Titulación: Grado en Ingeniería Informática e Ingeniería en

Sistemas de Información

Curso: 2022-2023. Convocatoria Ordinaria de Junio

Asignatura: Bases de Datos Avanzadas – Laboratorio

Practica 1: Arquitectura PostgreSQL y

almacenamiento físico

ALUMNO 1:
Nombre y Apellidos:
DNI:
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 <u>Suspensa – Cero</u> .
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

Plazos

ATTEMANTO.

Trabajo de Laboratorio: semana 30 enero, 6 febrero, 13 febrero, 20 febrero y 27 de

febrero.

Entrega de práctica: día 7 de marzo. Aula Virtual

una explicación hará que sea invalidada esa cuestión.

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. El fichero se

deberá llamar: DNIdelosAlumnos PL1.zip

AMBOS ALUMNOS DEBEN ENTREGAR EL FICHERO EN LA PLATAFORMA.

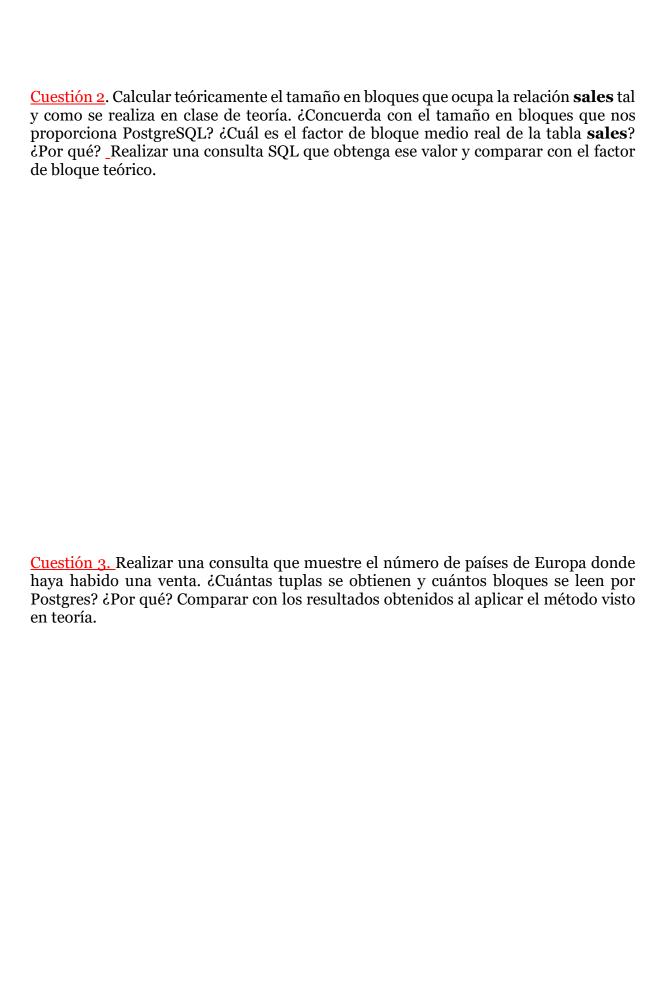
Introducción

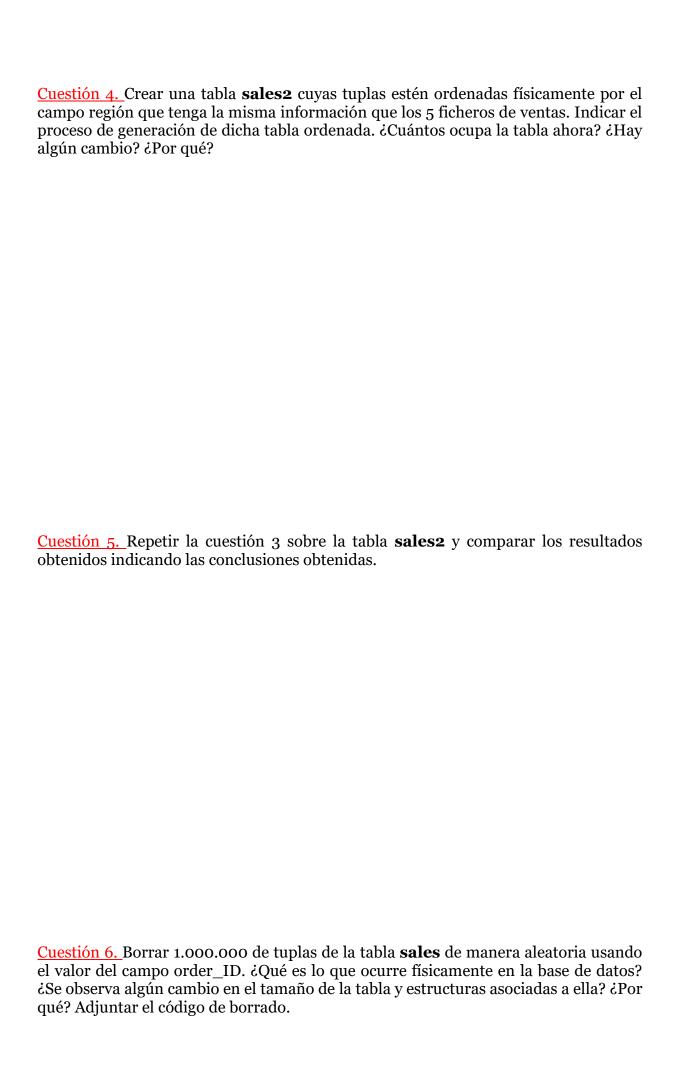
En esta primera práctica se introduce el sistema gestor de bases de datos PostgreSQL versión 15.1. Está compuesto básicamente de un motor servidor y de una serie de clientes que acceden al servidor y de otras herramientas externas. En esta primera práctica se entrará a fondo en la arquitectura de PostgreSQL, 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 o. Hay que resolver la práctica con consultas SQL.

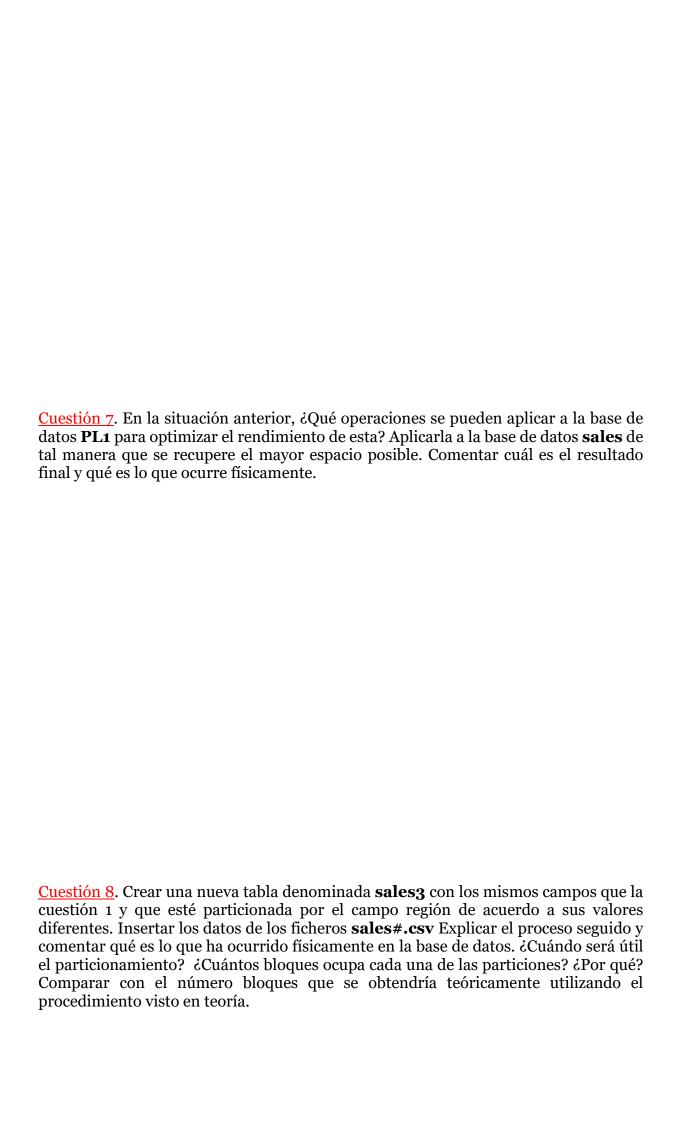
<u>Cuestión o.</u> Configurar el fichero de Error Reporting and Logging de PostgreSQL para que aparezcan recogidas las sentencias SQL DDL (Lenguaje de Definición de Datos) + DML (Lenguaje de Manipulación de Datos) generadas en dicho fichero. No se pide activar todas las sentencias. No activar la duración 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?

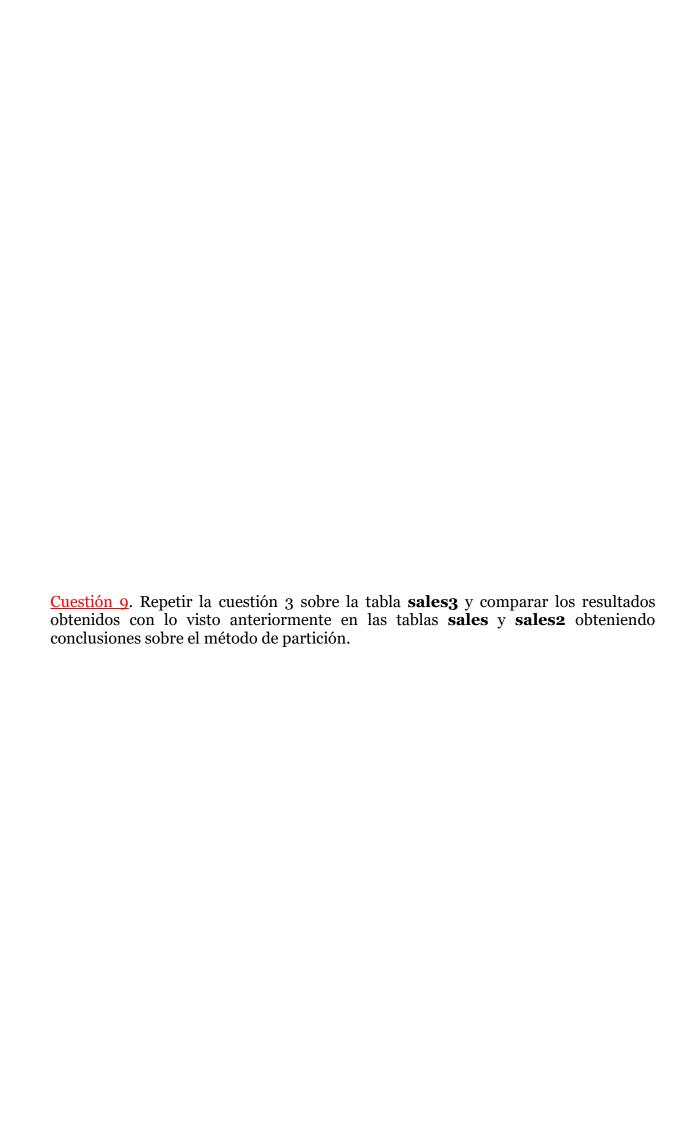
Organización de Archivos en PostgreSQL

<u>Cuestión 1</u>. Crear una nueva Base de Datos que se llame **PL1**. Después crear una tabla **sales** con los campos que vienen en la primera línea para poder cargar los datos correctamente. El campo order_ID es la Primary Key. Los tipos de datos de cada campo deben ser los más ajustados en capacidad de almacenamiento al dato que se inserta. Localizar los ficheros relacionados con la tabla. ¿cómo se localizan? ¿Cuánto ocupan y por qué?







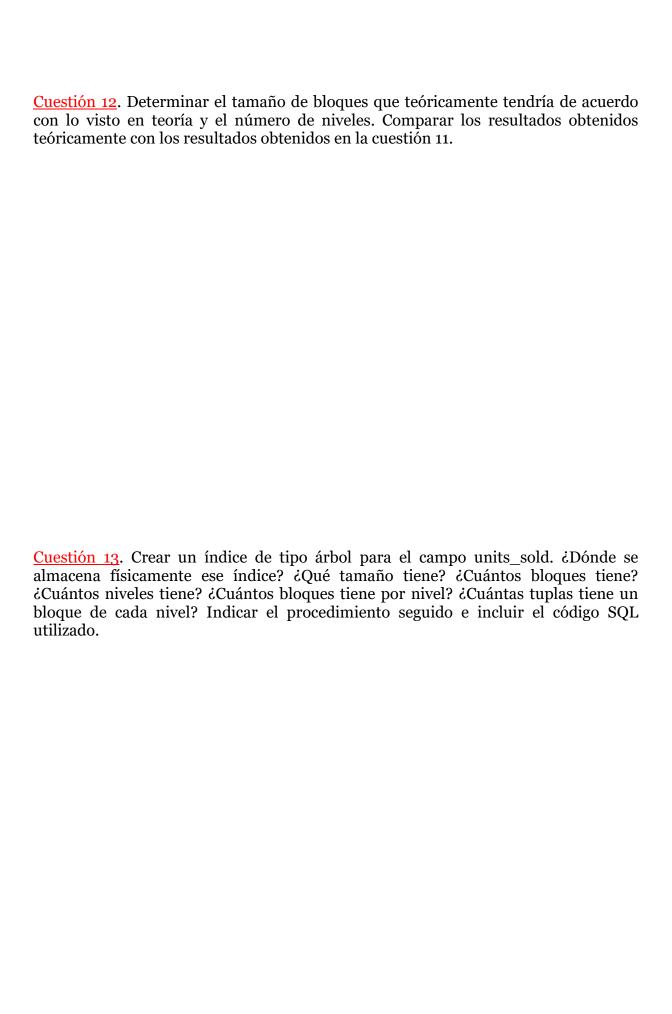


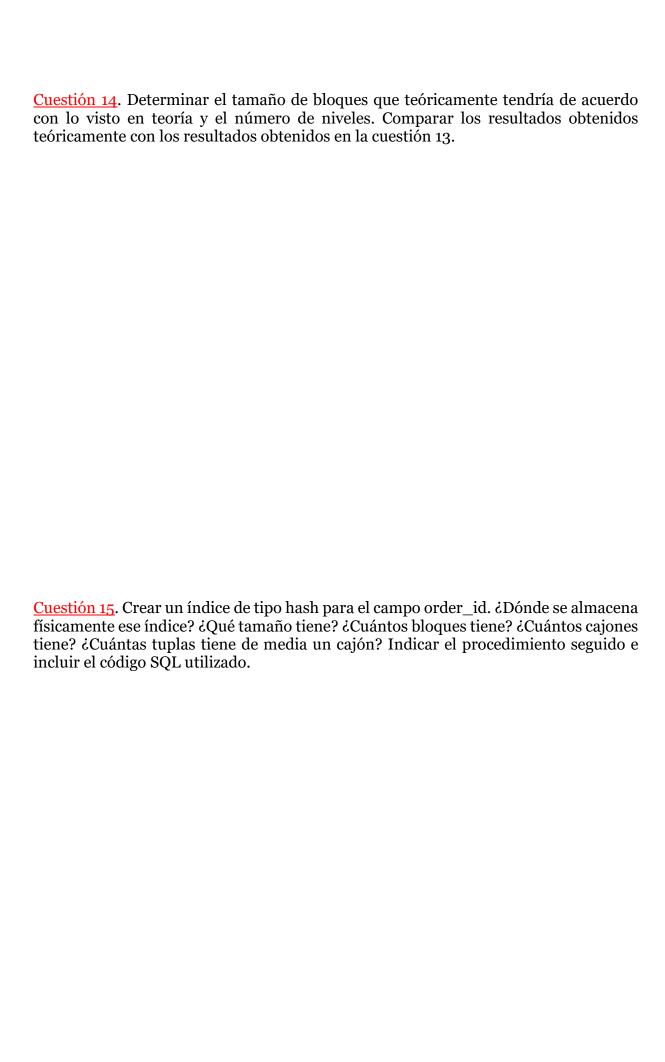
Indexación de PostgreSQL

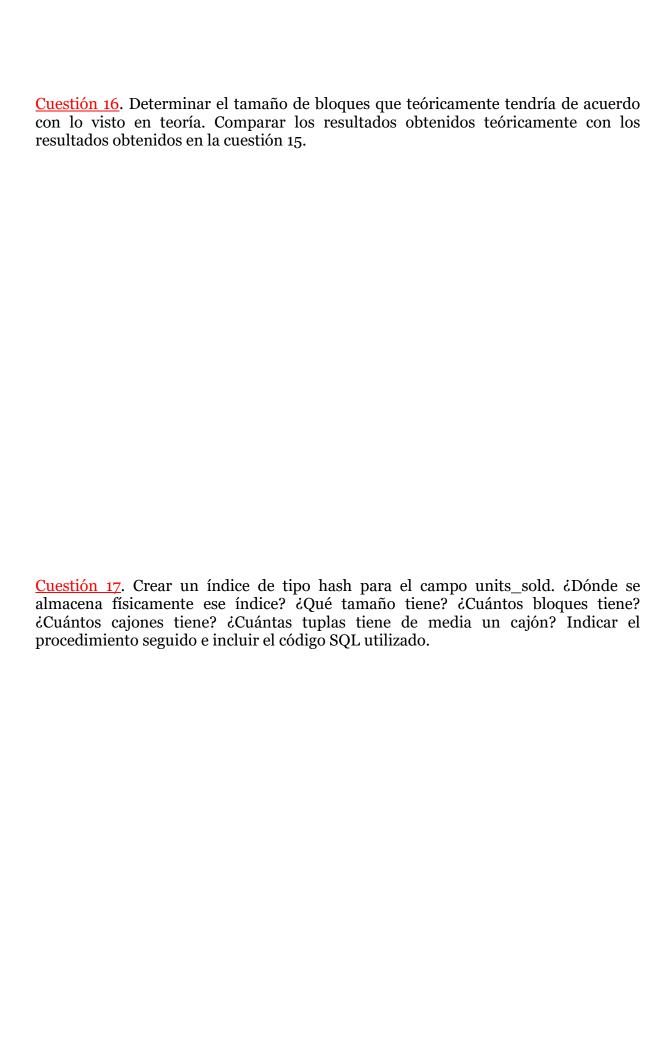
PostgreSQL soporta indexación definida por el usuario para ayudar a acelerar ciertas consultas. Entre otros tipos de índices soporta árboles y hash. En este apartado se va a trabajar sobre ambos tipos de índices, pudiendo observar cómo se organizan internamente y su funcionamiento.

<u>Cuestión 10</u>. Borrar todas las tablas **sales, sales2** y **sales3**. Crear una nueva tabla que se llama **sales** como en la cuestión 1 y que tenga cargados todos los datos de los ficheros **sales#.csv**

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









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 puede 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 **sales** como en la cuestión 1. Cargar los datos que se encuentran originalmente en los ficheros **sales**#.csv

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

<u>Cuestión 21</u>. 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? <u>Importante</u>, 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.

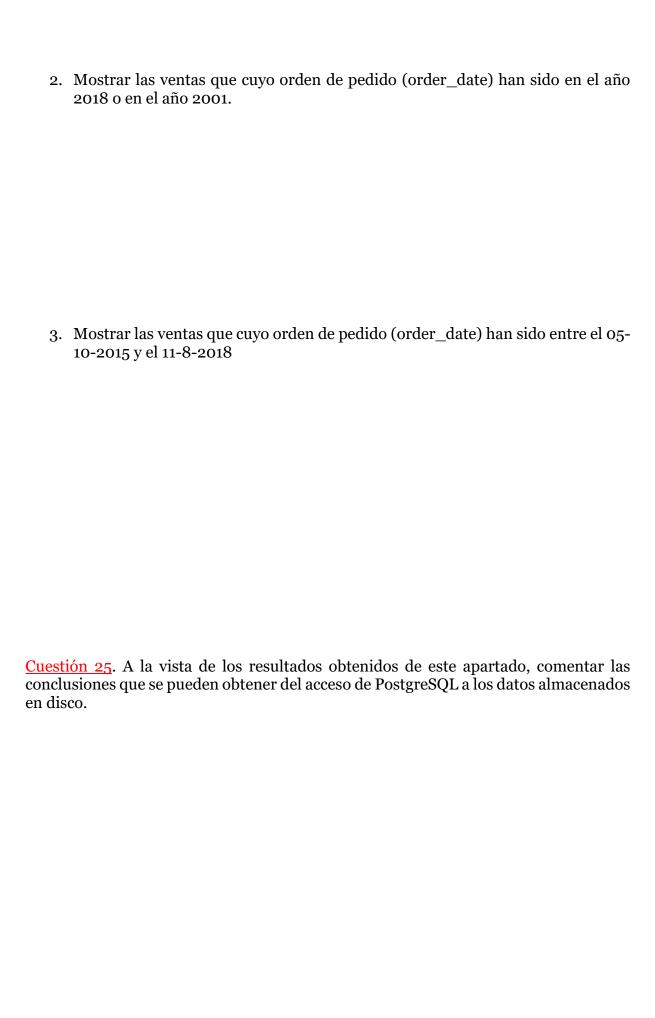
1. Mostrar la información de las tuplas con order id 100565711.

2.	Mostrar la información de las ventas con un coste total de 387.072,00 €.
વ	Contar el número de ventas que tienen coste total mayor de 32.000,00 € y
J.	menor de 100.000,00 €
1	Mostrar las ventas que son del país de Tanzania.
٦.	Prostrur las ventas que son der puis de l'anzuna.

5.	Actualizar la prioridad del pedido para que sea 'L' para todas las ventas que tienen un coste de 346.692,00 €
6.	Mostrar la información de los alumnos que tienen un order_id igual a 103903664 o 10390366
7.	Insertar una nueva venta en la tabla sales.

<u>Cuestión 22</u> . Crear un índice multiclave tipo árbol y otro hash sobre los campos región y country. 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 ventas totales que se han producido en Europa.
2. Mostrar el número de ventas que se han producido en la región de Asia y el país de Laos.

	Mostrar el número de tuplas del país México o de la región de "Aust Oceania".	ralia and
	Mostrar el número de tuplas de la región de "Sub-Saharan Africa" y que el número de unidades vendidas es de 2843.	e además
cada un obtene Comen leído de la activ	ón 24. Crear la tabla sales3 particionada por rango de años comple na de las consultas que se muestran a continuación, ¿Qué información er de los datos monitorizados por la base de datos al realizar la contar cómo se ha realizado la resolución de la consulta. ¿Cuántos bloque cada estructura? ¿Por qué? Importante, reinicializar los datos recolectidad de la base de datos antes de lanzar cada consulta. Mostrar las ventas que cuyo orden de pedido (order_date) han sido e 2022.	se puede consulta? es se han ctados de



Bibliografía (PostgreSQL 15)

- 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