

**UNIVERSIDAD DE LAS CIENCIAS INFORMÁTICAS**

**FACULTAD 4**

Título:

Yet Another TE - SQL Queries

**Tarea Extraclase de SBDII**

Autores: Joaquin Rivas Sánchez

Tutor: Luis Manuel Valera Perez

**La Habana, febrero 2024**

**“Año 65 de la Revolución”**

**Resumen**

El presente trabajo se centra en el análisis y comparación de estrategias de optimización de consultas en SQL, específicamente en el contexto de un caso de estudio basado en la Biblioteca Nacional. Se exploran situaciones reales donde se aplican funciones de SQL como FAV, WITH, e índices para mejorar la eficiencia en la gestión de bases de datos. Se presentan cinco problemáticas del caso de estudio, cada una con su implementación en SQL básico, FAV y WITH, junto con la medición de tiempos de ejecución. Se aborda la importancia de funciones, triggers y cursores en la implementación de consultas para garantizar la integridad de los datos. Este estudio proporciona una visión detallada de las estrategias de optimización de bases de datos, así como las funcionalidades en un entorno práctico y real, destacando la relevancia de utilizar herramientas avanzadas de SQL para mejorar el rendimiento y la eficacia en el manejo de grandes volúmenes de datos en entornos como el de una biblioteca nacional.

**Palabras clave:** Optimización de consultas, SQL, Biblioteca Nacional, Funciones avanzadas, Eficiencia en bases de datos.

**Introducción**

El presente trabajo aborda la optimización de bases de datos, centrándose en el análisis y comparación de estrategias para mejorar el rendimiento de consultas en SQL. En un entorno cada vez más orientado a la eficiencia y velocidad de acceso a la información, la optimización de bases de datos se convierte en un aspecto crucial para garantizar un adecuado funcionamiento de los sistemas de gestión de datos.

En este contexto, se exploran diferentes enfoques como el uso de SQL básico, FAV (window functions) y la cláusula WITH (common table expressions), así como la importancia de los índices en la mejora del rendimiento de consultas. Para ilustrar la aplicación práctica de estas estrategias, se plantea un caso de estudio basado en la Biblioteca Nacional, donde se evalúan los tiempos de ejecución de cada enfoque para determinar cuál ofrece la opción más eficiente en términos de optimización.

Se introducen otras herramientas como triggers, funciones y cursores. Los triggers son objetos de base de datos que se activan automáticamente en respuesta a ciertos eventos en una tabla, como inserciones, actualizaciones o eliminaciones de datos. Se utilizan para garantizar la integridad de los datos, aplicar reglas de negocio complejas o realizar acciones automáticas en cascada. Por ejemplo, un trigger puede ser empleado para auditar cambios en ciertas tablas, validar datos antes de su inserción o actualizar información en otras tablas relacionadas.

Las funciones nos permiten encapsular lógica de negocio y operaciones complejas que pueden ser reutilizadas en múltiples consultas. Estas funciones pueden aceptar parámetros de entrada y devolver resultados, facilitando la modularización del código y la simplificación de consultas repetitivas. Por ejemplo, una función puede calcular un valor específico, formatear datos o realizar operaciones matemáticas complejas.

Los cursores son estructuras que permiten recorrer filas de un conjunto de resultados de una consulta de manera secuencial. Aunque su uso puede impactar en el rendimiento debido a su naturaleza de procesamiento fila por fila, los cursores son útiles en situaciones donde se requiere un procesamiento detallado y controlado de los datos, como en operaciones de actualización condicional, manipulación de datos en lotes o procesamiento de datos en tiempo real.

**Capítulo I. Estado del arte**

**Introducción**

En el contexto del presente trabajo, el capítulo del estado del arte se erige como un pilar fundamental que sustenta esta investigación. Este capítulo no solo se limita a recopilar información existente, sino que también cumple la función de contextualizar el problema, establecer las bases teóricas y tecnológicas necesarias para comprender la problemática abordada y justificar la relevancia de la investigación.

El propósito de esta sección es proporcionar al lector una visión panorámica de las tendencias, avances y desafíos actuales en el campo de estudio. A través de un análisis exhaustivo de la literatura especializada, se pretende identificar el estado actual del conocimiento, las brechas existentes y las oportunidades de mejora que motivan la presente investigación.

Además, la introducción al estado del arte busca establecer un marco conceptual sólido que sirva de fundamento para la posterior discusión de la problemática, el análisis del modelo de datos, la caracterización de las tablas y la presentación de las soluciones propuestas. Se enfatiza la importancia de comprender el contexto en el que se desarrolla la investigación, así como la necesidad de situar el estudio dentro de un marco teórico y práctico que permita abordar de manera efectiva los desafíos planteados.

En este sentido, la introducción al estado del arte no solo presenta una visión general de los temas relevantes a tratar en el capítulo, sino que también establece las bases para una exploración profunda y rigurosa de los aspectos clave que guiarán el desarrollo de la investigación. A través de un enfoque crítico y reflexivo, se busca no solo informar al lector, sino también involucrarlo en el proceso de análisis y reflexión que caracteriza a la investigación académica de calidad

* 1. **Problemática del caso de estudio**

La problemática abordada en el presente estudio se centra en la optimización de bases de datos en el contexto de consultas SQL, con un enfoque particular en la eficiencia y el rendimiento de las operaciones de consulta. En un entorno cada vez más exigente en términos de velocidad y escalabilidad, la necesidad de mejorar el tiempo de respuesta de las consultas se convierte en un desafío crucial para garantizar un adecuado funcionamiento de los sistemas de bases de datos.

La complejidad de las consultas SQL, la cantidad de datos a procesar y la estructura de la base de datos pueden impactar significativamente en el rendimiento de las operaciones, afectando la experiencia del usuario y la eficacia de las aplicaciones que dependen de dichas consultas. Por tanto, la problemática se enfoca en identificar estrategias y técnicas de optimización que permitan reducir los tiempos de ejecución de las consultas, minimizar el uso de recursos y mejorar la eficiencia global del sistema.

Además, la problemática del caso de estudio se ve agravada por la necesidad de trabajar con un conjunto de datos realista y representativo, como el utilizado en el escenario de la Biblioteca Nacional. La diversidad de consultas, la complejidad de las relaciones entre las entidades y la variabilidad en los patrones de acceso a la información plantean desafíos adicionales que requieren soluciones específicas y adaptadas al contexto particular de la base de datos analizada.

En este sentido, la problemática del caso de estudio no solo se centra en la optimización de consultas SQL en un entorno genérico, sino que también considera las particularidades y restricciones propias del escenario de la Biblioteca Nacional, lo que añade un nivel de complejidad adicional a la búsqueda de soluciones efectivas y eficientes. La identificación y resolución de estos desafíos constituyen el núcleo de la investigación y orientan la exploración de estrategias innovadoras y prácticas para mejorar el rendimiento de las bases de datos en este contexto específico.

* 1. **Modelo de datos**

Ver el anexo para el diagrama ERD

* 1. **Caracterización de las tablas**

*La Biblioteca Nacional José Martí alberga una diversidad de colecciones, cada una representada por un identificador único, como, por ejemplo, "MED". Cada colección se encuentra ubicada en una sala específica, identificada por su correspondiente identificador. Un ejemplo es la Sala de Servicios Generales, que alberga la Colección Cubana. Esta colección cuenta con unos varios correos electrónicos a través de la cual se puede contactar a la biblioteca en coleccioncubana@bnjm.cu. La descripción de la colección proporciona información detallada sobre su contenido, brindando una visión integral de los recursos que alberga. Además, se clasifica como una colección especial, abarcando artículos de arte, fotos, música y mapas, lo que la distingue de las colecciones Generales, que incluyen libros, carteles, seriados y fichas. Este enfoque detallado y estructurado permite a los usuarios de la biblioteca acceder y explorar de manera eficiente las distintas colecciones, cada una con sus características y recursos únicos.*

*Cada* ***Sala*** *en la Biblioteca es identificada por un código único, por ejemplo, "CAR-2018-0070-C". Cada sala lleva consigo un nombre único, que destaca su propósito o contenido, como "Sala Circundante" o "Sala Rusa". La* ***localización*** *específica dentro del edificio es esencial para la ubicación física de los usuarios interesados. Los* ***teléfonos****, compuestos por números en una extensión específica por cada sala, ofrecen canales de comunicación directa con el personal de la sala, mientras que la lista de* ***correos*** *proporciona opciones adicionales para el contacto, permitiendo una comunicación efectiva con el personal encargado. La* ***descripción*** *de cada sala brinda una visión integral de su contenido y su importancia en el contexto bibliotecario. El* ***método de acceso*** *varía según el tipo de usuario y el nivel de consulta, con ejemplos como el uso de carnés de usuario o la condición de ser investigador. Además, cada sala ofrece una serie de* ***servicios complementarios****, que pueden incluir préstamos, consulta en línea, fotocopiado, digitalización, entre otros. Los usuarios interesados en explorar las salas y sus colecciones pueden obtener información detallada en el mostrador de recepción, donde se proporciona asistencia personalizada.*

Cada **Documento** es singularmente identificado por un número único de documento, facilitando una referencia precisa. Cada documento tiene un título distintivo que refleja su contenido o temática, como "Historia de la Literatura Cubana" o "Concierto de Beethoven". Los autores, responsables del contenido, se registran para dar crédito a quienes contribuyeron a la creación del documento. La fecha de publicación o creación permite ubicar el documento temporalmente, mientras que la editorial representa la entidad responsable de su producción y distribución.

La ubicación específica del documento en la biblioteca es esencial para su localización física, y el idioma en que está escrito se registra para facilitar a los usuarios la búsqueda de documentos en un idioma específico. El formato del documento, ya sea libro, revista, música, o arte, se especifica para adaptarse a la diversidad de materiales en la biblioteca. La materia principal del documento se clasifica según un vocabulario controlado o una clasificación establecida.

Para documentos específicos, como libros, se incluyen detalles adicionales como el ISBN, Dewey, género e ISSN. Para música, se registran datos como género, intérprete, compositor y duración. Obras de arte y carteles pueden tener información adicional como editorial, técnica y dimensiones. Se contempla la posibilidad de datos particulares para documentos de etnología, manuscritos, mapas, mediatecas, referencias, publicaciones seriadas, fotos y fichas.

Esta estructura proporciona una visión integral y flexible para gestionar la diversidad de documentos en la Biblioteca Nacional José Martí.

Las tablas específicas para diferentes tipos de documentos en la base de datos de la Biblioteca Nacional José Martí ofrecen una estructura detallada para gestionar y organizar documentos diversos.

Para los manuscritos destaca por incluir un atributo adicional llamado "periodo", que proporciona información sobre el período histórico al que pertenece cada manuscrito. Esta característica permite una clasificación más precisa y contextualización temporal de estos documentos escritos a mano.

En cuanto a los mapas, se enfoca en dimensiones físicas, registrando la altura y el ancho del mapa. Estos atributos específicos son fundamentales para aquellos documentos cartográficos, proporcionando detalles cruciales sobre sus proporciones físicas y facilitando la gestión de su almacenamiento y acceso.

Las imágenes, comparten un enfoque similar al de los mapas en términos de dimensiones físicas. Aquí, la altura y el ancho de las imágenes son registrados, brindando información esencial para la presentación y organización de estos documentos visuales.

La pintura, como forma artística, que incluye atributos únicos como "técnica", describiendo la técnica utilizada para crear la pintura. Además, se registran las dimensiones físicas, lo que añade profundidad a la descripción de estos documentos visuales.

La mediateca abarca contenido multimedia y se distingue por atributos como "género", "director", "productor" y "duración". Estos atributos son cruciales para la gestión eficiente de documentos multimedia, permitiendo una clasificación y búsqueda más precisa.

En el ámbito musical, se centra en detalles específicos de piezas musicales, como género, intérprete, compositor y duración. Estos atributos proporcionan una visión detallada de la información asociada con documentos musicales.

Para documentos de referencia, destaca con el atributo adicional "serie", que enumera los documentos de referencia de manera única, facilitando su identificación y recuperación.

Las revistas, representadas en la tabla "magazine", incluyen atributos distintivos como "editor" e "ISSN". Estos datos son esenciales para la identificación y gestión de revistas dentro de la biblioteca.

Finalmente, los libros y con atributos específicos como "género", "ISSN", "ISBN" y "Dewey". Estos detalles ofrecen una descripción completa de los libros, facilitando su clasificación y búsqueda

Existen miembros asociados a la y estos constituyen la diversidad de individuos que acceden a sus recursos y servicios. Cada miembro se distingue por un identificador único. La información asociada a cada miembro incluye su nombre, edad y una categoría que define su relación con la biblioteca.

La categoría del miembro es un componente clave, clasificando a los individuos en segmentos distintos según sus roles y propósitos dentro de la biblioteca. Esta categorización se divide en opciones específicas como "Investigador", "Profesional", "Transitorio" u "Otro". Cada categoría refleja los diferentes perfiles y necesidades de los miembros, facilitando así la adaptación de servicios y recursos de la biblioteca para satisfacer de manera más eficiente sus requisitos particulares.

Se necesita también almacenar información sobre los autores de diversos documentos presentes en su colección. Cada registro en esta tabla representa un autor identificado por un único código y está asociado con su respectivo nombre. Un autor está asociado con documentos específicos, como libros, artículos, música, y otros recursos presentes en la biblioteca.

El servicio de préstamo en la Biblioteca Nacional José Martí ofrece a los usuarios la oportunidad de llevarse a sus hogares ciertos documentos de la colección por un tiempo definido y bajo ciertas condiciones. Este servicio está disponible para los miembros categorizados como "Investigador" y "Profesional".

Para acceder al préstamo, es necesario estar registrado en la biblioteca y presentar el carné de lector, obtenido de forma gratuita en el mostrador de recepción. Este carné tiene una validez de un año y se puede renovar fácilmente. El préstamo está sujeto a la disponibilidad de documentos y a las normas establecidas por la biblioteca.

No todos los documentos son elegibles para préstamo; sólo aquellos con múltiples copias y que no forman parte de colecciones especiales pueden ser prestados. Cada usuario puede llevar prestados hasta tres documentos, con un plazo máximo de quince días, prorrogables por otros quince si no hay reservas pendientes. La devolución puntual y en buen estado es crucial, ya que el incumplimiento puede acarrear sanciones como multas o suspensión del servicio.

Además, se ofrece un servicio de préstamo interbibliotecario que permite a los usuarios solicitar documentos no disponibles en la Biblioteca Nacional José Martí, pero presentes en otras bibliotecas del país o del extranjero. Durante un préstamo interbibliotecario, se registran detalles sobre el documento solicitado, información de la biblioteca solicitante y fechas clave del proceso. Este servicio amplía el acceso a recursos más allá de la colección local.

* 1. **Soluciones dadas a las problemáticas de Sistemas de Bases de Datos**

Se presentan diversas estrategias para abordar los desafíos identificados en el estudio de optimización de bases de datos en la Biblioteca Nacional. Entre las soluciones propuestas se incluyen la optimización de consultas en SQL mediante la aplicación de técnicas como la optimización de consultas básicas, el uso de funciones de ventana (FAV) y la implementación de expresiones de tabla comunes (WITH). Estas estrategias se han evaluado comparando los tiempos de ejecución de cada enfoque para determinar la opción más eficiente en términos de rendimiento y eficacia. Además, se ha implementado un sistema de índices en la base de datos para acelerar la recuperación de datos y mejorar el rendimiento de las consultas, lo que ha demostrado ser efectivo en la optimización del acceso a la información. Los casos de estudio específicos abordados en el análisis han permitido identificar los beneficios de estas soluciones en la reducción de tiempos de respuesta, la mejora del rendimiento de las consultas y la eficiencia general del sistema.

Se han desarrollado funciones personalizadas para transformar y manipular datos de manera específica, como la transformación de entradas de tablas para mostrar información relevante de forma estructurada y comprensible. Estas funciones han contribuido a mejorar la presentación y accesibilidad de la información almacenada en la base de datos.

Los cursores se han empleado en situaciones donde se requiere un procesamiento detallado y personalizado de conjuntos de datos, como la corrección de errores en registros específicos o la transferencia de datos entre tablas de manera controlada. Por ejemplo, se han utilizado cursores para identificar y corregir errores en la información de los estudiantes de Cuba, así como para transferir datos de imágenes a una tabla específica con nuevas especificaciones. Estas acciones detalladas han permitido realizar operaciones complejas de manera eficiente y precisa en la base de datos.

La combinación de estas estrategias, incluyendo funciones, triggers y cursores, ha contribuido significativamente a la optimización y mejora del sistema de bases de datos en la Biblioteca Nacional, permitiendo una gestión más eficiente, un acceso más rápido a la información y una mayor integridad de los datos almacenados.

**Conclusiones parciales**

Tras un exhaustivo análisis del estado del arte en optimización de bases de datos, se ha podido identificar una variedad de estrategias y técnicas utilizadas para mejorar el rendimiento de consultas en SQL. La recopilación de información existente ha permitido establecer una base sólida para comprender la importancia de la optimización en entornos de gestión de datos. Se ha evidenciado la relevancia de utilizar herramientas avanzadas de SQL, como funciones, triggers y cursores, para garantizar la eficiencia y la integridad de los datos almacenados. Este análisis ha sentado las bases para la exploración de soluciones innovadoras y prácticas que contribuyan a mejorar el rendimiento de las bases de datos en diversos contextos.

**Capítulo II. Optimización de bases de datos**

**Introducción**

El presente capítulo se enfoca en analizar y comparar estrategias de optimización de consultas en SQL, centrándose en el uso de SQL básico, FAV (window functions), y la cláusula WITH (common table expressions) así como los índices. Se han planteado diversas problemáticas basadas en un caso de estudio (Biblioteca Nacional), evaluando los tiempos de ejecución de cada enfoque para determinar la opción más eficiente.

Se crearon 5 problemáticas del caso de estudio (Biblioteca Nacional) y de cada uno se mostrará una posible implementación con SQL básico, fav, y la cláusula with, en conjunto con los tiempos medidos

**2.1 Índices**

## Búsqueda de Texto

Se informó de quejas en la página web de la Biblioteca sobre la lentitud de las respuestas, luego de un análisis extenso se llegó a la conclusión que tiene que ver con el módulo de búsqueda de texto

Se podrá de ejemplo la tabla ‘author’ las consultas serán en este formato %substring%

Refiérase al anexo: , y , , para el resultado del explain analyze

Como se aprecia el método 1 no arrojó resultados favorables por lo que se eliminó, el método 2 arrojó los mejores, pero aumenta la complejidad al usar el método gin y diccionarios para hacer las búsquedas, el método 3 hace uso del método gin y usa trigramas (refiérase a la bibliografía), tiene como requisitos el crear una extensión, pero es el más completo y fácil de los 3

## Búsqueda de Texto con prefijo

Aquí como anotación se demuestra una tercera forma que es más rápida si se trata de búsquedas en la forma substring%

Refiérase al anexo , y anexo , para el resultado del explain analyze

## Consultas en Rango

La Biblioteca muy a menudo desea hacer censos sobre sus miembros asociados en cuanto a su edad, pero se cuenta con un enorme dataset que no es comúnmente actualizado, para optimizar este tipo de query se usó índices

Refiérase al anexo , y anexo , para el resultado del explain analyze

## Búsqueda directa

Se desea brindar una nueva funcionalidad en la página web a sus usuarios en la cual este pueda hacer una búsqueda por nombre exacto de la ubicación de una sala

Refiérase al anexo y anexo , para el resultado del explain analyze

**2.2 FAV & WITH**

**Ejemplo #1:**

Se necesita de los préstamos que se entregan en un día darle prioridad a los que han estado más tiempo prestados, para eso: Se desea mostrar los datos de los préstamos (id\_document, id\_service, fecha inicio, fecha entrega), así como la prioridad de los préstamos que se entregan en un mismo día con 1 la más alta que tendrá el que más tiempo a estado prestado (puede haber empates)

Revisar adjuntos: , , ,

Resultados: FAV esa la mejor opción por simplicidad y velocidad

**Ejemplo #2:**

Se quiere hacer un censo de los miembros que son estudiantes universitarios jóvenes asociados a la biblioteca, así como el promedio de edad por país de los estudiantes. De ellos se desea obtener los datos (nombre, edad, país, escuela) así como el promedio de edad

Revisar adjuntos**: , , ,**

Resultados: FAV esa la mejor opción por simplicidad y velocidad

**Ejemplo #3:**

Se desea conocer el título, fecha de creación, formato, género y promedio de duración de las últimas 5 canciones añadidas a su catálogo, y la duración total, según el género al que pertenezcan

Revisar adjuntos: , ,

Resultados: FAV esa la mejor opción por simplicidad y velocidad

**Ejemplo #4:**

Se desea conocer los datos de profesionales que tienen deudas y el total de deuda relacionada a empleados que trabajan para la misma empresa

Revisar adjuntos: , ,

Resultados: FAV y WITH están casi empatados dado que la diferencia es demasiado pequeña y las muestras fueron muy pocas

**Ejemplo #5:**

Se desea conocer el título, formato, lugar de publicación, así como el promedio del ancho y altura de las fotos por lugar de publicación

Revisar adjuntos: , ,

Resultados: FAV esa la mejor opción por simplicidad y velocidad

**Ejemplo #6:**

Se desea mostrar para todos los préstamos mostrar el id\_servicio, id\_document, fecha de inicio, fecha final y diferencia promedio entre la fecha de entrega y el plazo de entrega de todos los préstamos de la misma fecha de inicio

Revisar adjuntos: , , ,

Resultado: WITH fue la más rápida

**Ejemplo #7:**

Se desea mostrar los mapas y las dimensiones promedio de estos

Revisar adjuntos: , , ,

Resultado: En este caso parece ser que postgres está planificando y optimizando mejor la consulta de SQL básico que la fav y la with. Esto puede deberse al tamaño no muy extenso de la tabla

**Ejemplo #8:**

Se desea mostrar por cada documento de la editorial Halvorson LLC sus datos y el ano promedio de creación de estos si son de la misma editorial

Revisar adjuntos: , , ,

Resultado: Fav en este caso ofrece una mejora significativa respecto a su respectivo SQL básico y al with

**Ejemplo #9:**

Se desea mostrar el ranking de los miembros a partir de su edad según su país sin coinciden tendrán el mismo número

Revisar adjuntos: , ,

Resultado: El with y el fav quedan en velocidades muy similares por obvias razones, dado que dentro del with se usó un fav. En este caso se recomienda usar el FAV dado que ofrece el mejor rendimiento y simplicidad

**Ejemplo #10:**

Se desea mostrar los datos de las canciones y el promedio de la duración entre las que son del mismo género

Revisar adjuntos: , , ,

Resultado: El with es el más óptimo en este caso

**2.4 Otras técnicas de optimización**

En esta sección se hablará de algunas técnicas de optimizaciones adicionales que son muy interesantes y únicas a su manera

- Parallel Queries

- Partitioning

- Materialized Views

- Prepared Statements

**Parallel Queries**

Las consultas paralelas, o "parallel queries", son aquellas consultas SQL que se ejecutan de manera simultánea utilizando múltiples procesadores o núcleos del servidor de base de datos. Esta técnica de optimización se utiliza para acelerar el procesamiento de consultas complejas al dividir la carga de trabajo entre varios hilos de ejecución.

Es el trabajo del planificador de consultas determinar si es la mejor estrategia para usarse y cuantos núcleos puede usar, esto está definido en la configuración de postres bajo la variable

max\_parallel\_workers\_per\_gather: El número de workers que el planificador usará

Se caracteriza por:

- División de Tareas: Las tareas de la consulta se dividen en subconjuntos independientes que pueden ejecutarse simultáneamente.

- Uso de Recursos Paralelos: Se aprovechan múltiples núcleos o procesadores disponibles en el sistema para realizar operaciones en paralelo.

- Coordinación de Resultados: Al finalizar las operaciones paralelas, los resultados se combinan para producir el resultado final de la consulta.

- Eficiencia en Consultas Complejas: Se beneficia especialmente en consultas que involucran grandes conjuntos de datos u operaciones complejas que pueden dividirse en partes independientes.

**Partitioning**

El particionado es una técnica de optimización en bases de datos que consiste en dividir una tabla grande en secciones más pequeñas llamadas particiones. Cada partición tiene su propio conjunto de datos y, al dividir la tabla en partes más manejables, se pueden obtener beneficios en términos de rendimiento y mantenimiento. A continuación, se abordan los puntos clave sobre el particionado en SQL:

PostgreSQL tiene soporte para los siguientes 3 tipos de particiones:

- Range Partitioning

- List Partitioning

- Hash Partitioning

Tiene como características:

- Mejora en el rendimiento: Las consultas y operaciones que se centran en un subconjunto específico de datos pueden ser más rápidas, ya que el sistema solo necesita buscar en la partición relevante en lugar de toda la tabla.

- Facilita el Mantenimiento: Las operaciones de mantenimiento, como la carga de datos y la eliminación, pueden ser más eficientes al trabajar con particiones más pequeñas en lugar de la tabla completa.

- Optimización de Recursos: Al limitar el acceso a ciertas particiones, el sistema puede asignar recursos de manera más efectiva para gestionar operaciones concurrentes

Ver

**Materialized Views**

Las vistas materializadas (Materialized Views) son objetos de base de datos que almacenan los resultados de una consulta para permitir un acceso más rápido y eficiente a los datos. A diferencia de las visitas regulares, que son consultas almacenadas y recalculadas cada vez que se llaman, las vistas materializadas almacenan físicamente los resultados y se actualizan periódicamente.

Se caracteriza por:

- Almacenamiento Físico A diferencia de las visitas regulares, las vistas materializadas almacenan físicamente los datos en disco.

- Actualización Programada: Los datos en una vista materializada se actualizan periódicamente según una programación o en respuesta a eventos específicos.

- Acceso Rápido: Proporcionan un acceso más rápido a los resultados precalculados en comparación con la ejecución de la consulta original

Ver

**Conclusiones parciales**

Tras explorar las diversas estrategias de optimización de consultas en SQL en el contexto del caso de estudio basado en la Biblioteca Nacional, se ha evidenciado la importancia de implementar técnicas avanzadas para mejorar el rendimiento y la eficiencia de las operaciones de consulta. El uso de SQL básico, FAV (window functions) y la cláusula WITH (common table expressions) ha demostrado ser fundamental para optimizar los tiempos de ejecución y la gestión de grandes volúmenes de datos. La introducción de herramientas como triggers, funciones y cursores ha permitido garantizar la integridad de los datos, aplicar reglas de negocio complejas y facilitar la reutilización de lógica de negocio en múltiples consultas. Estas estrategias han contribuido significativamente a mejorar el rendimiento de las consultas y a optimizar la eficiencia global del sistema de bases de datos en el escenario específico de la Biblioteca Nacional.

**Capítulo III. Comportamiento activo y seguridad de bases de datos.**

**Introducción**

El comportamiento activo en bases de datos es una estrategia que permite automatizar acciones en respuesta a eventos específicos, brindando una mayor flexibilidad y dinamismo a los sistemas de gestión de datos. Al implementar el comportamiento activo, las bases de datos pueden responder de manera proactiva a cambios en los datos o en el entorno, mejorando la eficiencia y la capacidad de adaptación del sistema. En este contexto, se explorarán las ventajas y desafíos de incorporar el comportamiento activo en bases de datos, así como su impacto en la optimización del rendimiento y la gestión de la información.

**3.1 Funciones**

**Ejemplo 1**

Se desea una función que transforme las entradas de la tabla document a un formato de cuando fue creada y porqué editorial

Revisar anexo:

**Ejemplo 2, 3**

A Bob se le encomendó para su prueba técnica la tarea de hacer una función que devuelva la extensión + números de todos los números asociados a una sala específica

Revisar anexo:

**Ejemplo 4**

Dado un id de un miembro de la Biblioteca, se desea saber si está dentro del rango de jóvenes, o adultos o viejos

Revisar anexo:

**Ejemplo 5**

Se desea conocer el nombre de todas las bibliotecas a las cuales se les está prestando documentos

Revisar anexo:

**3.2 Triggers**

Entre las formas que puede adoptar un trigger están

Row-level triggers: Estos disparadores se invocan una vez para cada fila afectada por una declaración que disparó el trigger. Cuando se llama a una función PL/pgSQL como un trigger, se definen automáticamente varias variables especiales para describir la condición que provocó la llamada.

Statement-level triggers: A diferencia de los disparadores a nivel de fila, los disparadores a nivel de declaración se invocan una vez para cada declaración, independientemente del número de filas afectadas. Operan en todo el conjunto de filas afectadas por la declaración.

## Triggers en cascada

Se desea garantizar que al eliminar una tupla de la tabla phone\_room se elimine también el respectivo teléfono

Revisar anexo

Se desea garantizar que al eliminar una sala sean también eliminas las respectivas conexiones service\_email, service\_phone y si existe algún miembro que este actualmente usando un servicio de esa sala esta no sea eliminada alertando al usuario de que no se puede llevar a cabo la operación

Revisar anexo

## Triggers para auditoría

Se desea registrar los cambios hechos por un usuario al monto a pagar de una multa

Revisar anexo

Se desea registrar la actividad sobre los libros de cualquier operación insert/update/delete

Revisar anexo

## Triggers para validación

Para asegurar que no sea posible imponer un monto de multa mayor a 1000 pesos con un mínimo de 10 pesos se creó un trigger function que valida la creación de una multa

Revisar anexo

Se quiere garantizar que no sea posible registrar un préstamo a un miembro trabajador profesional de un documento que sea patrimonio

Revisar anexo

Se quiere garantizar que al asociar un correo a una librería ambos existan, y así mantener la integridad referencial

Revisar anexo

## Triggers para metadatos

Se quiere quitar complejidad del backend de la aplicación a la hora de garantizar la fecha de creación de los préstamos, además de garantizar una mayor coherencia del formato, para ello se usó un trigger

Revisar anexo

**3.3 Cursores**

A continuación, se presentan situaciones hipotéticas que se asemejan a posibles situaciones reales en las cuales se podría usar cursores.

**Situación 1:**

Hubo un problema con el frontend que causa que la edad de los estudiantes de Cuba al llenar el formulario se registrase 1 año menos, adicionalmente mostrar los ids de estos para arreglar esto se hizo uso de cursores

Revisar anexo:

**Situación 2:**

Se ha detectado que las imágenes registradas por la editorial "Halvorson LLC" son en realidad imágenes de mapas por lo que estas deben ser transferidas a la tabla mapas y su nueva escala será 1000x1000 y el tipo de mapa será físico además se desea retornar los ids de estas fotos para ser revalidadas por la cache de la aplicación cliente

Revisar anexo:

**Situación 3:**

Se desea subir el precio de la multa a todos los préstamos que han estado más de 100 días sin ser devueltos pasado la fecha de entrega

Revisar anexo:

**Situación 4:**

Bob es un empleado que era el encargado de registrar los préstamos y de estos el plazo de entrega ("term") pero algunas personas tienen un plazo que acaba un día feriado quitándole la posibilidad a la persona de poder emplear al completo su plazo entonces se requiere que implementes un cursor que actualice el plazo de las personas cuya fecha de inicio + plazo caigan en el día feriado determinado. (PD: Bob obviamente fue despedido)

Revisar anexo:

**Situación 5:**

Se desea para el frontend de la Biblioteca Nacional realizar una paginación dado un límite de página y una página. Esta vez tuvieron que re-contratar a Bob dado que es el único que sabe de cursores

Bob completo la asignación y posteriormente fue despedido porque nadie perdona su error anterior

Revisar anexo:

**Conclusiones parciales**

El análisis del comportamiento activo en bases de datos ha revelado su potencial para automatizar procesos, mejorar la eficiencia operativa y facilitar la toma de decisiones en tiempo real. La implementación de reglas y disparadores activos ha demostrado ser una estrategia efectiva para gestionar eventos y desencadenar acciones específicas en función de condiciones predefinidas. La introducción del comportamiento activo en bases de datos promete optimizar la respuesta a cambios en los datos y en el entorno, contribuyendo a una gestión más dinámica y adaptable de la información.

**Conclusiones generales**

En conclusión, la optimización de bases de datos a través de estrategias avanzadas en SQL y la incorporación del comportamiento activo representan enfoques clave para mejorar el rendimiento, la eficiencia y la integridad de los sistemas de gestión de datos. La combinación de herramientas avanzadas de SQL, como funciones personalizadas y cursores, junto con la automatización de acciones mediante el comportamiento activo, ha demostrado ser fundamental para garantizar un acceso rápido y eficaz a la información almacenada. Estas innovaciones no solo optimizan el rendimiento de las consultas, sino que también contribuyen a una gestión más eficiente y dinámica de las bases de datos en entornos complejos y cambi

**Bibliografía Consultada**

*39.1. Overview of Trigger Behavior*. (2024, February 8). PostgreSQL Documentation. <https://www.postgresql.org/docs/current/trigger-definition.html>

*39.1. Overview of Trigger Behavior*. (2024b, February 8). PostgreSQL Documentation. <https://www.postgresql.org/docs/current/trigger-definition.html>

*11.2. Index types*. (2024, February 8). PostgreSQL Documentation. <https://www.postgresql.org/docs/current/indexes-types.html>

*Chapter 70. GIN Indexes*. (2024, February 8). PostgreSQL Documentation. <https://www.postgresql.org/docs/current/gin.html>

Cockroach Labs Documentation Team. (n.d.). *Trigram indexes*. CockroachDB Docs. <https://www.cockroachlabs.com/docs/stable/trigram-indexes#:~:text=A%20trigram%20index%20is%20a%20type%20of%20inverted,large%20tables%20without%20providing%20an%20exact%20search%20term>.

*CREATE INDEX*. (2024, February 8). PostgreSQL Documentation. <https://www.postgresql.org/docs/current/sql-createindex.html>

*43.10. Trigger Functions*. (2024, February 8). PostgreSQL Documentation. <https://www.postgresql.org/docs/current/plpgsql-trigger.html#:~:text=PL%2FpgSQL%20can%20be%20used%20to%20define%20trigger%20functions,change%20triggers%29%20or%20event_trigger%20%28for%20database%20event%20triggers%29>

**Anexos**

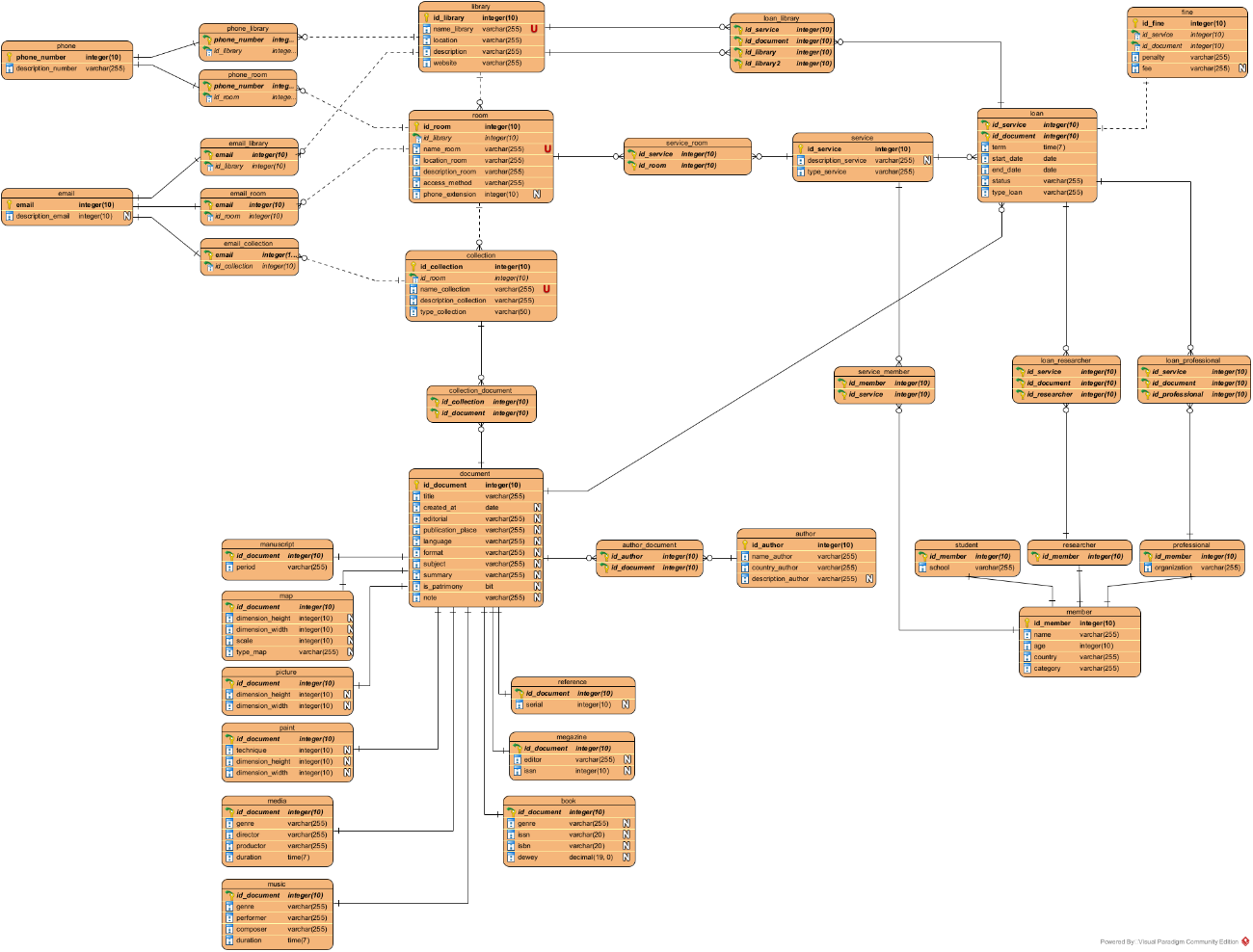


Figura 0: ERD

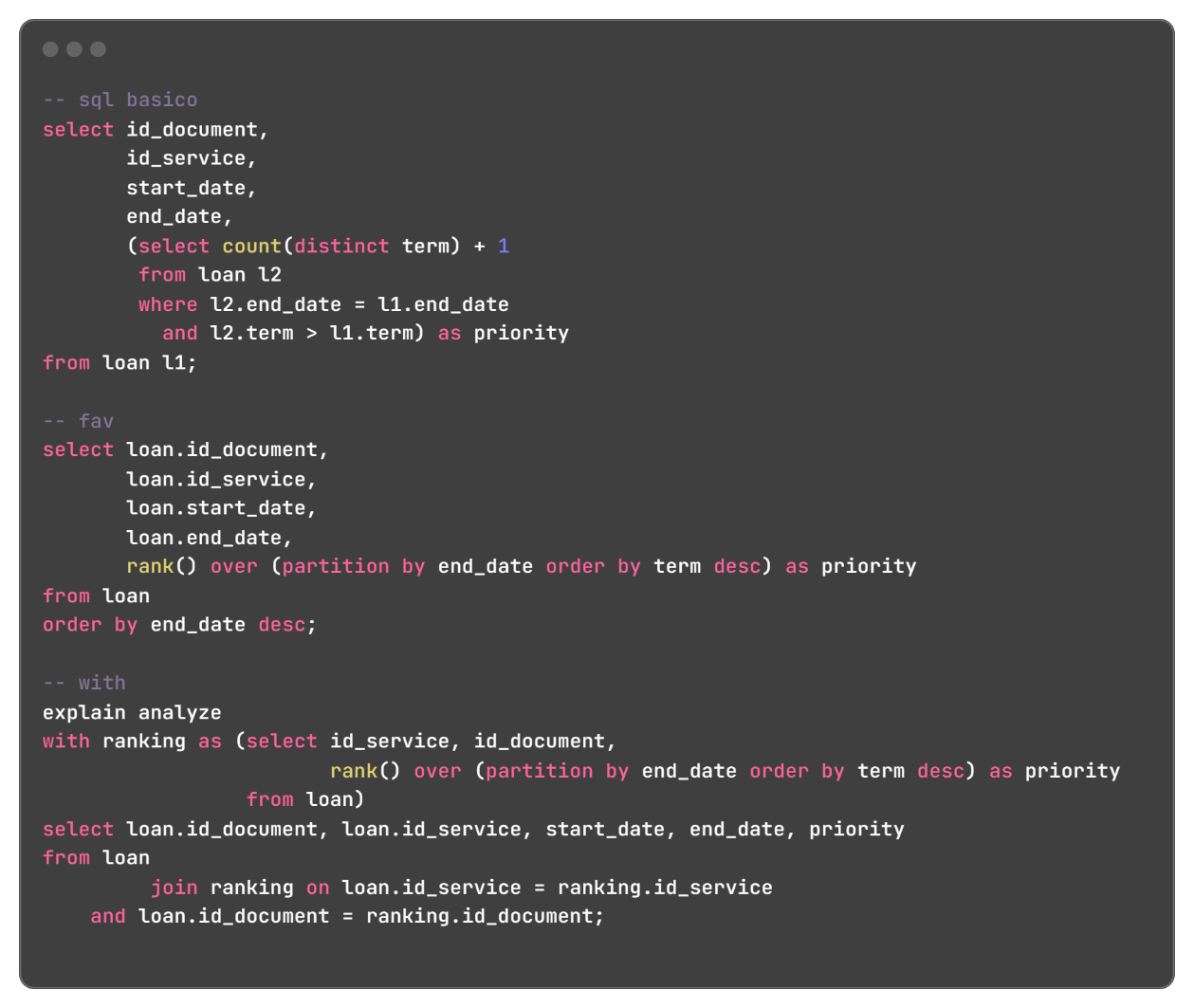


Figura 1: Ejemplo

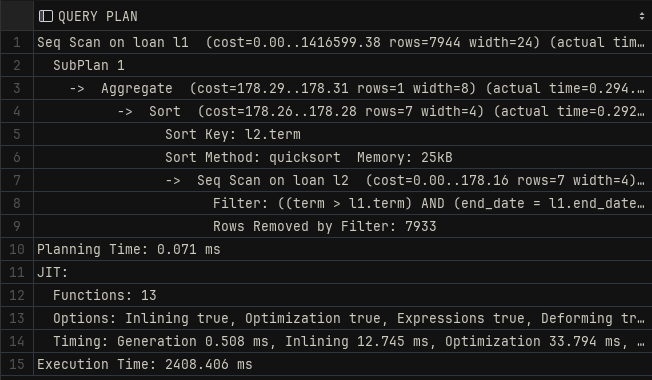


Figura 2: SQL Básico

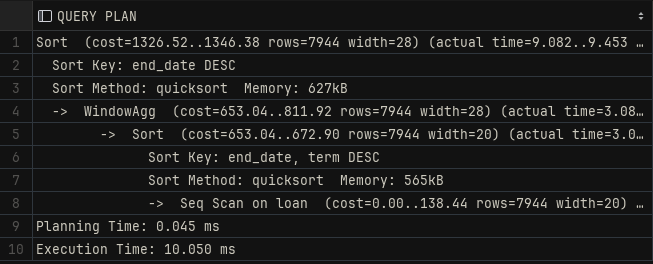


Figura 3: Fav

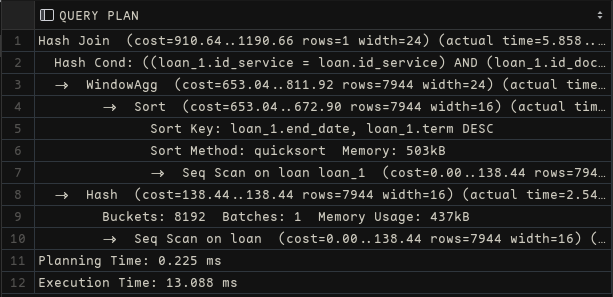


Figura 4: WITH



Figura 5: Ejemplo

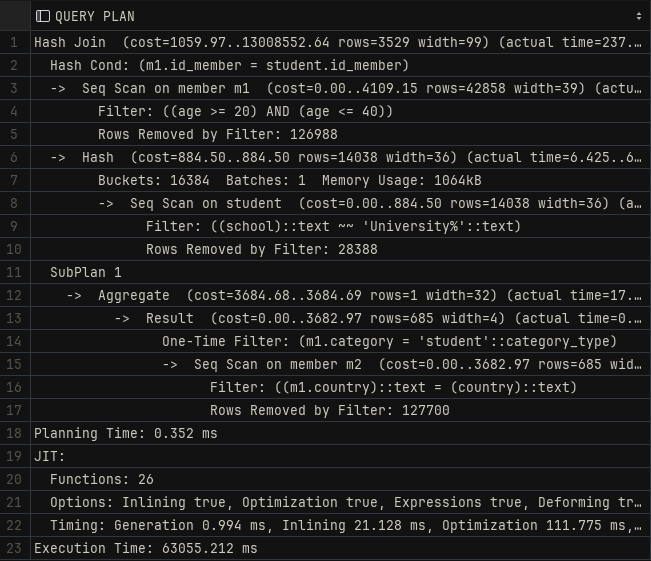


Figura 6: SQL Básico

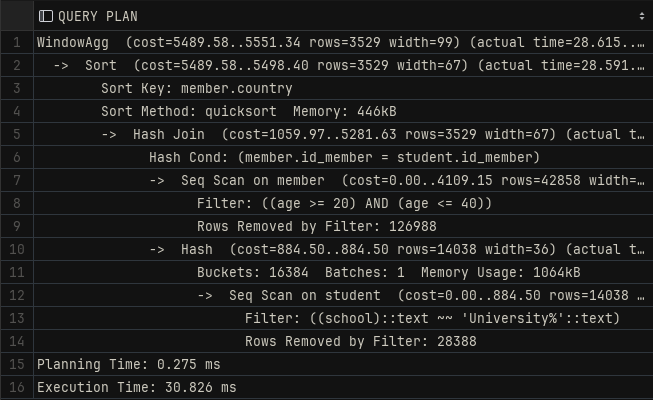


Figura 7: Fav

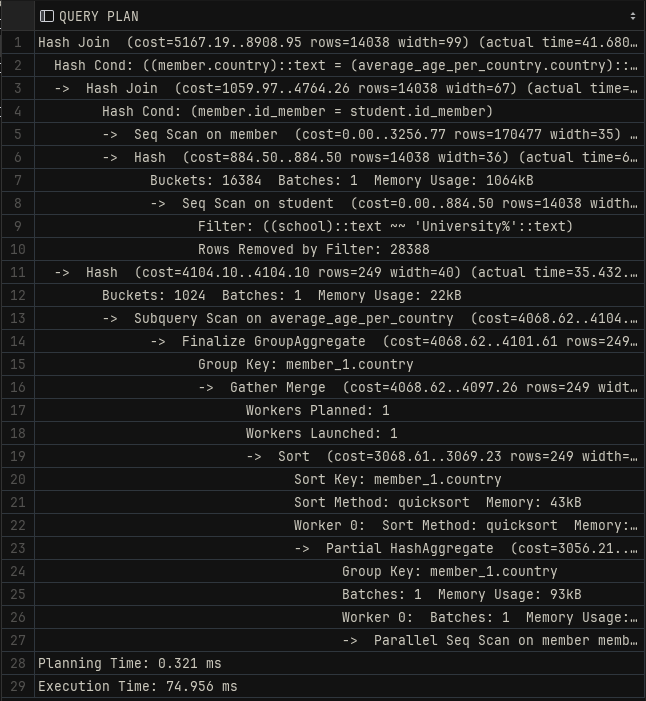


Figura 8: WITH



Figura 9: Ejemplo

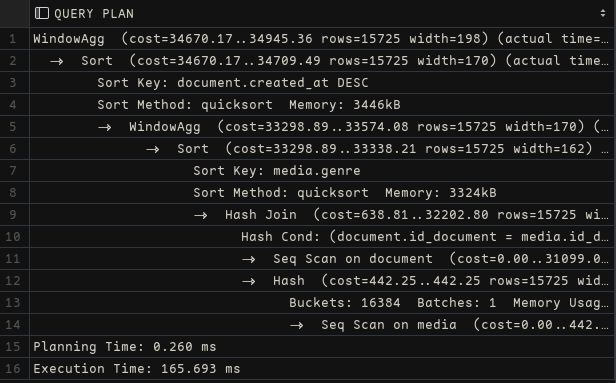


Figura 10: Fav

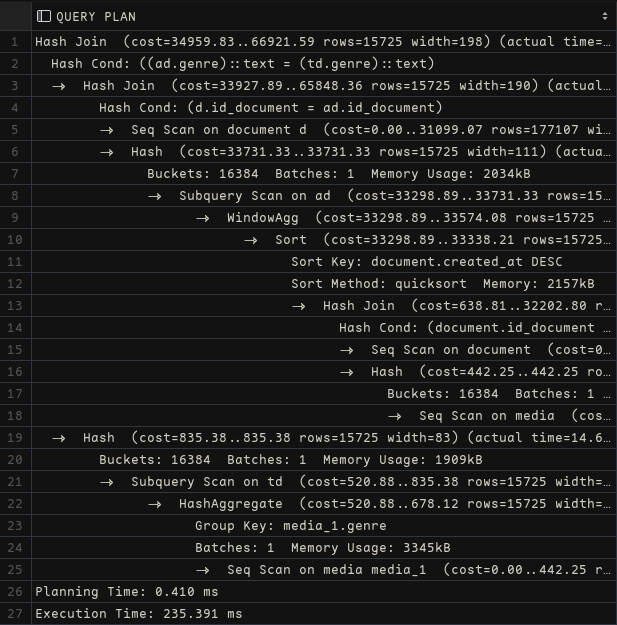


Figura 11: WITH



Figura 12: Ejemplo

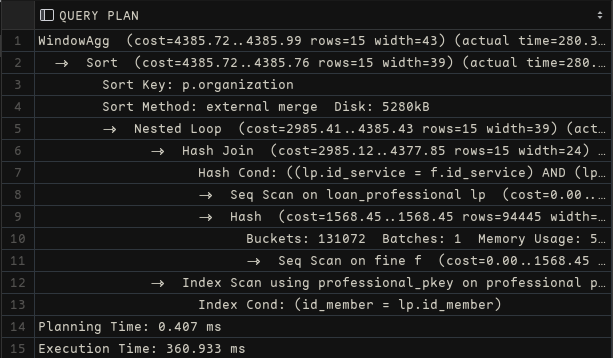


Figura 13: Fav



Figura 14: WITH



Figura 15: Ejemplo 5

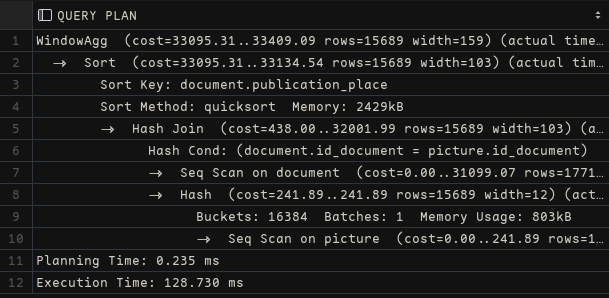


Figura 16: Fav

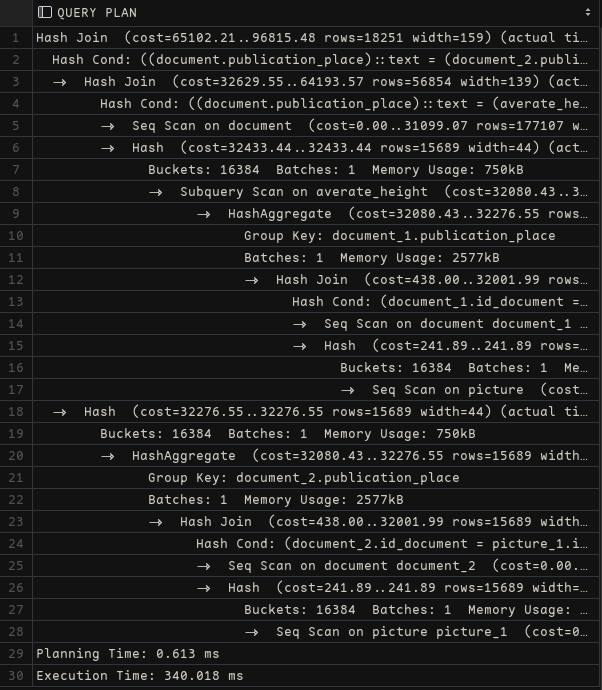


Figura 17: WITH

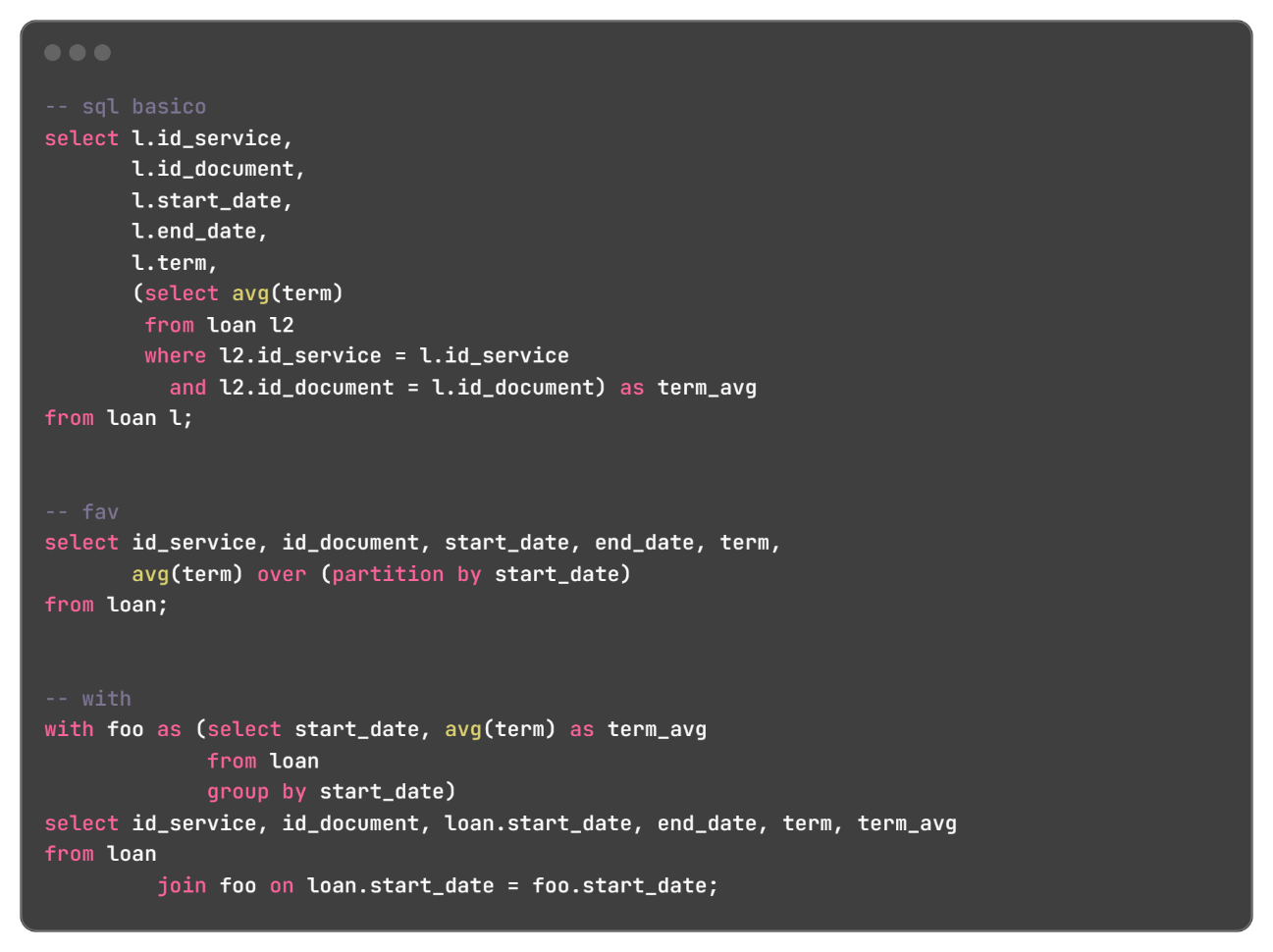


Figura 18: Ejemplo 6

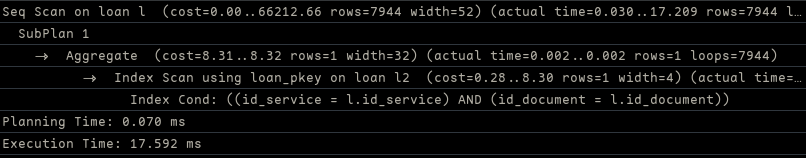


Figura 19: SQL Básico

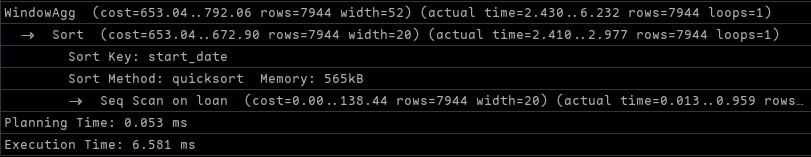


Figura 20: Fav

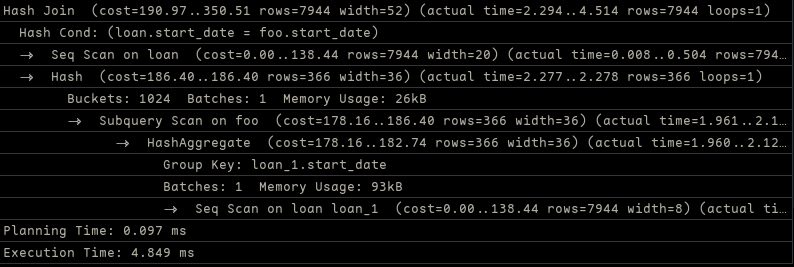


Figura 21: WITH



Figura 22: Ejemplo 7

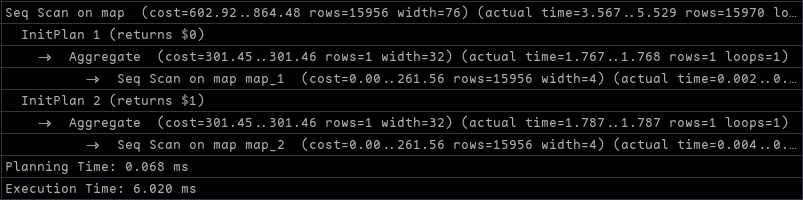


Figura 23: SQL Básico

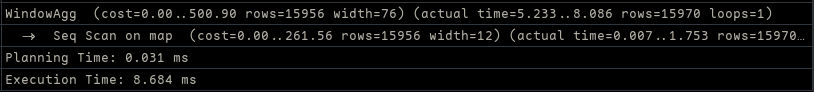


Figura 24: Fav

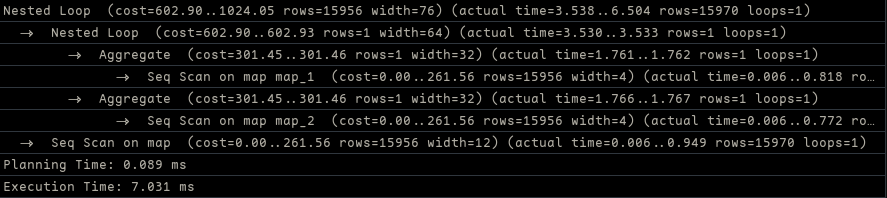


Figura 25: WITH



Figura 26: Ejemplo 8

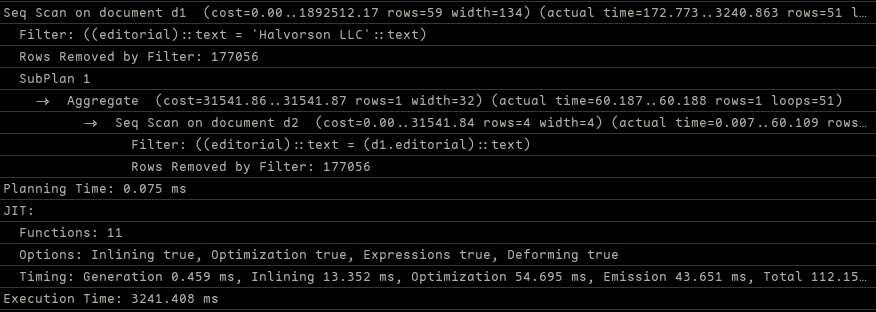


Figura 27: SQL Básico

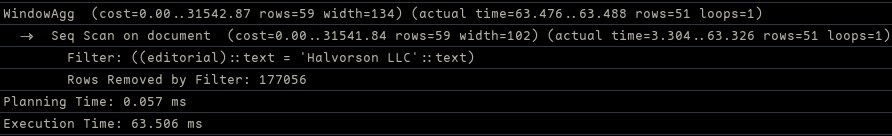


Figura 28: Fav

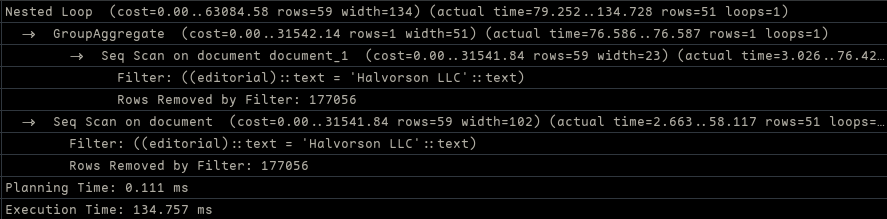


Figura 29: WITH



Figura 30: Ejemplo 9

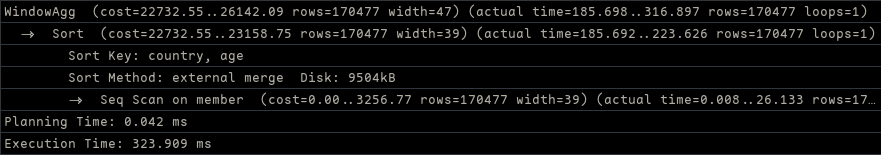


Figura 31: Fav

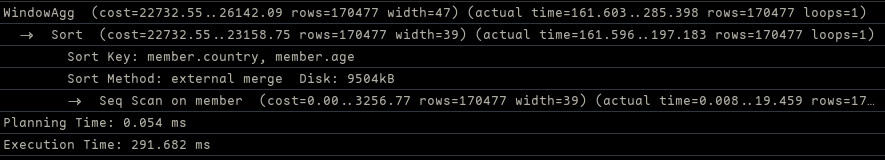


Figura 32: WITH



Figura 33: Ejemplo 10

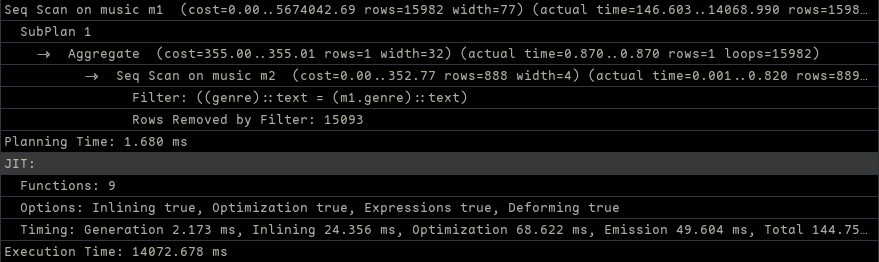


Figura 34: SQL Básico

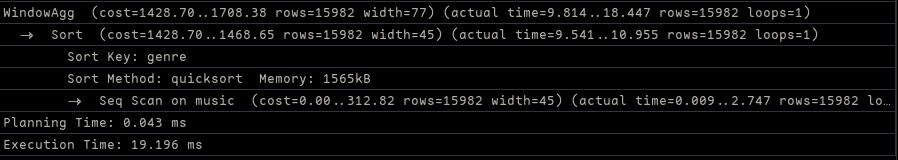


Figura 35: Fav

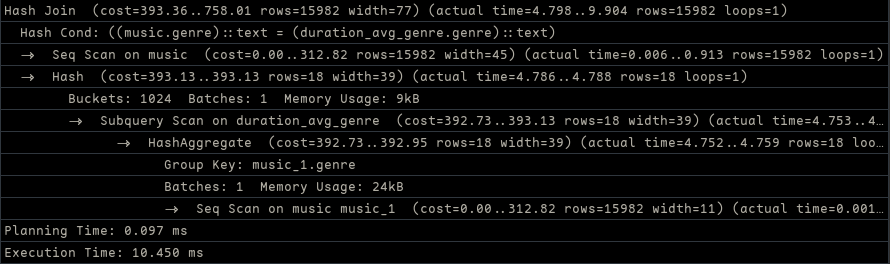


Figura 36: WITH

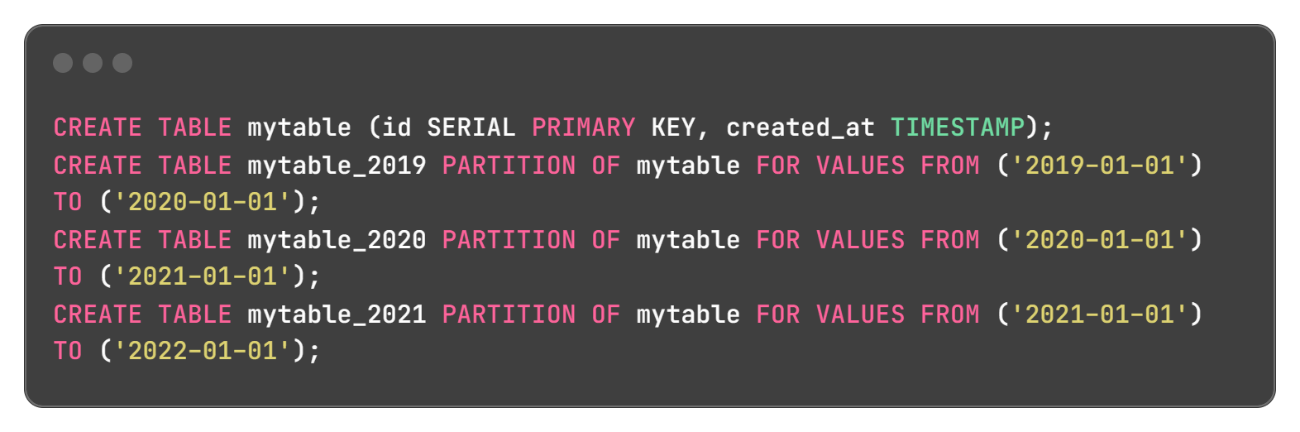


Figura 37: Partitioning

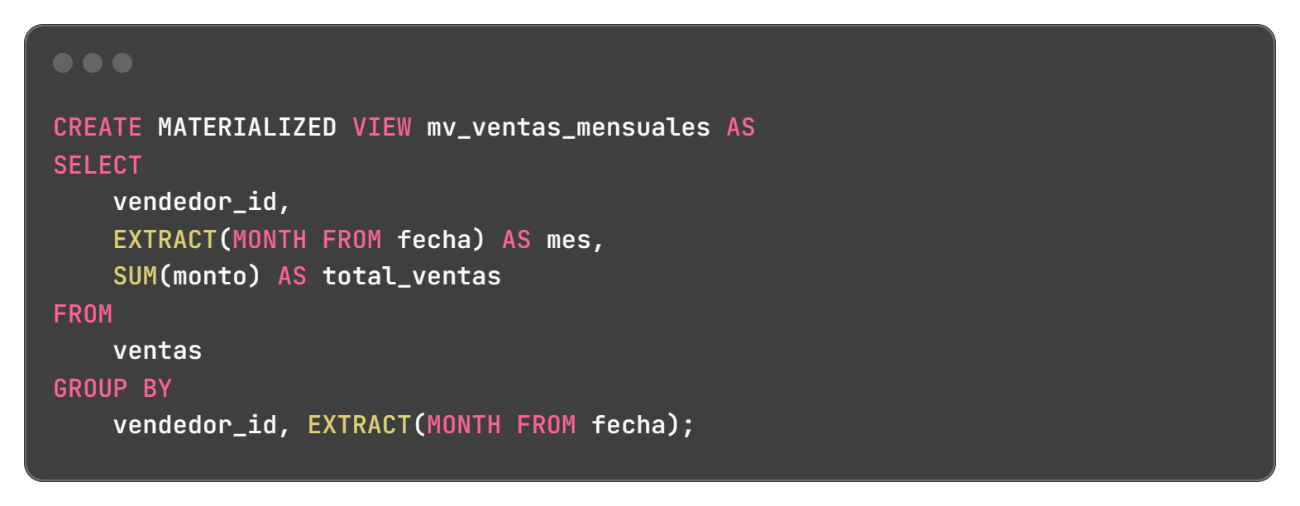


Figura 38: materialized views

**Figura: Búsqueda de texto

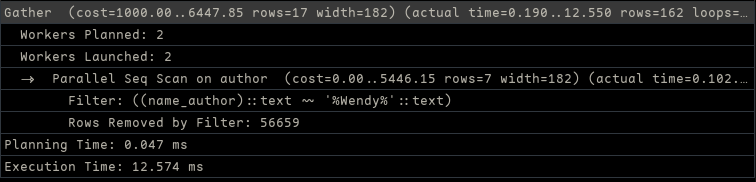


Figura: explain analyse sin índices



Figura: explain analyse con índices método 2

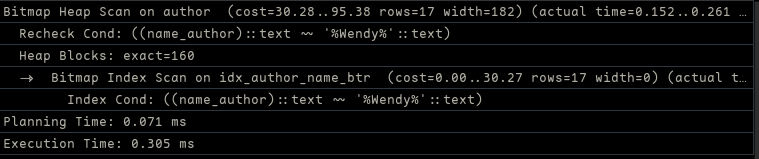


Figura: explain analyse con índices método 3

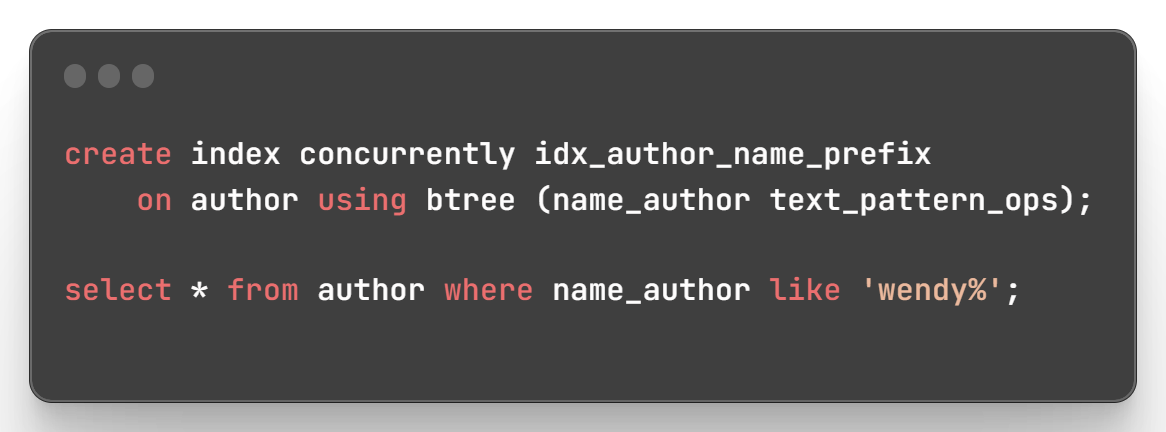


Figura: Búsqueda de texto en la forma substring%

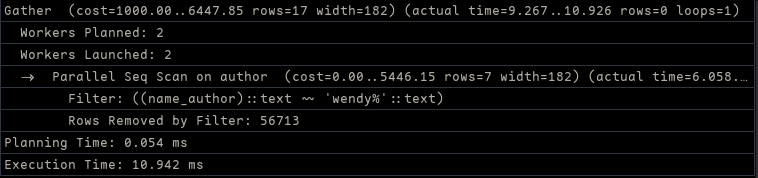
****

Figura: Búsqueda de texto en la forma substring% sin índice

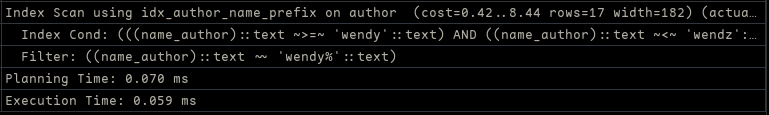
****

Figura: Búsqueda de texto en la forma substring% con índice

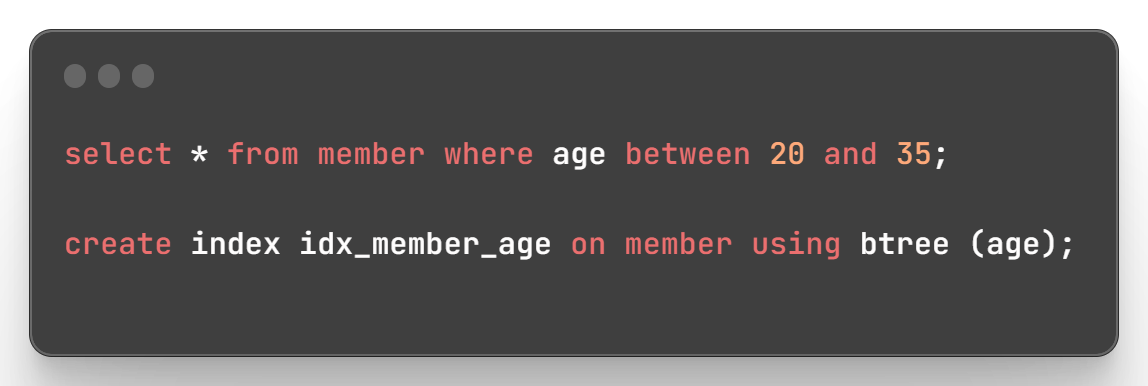
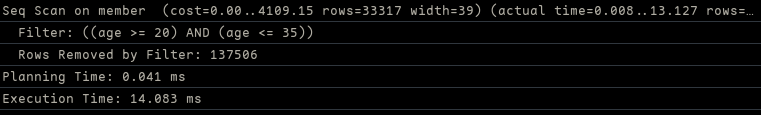
****

Figura: Búsqueda de un intervalo de números

****Figura: Búsqueda de un intervalo de números sin índice

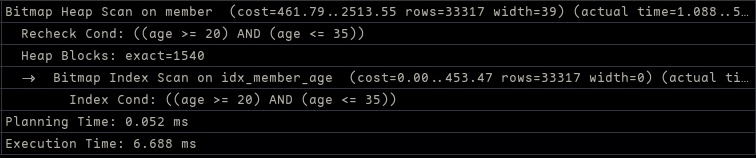


Figura: Búsqueda de un intervalo de números con índice

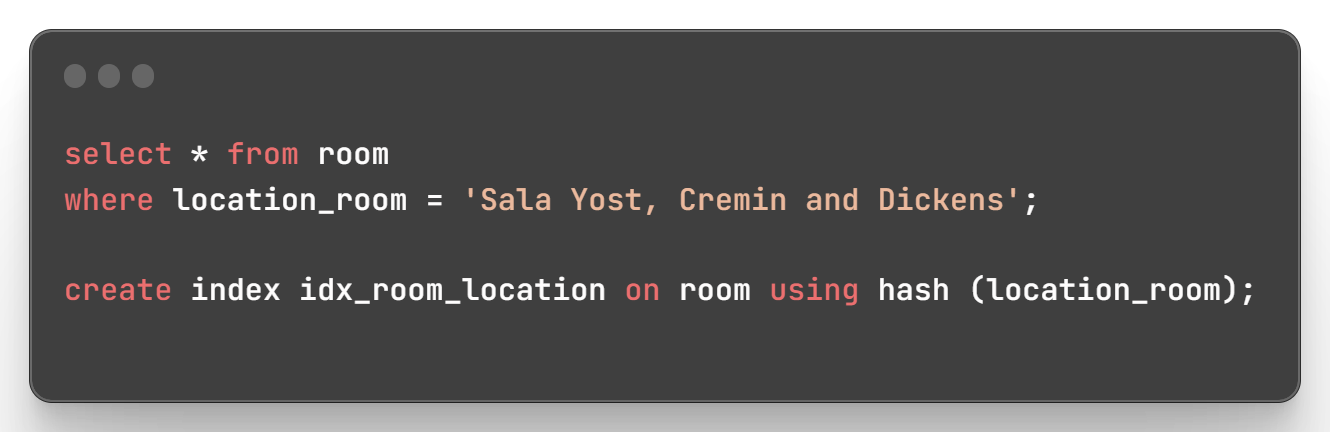
****

Figura: Búsqueda de texto exacto

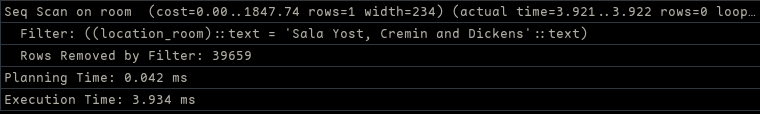
****

Figura: Búsqueda de texto exacto sin índice

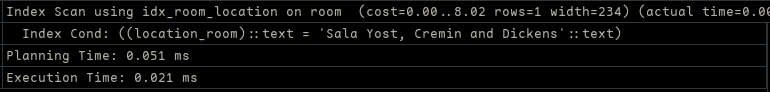


Figura: Búsqueda de texto exacto con índice

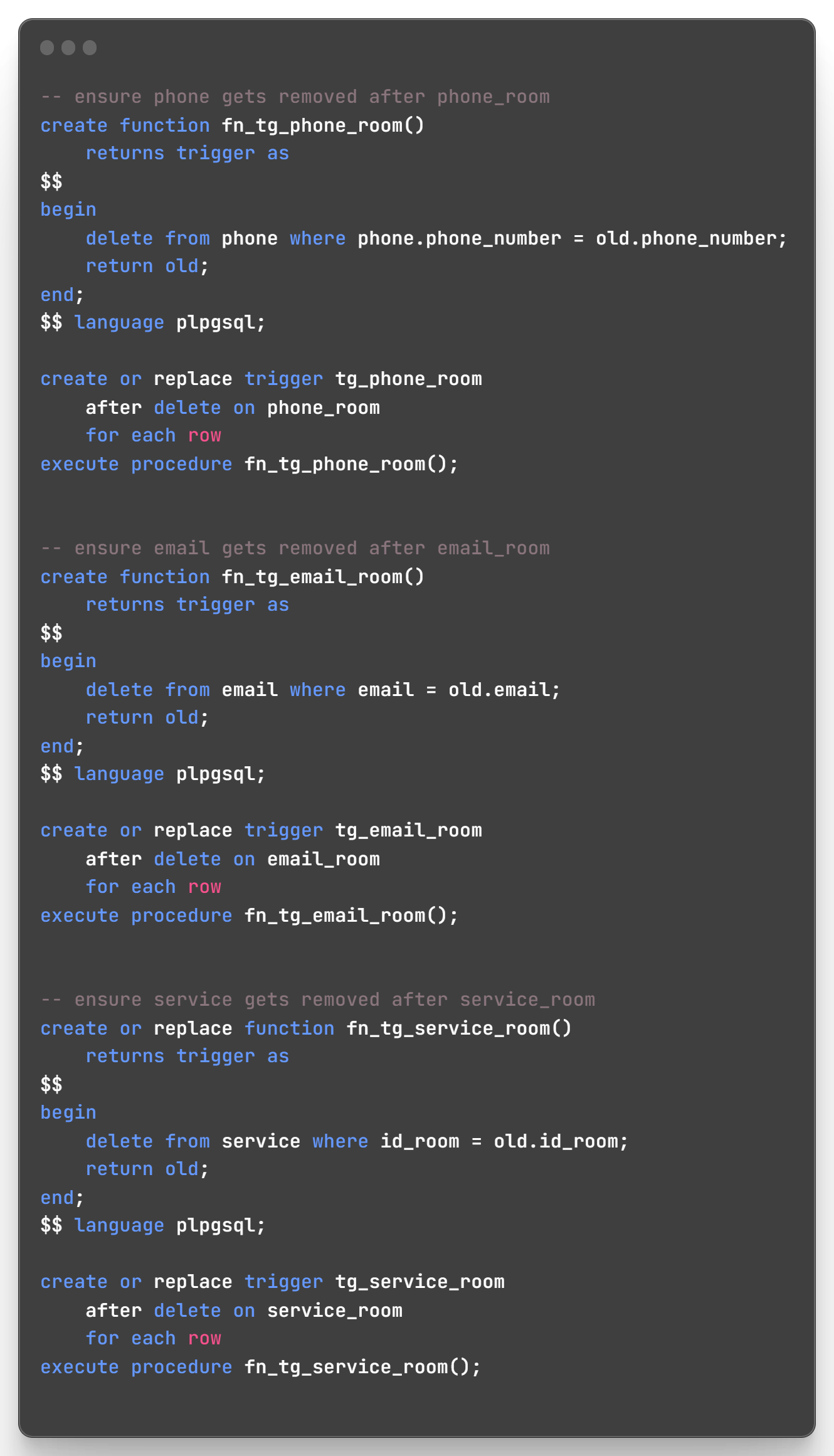
****

Figura: Eliminar después

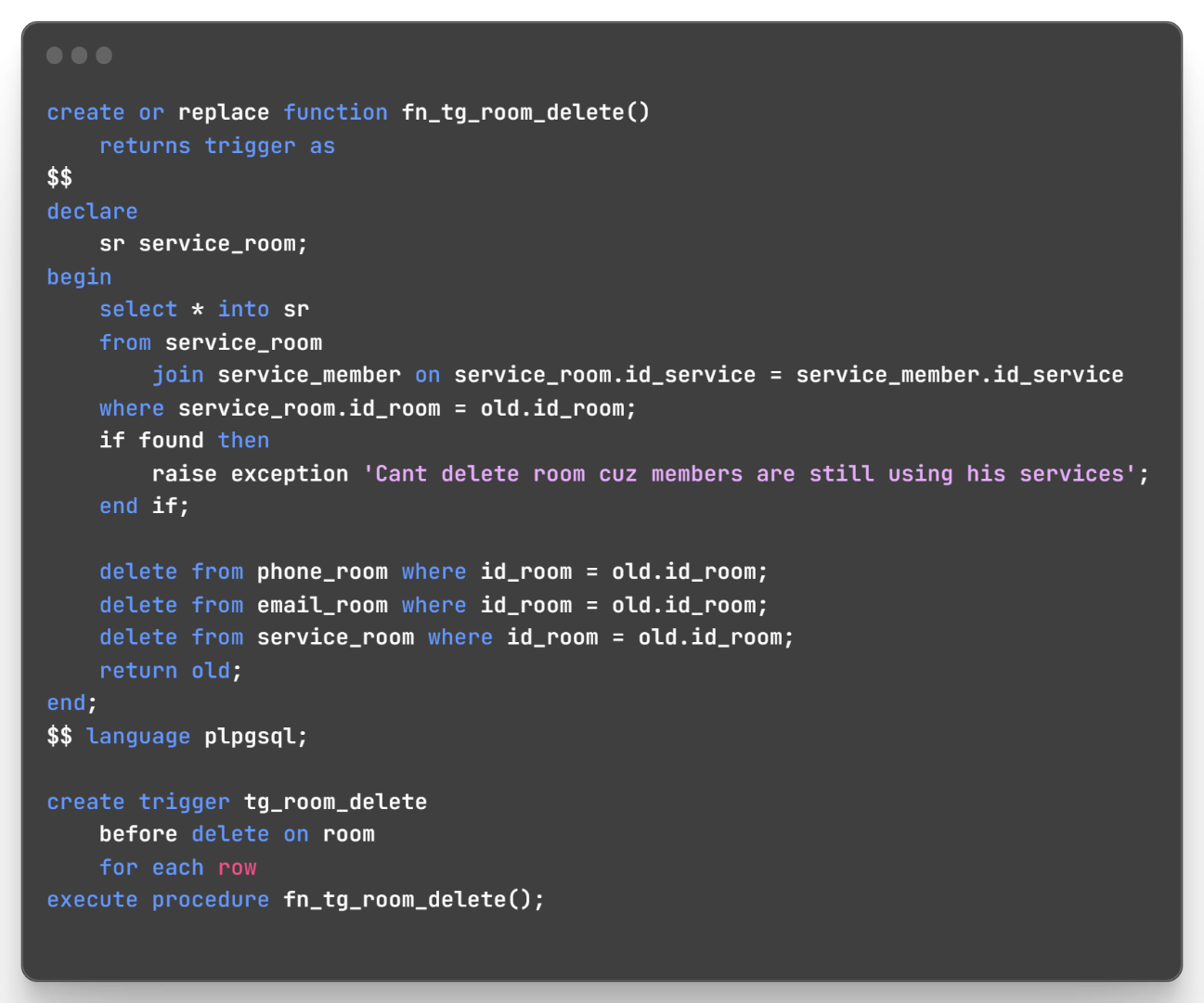
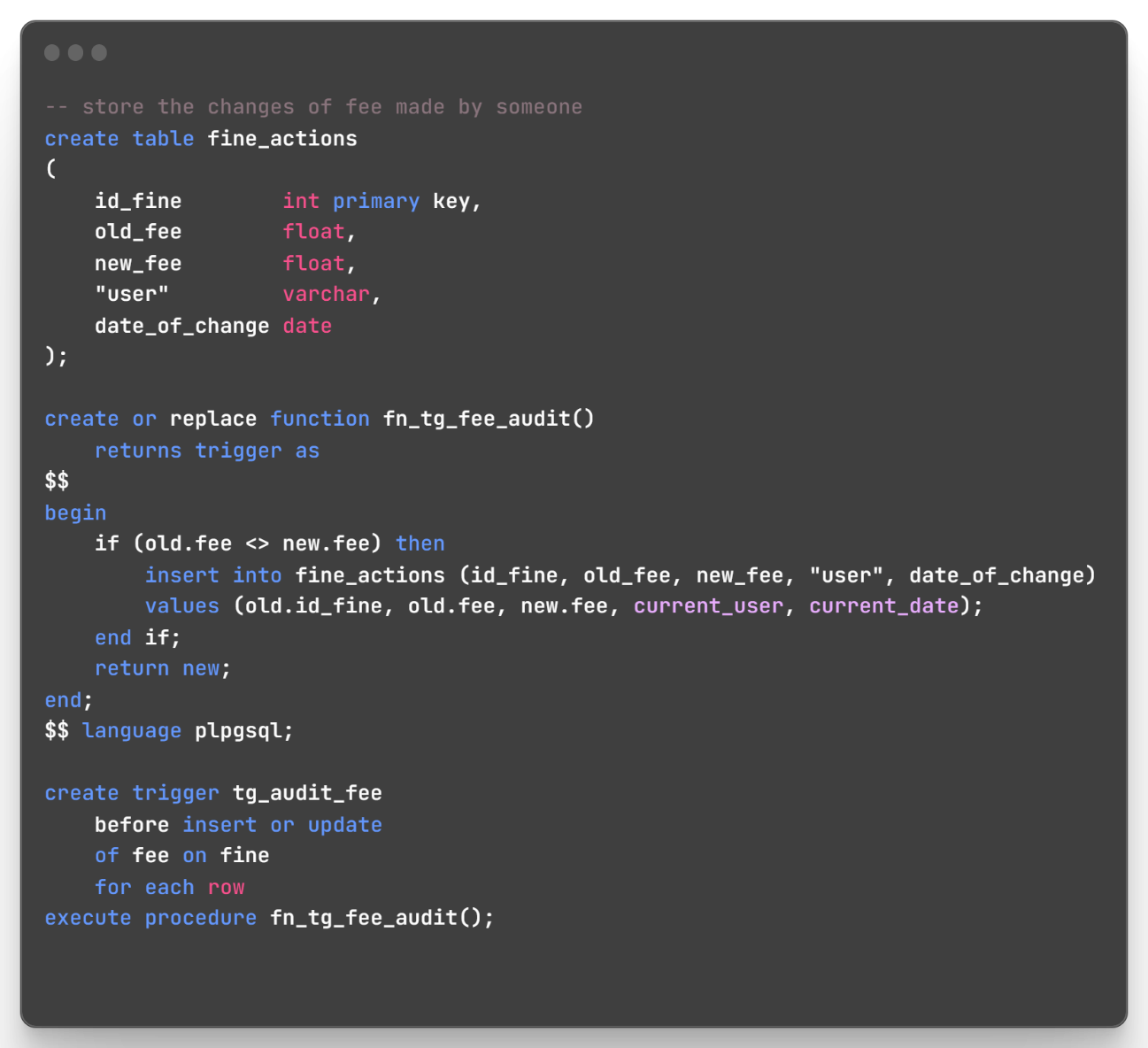
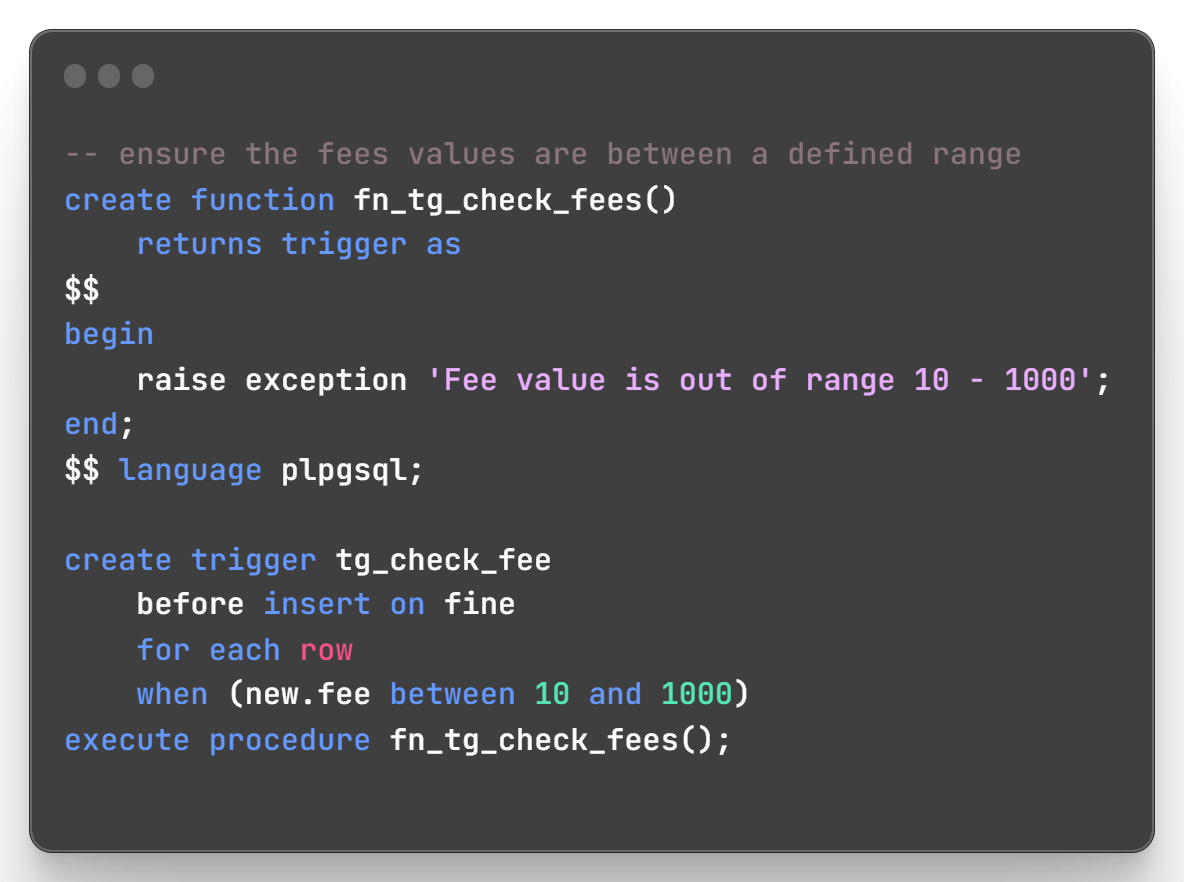
****

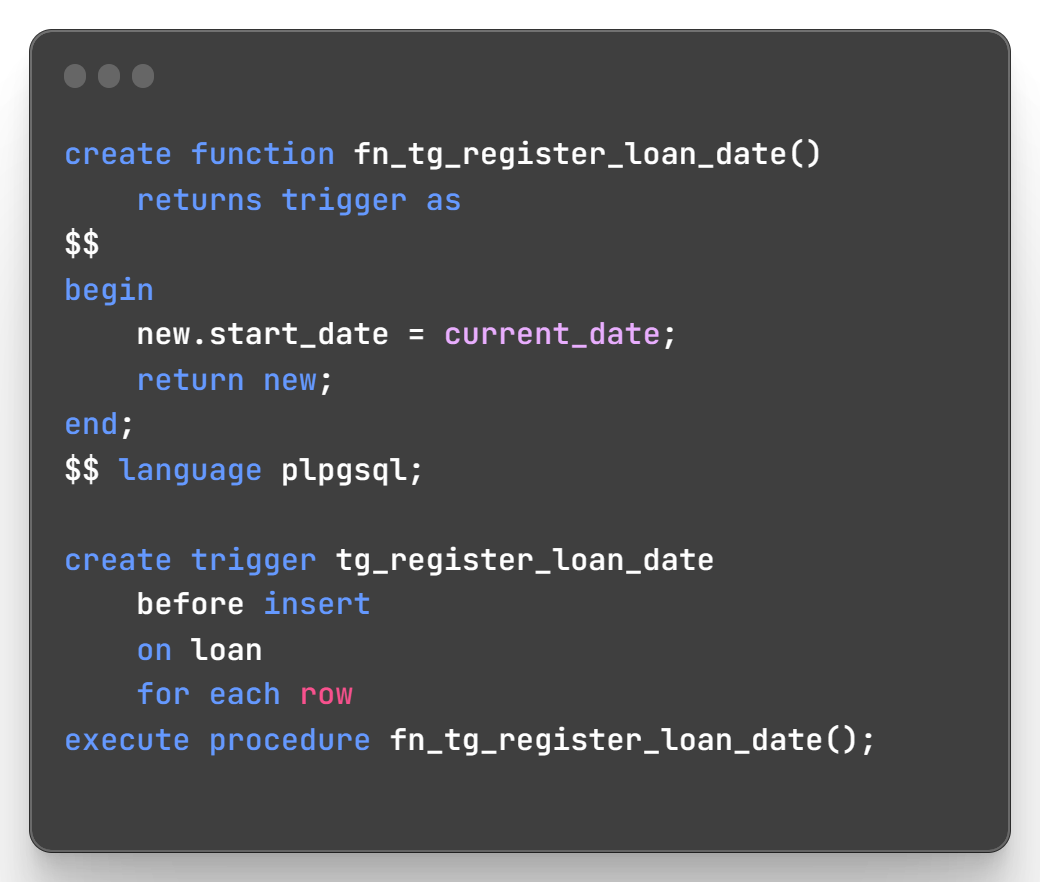
Figura: Eliminar relacionadas

****

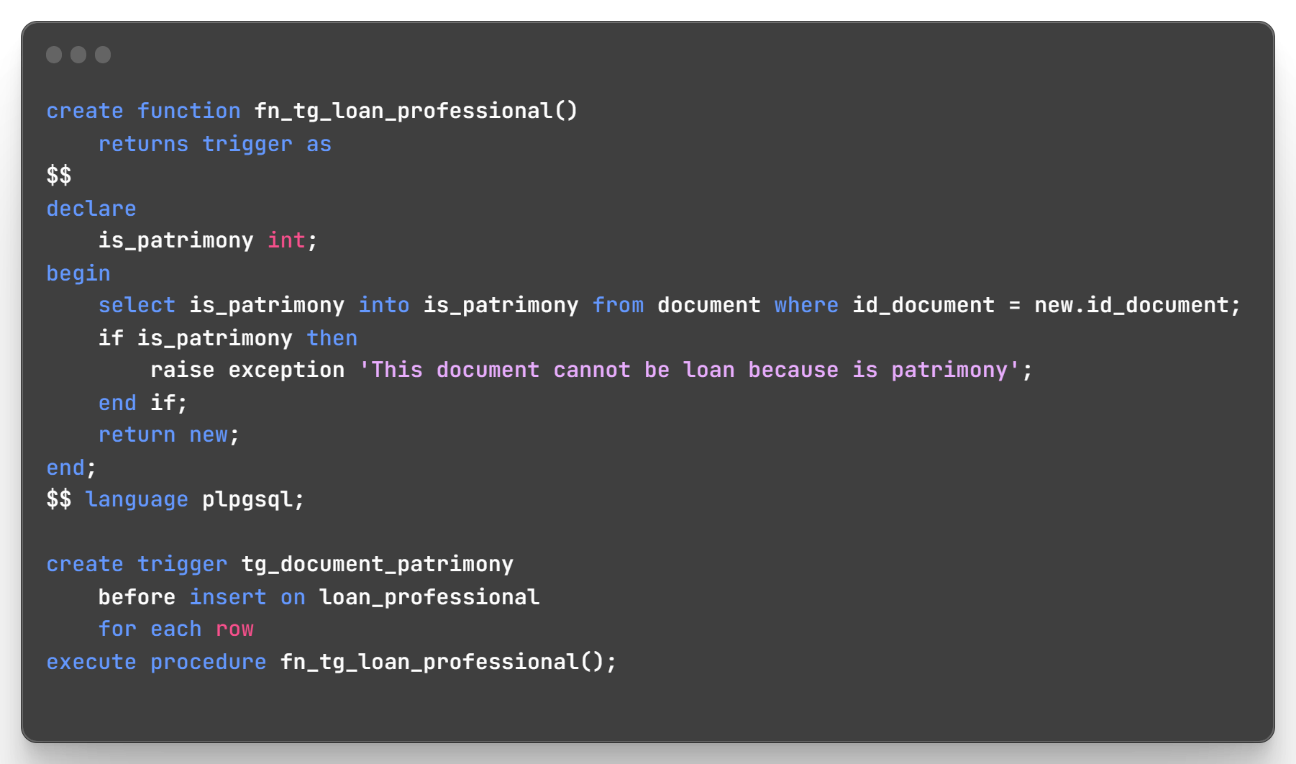
**Figura:**

****

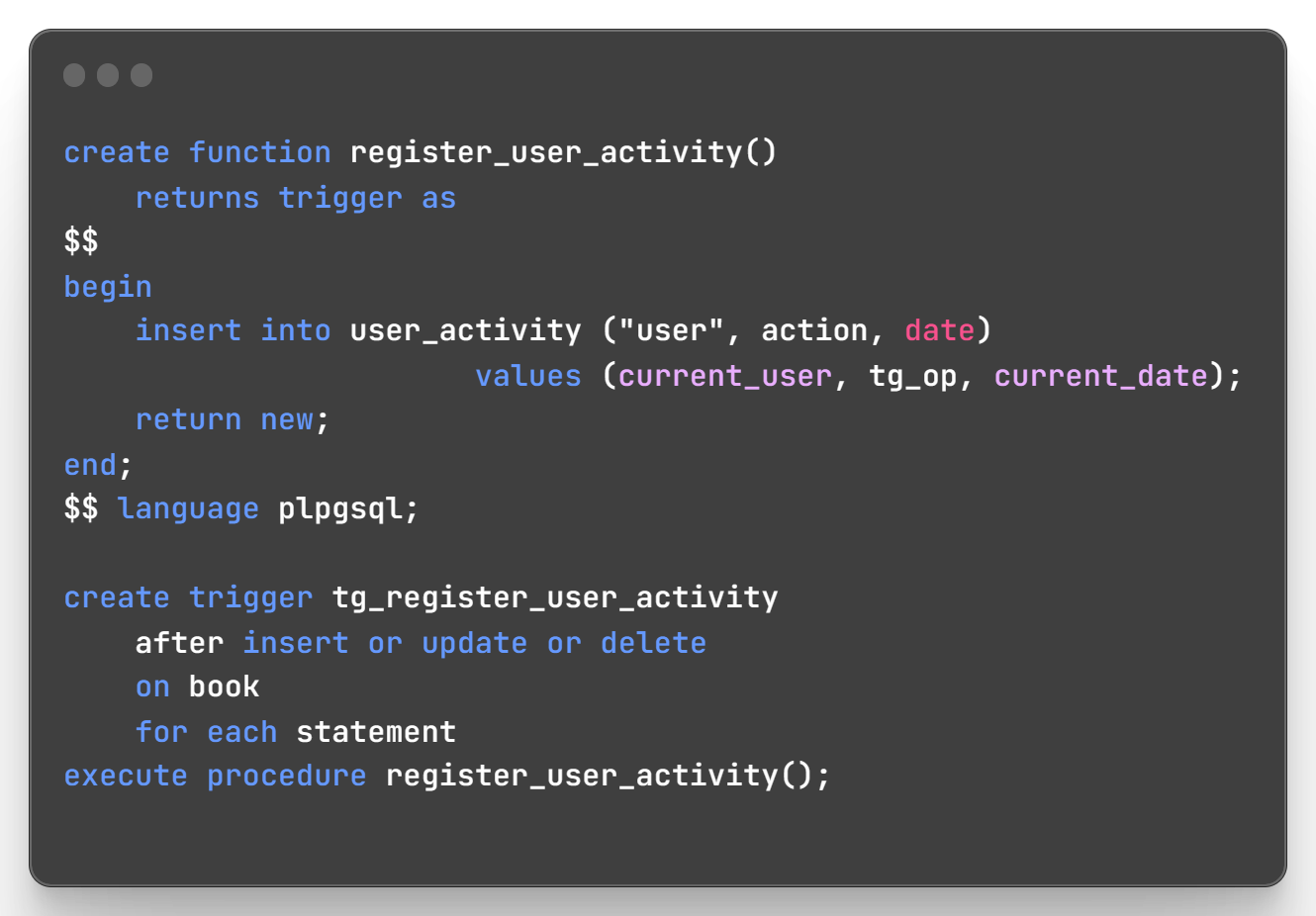
**Figura**

****

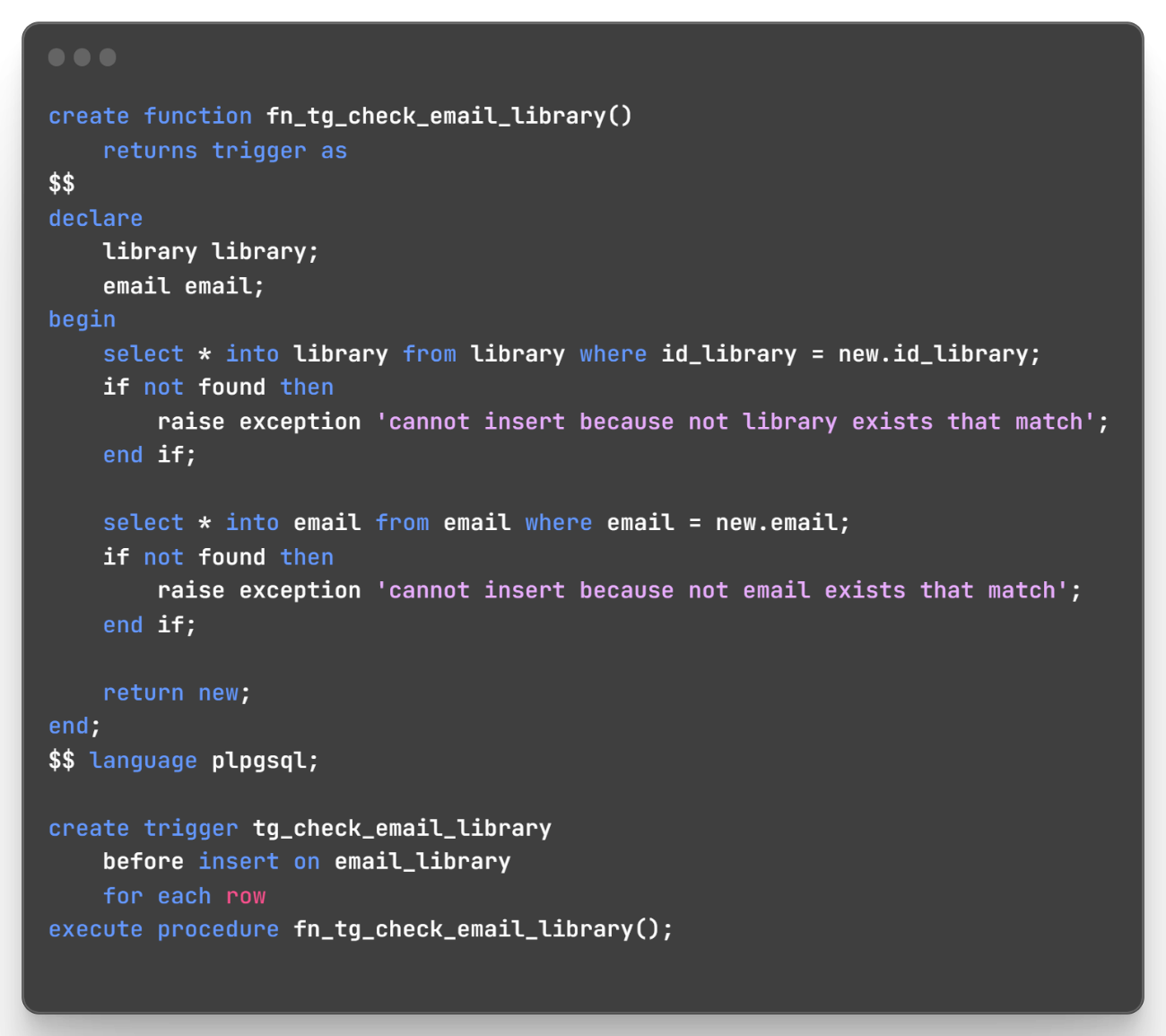
**Figura**

****

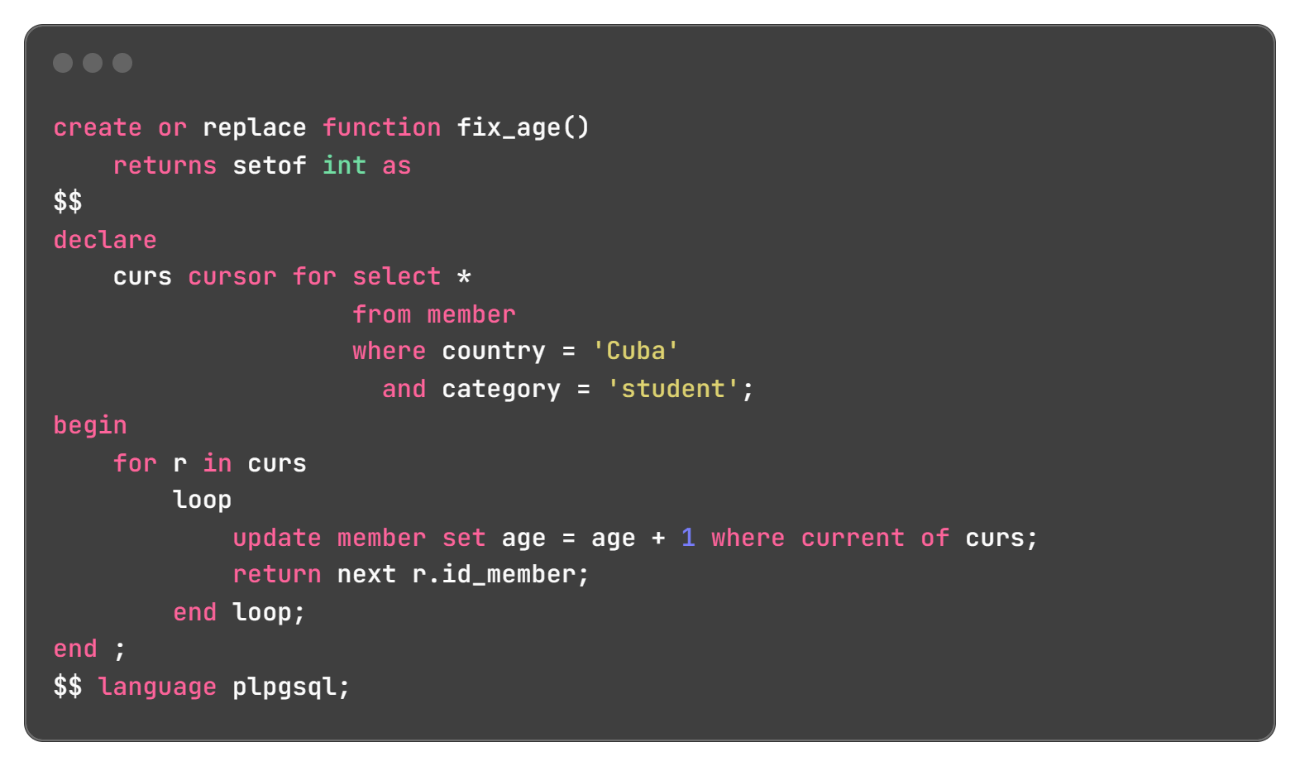
**Figura**

****

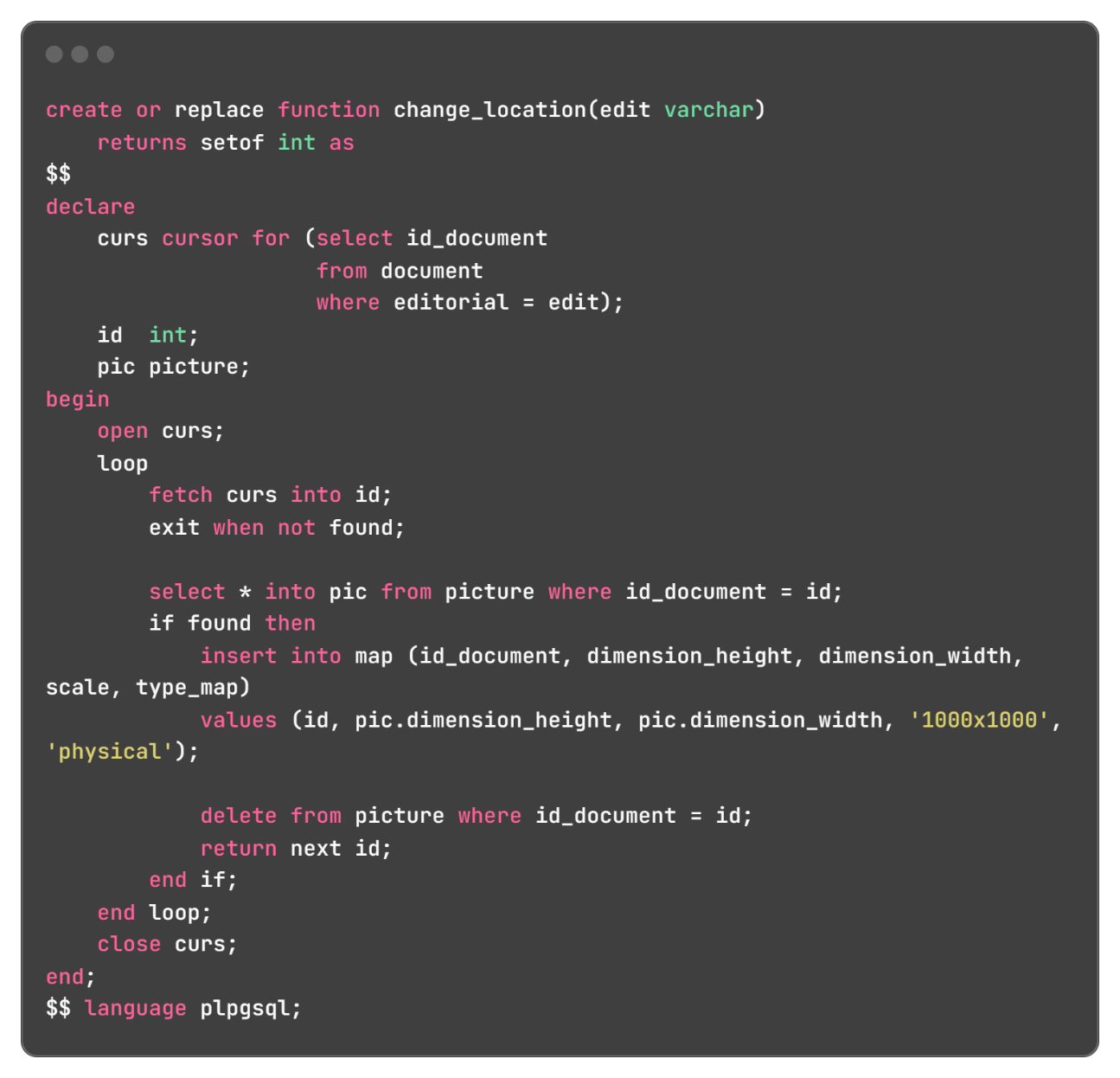
**Figura**

****

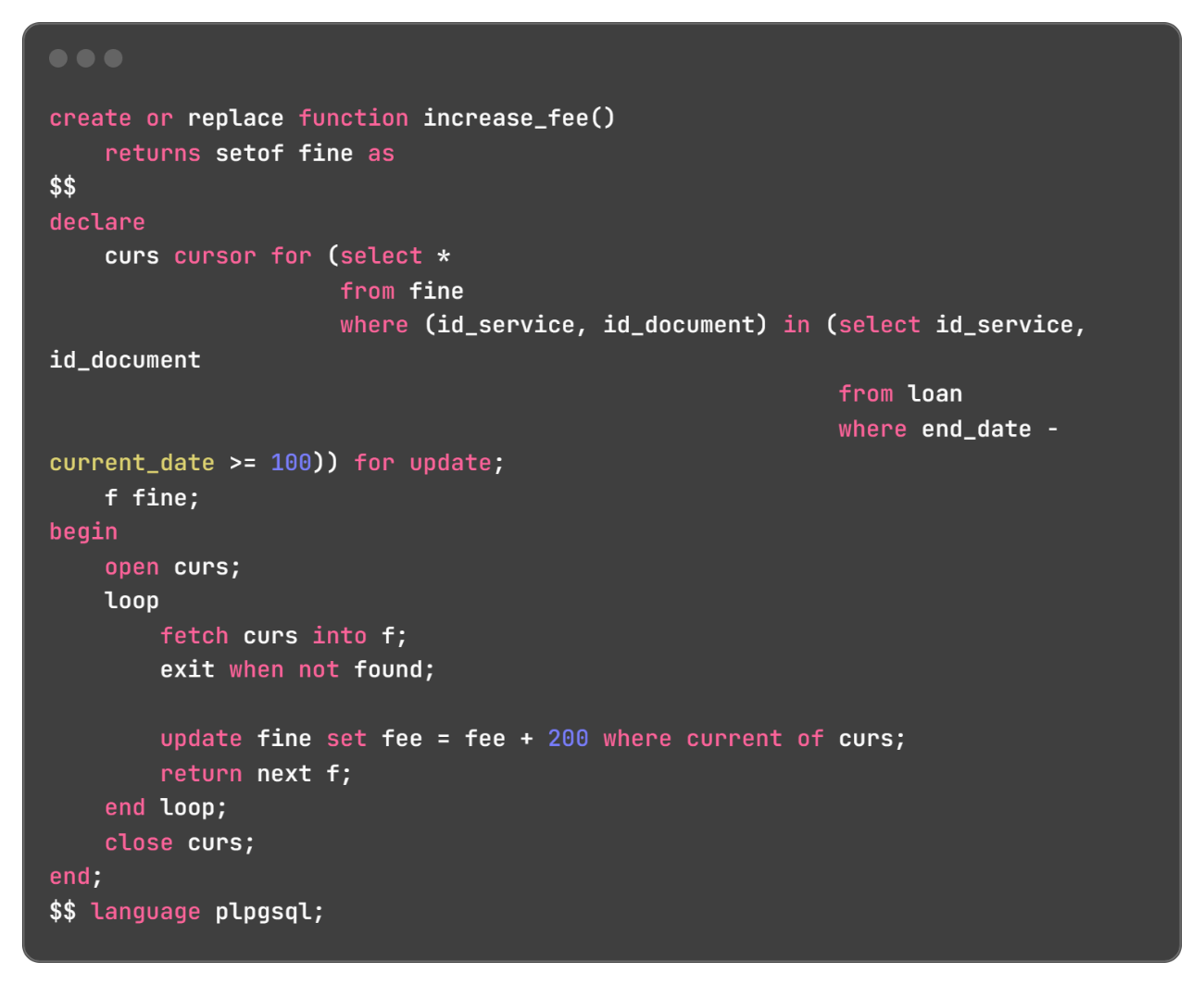
**Figura**



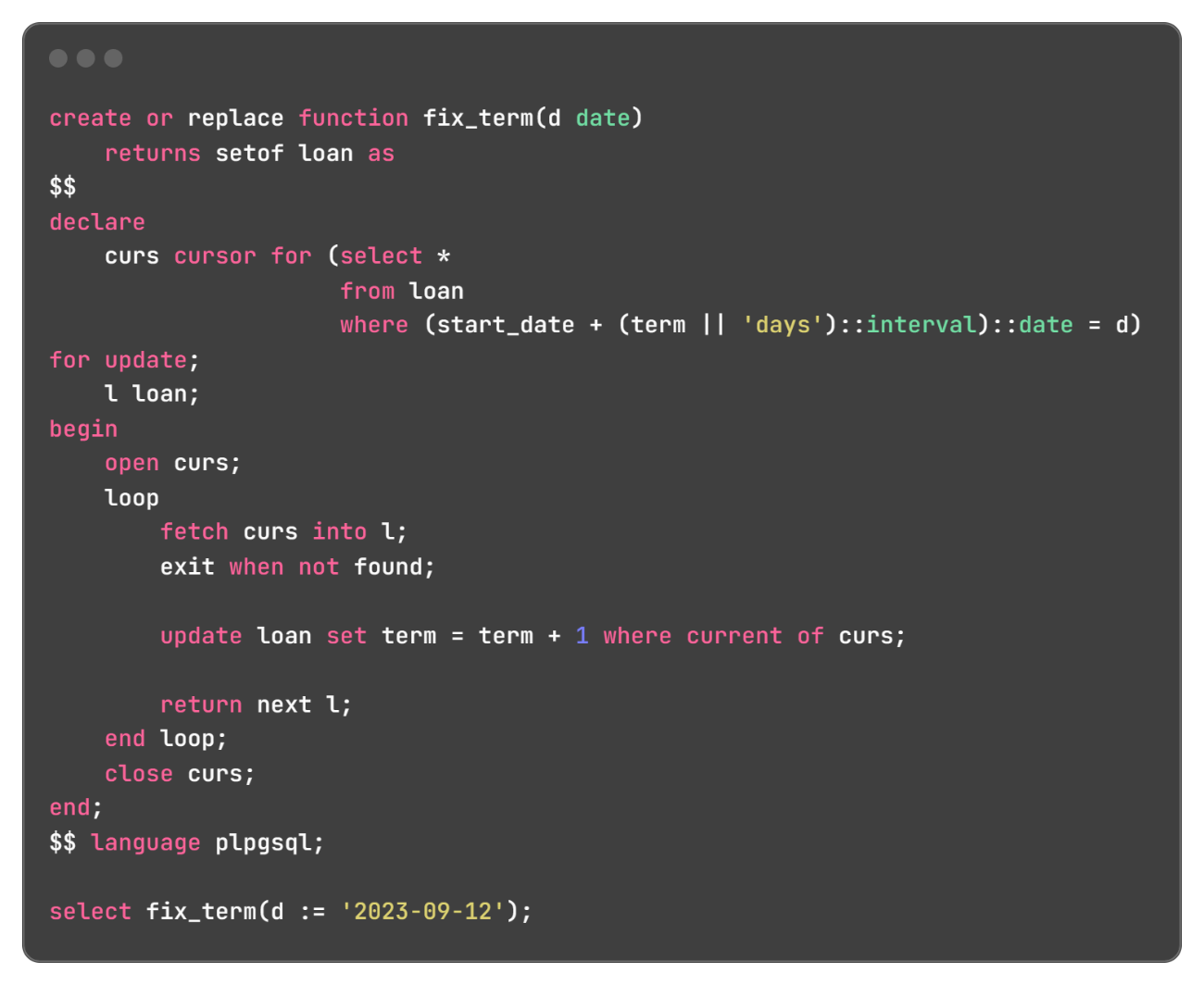
**Figura: Cursores 1**



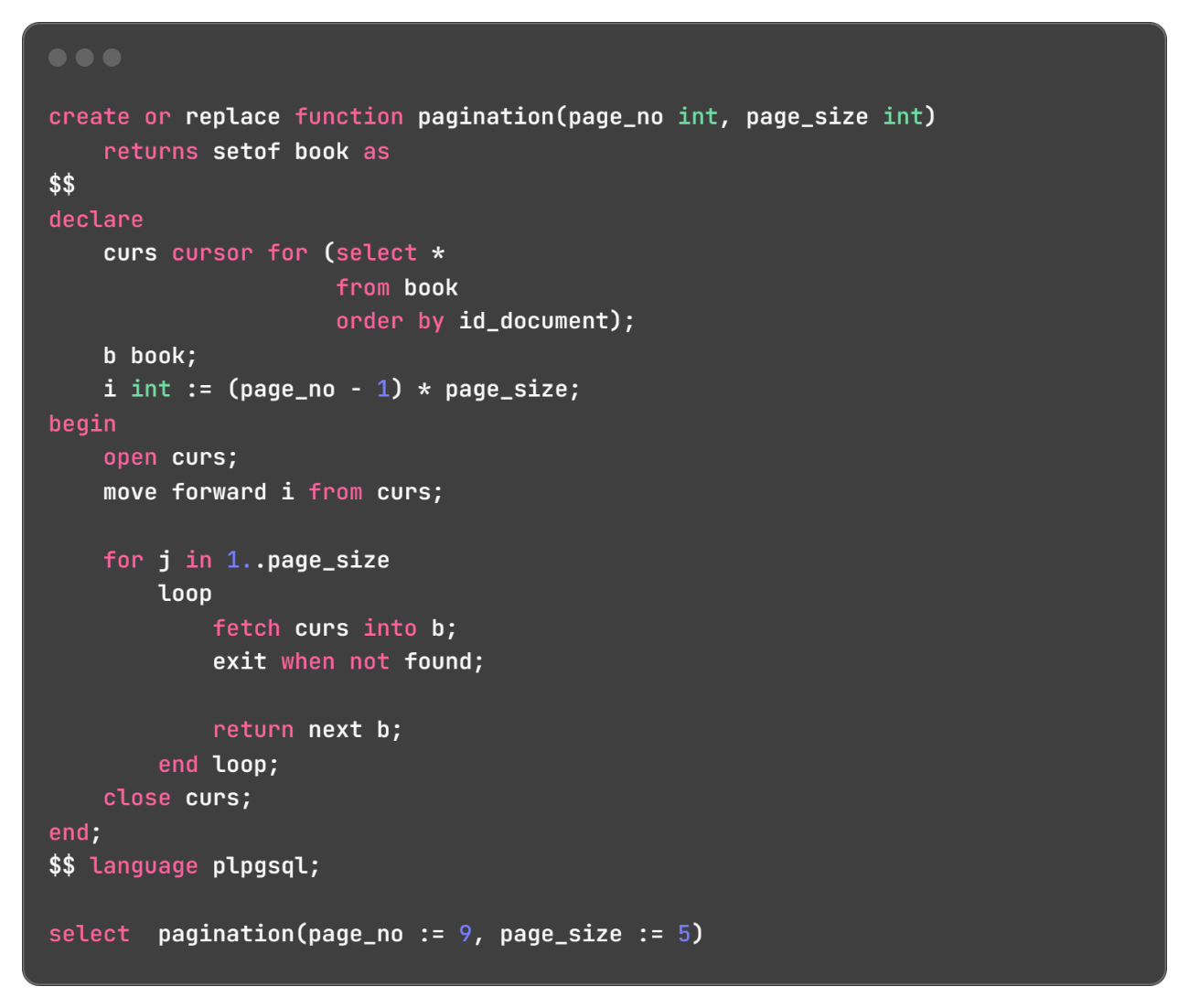
**Figura: Cursores 2**



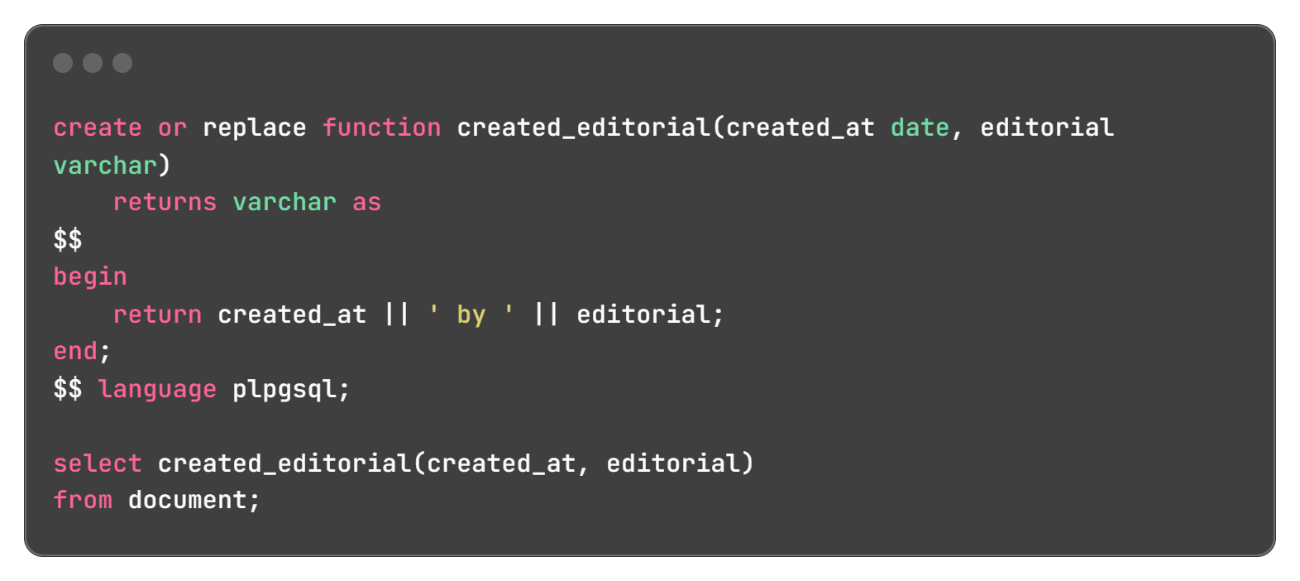
**Figura: Cursores 3**



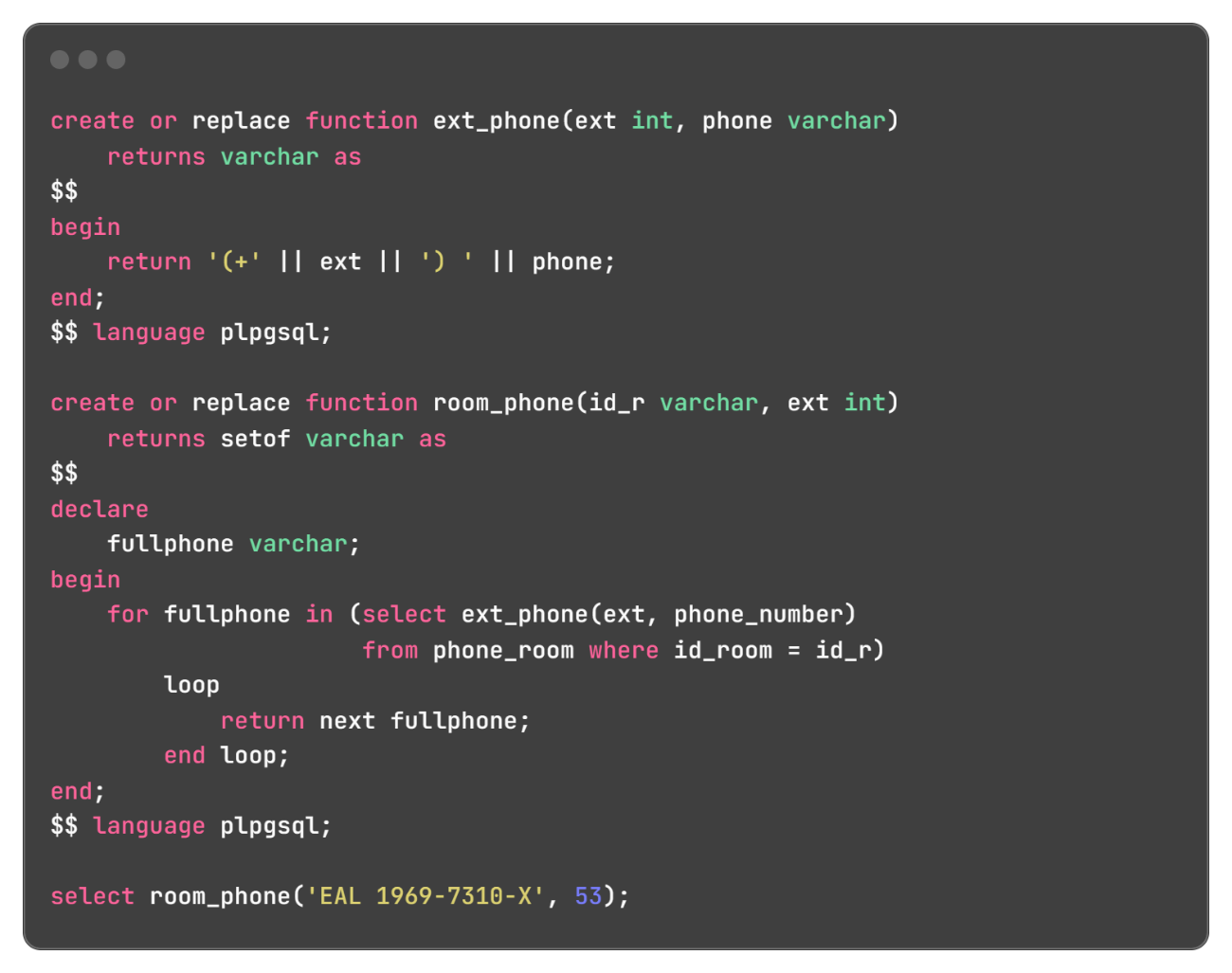
**Figura: Cursores 4**



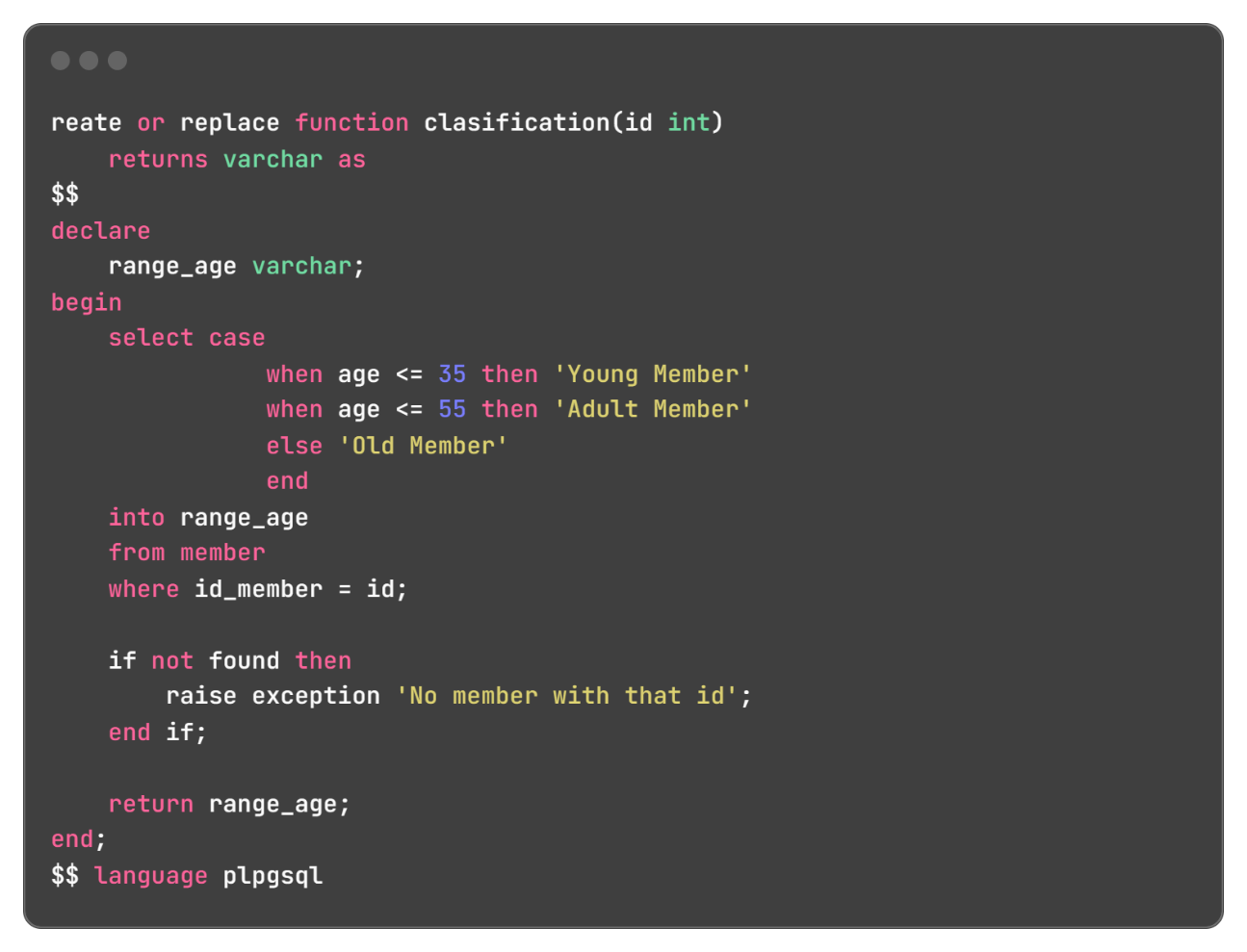
**Figura: Cursores 5**



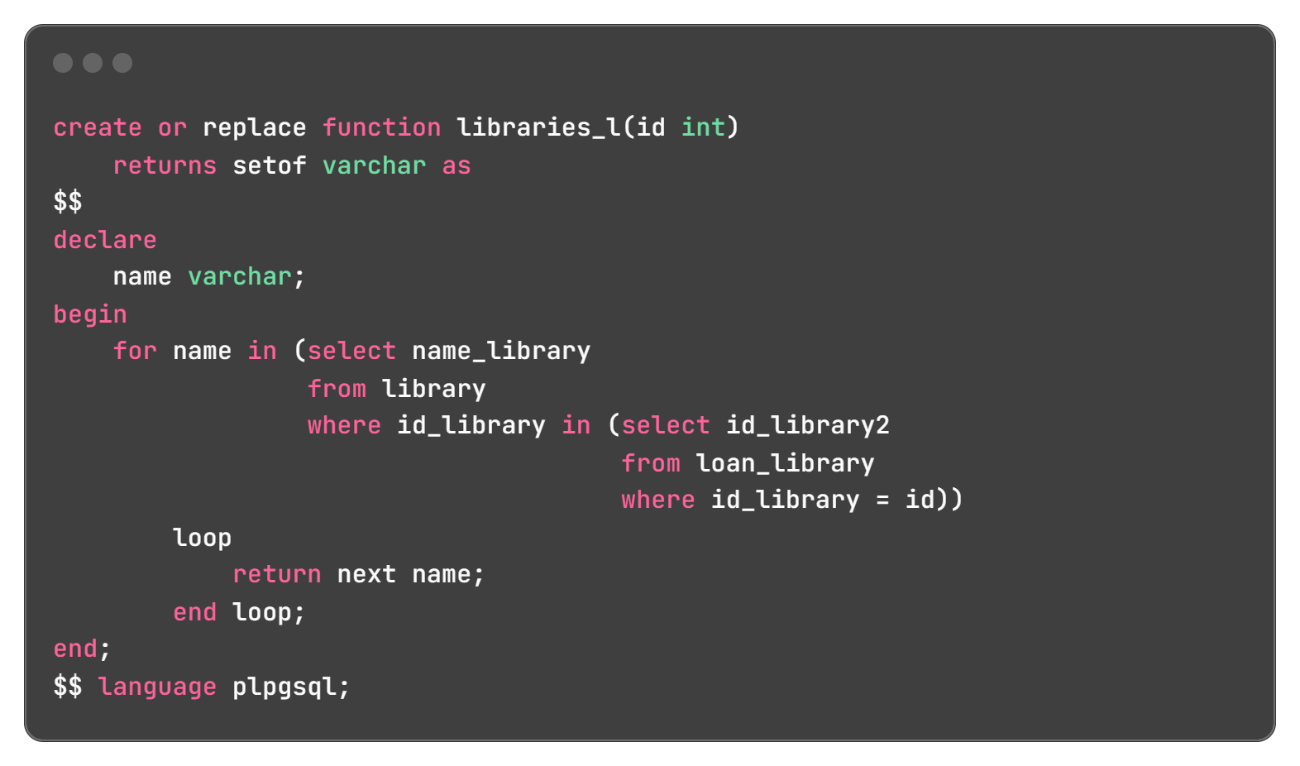
**Figura: Funciones 1**



**Figura: Funciones 2 y 3**

****

**Figura: Funciones 4**



**Figura: Funciones 5**