombre_asignatura character varying (100)	cantidad_alumnos bigint
1	Metabolismo y biosíntesis de biomoléculas	0
2	Matemáticas I	0
3	Teoría de códigos y criptografía	0
4	Virología	0
5	Biología microbiana	0
6	Introducción a la programación	3
7	Modelado y Diseño del Software 2	0
8	Seguridad Informática	0
9	Inmunología	0
10	Rioteología venefal	0

Total rows: 92 of 92. Query complete 00:00:00.101

- Mostrar todos los alumnos que no se han matriculado en ninguna asignatura.

The screenshot shows the pgAdmin 4 interface. The left sidebar displays the database structure for 'PostgreSQL 14' under the 'Universidad' database. The main pane shows a SQL query:

```
1 SELECT persona.nombre AS nombre_alumno
2 FROM persona
3 LEFT JOIN alumno_se_matricula_asignatura am ON persona.id = am.id_alumno
4 WHERE persona.tipo = 'alumno' AND am.id_alumno IS NULL;
```

The 'Data Output' tab shows the results of the query:

	nombre_alumno character varying (25)
1	José
2	Ismael
3	Ramón
4	Daniel
5	Juan
6	Antonio

- Obtener el nombre del profesor con la asignatura que imparte, junto con el nombre de sus estudiantes de esa asignatura y el grado al que pertenecen.

The screenshot shows the pgAdmin 4 interface with a SQL query executed. The query is as follows:

```

1 SELECT persona.nombre AS nombre_profesor, asignatura.nombre AS nombre_asignatura, persona.nombre AS nombre_estudiante, grado
2 FROM profesor JOIN persona ON profesor.id_profesor = persona.id
3 JOIN asignatura ON profesor.id_profesor = asignatura.id_profesor
4 JOIN alumno_se_matricula_asignatura am ON asignatura.id = am.id_asignatura JOIN grado ON asignatura.id_grado = grado.id ;

```

The results are displayed in a table with the following columns: nombre_profesor, nombre_asignatura, nombre_estudiante, and nombre. The table contains 39 rows of data. The status bar indicates: "Total rows: 39 of 39 Query complete 00:00:00.195 Successfully run. Total query runtime: 195 msec. 39 rows affected."

nombre_profesor	nombre_asignatura	nombre_estudiante	nombre
Zoe	Álgebra lineal y matemática discreta	Zoe	Grado en Ingeniería Informática (Plan 2015)
Manolo	Cálculo	Manolo	Grado en Ingeniería Informática (Plan 2015)
Zoe	Física para informática	Zoe	Grado en Ingeniería Informática (Plan 2015)
Zoe	Álgebra lineal y matemática discreta	Zoe	Grado en Ingeniería Informática (Plan 2015)
Manolo	Cálculo	Manolo	Grado en Ingeniería Informática (Plan 2015)
Zoe	Física para informática	Zoe	Grado en Ingeniería Informática (Plan 2015)
Zoe	Álgebra lineal y matemática discreta	Zoe	Grado en Ingeniería Informática (Plan 2015)
Manolo	Cálculo	Manolo	Grado en Ingeniería Informática (Plan 2015)
Zoe	Física para informática	Zoe	Grado en Ingeniería Informática (Plan 2015)
Zoe	Álgebra lineal y matemática discreta	Zoe	Grado en Ingeniería Informática (Plan 2015)

- Obtener todos los alumnos (nombre, apellido1, apellido2) que pertenecen al grado (nombre del grado).

The screenshot shows the pgAdmin 4 interface with a SQL query executed. The query is as follows:

```

1 SELECT DISTINCT persona.nombre, persona.apellido1, persona.apellido2
2 FROM persona JOIN alumno_se_matricula_asignatura am ON persona.id = am.id_alumno
3 JOIN asignatura ON am.id_asignatura = asignatura.id JOIN grado ON asignatura.id_grado = grado.id
4 WHERE grado.nombre = 'Grado en Ingeniería Informática (Plan 2015)';

```

The results are displayed in a table with the following columns: nombre, apellido1, and apellido2. The table contains 6 rows of data. The status bar indicates: "Total rows: 6 of 6 Query complete 00:00:00.168 Ln 3, Col 53".

nombre	apellido1	apellido2
Inma	Lakin	Yundt
Irene	Hernández	Martínez
Juan	Saez	Vega
Pedro	Heller	Pagac
Salvador	Sánchez	Pérez
Sonia	Gea	Ruiz

2. Realice una comparación de los sistemas manejadores de bases de datos (SGBD) NO RELACIONALES existentes en el mercado VS los manejadores de base de datos RELACIONALES.

BASES DE DATOS

SQL

- es relacional, los datos se organizan en tablas, y se pueden establecer relaciones entre las tablas utilizando claves primarias y claves foráneas.
- presenta cada registro un identificador único.
- rendimiento: mas complejo = menos veloz.
- escalabilidad vertical (mejorar potencia del servidor para obtener mejores resultados)
- permite unir tablas con JOIN.

NOSQL

- No relacional, no requiere un esquema fijo y permite almacenar y representar información en formatos diversos, como en documentos, pares clave-valor.
- No necesita identificador en los datos.
- rendimiento: menos complejo = mas veloz.
- escalabilidad horizontal (repartir/distribuir base de datos en diferentes servidores).
- No permite el uso de de JOINS o están muy limitados.

CUADRO COMPARATIVO			
SGBD	DESCRIPCION	VENTAJAS	DESVENTAJAS
mysql	MySQL es un sistema de bases de datos de Oracle que se utiliza en todo el mundo para gestionar bases de datos.	<ul style="list-style-type: none"> • Instalación sencilla • buena integración con php • posee conectividad segura 	<ul style="list-style-type: none"> • Cuando se debe modificar la estructura de Base de datos puede existir ligeros fallos.
postgresql	es un sistema de gestión de datos de base relacional de código abierto.	<ul style="list-style-type: none"> • Gran escalabilidad • instalación ilimitada y segura 	<ul style="list-style-type: none"> • complejidad • menor popularidad
oracle database	es un sistema de gestión de datos de base relacional de código abierto.	<ul style="list-style-type: none"> • fácil entendimiento • potencial de desempeño 	<ul style="list-style-type: none"> • precio • falta de información sobre su uso
Mongodb	es una base de datos NoSQL orientada a documentos que apareció a mediados de la década de 2000.	<ul style="list-style-type: none"> • tiene una gran documentación. • es un complemento para javascript. 	<ul style="list-style-type: none"> • no tiene joins para consultas. • No es una base de datos adecuada para aplicaciones con transacciones complejas.
Cassandra	Cassandra se define como una base de datos NoSQL distribuida y masivamente escalable	<ul style="list-style-type: none"> • alta disponibilidad. • tolerancia a particiones y escalado. 	<ul style="list-style-type: none"> • alta disponibilidad. • alto costo de mantenimiento.
Neo4j	Neo4j es una BD orientada a grafos que nos permite un alto rendimiento, escalabilidad, persistencia de datos, seguridad.	<ul style="list-style-type: none"> • pioneros en la revolucion de los grafos. • amplia comunidad de usuarios. 	<ul style="list-style-type: none"> • modelos de datos no estandarizado. • Replicación de grafos no de subgrafos.

3. ¿Qué importancia tiene las bases de datos relacionales en la actualidad?

Las bases de datos relacionales son de gran importancia porque permiten organizar y gestionar grandes cantidades de información de manera eficiente, establecer relaciones entre los datos y garantizar la integridad de la información almacenada. Son las más usadas en el mundo, todos las hemos utilizado ya sea en el ámbito académico o empresarial. Son altamente compatibles para el uso de análisis de datos, un ejemplo claro es PostgreSQL es una herramienta potente para este tipo de situaciones.

4. ¿Qué papel juegan las bases de datos relacionales en el Big data?

Juegan un papel importante ya que el Big Data maneja conjuntos de datos masivos y complejos, ahí es donde entran las bases de datos relacionales para proporcionar una estructura clara y organización que facilita la gestión de esos datos mediante tablas, filas, columnas. Además, podemos utilizar el lenguaje SQL que permite hacer consultas específicas para obtener respuestas rápidas. Otra ventaja es que ayuda en la integración de datos, si tenemos los datos dispersos en diferentes lugares, existe la posibilidad de que haya información duplicada, esto significa que puede haber información repetida en archivos o bases de datos, entonces al tener todos los datos en un solo lugar, se vuelve más fácil analizarlos y extraer información valiosa.

5. ¿Qué motor de bases recomendaría? MySQL o PostgreSQL?

Recomendaría MySQL porque es la más demandada, Tiene una gran comunidad de usuarios y desarrolladores, lo que significa que encontraremos muchos recursos, tutoriales y soluciones a problemas comunes. Es fácil de usar, no se requiere un conocimiento profundo de bases de datos, tenemos que aprender unos pocos comandos básicos de SQL para crear tablas, insertar datos, realizar consultas. Es gratuito, Al ser de código abierto no se requiere pagar una licencia por utilizarlo en los proyectos. Tiene buen rendimiento en términos de velocidad y una gran capacidad para procesar consultas y transacciones de manera rápida.

6. Realice un video modo tutorial especificando paso a paso de los comandos para subir la evidencia al repositorio "act_pedagogica_2451627" de GITHUB.

Conclusión

Luego de realizar el presente trabajo se llega a la conclusión que es de suma importancia aprobar la siguiente competencia: **Diseñar el sistema de acuerdo con los requisitos del cliente.** específicamente en el programa Bases de datos para poder seguir con el ciclo de formación en el Sena. Este trabajo me ayudo a reforzar mis conocimientos en bases de datos Sql y Nosql , y hacer consultas en la herramienta postgres.