1 Describir el enfoque que tomarías para identificar el problema específico en las consultas (e.g., herramientas, métricas, o técnicas para análisis de rendimiento).

Existen varias herramientas que nos permiten visualizar de mejor manera los detalles de las diversas consultas. Sin embargo, dichas herramientas tienden construidas para funcionar específicamente con algún lenguaje de consultas. Es por eso que considero más oportuno utilizar las cláusulas de EXPLAIN ANALYZE que existe en la gran mayoría de las bases de datos relacionales. Si nosotros agregamos esta cláusula al principio de nuestra consulta, lo que se nos retornará es un análisis breve de la consulta seleccionada. Este análisis nos dirá la altura y anchura de los datos analizados (cantidad de filas y cantidad de columnas respectivamente) y también nos dirá el tiempo de planeación y ejecución de nuestra consulta. Lo que termina siendo más importante es que si nuestra consulta contiene algún JOIN o alguna sub-consulta, EXPLAIN ANALYZE nos dará el detalle de cuánto tiempo le tomó hacer cada operación. Esto entonces nos dará la oportunidad de entender de mejor manera cuales son las operaciones que tenemos que optimizar/editar para evitar que nuestras consultas tomen más tiempo del necesario.

2. Proponer una estrategia general para mejorar el rendimiento en este caso, explicando los pasos y cambios necesarios en la arquitectura o las consultas.

Indexing

Siempre que se habla de optimización de base de datos, se hace mención a cómo hay que optimizar tanto el uso de espacio como la velocidad de consulta. Sin embargo, conforme va creciendo la cantidad de filas de una tabla, disminuyen la cantidad de métodos que se pueden aplicar para optimizar la velocidad de consulta. Es por esto que el Indexing termina siendo un método muy popular de optimización. Lo que hace este método es mejorar el tiempo de consulta a coste de espacio en la base de datos. Al crear una especie de “hash-table”, le permitimos a nuestras consultas saber de manera inmediata la ubicación de ciertas filas sin tener que leer tablas enteras. El único problema que puede presentar este método es que incrementa las acciones en cascada. Escribir, borrar y actualizar datos ahora tomará un tiempo

más elevado debido a también tener que ejecutar la acción de indexado. Es por esto que también es importante darle prioridad a columnas que son frecuentemente utilizadas en consultas y que también son frecuentemente utilizadas como referencia desde otras columnas.

Evitar el uso de SELECT *

Durante varios proyectos en los que he trabajado previamente, he tenido la oportunidad de desarrollar varios endpoints. Al analizar los endpoints previamente desarrollados en el proyecto para darle continuidad al formato y estructura utilizados, me he dado cuenta que en varias consultas se utiliza SELECT *. Esto primordialmente ocurría porque al momento del desarrollo de la consulta se utilizaban todos los valores de la tabla, pero con el paso del tiempo se fueron agregando nuevas columnas. Esto no niega que existieron casos en los que simplemente se agregó un SELECT * sin necesitar todos los valores de la columna. En ambos casos hay que asegurar que las columnas seleccionadas sean únicamente las utilizadas en la aplicación.

Paginación

El tema de la paginación es uno de los más hablados al optimizar consultas en una base de datos. Esto es un proceso en el que estaremos limitando la cantidad de filas devueltas por nuestras consultas para de la misma manera limitar el tiempo que le tomará a nuestra consulta obtener dichas filas. Para este proceso estaremos haciendo uso de las cláusulas de LIMIT y OFFSET en nuestras consultas de datos. Los valores que estaremos utilizando para LIMIT y OFFSET los obtendremos de nuestra aplicación principal. En dicha aplicación estaremos manteniendo registro de cuantas filas ya hemos obtenido para utilizar ese valor como OFFSET. En cuanto al valor de LIMIT, podemos dejar un valor fijo o inclusive darle al usuario unas opciones de cuantas filas le gustaría ver por página.

Normalizacion

El proceso de normalización es uno que no siempre es aplicado al momento del diseño de nuevas tablas o el mantenimiento de estas. Esto ha llevado a que muchas tablas terminen teniendo columnas que no le deberían pertenecer en un primer lugar. En casos de tablas que están teniendo una cantidad alta de filas, estamos agregando una anchura que hará consultas más lentas sobre todo si es una que utiliza SELECT *. De la misma manera, hay que evitar

normalizar de manera excesiva nuestras tablas ya que esto llevaría a que nuestras consultas tengan que cada vez unir más tablas de manera innecesaria. Es por esto que es muy importante tomarse el tiempo para normalizar las tablas de la base de datos de manera apropiada para evitar tener que hacer refactorización innecesaria en un futuro.

3 Justificar por qué tu propuesta sería efectiva y cómo mejoraría los tiempos de respuesta en el sistema.

El análisis de consultas y los diversos métodos de optimización mencionados anteriormente, por más generales o superficiales que puedan llegar a ser, tienden a ser de los más populares en cuanto al desarrollo de bases de datos debido a la sinergia que tienen el uno con el otro. Cada uno de estos métodos termina siendo una aproximación diferente a la problemática. Ya sea cambiando el balance entre tiempo de consulta y espacio utilizado, elevando la complejidad de relaciones o simplemente limitando manualmente las filas obtenidas, cada parte del proceso de una consulta termina siendo desestructurado para ser optimizado. Aunque no exista una manera de obtener la mejora de rendimiento físico desde un punto de vista puramente teórico, podemos asumir que estos metodos seran como minimo efectivos debido a que ampliamente conocidos como buenas prácticas en cuanto al diseño y construcción de bases de datos y sus respectivas consultas, llegando a inclusive ser un requerimiento del desarrollo en varias ocasiones.