Titulación: Grado en Ingeniería Informática, Ingeniería en Sistemas de Información e InfoAde
Curso: 2023-2024. Convocatoria Ordinaria de Mayo
Asignatura: Bases de Datos Avanzadas – Laboratorio

Practica 1: Arquitectura PostgreSQL y almacenamiento físico

ALUMNO 1:
Nombre y Apellidos: Sergio Camero Flores
DNI: 09086735X

ALUMNO 2:
Nombre y Apellidos:
DNI:

Fecha:

Profesor Responsable:

Mediante la entrega de este fichero los alumnos aseguran que cumplen con la normativa de autoría de trabajos de la Universidad de Alcalá, y declaran éste como un trabajo original y propio.

En caso de ser detectada copia, se calificará la asignatura como Suspensa – Cero.

Es obligatorio proporcionar una explicación a lo que está ocurriendo en PostgreSQL cuando así se indica en la cuestión. No solo vale poner un pantallazo. La ausencia de una explicación hará que sea invalidada esa cuestión.

Plazos

Trabajo de Laboratorio: semana 29 enero, 5 febrero, 12 febrero, 19 febrero y 26 de febrero.

Entrega de práctica: día 3 de marzo. Aula Virtual

Documento a entregar: un fichero con formato ZIP con las respuestas a las cuestiones planteadas, así como los ficheros de log de postgresql relacionados con la resolución de la práctica y el fichero con el código de generación de los datos de la cuestión 1. El fichero se deberá llamar: `DNIdelosAlumnos_PL1.zip`

AMBOS ALUMNOS DEBEN ENTREGAR EL FICHERO EN LA PLATAFORMA.

Introducción

En esta primera práctica se introduce el sistema gestor de bases de datos PostgreSQL (16.1 la última). Está compuesto básicamente de un motor servidor y de una serie de clientes que acceden al servidor y de otras herramientas externas. En esta primera práctica se entrará a fondo en la arquitectura de PostgreSQL, sobre todo en el almacenamiento físico de los datos y del acceso a los mismos. Antes de comenzar es obligatorio configurar lo que se comenta en la cuestión. Hay que resolver la práctica con consultas SQL.

Cuestión 0. Configurar el fichero de Error Reporting and Logging de PostgreSQL para que aparezcan recogidas las sentencias SQL DDL (Lenguaje de Definición de Datos) + DML (Lenguaje de Manipulación de Datos) generadas en dicho fichero. No se pide activar todas las sentencias. No activar la duración de la consulta. También se debe de configurar el log para que en el comienzo de la línea de registro de la información del log ("line prefix") aparezcan vuestros DNI's y el nombre del host con su puerto. ¿Cómo se ha realizado la configuración?

Organización de Archivos en PostgreSQL

Cuestión 1. Crear una nueva Base de Datos que se llame **PL1**. Después crear una tabla **camiones** con los siguientes campos:

- id_camion: que debe ser el identificador del camión comenzando por 1.
- Matricula: guarda la matrícula española del camión y no se debe repetir.
- Empresa: guarda la empresa de transportes a la que pertenece el camión.
- Kilómetros: guarda los kilómetros que tiene actualmente el camión.

Crear un programa que permita generar 20 millones de registros en un fichero de texto que pueda ser cargado en la tabla (preferiblemente en Python) con las siguientes propiedades para los siguientes campos, cuyos valores se deben generar aleatoriamente.

- Empresa: deben ser valores aleatorios generados de 10000 empresas disponibles. Una de ellas debe ser UPS.
- Kilómetros: deben ser valores aleatorios generados entre 0 y 500000 km.

Cargar los datos en la tabla y localizar los ficheros relacionados con la tabla. ¿cómo se localizan? ¿Cuánto ocupan? ¿por qué?

Cuestión 2. Calcular teóricamente el tamaño en bloques que ocupa la relación **camiones** tal y como se realiza en clase de teoría. ¿Concuerda con el tamaño en bloques que nos proporciona PostgreSQL? ¿Cuál es el factor de bloque medio real de la tabla **camiones**? ¿Por qué? Realizar una consulta SQL que obtenga ese valor y comparar con el factor de bloque teórico.

Cuestión 3. Realizar una consulta que muestre la matrícula de los camiones que tengan 200000 km. ¿Cuántas tuplas se obtienen y cuántos bloques se leen por Postgres? ¿Por qué? Comparar con los resultados obtenidos al aplicar el método visto en teoría.

Cuestión 4. Crear una tabla **camiones2** cuyas tuplas estén ordenadas físicamente por el campo km de menor a mayor y que tenga la misma información. Cargar el mismo fichero de datos creado en la cuestión1. Indicar el proceso de generación de dicha tabla ordenada. ¿Cuántos bloques ocupa la tabla ahora? ¿Hay algún cambio? ¿Por qué?

Cuestión 5. Repetir la cuestión 3 sobre la tabla **camiones2** y comparar los resultados obtenidos indicando las conclusiones obtenidas.

Cuestión 6. Borrar 2000000 tuplas de la tabla **camiones** de manera aleatoria usando el valor del campo id_camion. ¿Qué es lo que ocurre físicamente en la base de datos? ¿Se observa algún cambio en el tamaño de la tabla y estructuras asociadas a ella? ¿Por qué? Adjuntar el código de borrado.

Cuestión 7. En la situación anterior, ¿Qué operaciones se pueden aplicar a la base de datos **PL1** para optimizar el rendimiento de esta? Aplicarlas de tal manera que se recupere el mayor espacio posible. Comentar cuál es el resultado final y qué es lo que ocurre físicamente.

Cuestión 8. Crear una nueva tabla denominada **camiones3** con los mismos campos que la cuestión 1 y que esté particionada por el campo kilómetros con la función hash **kilómetros mod 20**. Insertar los datos del fichero de datos generado en la cuestión 1. Explicar el proceso seguido y comentar qué es lo que ha ocurrido físicamente en la base de datos. ¿Cuándo será útil el particionamiento? ¿Cuántos bloques ocupa cada una de las particiones? ¿Por qué? Comparar con el número bloques que se obtendría teóricamente utilizando el procedimiento visto en teoría.

Cuestión 9. Repetir la cuestión 3 sobre la tabla **camiones3** y comparar los resultados obtenidos con lo visto anteriormente en las tablas **camiones** y **camiones2** obteniendo conclusiones sobre el método de partición.

Indexación de PostgreSQL

PostgreSQL soporta indexación definida por el usuario para ayudar a acelerar ciertas consultas. Entre otros tipos de índices soporta árboles y hash. En este apartado se va a trabajar sobre ambos tipos de índices, pudiendo observar cómo se organizan internamente y su funcionamiento.

Cuestión 10. Borrar todas las tablas **camiones, camiones2** y **camiones3**. Crear una nueva tabla que se llama **camiones** como en la cuestión 1 y que tenga cargados todos los datos del fichero de texto generado.

Cuestión 11. Crear un índice de tipo árbol para kilómetros. ¿Dónde se almacena físicamente ese índice? ¿Qué tamaño tiene? ¿Cuántos bloques tiene? ¿Cuántos niveles tiene? ¿Cuántos bloques tiene por nivel? ¿Cuántas tuplas tiene un bloque de cada nivel? Indicar el procedimiento seguido e incluir el código SQL utilizado.

Cuestión 12. Determinar el tamaño de bloques que teóricamente tendría de acuerdo con lo visto en teoría y el número de niveles. Comparar los resultados obtenidos teóricamente con los resultados obtenidos en la cuestión 11.

Cuestión 13. Crear un índice de tipo hash para el campo kilómetros. Dónde se almacena físicamente ese índice? ¿Qué tamaño tiene? ¿Cuántos bloques tiene? ¿Cuántos cajones tiene? ¿Cuántas tuplas tiene de media un cajón? Indicar el procedimiento seguido e incluir el código SQL utilizado.

Cuestión 14. Determinar el tamaño de bloques que teóricamente tendría de acuerdo con lo visto en teoría y el número de niveles. Comparar los resultados obtenidos teóricamente con los resultados obtenidos en la cuestión 13.

Cuestión 15. Crear un índice de tipo árbol para el campo matrícula. ¿Dónde se almacena físicamente ese índice? ¿Qué tamaño tiene? ¿Cuántos bloques tiene? ¿Cuántos niveles tiene? ¿Cuántos bloques tiene por nivel? ¿Cuántas tuplas tiene un bloque de cada nivel? Indicar el procedimiento seguido e incluir el código SQL utilizado.

Cuestión 16. Determinar el tamaño de bloques que teóricamente tendría de acuerdo con lo visto en teoría. Comparar los resultados obtenidos teóricamente con los resultados obtenidos en la cuestión 15.

Cuestión 17. Crear un índice de tipo hash para el campo matrícula. ¿Dónde se almacena físicamente ese índice? ¿Qué tamaño tiene? ¿Cuántos bloques tiene? ¿Cuántos cajones tiene? ¿Cuántas tuplas tiene de media un cajón? Indicar el procedimiento seguido e incluir el código SQL utilizado.

Cuestión 18. Determinar el tamaño de bloques que teóricamente tendría de acuerdo con lo visto en teoría. Comparar los resultados obtenidos teóricamente con los resultados obtenidos en la cuestión 17.

Cuestión 19. ¿Qué conclusiones se puede obtener de la gestión y organización de PostgreSQL sobre los dos índices árbol y hash que se han creado y han sido analizados? ¿Por qué? Comparar con lo visto en teoría.

Monitorización de la actividad de la base de datos

En este último apartado se mostrará el acceso a los datos con una serie de consultas sobre la tabla original. En este apartado se pretende mostrar cómo es el acceso a los datos para diferentes tipos de consultas.

PostgreSQL suministra varias vistas estadísticas que se pueden usar para monitorizar los bloques leídos (tipo statio) de cada una de las estructuras creadas en la base de datos. En este apartado se deben usar esas vistas y está prohibido el uso de otro comando para este fin (Table 28.2).

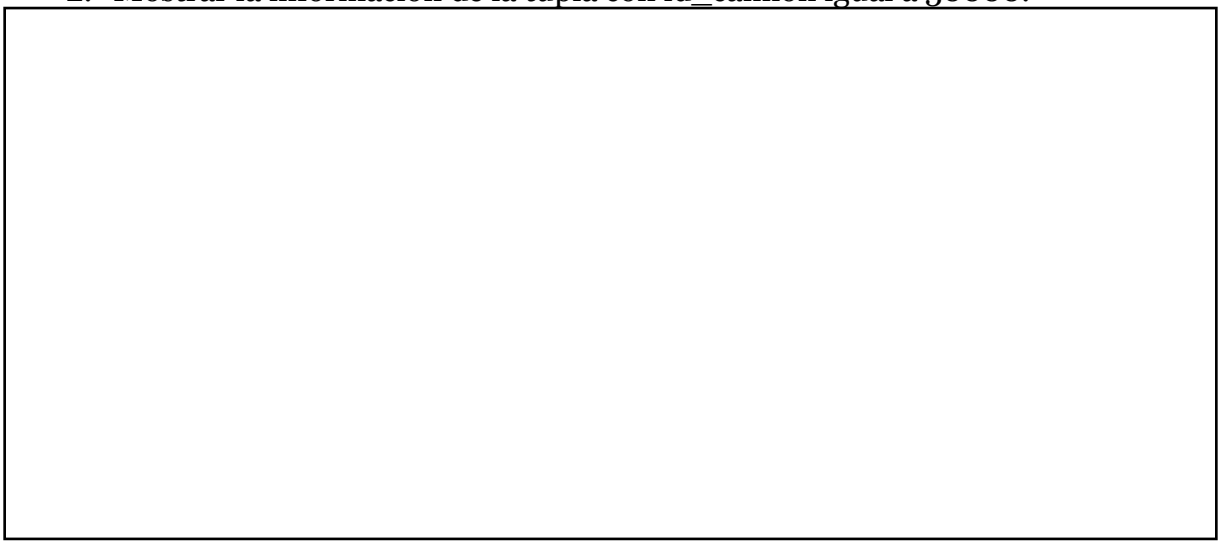
Para ello, borrar todas las tablas creadas y volver a crear la tabla **camiones** como en la cuestión 1. Cargar los datos que se encuentran originalmente en el fichero generado en la cuestión 1.

Cuestión 20. Crear un índice primario tipo árbol sobre el campo kilómetros. También crear un índice hash sobre el campo id_camion y otro sobre kilómetros. ¿Cuál ha sido el proceso seguido para crear cada tipo de índice? Incluir el código SQL utilizado para ello.

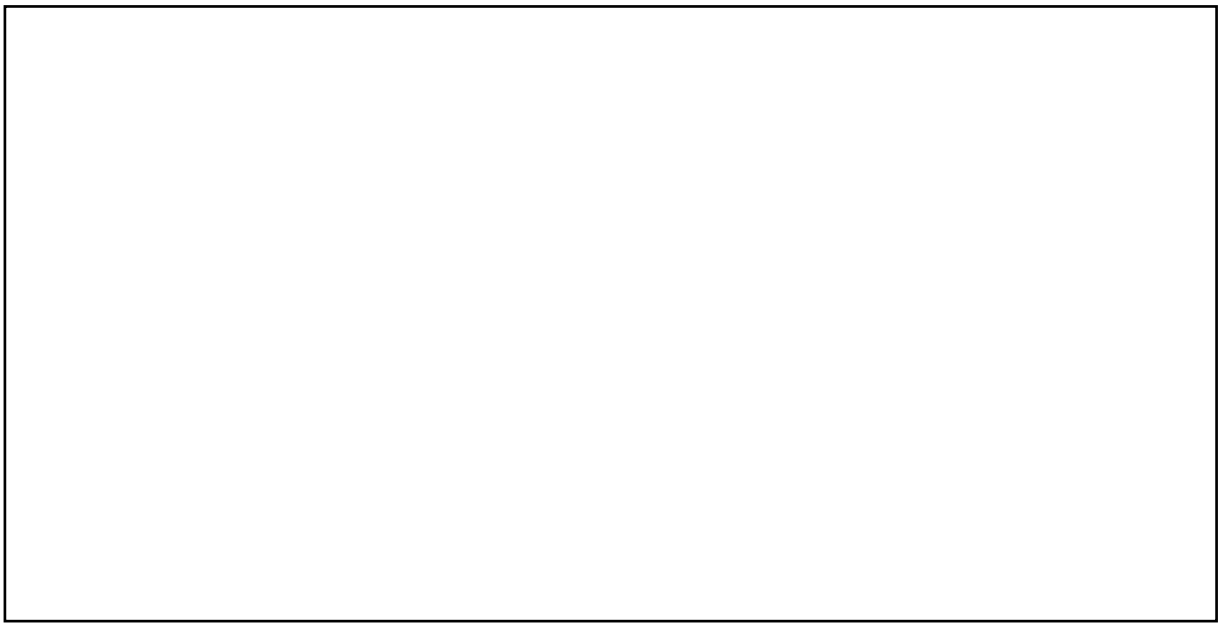
Cuestión 21. Para cada una de las consultas que se muestran a continuación, ¿Qué información se puede obtener de los datos monitorizados por la base de datos al realizar la consulta? Comentar cómo se ha realizado la resolución de la consulta. ¿Cuántos bloques se han leído de cada estructura? ¿Por qué? Comparar con lo visto en teoría. Importante, reinicializar los datos recolectados de la actividad de la base de datos antes de lanzar cada consulta. Se recuerda que solo se pueden usar vistas sobre las estadísticas de la base de datos.

1. Mostrar la información de las tuplas con 50000 km.

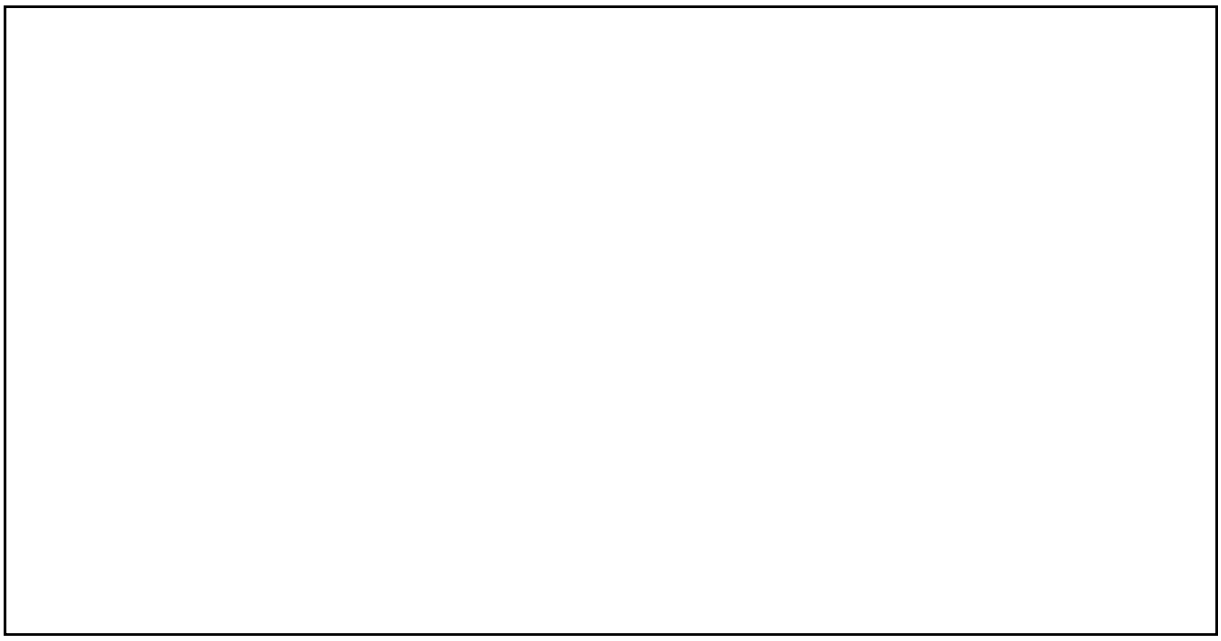
2. Mostrar la información de la tupla con id_camion igual a 30000.



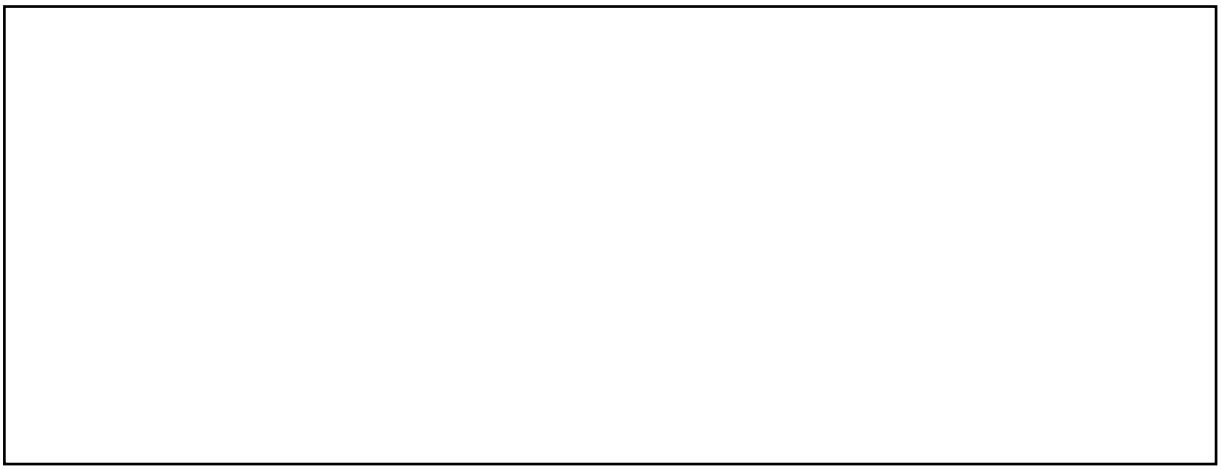
3. Contar el número de camiones que tienen más de 400000 km.



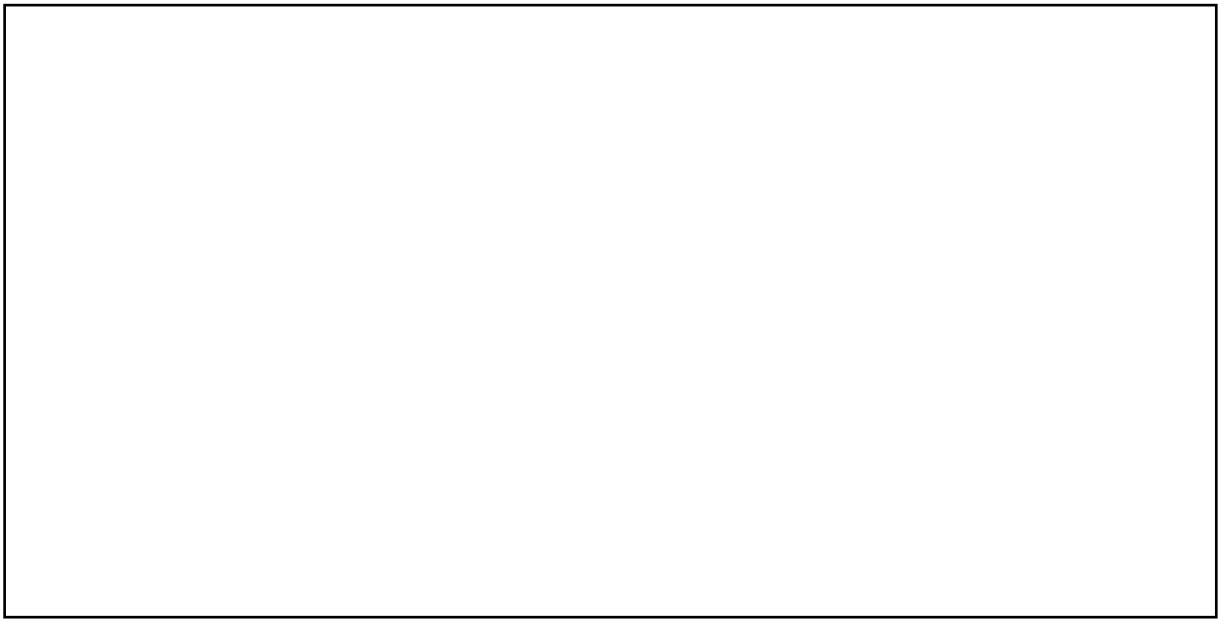
4. Mostrar el número de camiones de cada empresa.



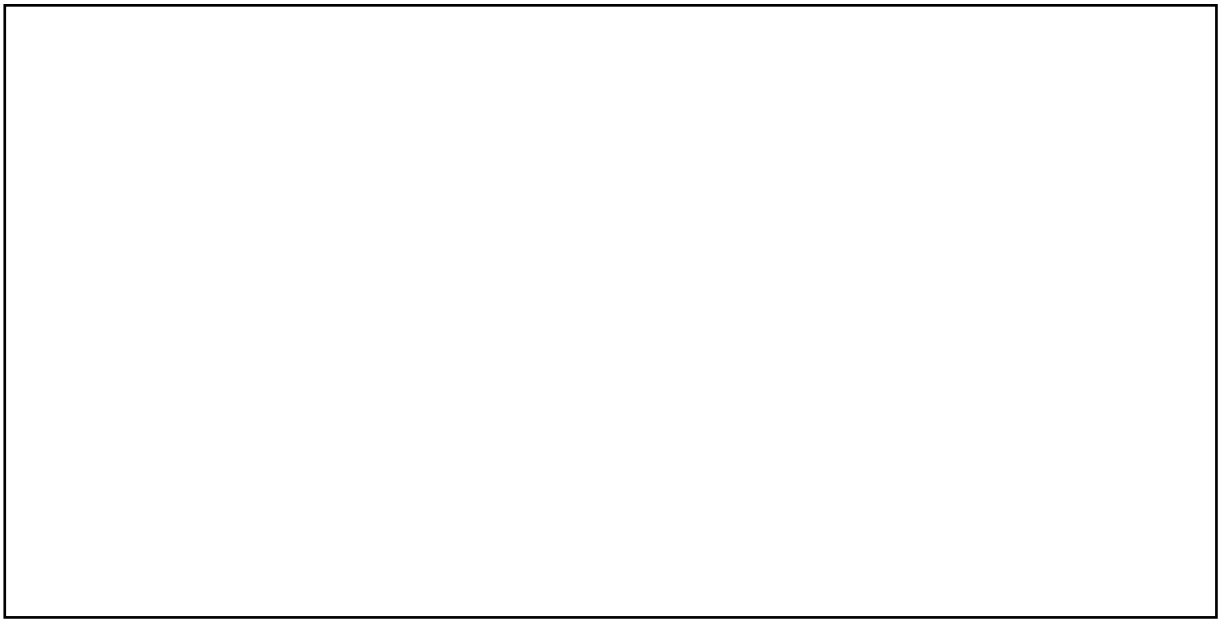
5. Insertar un nuevo camión en la tabla camiones con 30000 km.



6. Actualizar los kilómetros del camión insertado anteriormente para cambiar de 30000 a 20000 km.



7. Mostrar los datos de los camiones que tienen un id_camion entre 80000 y 100000.



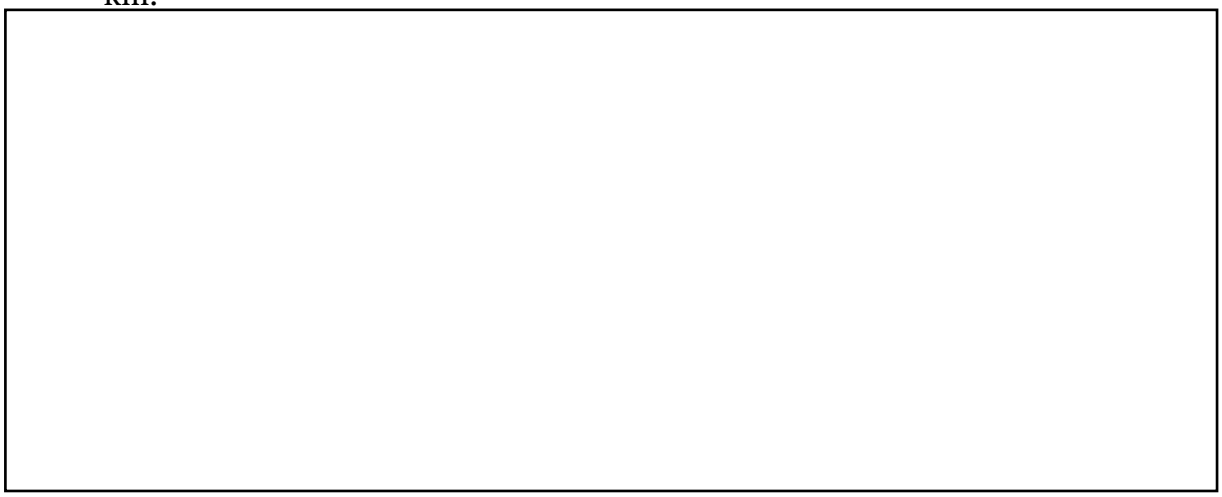
Cuestión 22. Borrar los índices creados en la cuestión 20. Crear un índice multiclave tipo árbol sobre los campos empresa y kilómetros. Incluir el código SQL utilizado para ello.

Cuestión 23. Para cada una de las consultas que se muestran a continuación, ¿Qué información se puede obtener de los datos monitorizados por la base de datos al realizar la consulta? Comentar cómo se ha realizado la resolución de la consulta. ¿Cuántos bloques se han leído de cada estructura? ¿Por qué? Importante, reinicializar los datos recolectados de la actividad de la base de datos antes de lanzar cada consulta:

1. Mostrar el número de camiones que tiene la empresa UPS.

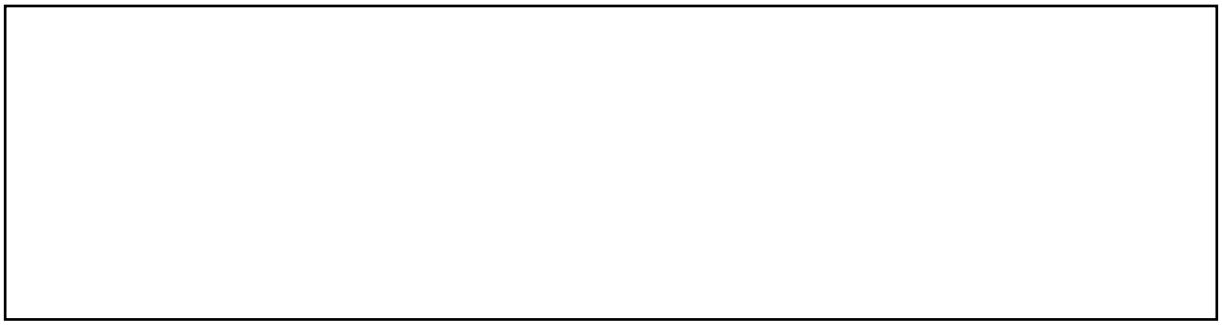
2. Mostrar la información de los camiones que son de la empresa UPS o que tienen 90000 km.

3. Mostrar la información de los camiones de la empresa UPS que tienen 60000 km.

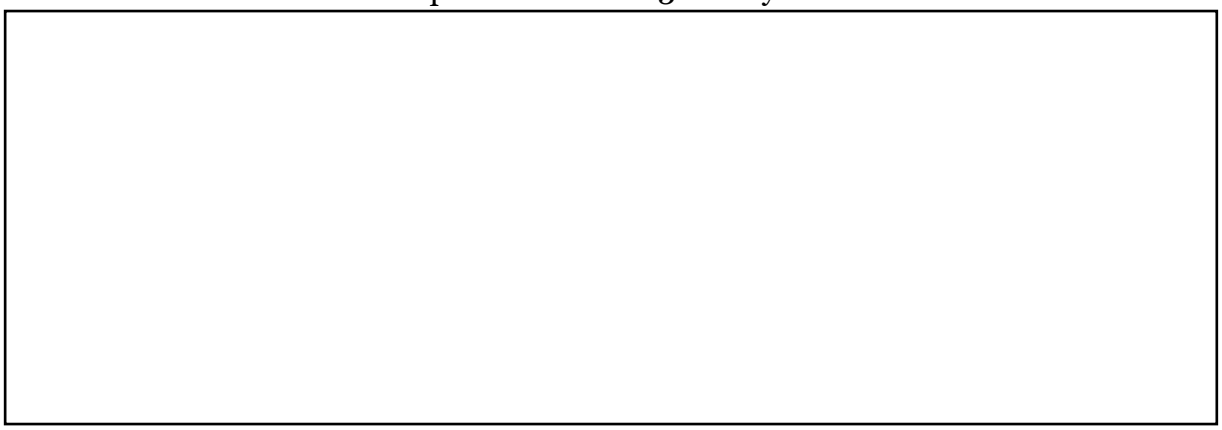


Cuestión 24. Crear la tabla **camiones3** particionada por rangos de 50.000 en 50.000 km hasta un máximo de 500000 km. Para cada una de las consultas que se muestran a continuación, ¿Qué información se puede obtener de los datos monitorizados por la base de datos al realizar la consulta? Comentar cómo se ha realizado la resolución de la consulta. ¿Cuántos bloques se han leído de cada estructura? ¿Por qué? Comparar con la teoría. Importante, reinicializar los datos recolectados de la actividad de la base de datos antes de lanzar cada consulta.

1. Mostrar el número de camiones con más de 600000 km.



2. Mostrar los camiones que tienen entre 30000 y 80000 km.



3. Mostrar los camiones con 400000 km.

Cuestión 25. A la vista de los resultados obtenidos de este apartado, comentar las conclusiones que se pueden obtener del acceso de PostgreSQL a los datos almacenados en disco.

Bibliografía (PostgreSQL 16)

- Capítulo 1: Getting Started.
- Capítulo 5: 5.5 System Columns.
- Capítulo 5: 5.11 Table Partitioning.
- Capítulo 11: Indexes.
- Capítulo 20: Server Configuration.
- Capítulo 25: Routine Database Maintenance Tasks.
- Capítulo 28: Monitoring Database Activity. The statistics Collector
- Capítulo 29: Monitoring Disk Usage. Determining Disk Usage
- Capítulo VI.II: PostgresSQL Client Applications.

- Capítulo VI.III: PostgresSQL Server Applications.
- Capítulo 53: System Catalogs.
- Capítulo 67: B-Tree Indexes.
- Capítulo 73: Database Physical Storage.
- Apéndice F: Additional Supplied Modules. Pageinspect, pgstatutuple
- Apéndice G: Additional Supplied Programs. Oid2name