

**Titulación:** Grado en Ingeniería Informática e Ingeniería en Sistemas de Información  
**Curso:** 2021-2022. Convocatoria Ordinaria de Junio  
**Asignatura:** Bases de Datos Avanzadas – Laboratorio

## **Practica 2: Carga Masiva de Datos, Procesamiento y Optimización de Consultas**

**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 Suspenso – Cero.

### **Plazos**

**Tarea en Laboratorio:** Semana 14 de Marzo, Semana 21 de Marzo, Semana 28 de Marzo y semana 4 de Abril.

**Entrega de práctica:** Semana 19 de Abril (Martes). Aula Virtual

**Documento a entregar:** Este mismo fichero con las respuestas a las cuestiones planteadas, el programa que genera los datos de carga de la base de datos en Python, el fichero history del pgcli y los ficheros de log de PostgreSQL de esta práctica. No se piden los ficheros de los datos en bruto de la base de datos. Se entregará en un ZIP comprimido llamado: **DNI'sdelosAlumnos\_PECL2.zip**

**AMBOS ALUMNOS DEBEN ENTREGAR EL FICHERO EN LA PLATAFORMA.**

## Introducción

El contenido de esta práctica versa sobre la monitorización de la base de datos, manipulación de datos, técnicas para una correcta gestión de los mismos, así como tareas de mantenimiento relacionadas con el acceso y gestión de los datos. También se trata el tema de procesamiento y optimización de consultas realizadas por PostgreSQL (14.x). Se analizará PostgreSQL en el proceso de carga masiva y optimización de consultas.

En general, la monitorización de la base de datos es de vital importancia para la correcta implantación de una base de datos, y se suele utilizar en distintos entornos:

- **Depuración de aplicaciones:** Cuando se desarrollan aplicaciones empresariales no se suele acceder a la base de datos a bajo nivel, sino que se utilizan librerías de alto nivel y mapeadores ORM (Hibernate, Spring Data, MyBatis...) que se encargan de crear y ejecutar consultas para que el programador pueda realizar su trabajo más rápido. El problema en estos entornos está en que se pierde el control de qué están haciendo las librerías en la base de datos, cuántas consultas ejecutan, y con qué parámetros, por lo que la monitorización en estos entornos es vital para saber qué consultas se están realizando y poder optimizar la base de datos y los programas en función de los resultados obtenidos.
- **Entornos de prueba y test de rendimiento:** Cuando una base de datos ha sido diseñada y se le cargan datos de prueba, una de las primeras tareas a realizar es probar que todos los datos que almacenan son consistentes y que las estructuras de datos dan un rendimiento adecuado a la carga esperada. Para ello se desarrollan programas que simulen la ejecución de aquellas consultas que se consideren de interés para evaluar el tiempo que le lleva a la base de datos devolver los resultados, de cara a buscar optimizaciones, tanto en la estructura de la base de datos como en las propias consultas a realizar.
- **Monitorización pasiva/activa en producción:** Una vez la base de datos ha superado las pruebas y entra en producción, el principal trabajo del administrador de base de datos es mantener la monitorización pasiva de la base de datos. Mediante esta monitorización el administrador verifica que los parámetros de operación de la base de datos se mantienen dentro de lo esperado (pasivo), y en caso de que algún parámetro salga de estos parámetros ejecuta acciones correctoras (reactivo). Así mismo, el administrador puede evaluar nuevas maneras de acceso para mejorar aquellos procesos y tiempos de ejecución que, pese a estar dentro de los parámetros, muestren una desviación tal que puedan suponer un problema en el futuro (activo).

Para la realización de esta práctica será necesario generar una muestra de datos de cierta índole en cuanto a su volumen de datos. Para ello se generarán, dependiendo del modelo de datos suministrado, para una base de datos denominada **BECAS**. Básicamente la base de datos guarda las becas que se ha concedido a cada uno de los estudiantes que quieren realizar una estancia en una Universidad de destino diferente a donde están matriculado. La tabla estudiantes guarda la información de cada estudiante, la tabla universidades guarda la información de cada Universidad que participa en el programa, la tabla asignaturas guarda las asignaturas ofertadas a los estudiantes y que pertenecen a cada Universidad, la tabla estancias guarda la información sobre la beca concedida a cada estudiantes; y la tabla convalidaciones

guarda la relación de las asignaturas que se estudian en destino junto con la asignatura que se convalida en origen, para cada estancia. Se suministra el modelo relacional construido en **pgmodeler** donde dentro de cada tabla se comenta lo que se guarda, así como la descripción de cada campo.

Los datos referidos al año 2021 que hay que generar deben de ser los siguientes:

- Hay un total de 100.000 universidades dadas de alta. Hay 50 países almacenados y uno de ellos debe ser "Spain". Los países se asignan a la Universidad de manera aleatoria entre los 50.
- Hay 5.000.000 de estudiantes dados de alta. A cada estudiante se le asigna una universidad de origen de manera aleatoria. Los créditos aprobados se definen entre 60 y 200 (enteros e incluidos) ECTS y deben de estar generados de manera aleatoria.
- Cada universidad oferta entre 15 y 20 asignaturas que pueden cursar los estudiantes en su estancia. Esas asignaturas pueden ser de tipo "básica", "obligatoria", "transversal" y "optativa". Se asigna el tipo de manera aleatoria a cada asignatura. El número de créditos ECTS varía entre 4 y 12 de manera aleatoria.
- Cada estudiante ha realizado una estancia, y la universidad de destino se elige de manera aleatoria sin que sea la propia universidad donde estudia el alumno. El tipo de beca puede ser: "KA103", "KA107", "MIT", "Franklin", "SICUE" y "Global". La duración puede ser primer cuatrimestre "1", segundo cuatrimestre "2" o anual "A". La cuantía de las becas asciende a valores entre 500 y 2000 euros. Todas las asignaciones se hacen de manera aleatoria.
- Cada estancia tiene asignadas entre 3 y 5 asignaturas si la estancia se realiza el primer (1) o el segundo cuatrimestre (2); y entre 6 y 10 asignaturas si la estancia es anual (A) de la universidad de destino elegida. La nota de origen puede ser: A+, A, A-, B+, B, B-, C+, C, C-, D+, D, D-, E o F. La nota en origen realizada una vez la convalidación tendrá que ver con la siguiente tabla de equivalencia. Todas las asignaciones se hacen de manera aleatoria

Nota Destino	Nota Origen
A+	10
A	9.8
A-	9.5
B+	9.0
B	8.5
B-	8.0
C+	7.5
C	7.3
C-	7.0
D+	6.5
D	6.0
D-	5.5
E	5.0
F	4.0

## Actividades y Cuestiones

**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") aparezca el DNI de los alumnos que realizan la práctica (ambos), el nombre del host con su puerto, y la fecha y hora de la operación que se ha realizado. Se ha de configurar también el servidor para que no use el procesamiento paralelo de consultas.

**Cuestión 1:** ¿Tiene el servidor postgres un recolector de estadísticas sobre el contenido de las tablas de datos? Si es así, ¿Qué tipos de estadísticas se recolectan y donde se guardan?

**Cuestión 2:** Crear una nueva base de datos llamada **matricula** y que tenga las siguientes tablas con los siguientes campos y características:

- estudiantes(carnet tipo numeric PRIMARY KEY, nombre tipo text, apellidos tipo text, credits tipo numeric)
- asignaturas(codigo tipo numeric PRIMARY KEY, nombre tipo text, caracter tipo text, credits tipo numeric)
- matriculas(carnet\_estu tipo numeric que sea FOREIGN KEY del campo carnet de la tabla estudiantes con restricciones de tipo RESTRICT en sus operaciones, codigo\_asig tipo numeric que sea FOREIGN KEY del campo codigo de la tabla asignaturas con restricciones de tipo RESTRICT en sus operaciones, nota de tipo numeric. La PRIMARY KEY debe ser compuesta de carnet\_estu y codigo\_asig.

Se pide:

- Indicar el proceso seguido para generar esta base de datos.
- Cargar la información del fichero datos\_estudiantes.csv, datos\_asignaturas.csv y datos\_matriculas.csv en dichas tablas de tal manera que sea lo más eficiente posible, asegurando la integridad y consistencia de la base de datos.
- Indicar los tiempos de carga.

**Cuestión 3:** Mostrar las estadísticas obtenidas en este momento para cada tabla. ¿Qué se almacena? ¿Son correctas? Si no son correctas, ¿cómo se pueden actualizar?

**Cuestión 4:** Aplicar el comando EXPLAIN a una consulta que obtenga el número de alumnos con créditos de más de 200 créditos ECTS. ¿Son correctos los resultados del comando EXPLAIN? ¿Por qué? Comparar con lo que se obtendría con el procedimiento visto en teoría obteniendo las estadísticas de postgres.

**Cuestión 5:** Aplicar el comando EXPLAIN a una consulta que obtenga el nombre de los alumnos que tienen 180 créditos y además tienen por lo menos una asignatura suspensa. ¿Son correctos los resultados del comando EXPLAIN? ¿Por qué? Comparar con lo que se obtendría con el procedimiento visto en teoría obteniendo las estadísticas de postgres.

**Cuestión 6:** Aplicar el comando EXPLAIN a una consulta que obtenga el nombre de las asignaturas de 6 créditos en los que los alumnos hayan obtenido un 8 de nota y los alumnos tengan un número de 60 créditos. ¿Son correctos los resultados del comando EXPLAIN? ¿Por qué? Comparar con lo que se obtendría con lo visto en teoría obteniendo las estadísticas de postgres.

**Cuestión 7:** Generar los datos solicitados al comienzo de la práctica para la base de datos **BECAS** usando un programa en Python que deberá de estar en un único fichero y comentado. Pegar el código del fichero en el cuadro de texto que se adjunta a continuación.

**Cuestión 8:** Realizar la carga masiva de los datos generados en la cuestión 7 en la base de datos **BECAS**. Indicar el proceso seguido y el orden de carga de las tablas, explicando el porqué de dicho orden; **y asegurando la consistencia e integridad de los datos cargados**. Comparar los tiempos en las tablas implicadas y explicar a qué es debida la diferencia. ¿Existe diferencia entre los tiempos que ha obtenido y los que aparecen en el LOG de operaciones de postgresQL? ¿Por qué?

Tabla	Tiempo (seg)



A partir de este momento en adelante, se deben de realizar las siguientes cuestiones con la base de datos que tiene la integridad referencial activada. Es obligatorio y queda prohibido cambiar la integridad referencial de la base de datos.

**Cuestión 9:** Realizar una consulta SQL que muestre “el nombre de las universidades españolas junto con el porcentaje de alumnos que tienen que se han ido de estancia con una beca de tipo “KA103” o “KA107” teniendo superados un número de créditos de más de 120 ECTS a Universidades de destino que no son españolas, habiéndose llevado asignaturas de tipo “obligatorio” o “básicas” y que hayan tenido una nota en destino de tipo B+,B o B- “.

Obtener el plan de ejecución con el resultado del comando EXPLAIN en forma de árbol de álgebra relacional. Explicar la información obtenida en el plan de ejecución de postgresQL. Comparar el árbol obtenido por nosotros al traducir la consulta original al álgebra relacional y el que obtiene postgresQL. Comentar las posibles diferencias entre ambos árboles.

Consulta SQL creada:

Resultado comando EXPLAIN

Comentarios y explicaciones.

**Cuestión 10:** Usando PostgreSQL, y a raíz de los resultados de la cuestión anterior, ¿qué modificaciones realizaría para mejorar el rendimiento de la misma y por qué? Obtener la información pedida de la cuestión 9 y explicar los resultados. Obtener el plan de ejecución con el resultado del comando EXPLAIN en forma de árbol de algebra relacional. Comentar los resultados obtenidos y comparar con la cuestión anterior.

Consulta SQL creada:

Resultado comando EXPLAIN

Comentarios y explicaciones.

**Cuestión 11:** Usando PostgreSQL, borre el 40% de las universidades almacenadas de manera aleatoria y todos sus datos relacionados de la manera más eficiente posible, ¿cuál ha sido el proceso seguido? ¿Y el tiempo empleado en el borrado? **Prohibido modificar la Integridad Referencial.**

Cuestión 12: Ejecute la consulta que pide los datos de la cuestión 9 de nuevo. Obtener el plan de ejecución con el resultado del comando EXPLAIN en forma de árbol de álgebra relacional. Comparar con los resultados anteriores.

Consulta SQL creada:

Resultado comando EXPLAIN

Comentarios y explicaciones.

**Cuestión 13:** ¿Qué optimización/mejoras de la BD propondría para mejorar los resultados de dicho plan sin modificar el código de la consulta? ¿Por qué?

**Cuestión 14:** Usando PostgreSQL, lleve a cabo las operaciones propuestas en la cuestión anterior y ejecute el plan de ejecución de la consulta que pide los datos de la cuestión 9. Obtener el plan de ejecución con el resultado del comando EXPLAIN en forma de árbol de álgebra relacional. Compare los resultados del plan de ejecución con los de los apartados anteriores. Coméntelos.

Consulta SQL creada:

Resultado comando EXPLAIN

Comentarios y explicaciones.

**Cuestión 15:** A partir de lo visto y recopilado en toda la práctica. Describir y comentar cómo es el proceso de procesamiento y optimización que realiza PostgreSQL en las consultas del usuario.

## **Bibliografía**

### PostgreSQL (14.x)

- Capítulo 14: Performance Tips.
- Capítulo 20: Server Configuration.
- Capítulo 15: Parallel Query.
- Capítulo 25: Routine Database Maintenance Tasks.
- Capítulo 51: Overview of PostgreSQL Internals.
- Capítulo 72: How the Planner Uses Statistics.
- <https://pgtune.leopard.in.ua/>
- <https://explain.dalibo.com/>