



7. RENDIMIENTO EN APLICACIONES.

7.1. METODOLOGÍA A EMPLEAR.

En este apartado, se va a realizar un estudio sobre el funcionamiento de las bases de datos a estudio sobre aplicaciones reales. Este estudio se abordará desde dos perspectivas diferentes. En primer lugar, se ha desarrollado una aplicación en HTML/PHP con interfaz Web, que consiste en un analizador de bases de datos, a través del cual se pueden medir los tiempos de respuesta para diversas operaciones sobre las bases de datos a estudio, todo ello a través de una interfaz Web sencilla e intuitiva. En segundo lugar, se ha buscado una aplicación que empleara las bases de datos a estudio. Esta aplicación consiste en un gestor de tareas y mensajes, muy apropiada para el ámbito empresarial. Esta aplicación se ha adaptado para que funcione con los sistemas gestores de bases de datos a estudio, y se han aislado las partes del código fuente correspondientes a las tareas de acceso a la base de datos, para analizarlas por separado con un banco de pruebas específico, con objeto de determinar qué sistema gestor es más apropiado según la carga de usuarios prevista en el sistema. Esta aplicación está desarrollada en HTML/PHP.

El hecho de escoger aplicaciones con interfaz Web para analizar casos reales se debe a que gran parte de las aplicaciones existentes que hacen uso de sistemas gestores de bases de datos se emplean para el acceso a dichos sistemas gestores de bases de datos a través de algún tipo de interfaz Web.

En los siguientes apartados se mostrarán con más detalle dichas aplicaciones y los resultados de su utilización.

7.1.1. Ambiente de las pruebas.

7.1.1.1. Igualdad de recursos.

Con el fin de que las pruebas se realicen en un entorno de total igualdad para ambos sistemas gestores de bases de datos, se utilizarán los mismos recursos físicos y lógicos ejecutándose las aplicaciones en la misma máquina, con el mismo sistema operativo, y configurando cada uno con la misma cantidad de utilización recursos lógicos.

7.1.1.2. Recursos físicos.

Partiendo de las características mínimas de instalación requeridas para un buen funcionamiento de cada uno de los dos sistemas gestores de bases de datos a estudio, se ejecutarán las aplicaciones en un equipo de mejores recursos físicos, con el fin de asegurar la estabilidad del sistema para todos los test que se van a ejecutar. De este modo, los recursos físicos empleados en el desarrollo de las pruebas han sido los mismos que para los análisis mostrados en apartados anteriores (bancos de pruebas). Recuerde que dicha configuración de recursos físicos era la siguiente:

- Procesador Intel Pentium IV HT a 3.0 GHz.
- 1024 MB de memoria RAM.
- Disco duro con aprox. 20 GB de espacio libre para la realización de las pruebas.



Esta configuración de hardware asegura, según los fabricantes de ambos sistemas gestores de bases de datos, el correcto funcionamiento de cada uno.

7.1.1.3. Recursos lógicos.

El software que se ha empleado para el desarrollo de las aplicaciones y para su correcta utilización se muestra en el siguiente listado:

- Sistema Operativo Linux. Distribución FEDORA CORE 4. Versión *kernel* 2.6.11.
- MySQL v4.1.11.
- PostgreSQL v8.1.3.
- Servidor Web Apache v2.0.54 para albergar las aplicaciones Web.
- Lenguaje de programación PHP para el desarrollo y adaptación de las aplicaciones.
- Editor de textos *Emacs* para codificar las aplicaciones HTML/PHP.
- Apache Bench v2.0.41 para realizar las pruebas de concurrencia sobre la aplicación.

La configuración adoptada para los dos sistemas gestores de bases de datos a estudio es la misma que la que se empleó en el apartado de bancos de pruebas.

7.1.1.4. Igualdad en el diseño.

Las dos aplicaciones a emplear, se han diseñado/modificado para que accedan de igual manera a las bases de datos (salvando pequeñas diferencias en cuanto a la sintaxis SQL), para que de este modo funcionen bajo igualdad de condiciones cuando se empleen con uno u otro sistema gestor de bases de datos. Además, tanto las tablas creadas por dichas aplicaciones, como los datos insertados/modificados/eliminados en esas aplicaciones son los mismos para cada uno de los sistemas a estudio.

7.1.2. Concurrencia

En este apartado también se realizarán pruebas de concurrencia para ver cómo afecta el número de usuarios concurrentes al rendimiento de una determinada aplicación. Para este tipo de pruebas se ha empleado el banco de pruebas apache sobre la aplicación gestora de tareas y mensajes que se expone más adelante en este capítulo.

7.2. ANALIZADOR WEB DE BASES DE DATOS.

Para evaluar el rendimiento de los sistemas gestores de bases de datos a estudio, se ha desarrollado una aplicación consistente en un analizador de bases de datos con interfaz Web. Mediante esta aplicación se podrán medir los tiempos de respuesta de cada uno de los sistemas gestores de bases de datos ante una serie de operaciones solicitadas a través de la interfaz Web. Este tipo de operaciones son las más comunes que suele emplear cualquier aplicación que acceda a un sistema gestor de bases de datos a través de una interfaz Web.

Se ha buscado en Internet alguna aplicación parecida, pero la mayoría de los bancos de pruebas y analizadores encontrados eran de características similares a los que se



emplearon en el Capítulo 6, y ninguno poseía interfaz Web. Se encontraron algunos analizadores Web comerciales, pero ninguno de ellos hacía especial énfasis en el acceso a bases de datos, sino que simplemente se limitaban a evaluar el rendimiento de una aplicación Web genérica. Es por este motivo por lo que se pensó en desarrollar el banco de pruebas con interfaz Web aquí expuesto. Además, este banco de pruebas tiene una ventaja respecto a los utilizados anteriormente en el Capítulo 6 y otros similares, que es el hecho de que no es necesaria instalación de ningún tipo, sino que simplemente accediendo a la página Web en la que se aloje se podrá evaluar el rendimiento de cualquier sistema gestor de bases de datos de los estudiados, sea local o remoto.

El diseño, implementación y pruebas de la aplicación se han llevado a cabo en unas 100 horas, aproximadamente. Para desarrollarlo, se ha empleado los lenguajes PHP y HTML, y para las pruebas y depuración se ha alojado la aplicación en un servidor Apache.

El diseño de la aplicación es bastante modular, lo que facilita notablemente la expansión de la misma para incluir otros tipos de pruebas y otros sistemas gestores de bases de datos diferentes de los estudiados en este Proyecto.

En la figura 7.1 se muestra el esquema de la aplicación.

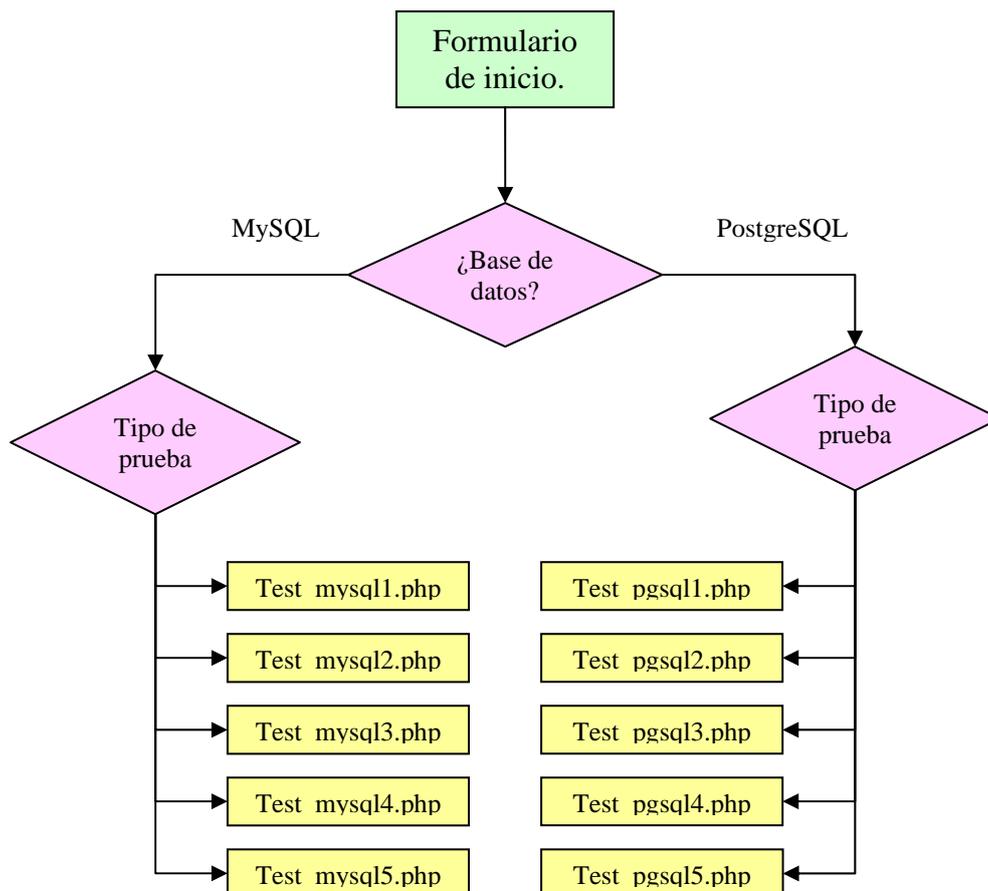


Figura 7.1. Diagrama de Flujo del analizador Web de bases de datos.



Como puede verse en la figura 7.1, el diseño de la aplicación es bastante modular, con lo que fácilmente se le pueden añadir más tipos de pruebas e incluir otros sistemas gestores de bases de datos.

Por lo que respecta a los sistemas gestores de bases de datos a los cuales se podría extender la aplicación, la única restricción que tienen que cumplir es que dispongan de las funciones apropiadas para su conexión y acceso en el lenguaje PHP. De esta forma, algunos de los sistemas gestores de bases de datos que se podrían incluir en la aplicación serían FrontBase, IBase, Informix, Ingres, mSQL, msSQL, Oracle y Sybase. Además, también podrían incluirse otros sistemas gestores que proporcionasen conectores ODBC, ya que PHP proporciona una serie de funciones para manejar este tipo de conexiones.

En los siguientes apartados se describirán las características y el alcance de la aplicación, así como los resultados obtenidos al ejecutar dicha aplicación sobre cada sistema gestor de bases de datos a estudio.

7.2.1. Descripción de la aplicación.

El analizador de bases de datos desarrollado, se ha codificado empleando los lenguajes HTML y PHP. El código fuente de la aplicación se mostrará en el ANEXO C. Esta aplicación posee una interfaz Web bastante sencilla e intuitiva, y puede alojarse en cualquier servidor Web. A través de ella se puede analizar cualquier sistema gestor de bases de datos MySQL o PostgreSQL, con independencia de que estos estén alojados en la misma máquina o no. La ventana principal de la aplicación se muestra en la figura 7.2.

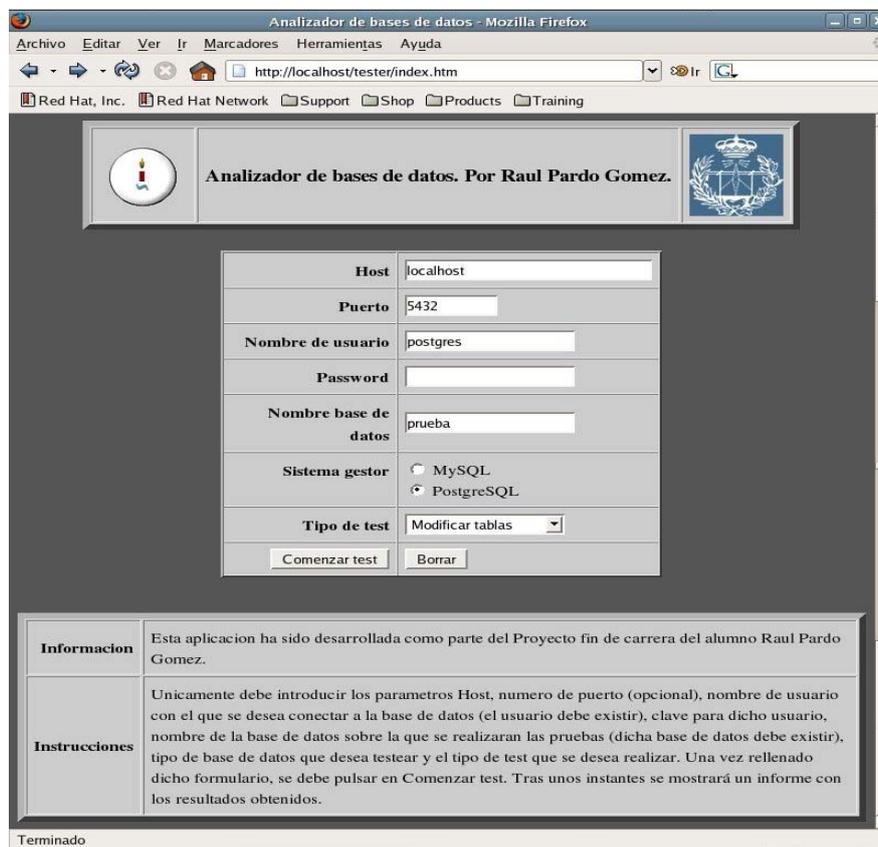


Figura 7.2. Interfaz del analizador de bases de datos.



Como puede verse, el uso de esta aplicación es bastante sencillo. En la parte inferior de la pantalla se muestran las instrucciones para ejecutar esta aplicación. Para ello, simplemente habrá que rellenar los parámetros de entrada del formulario mostrado y pulsar el botón “Comenzar test”. Los parámetros de entrada de dicho formulario se describen a continuación:

Host

En este cuadro de texto se deberá introducir el nombre del servidor donde se encuentra alojado el sistema gestor de bases de datos que se desea analizar. Si este sistema gestor de bases de datos está en la misma máquina en la que se encuentra el servidor Web, se deberá especificar “localhost”.

Puerto

Este parámetro es opcional, y se debe introducir en caso de que la conexión al sistema gestor de bases de datos se haga a través de un puerto diferente al establecido por defecto.

Nombre de usuario

En este parámetro se debe introducir un nombre de usuario válido para la base de datos que se desea analizar. Dicho usuario debe existir y poseer los permisos necesarios para la base de datos empleada en la realización de las pruebas.

Password

Aquí se debe especificar la clave de usuario empleada para el acceso al sistema gestor de bases de datos que se desea analizar, en caso de que el acceso de dicho usuario este restringido por una clave.

Nombre base de datos

En este cuadro de texto se debe especificar el nombre de la base de datos sobre la que se realizarán las pruebas. Dicha base de datos debe existir. Las pruebas se han diseñado de forma que todas las tablas creadas sean eliminadas al final de cada prueba, devolviendo la base de datos a su estado inicial.

Sistema gestor

En la parte correspondiente al sistema gestor se debe seleccionar el sistema gestor de bases de datos que se desea analizar de entre los dos disponibles: MySQL (como motor de almacenamiento MyISAM) y PostgreSQL.

Tipo de test

Por último, en este apartado se puede seleccionar el tipo de prueba que se desea realizar sobre la base de datos seleccionada. Las pruebas disponibles son cinco:

- Conexión.
- Creación de tablas.
- Inserción de datos.
- Consultas SELECT.
- Modificación de tablas.



En estas cinco pruebas se engloban las operaciones más habituales que ejecuta una aplicación determinada sobre bases de datos. Más adelante se verán en detalle las operaciones realizadas en cada una de las cinco pruebas.

Una vez que se hayan introducido en el formulario de entrada todos los parámetros de entrada necesarios se deberá pulsar en el botón “Comenzar test”. Tras unos instantes, se mostrarán los resultados obtenidos. Si alguno de los parámetros introducidos no es válido, o si ha habido problema con la conexión a la base de datos, se mostrará un mensaje de error por pantalla.

Un ejemplo de pantalla de resultados se muestra en la figura 7.3.

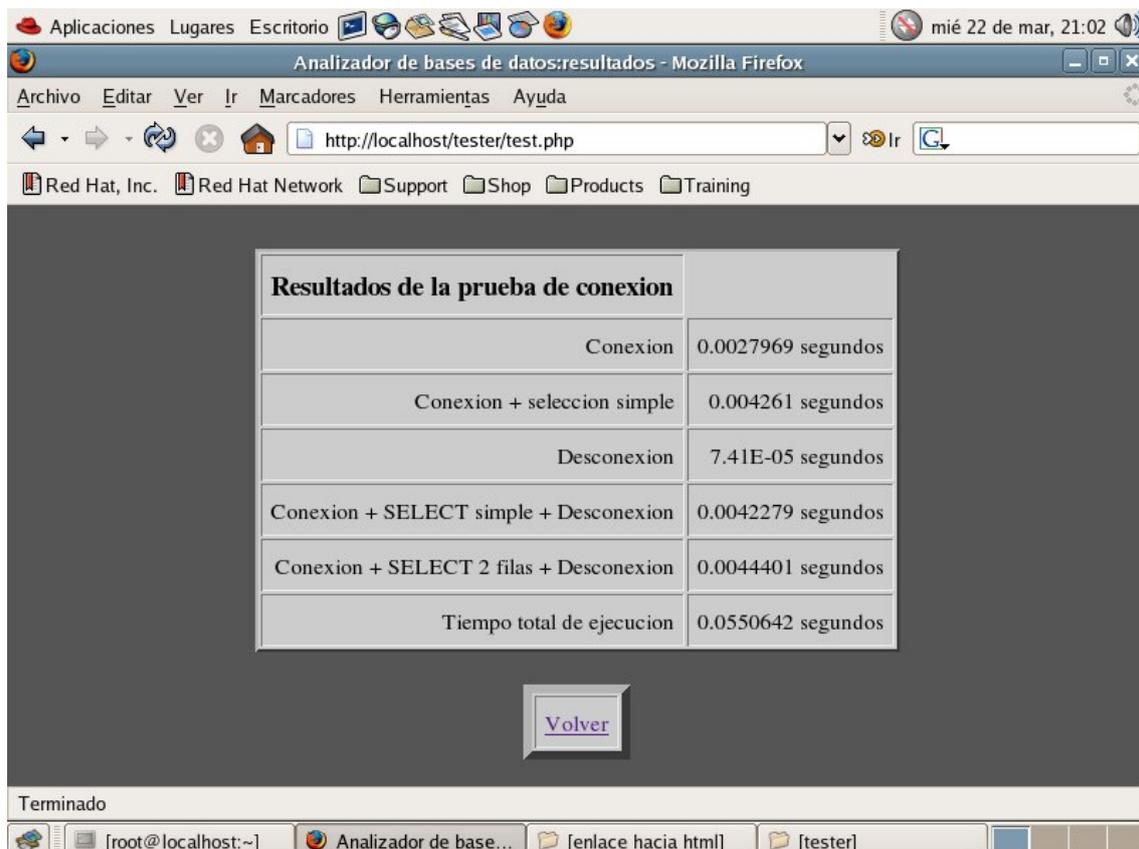


Figura 7.3. Ventana de resultados del analizador de bases de datos.

Como puede verse en la imagen anterior, se muestran todos los resultados obtenidos para la prueba especificada debidamente tabulados. Este formato de presentación de resultados es similar para todas las pruebas.

Como se comentó antes, las pruebas disponibles son cinco. A continuación se va a hacer una descripción de cada una de ellas.

7.2.1.1. Pruebas de conexión.

Mediante estas pruebas se evalúan los tiempos empleados asociados a tareas de conexión/desconexión. Los tiempos que se miden son los siguientes:

- Tiempo empleado en la conexión a la base de datos.



- Tiempo empleado en realizar una conexión y una selección simple.
- Tiempo empleado en la desconexión a la base de datos.
- Tiempo empleado en realizar una conexión, una selección simple, y una desconexión a la base de datos.
- Tiempo empleado en realizar una conexión, una selección que dé como resultado dos filas, y una desconexión a la base de datos.

7.2.1.2.Pruebas de creación de tablas.

Con estas pruebas, se pretende evaluar el comportamiento del sistema gestor de bases de datos bajo análisis ante tareas de creación de tablas. Se miden los tiempos asociados a las siguientes operaciones:

- Tiempo empleado en crear 100 tablas.
- Tiempo empleado en eliminar 100 tablas.
- Tiempo empleado en crear y eliminar una tabla.
- Tiempo empleado en crear y eliminar una tabla con muchos índices.

7.2.1.3.Pruebas de inserción de datos.

En esta prueba se obtienen los tiempos empleados en realizar tareas de inserción, eliminación y actualización de registros en tablas. Los resultados que se obtienen son los siguientes:

- Tiempo empleado en insertar 1000 registros en una tabla.
- Tiempo empleado en insertar 1000 registros en orden en una tabla indexada.
- Tiempo empleado en insertar 1000 registros en orden inverso en una tabla indexada.
- Tiempo empleado en insertar 1000 registros en orden aleatorio en una tabla indexada.
- Si se permite o no la inserción de datos con ids duplicadas.
- Tiempo empleado en la actualización de todos los registros de una tabla.
- Tiempo empleado en la actualización de parte de los registros de una tabla.
- Tiempo empleado en insertar datos en una tabla con un índice a partir de una consulta SELECT.
- Tiempo empleado en insertar datos en una tabla con dos índices a partir de una consulta SELECT.
- Tiempo empleado en eliminar todos los registros de una tabla.
- Tiempo empleado en eliminar un registro de una tabla.
- Tiempo empleado en eliminar un rango de registros en una tabla.
- Tiempo empleado en insertar 1000 registros en una tabla con muchos índices.
- Tiempo empleado en insertar un registro en una tabla con muchos índices.
- Tiempo empleado en actualizar un registro en una tabla con muchos índices.
- Tiempo empleado en actualizar varios registros en una tabla con muchos índices.
- Tiempo empleado en eliminar un registro en una tabla con muchos índices.
- Tiempo empleado en eliminar varios registros en una tabla con muchos índices.

7.2.1.4.Pruebas de consultas SELECT.



Mediante esta prueba se evalúan los tiempos empleados en realizar determinadas consultas de selección con SELECT. Dichas consultas con las siguientes:

- SELECT todos los elementos de una tabla.
- SELECT parte de los elementos de una tabla.
- SELECT parte de los elementos de una tabla con 2 condiciones.
- SELECT parte de los elementos de una tabla con 2 condiciones desde la caché.
- Consultas SELECT grandes.
- SELECT con funciones sum().
- SELECT con funciones max(), min() y count().
- SELECT + ORDER BY 1 columna.
- SELECT + ORDER BY 2 columnas.
- SELECT + GROUP BY 1 columna.
- SELECT + GROUP BY 2 columnas.
- Consulta UNION.
- Consulta con producto cartesiano simple.
- Consultas JOIN.

7.2.1.5. Pruebas de modificación de tablas.

Con la ejecución de esta prueba se obtienen los tiempos empleados en realizar diversas operaciones de modificación de tablas. Los tiempos que se muestran en la salida de la prueba son los siguientes:

- Tiempo empleado en añadir 10 campos a una tabla.
- Tiempo empleado en añadir 5 índices a una tabla.
- Tiempo empleado en eliminar 5 índices de una tabla.
- Tiempo empleado en eliminar 10 campos de una tabla.
- Tiempo empleado en renombrar una tabla.

Por tanto, con la ejecución de estas pruebas se puede obtener una idea del rendimiento de cada sistema gestor de bases de datos ante operaciones y situaciones en un ambiente real de ejecución. Los resultados obtenidos en la realización de dichas pruebas para la configuración de Hardware/software empleada a lo largo de todo el proyecto se mostrarán en el siguiente apartado.

7.2.2. Resultados obtenidos.

La aplicación desarrollada se ha ejecutado bajo el ambiente de pruebas ya descrito en otros apartados. La aplicación se ha alojado en un servidor Apache, corriendo en la misma máquina que los servidores de bases de datos bajo estudio. No obstante, cabe resaltar que la aplicación se ha diseñado para que se puedan analizar sistemas gestores de bases de datos corriendo en la misma máquina que el servidor Web o en una máquina diferente.

Los resultados obtenidos para cada una de las diferentes pruebas se muestran en las figuras que siguen.



Prueba de conexión

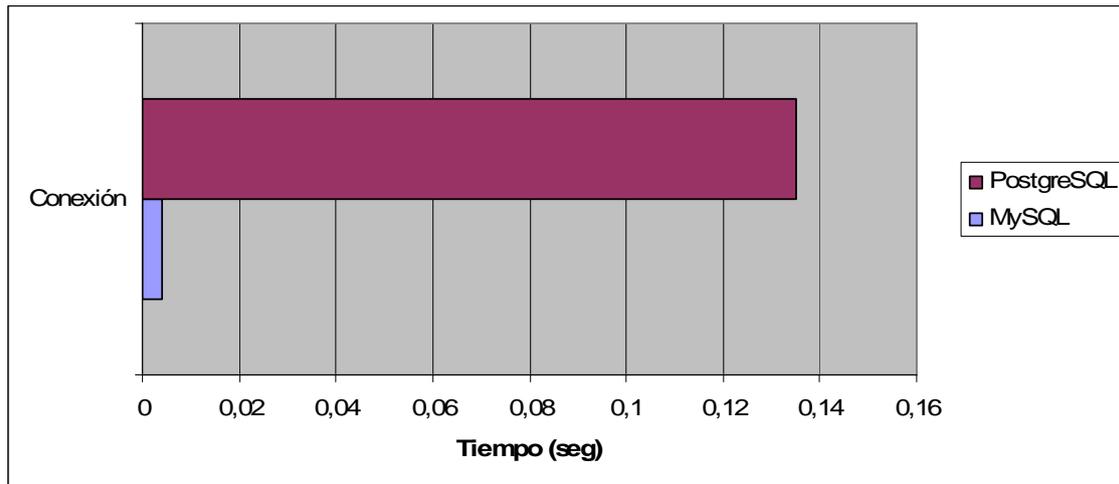


Figura 7.4. Tiempo empleado en la conexión al sistema.

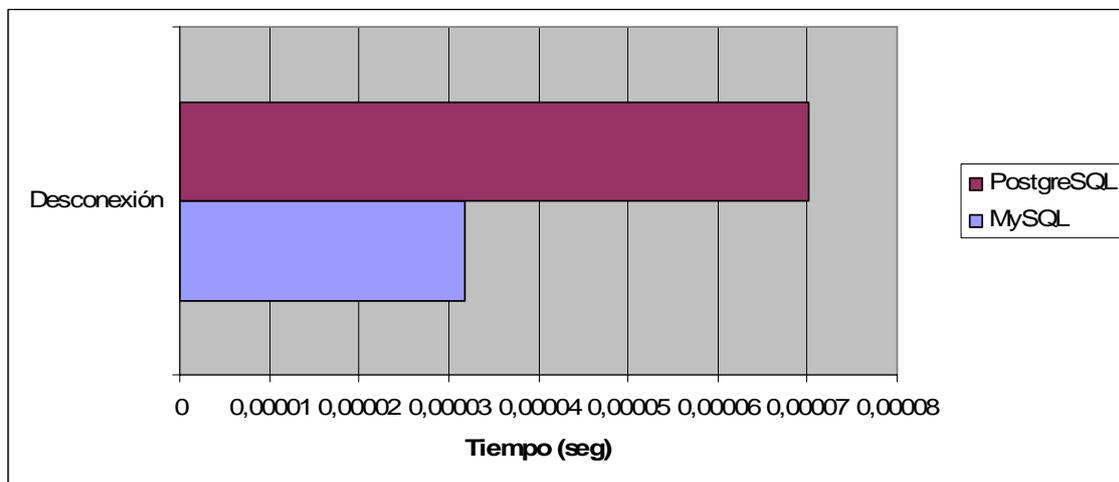


Figura 7.5. Tiempo empleado en la desconexión al sistema.

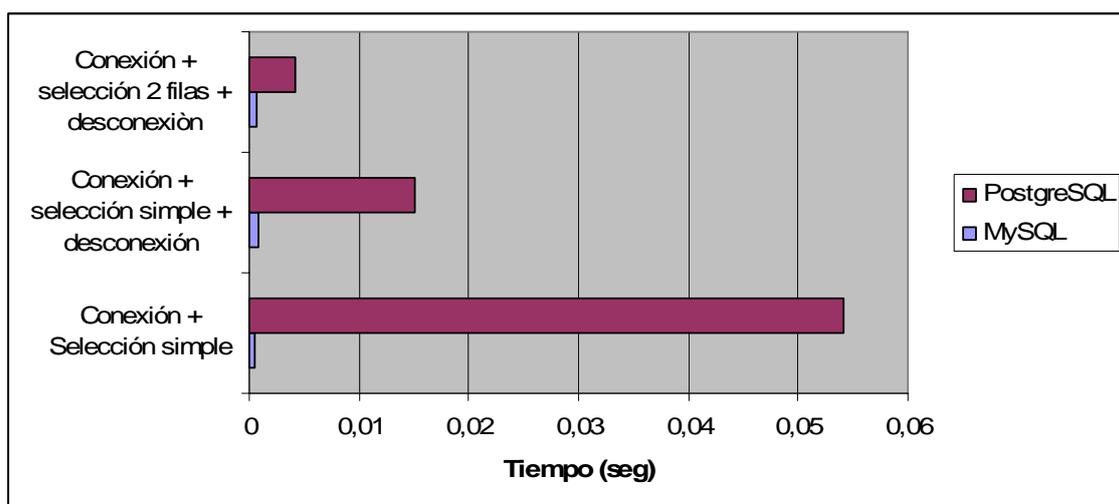


Figura 7.6. Tiempo empleado en tareas de conexión/selección/desconexión.



Prueba de creación de tablas

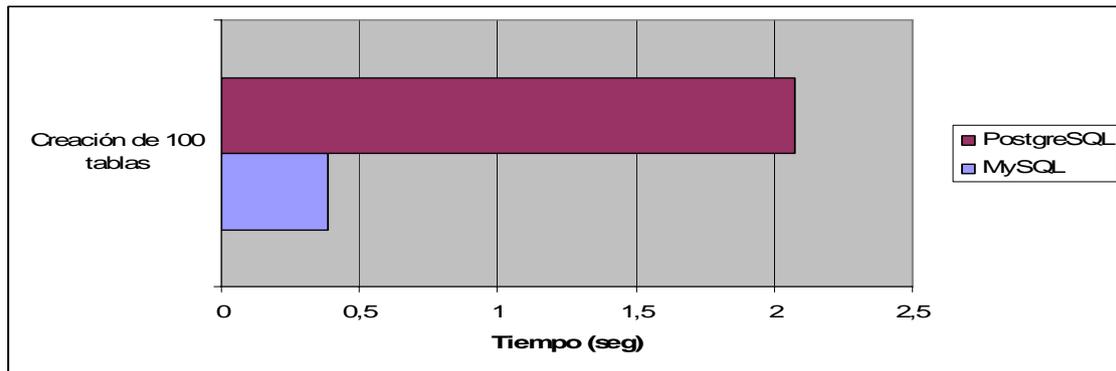


Figura 7.7. Tiempo empleado en tareas de creación de tablas (I).

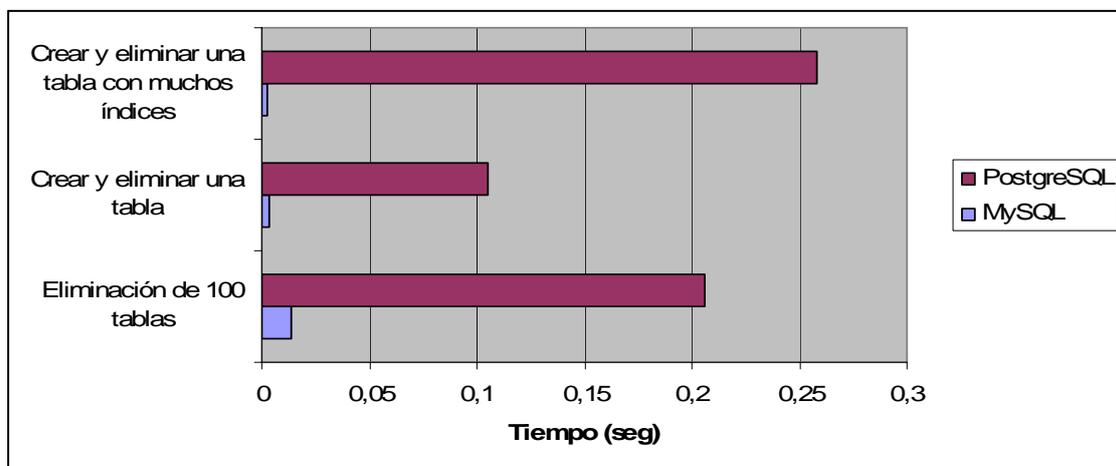


Figura 7.8. Tiempo empleado en tareas de creación de tablas (I).

Prueba de inserción de datos

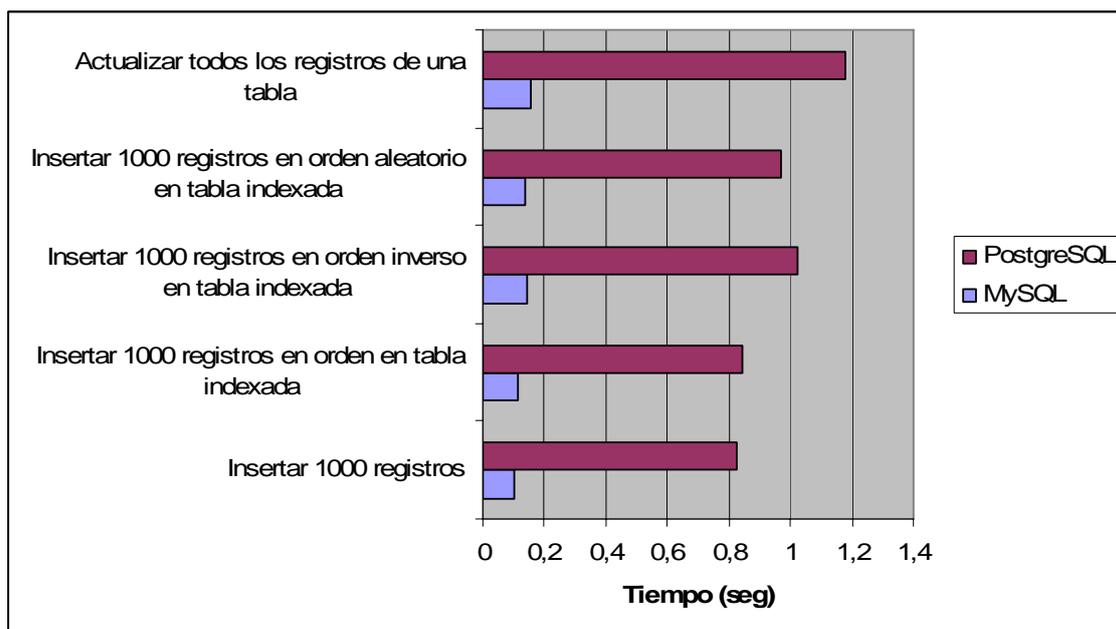


Figura 7.9. Tiempo empleado en tareas de inserción de datos (I).

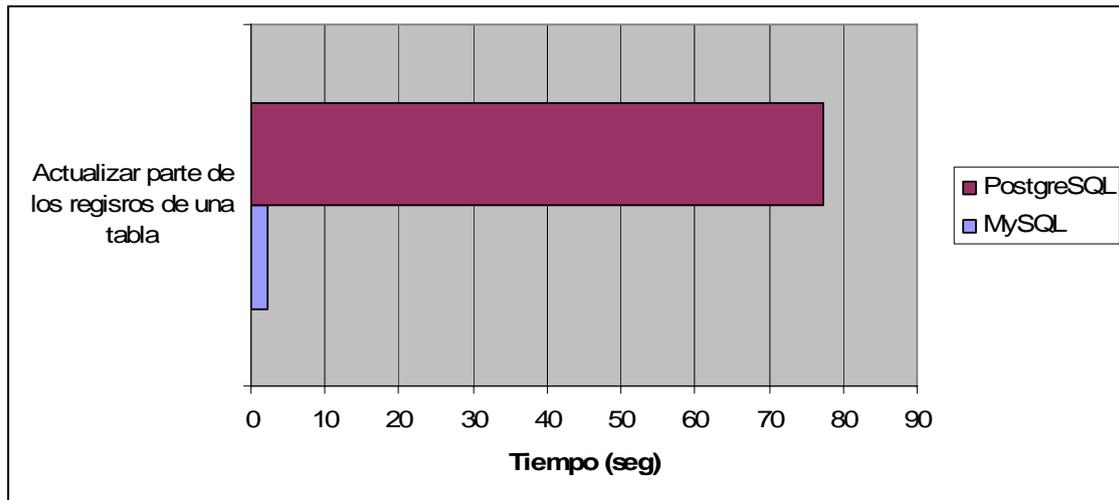


Figura 7.10. Tiempo empleado en tareas de inserción de datos (II).

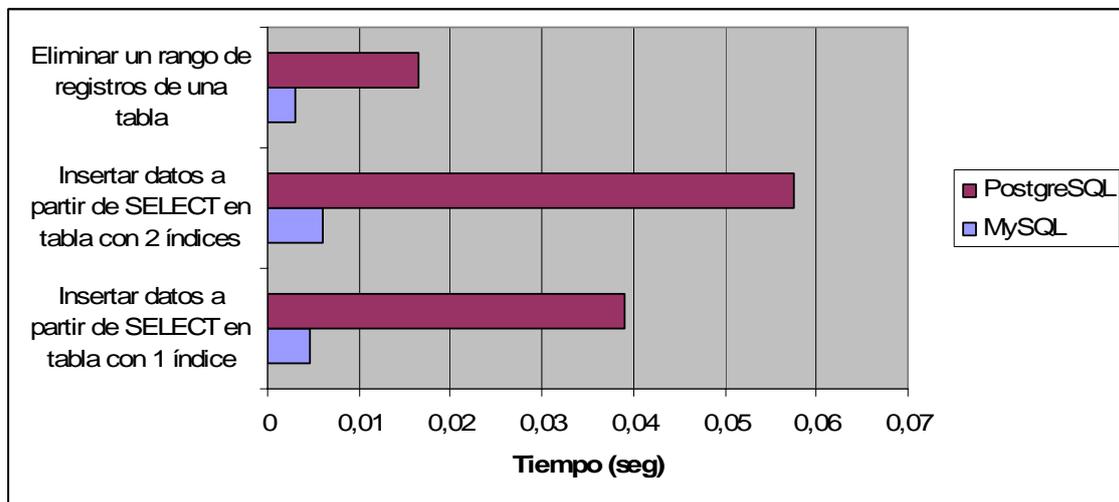


Figura 7.11. Tiempo empleado en tareas de inserción de datos (III).

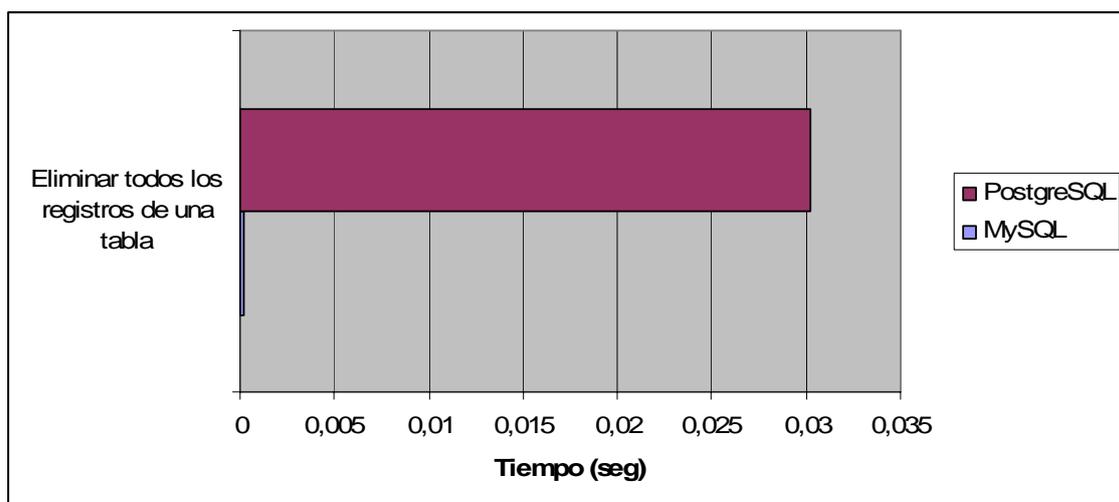


Figura 7.12. Tiempo empleado en tareas de inserción de datos (IV).

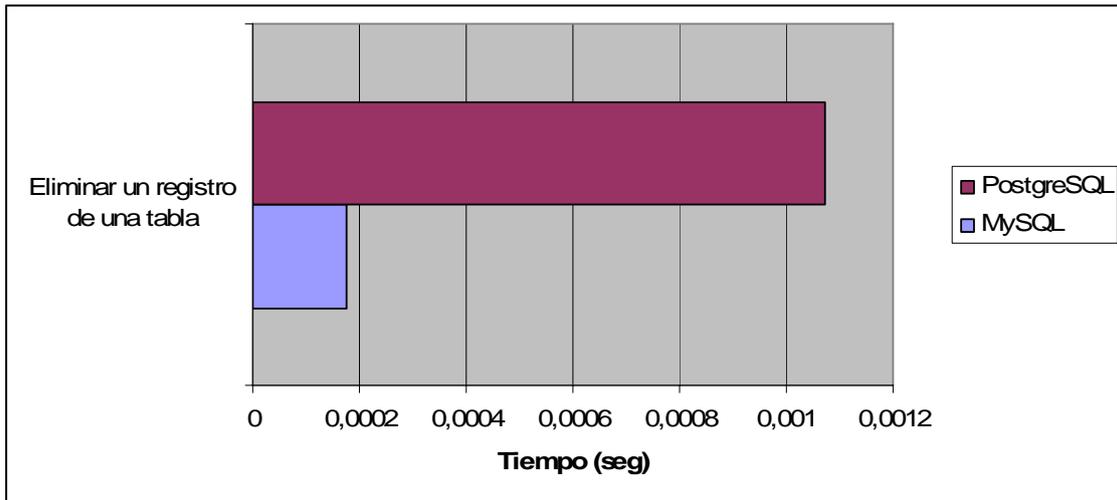


Figura 7.13. Tiempo empleado en tareas de inserción de datos (V).

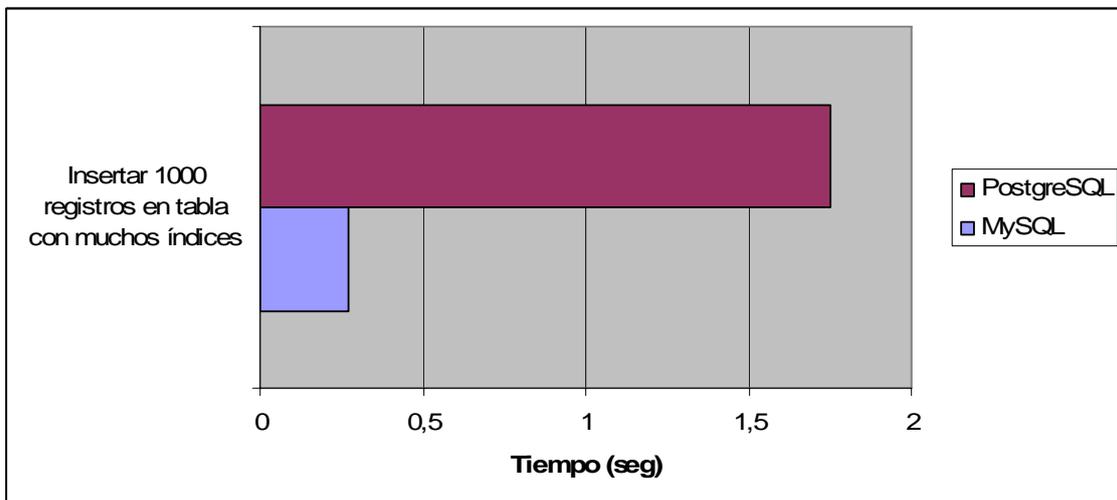


Figura 7.14. Tiempo empleado en tareas de inserción de datos (VI).

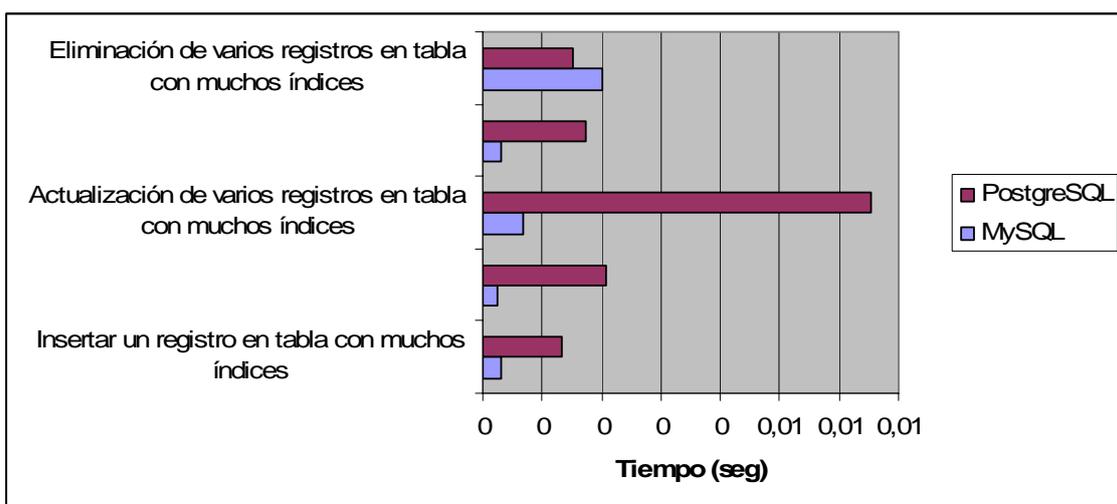


Figura 7.15. Tiempo empleado en tareas de inserción de datos (VII).



Pruebas de consultas SELECT

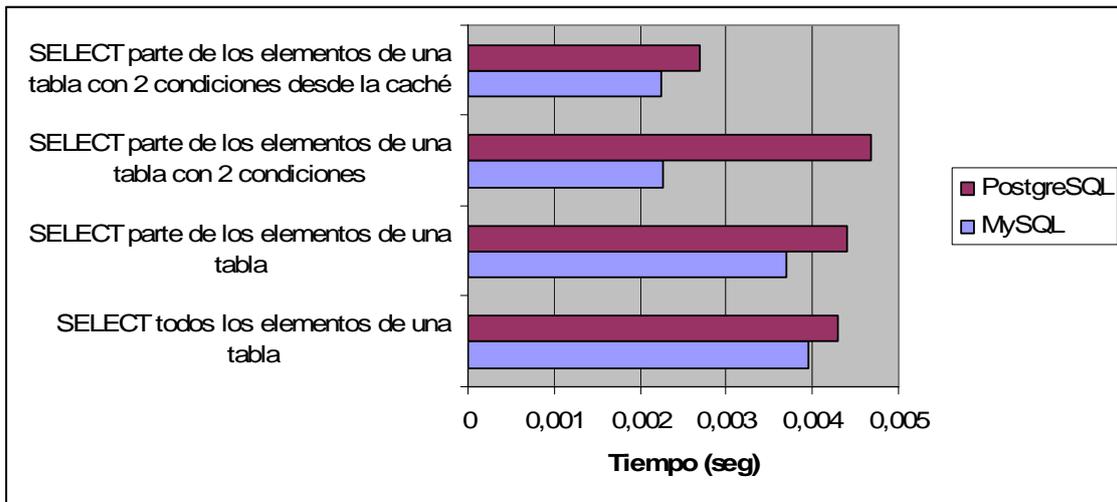


Figura 7.16. Tiempo empleado en consultas de selección (I).

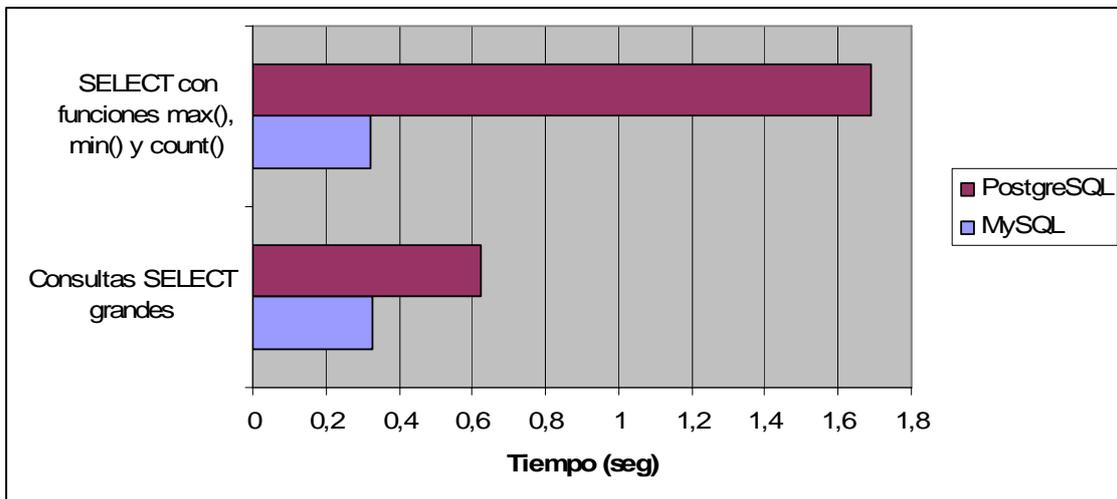


Figura 7.17. Tiempo empleado en consultas de selección (II).

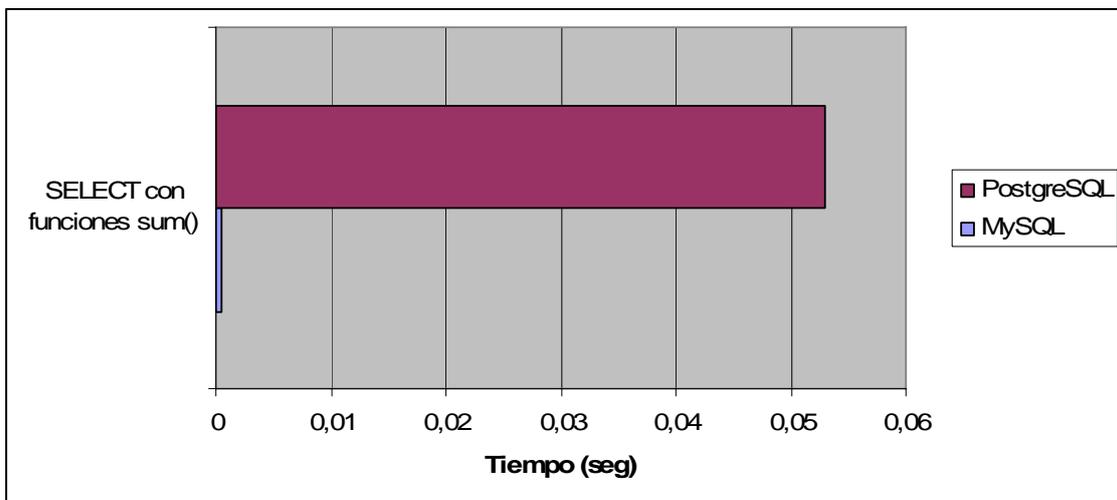


Figura 7.18. Tiempo empleado en consultas de selección (III).

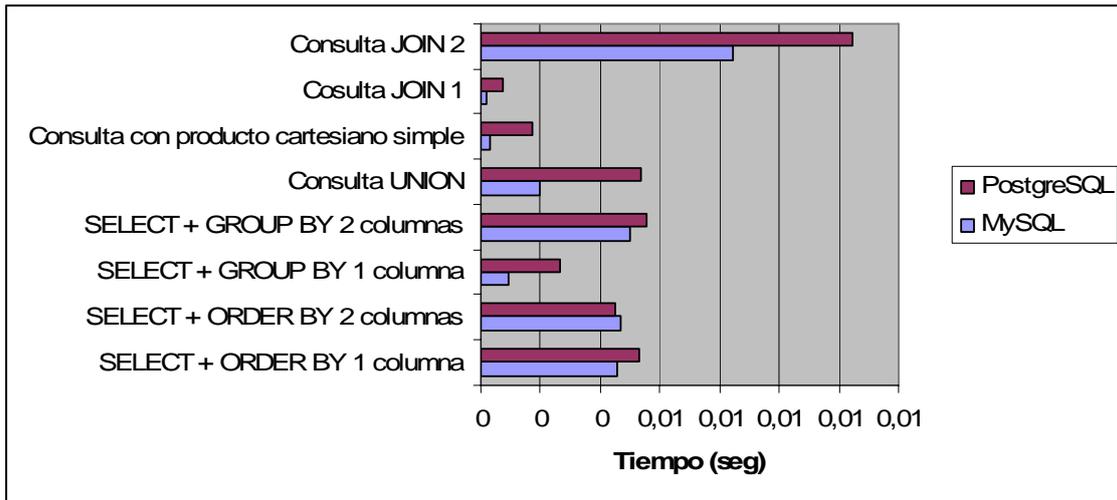


Figura 7.19. Tiempo empleado en consultas de selección (IV).

Pruebas de modificación de tablas

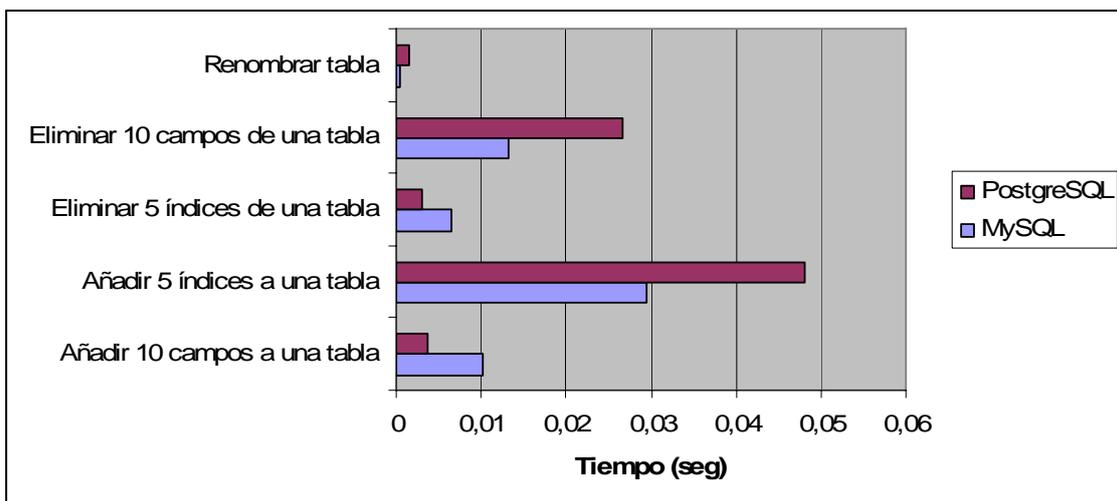


Figura 7.20. Tiempo empleado en tareas de modificación de tablas.

7.2.3. Comentarios.

A la vista de los resultados obtenidos en cada una de las pruebas, se puede concluir que MySQL obtiene mejores tiempos de respuesta en la mayoría de las operaciones. El único test en el que PostgreSQL obtuvo resultados más cercanos a los de MySQL (aún así inferiores) fueron en las pruebas de consultas SELECT. Por tanto, en una aplicación en la que predominaran los accesos a la base de datos a través de sentencias de selección (SELECT), no resultaría desaconsejable el uso de PostgreSQL en el caso de que se quisieran aprovechar las funcionalidades adicionales que este sistema gestor de bases de datos ofrece. De otro modo, si se quisiera una aplicación en la que los tiempos de respuesta fueran un factor primordial, MySQL sería la opción más acertada.

Cabe comentar que estos resultados se han obtenido para un solo usuario conectado a la base de datos, por lo que estas conclusiones serían adecuadas para sistemas con una



carga de usuarios baja. Para ver como afecta el crecimiento de solicitudes concurrentes al rendimiento del sistema se evaluará otra aplicación en el apartado siguiente.

7.3. GESTOR DE TAREAS.

Para continuar con el análisis del rendimiento de los sistemas gestores de bases de datos en aplicaciones reales, se ha buscado una aplicación que basara su funcionamiento en el acceso bases de datos. Esta aplicación consiste en un gestor de tareas, que hace uso de unas librerías existentes, escritas en PHP, para el acceso a bases de datos PostgreSQL y MySQL, por lo que adaptar dicha aplicación para que funcione con ambos sistemas gestores de bases de datos es bastante sencillo. En las líneas siguientes se verá una descripción de la aplicación, así como las medidas de rendimiento de la misma para diferentes valores de carga de usuarios.

7.3.1. Descripción de la aplicación.

La aplicación escogida se trata de un gestor de tareas con interfaz Web. Dicho gestor de tareas está codificado empleando los lenguajes HTML y PHP. Para obtener una idea de las características y funcionalidades de dicha aplicación se puede observar la figura 7.21, que muestra un esquema de la misma.

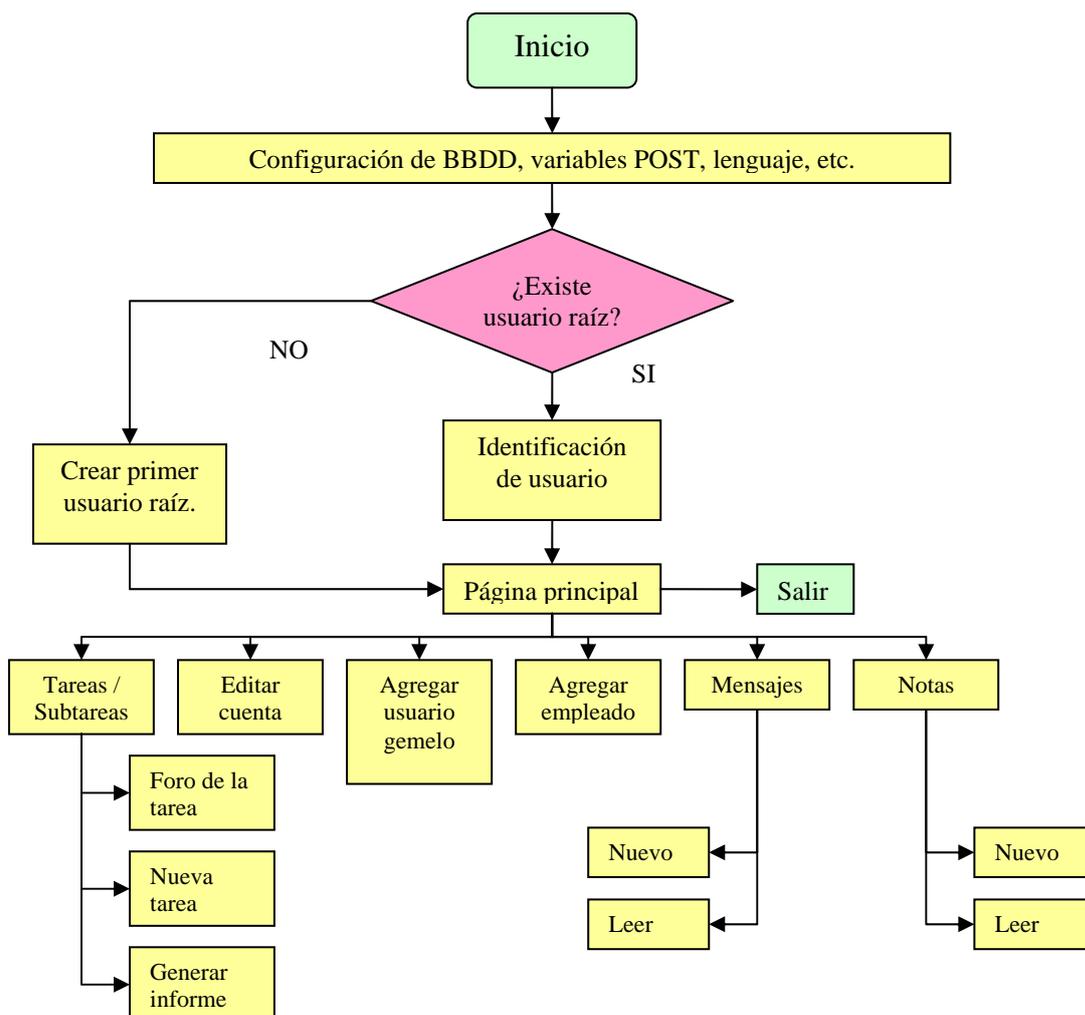


Figura 7.21. Esquema del gestor de tareas.



A la vista del diagrama anterior, las tareas que se pueden llevar a cabo con la aplicación son muy diversas (gestión de empleados, gestión de tareas, mensajería, etc.). Toda la información relativa a los empleados, las tareas, las notas, los mensajes y los foros se almacenan en las bases de datos en diferentes tablas. A continuación se muestran las tablas necesarias para el desempeño de estas funciones.

Tabla *forum*

Esta tabla contiene toda la información referente a los foros asociados a las tareas.

Campo	Tipo	Null	Key	Por defecto	Extra	Descripción
Id	Int(11)		PRI	NULL	Auto-increment	Id del mensaje del foro.
Parent	Int(11)			0		Especifica si es mensaje principal.
Task	Int(11)	SI		NULL		Id de la tarea asociada al hilo del foro.
Autor	Varchar(32)					Autor del mensaje.
Creation_date	Date			0000-00-00		Fecha de creación del post.
Title	Varchar(100)					Título del post.
Text	Text	SI		NULL		Texto del mensaje.

Tabla 7.1. Campos de la tabla "forum".

Tabla *messages*

En esta tabla se almacenan todos los mensajes del servicio de mensajería entre usuarios.

Campo	Tipo	Null	Key	Por defecto	Extra	Descripción
Message_id	Int(11)		PRI	NULL	Auto-increment	Id del mensaje.
From_id	Varchar(32)			0		Id del usuario remitente.
To_id	Varchar(32)	SI		NULL		Id del usuario destinatario.
subject	Varchar(100)	SI		NULL		Asunto.
Message	Text	SI		NULL		Cuerpo del mensaje.
Creation_date	Datetime			0000-00-00 00:00		Fecha y hora de creación del mensaje.

Tabla 7.2. Campos de la tabla "messages".

Tabla *notes*

En esta tabla se almacenan todas las notas, tanto las asociadas a las tareas como las independientes de cada usuario.

Campo	Tipo	Null	Key	Por defecto	Extra	Descripción
Note_id	Int(11)		PRI	NULL	Auto-increment	Id de la nota.
Title	Varchar(100)	SI		NULL		Título de la nota.
Text	Text					Cuerpo de la nota.
Task_id	Int(11)	SI		NULL		Id de la tarea asociada a la



Campo	Tipo	Null	Key	Por defecto	Extra	Descripción
						nota.
User_id	Varchar(32)					Id del usuario que crea la nota.

Tabla 7.3. Campos de la tabla “notes”.

Tabla tree

Esta tabla contiene toda la información referente a la jerarquía de usuarios (en forma de árbol).

Campo	Tipo	Null	Key	Por defecto	Extra	Descripción
Node	Varchar(50)					Nodo del usuario en el árbol de usuarios.
Leaf	Int(11)	SI		NULL		Hoja del usuario en el árbol de usuarios.
Id	Varchar(32)		PRI			Id del usuario.

Tabla 7.4. Campos de la tabla “tree”.

Tabla users

Esta tabla contiene toda la información referente a los usuarios registrados en la aplicación.

Campo	Tipo	Null	Key	Por defecto	Extra	Descripción
Id	Varchar(32)		PRI	0		Id del usuario.
Username	Varchar(16)					Identificador del usuario.
Password	Varchar(16)					Clave del usuario.
Name	Varchar(50)					Nombre del usuario.
Email	Varchar(100)					Correo electrónico del usuario.
Can_add_user	Int(11)			0		¿Puede añadir empleados el usuario?
Active	Int(11)			1		Usuario activo o no activo.

Tabla 7.5. Campos de la tabla “users”.

Tabla tasks

En esta tabla almacena toda la información referente a las tareas

Campo	Tipo	Null	Key	Por defecto	Extra	Descripción
Task_id	Int(11)		PRI	NULL	Auto-increment.	Id de la tarea.
Node	Varchar(20)	SI		NULL		Nodo de la tarea en el árbol de tareas.
Leaf	Int(11)	SI		NULL		Hoja de la tarea en el árbol de tareas.
Master_id	Varchar(32)					Id del usuario que asigna la tarea.
Slave_id	Varchar(32)					Id del usuario al que se le asigna la tarea.
Title	Varchar(120)					Título de la tarea.
Description	Text					Descripción de la tarea.



Campo	Tipo	Null	Key	Por defecto	Extra	Descripción
Inicial_date	Date			0000-00-00		Fecha de inicio prevista de la tarea.
Due_date	Date			0000-00-00		Fecha de finalización prevista de la tarea.
Initiated_date	Date	SI		NULL		Fecha de inicio de la tarea.
Terminated_date	Date	SI		NULL		Fecha de finalización de la tarea.
State	Varchar(10)					Estado de la tarea.
State_feedback	Text	SI		NULL		Descripción del estado de la tarea.
Priority	Varchar(20)					Prioridad de la tarea.
Private	Char(1)	SI		NULL		¿Tarea privada?

Tabla 7.6. Campos de la tabla "tasks".

Dada la estructura de la base de datos a emplear, mostrada en las tablas anteriores, dicha aplicación se puede implementar sin ningún tipo de problema en los dos sistemas a estudio. La diferencia radicará, por tanto, en el rendimiento de la aplicación cuando se emplea con cada uno de los sistemas gestores de base de datos a estudio.

Por tanto, se ha adaptado el código de la aplicación para que la misma sea funcional con ambos sistemas gestores de bases de datos. Esta adaptación simplemente consiste en la utilización de unas librerías existentes en la aplicación, escritas en PHP, para el manejo de bases de datos.

A continuación se muestran una serie de capturas de la aplicación en funcionamiento.

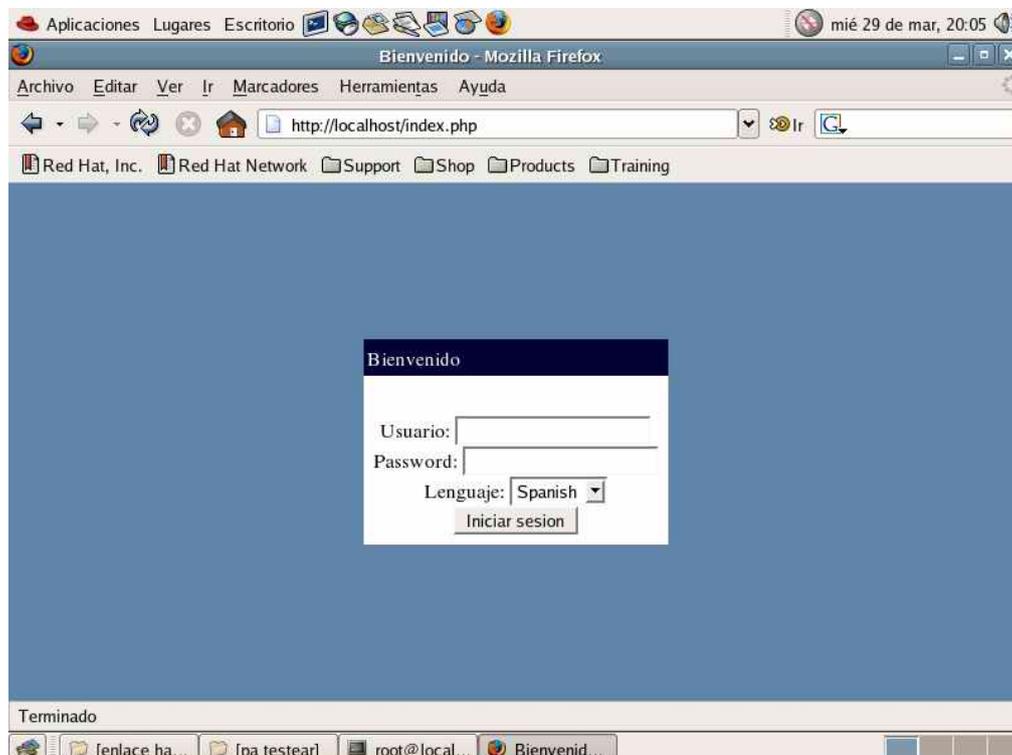


Figura 7.22. Pantalla de login.

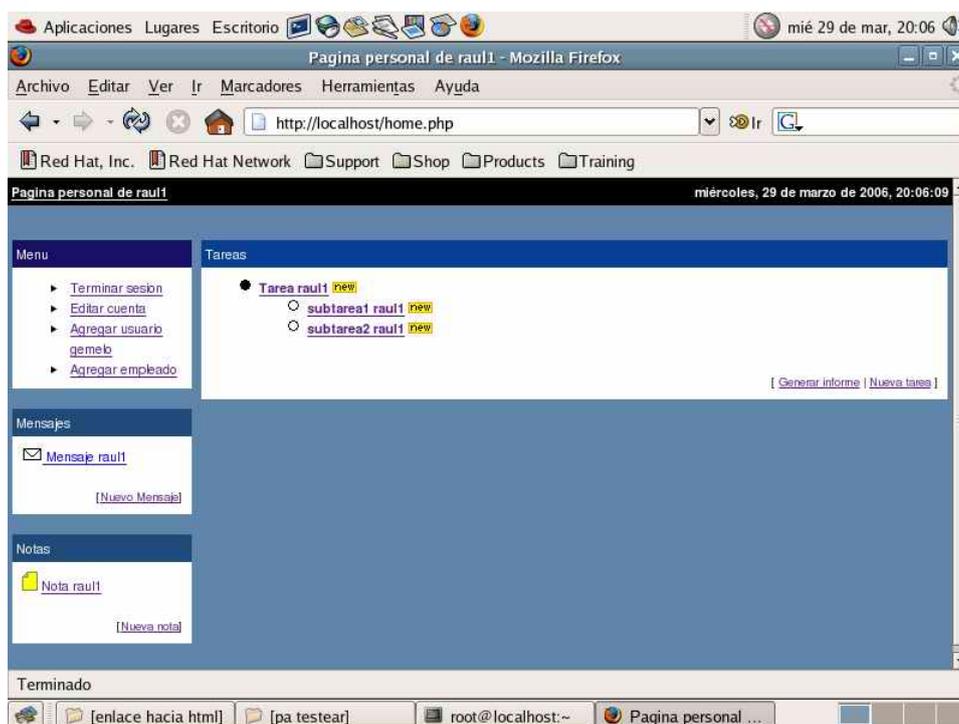


Figura 7.23. Pantalla principal de la aplicación.

Desde esta pantalla principal, se gestionan todas las operaciones que se pueden realizar sobre las tareas, los usuarios, las notas, etc.

Como puede verse en la figura 7.23, el funcionamiento de la aplicación es bastante sencillo e intuitivo. Dicha aplicación ha sido probada con los dos sistemas gestores de bases de datos a estudio (MySQL con motor de almacenamiento MyISAM y PostgreSQL), funcionando correctamente con ambos sistemas. Por tanto, partiendo de que la aplicación es completamente funcional para los dos sistemas gestores de bases de datos, se va a ver cómo se comporta la misma cuando aumenta la carga de usuarios, para así poder decidir qué sistema gestor de bases de datos es más apropiado para emplearlo como *backend* de la aplicación. Esto se discute en el siguiente apartado.

7.3.2. Pruebas de concurrencia.

Anteriormente se vio que la aplicación elegida es apta para el funcionamiento con los sistemas gestores de bases de datos a estudio, y se ha probado dicho funcionamiento, resultando una aplicación completamente operativa. Las situaciones descritas en el apartado anterior se basan en el funcionamiento de la aplicación con un único usuario conectado al sistema.

Dado el carácter y las funcionalidades de la aplicación, es lógico pensar que la misma puede usarse en un ambiente empresarial, de forma que serían muchos los usuarios trabajando simultáneamente con la aplicación. Esto lleva a pensar que no sólo es necesario que la aplicación funcione con un sistema gestor de base de datos, sino que también será deseable que se rendimiento no se vea empobrecido cuando sean muchos los usuarios trabajando simultáneamente con la misma. Por ello, se ha diseñado una batería de pruebas para evaluar el comportamiento de la aplicación ante un número variable de usuarios concurrentes.



7.3.2.1. Diseño de las pruebas.

Para realizar el análisis de concurrencia de la aplicación se ha empleado la utilidad Apache Bench. Apache Bench es un programa que viene incluido con el servidor Web Apache, y que permite evaluar el rendimiento de aplicaciones Web alojadas en un servidor Apache. La forma de proceder ha sido la siguiente:

- 1) Aislar las partes del código de la aplicación que acceden a la base de datos.
- 2) Subir dichas partes al servidor Apache, y comprobar que son operativas.
- 3) Ejecutar el programa Apache Bench sobre los fragmentos de código aislado, para varios valores de número de usuarios concurrentes.
- 4) Tabular y presentar los resultados.

Tanto en el apartado anterior como en este, se ha empleado el servidor Web Apache para albergar tanto la aplicación como los fragmentos aislados de la misma que se van a estudiar.

7.3.2.2. Aislamiento del código de la aplicación.

Para poder evaluar la aplicación con el programa Apache Bench, se ha aislado de la misma la parte del código que accede a bases de datos. Dichas secciones de códigos han sido las mismas para los dos sistemas gestores de bases de datos a estudio, para hacer de esta forma las pruebas con ambos sistemas en las mismas condiciones. Se han suprimido también aquellas partes de código que insertan algún tipo de valor aleatorio en tablas, de forma que los datos empleados en los análisis han sido exactamente los mismos en ambos casos.

Con todo esto, se obtienen una serie de ficheros en PHP que serán “atacados” por el banco de pruebas Apache Bench. La estructura resultante de ficheros y directorios que contienen las partes aisladas del código para su análisis se muestra en la figura 7.24.

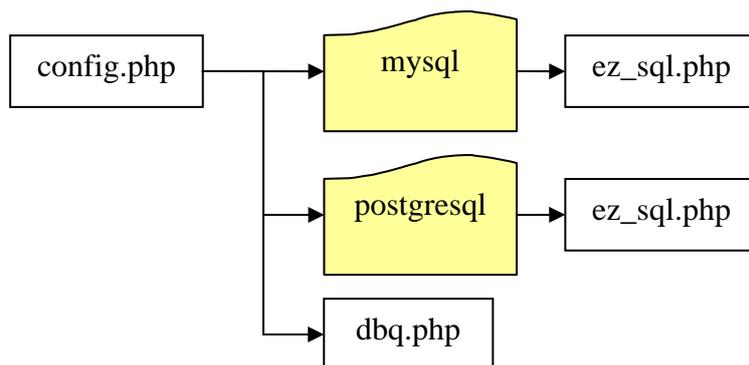


Figura 7.24. Esquema del código aislado de la aplicación

El contenido de los ficheros mostrados en la figura 7.24, que alberga el código aislado de la aplicación, se puede consultar en el CDrom que acompaña a este documento. De igual manera, en dicho CDrom también se incluye el código fuente de la aplicación.

Los archivos `ez_sql.php`, contenidos en las carpetas `mysql` y `postgresql`, contienen el código en PHP que define una clase con una serie de funciones miembro para el manejo de bases de datos. Este conjunto de funciones constituye una librería bastante completa



para la realización de tareas sobre bases de datos. Además, en dicho archivo se crea una instancia de la clase, llamada *db*, para ser empleada por los demás ficheros PHP.

El fichero *dbq.php* contiene el código de las funciones empleadas en *config.php*. Este código hace uso de las funciones miembro de la clase definida en el archivo *ez_sql.php*.

Si se observa el contenido del fichero *config.php* se puede ver que dicho fichero llama a todas las funciones que acceden a la base de datos con sus correspondientes parámetros. Por tanto, al ejecutar este fichero se ejecutan una vez todas las operaciones disponibles en la aplicación que acceden a la base de datos. Este fichero será evaluado por el programa Apache Bench, que simulará un determinado número de usuarios ejecutándolo a la vez.

Esta estructura de ficheros y directorios se ha subido al servidor apache existente en la máquina en la que se van a ejecutar las pruebas, con objeto de ejecutar y evaluar el comportamiento de dichos *scripts* en PHP con el banco de pruebas del servidor Apache.

7.3.2.3. Apache Bench.

Apache Bench es un banco de pruebas que viene instalado en la mayoría de las distribuciones de Unix/Linux que contienen el servidor Apache. Apache Bench permite analizar el rendimiento de aplicaciones Web ante diferentes valores de carga de usuarios, dando una indicación de cuántas solicitudes por segundo será capaz de atender el servidor. Este programa se deberá utilizar desde la línea de comandos, y su sintaxis es la siguiente:

```
ab [opciones] [http://]nombre_host[:port]/ruta/fichero
```

En la línea anterior, *nombre_host* es el nombre del host del servidor apache, *port* es el puerto que emplea dicho servidor, *ruta* es el camino hacia el fichero *fichero* que se desea evaluar. Este fichero puede ser, por ejemplo, un fichero con extensión *htm*, *html*, *php*, etc.

Las opciones disponibles para este comando se muestran en la tabla 7.7.

Opción	Descripción
-k	Habilita la característica HTTP <i>KeepAlive</i> para realizar múltiples solicitudes con una sola sesión. Por defecto esta desactivado.
-i	Usa una HTTP 'HEAD' en vez del método GET. No puede ser mezclado con POST.
-n <i>solicitudes</i>	El número de solicitudes que se harán en cada sesión. Por defecto solo una.
-t <i>tiempolimito</i>	Tiempo límite de ejecución de las pruebas. Por defecto ilimitado.
-c <i>conurrencia</i>	Numero de solicitudes simultáneas que se realizarán. Por defecto sólo una.
-p <i>ficheroPOST</i>	Fichero que contendrá los datos que se enviarán al servidor Apache por el método POST.
-A <i>Autorizacion nombre_usuario:Password</i>	Credenciales de autorización para el acceso al servidor.
-P <i>Autorización-Proxy nombre_usuario:Password</i>	Credenciales de autorización para el acceso al servidor a través de Proxy.
-C <i>Cookie nombre=valor</i>	Añade una línea 'Cookie:' a la solicitud.



Opción	Descripción
-H <i>Header cadena</i>	Añade una cabecera extra a la solicitud.
-T <i>content-type</i>	La cabecera <i>content-type</i> para usar con POST.
-v	Establece el nivel de depuración.
-w	Imprime los resultados en tablas HTML.
-x <i>atributos</i>	Cadena para usar como atributos para las tablas.
-X <i>proxy:port</i>	Usa el servidor Proxy especificado, corriendo en el puerto especificado.
-y <i>atributos</i>	Cadena de atributos para <tr>.
-z <i>atributes</i>	Cadena de atributos para <td>.
-V	Muestra el número de versión y sale.
-h	Muestra la ayuda del comando.

Tabla 7.7. Opciones de Apache Bench.

Para las pruebas que se han realizado sobre la aplicación, se ha escogido un valor de 100 para el número de solicitudes que se realizan en cada conexión (parámetro opcional -n) y el número de usuarios concurrentes se ha variado entre 1 usuario y 300 usuarios, con un incremento de 50. En el apartado siguiente se mostrarán los resultados obtenidos.

Al introducir el comando en el *shell* de Linux, tras unos instantes se mostrarán por pantalla los resultados del test. La información que se muestra es la siguiente:

- Ruta del fichero.
- Tamaño del fichero analizado.
- Nivel de concurrencia (parámetro de entrada).
- Solicitudes fallidas.
- Errores de escritura.
- Tamaño total transferido.
- HTML transferido.
- Solicitudes por segundo.
- Tiempos por consulta.
- Tasa de transferencia.
- Tiempos de conexión al servidor Web.
- Porcentaje de solicitudes atendidas en un tiempo determinado.

Como se puede ver, se muestra información bastante completa de lo que ocurre en el servidor. En el siguiente apartado se mostrarán los resultados obtenidos en las diferentes pruebas.

7.3.2.4. Resultados obtenidos.

Para evaluar el rendimiento de la aplicación con uno u otro sistema gestor de bases de datos, se van a realizar una serie de pruebas con la aplicación Apache Bench sobre los fragmentos de código de la aplicación. Recuerde que dicha porción de código contiene todos los accesos que se realizan a la base de datos. Se ha escogido que cada usuario concurrente haga un total de 100 solicitudes. El número de usuarios se variará entre 1 y 300. En el ANEXO D se muestran los listados completos de los resultados obtenidos para cada nivel de concurrencia.

Los resultados obtenidos se muestran en las siguientes gráficas:

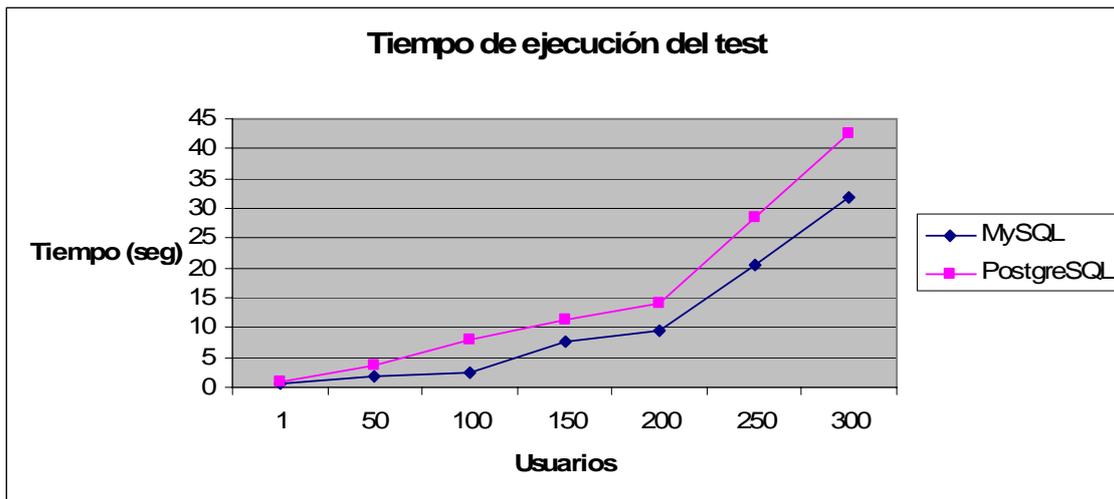


Figura 7.25. Tiempo empleado en la ejecución de la prueba.

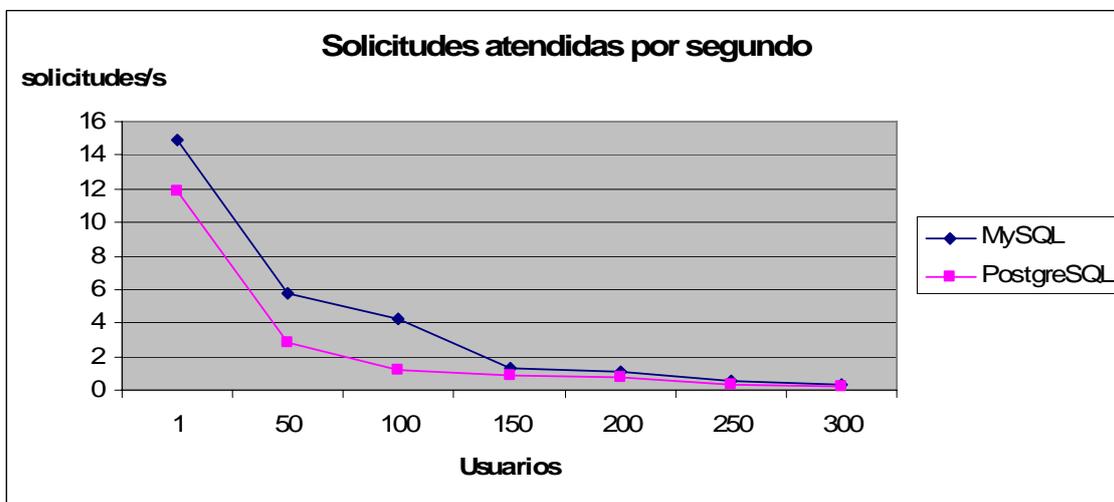


Figura 7.26. Solicitudes atendidas por segundo en función del número de usuarios.

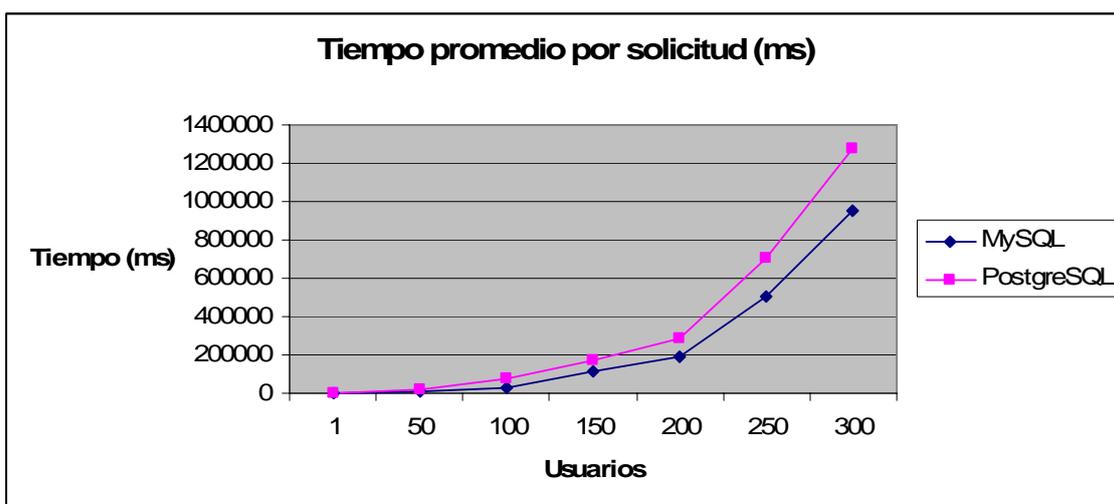


Figura 7.27. Tiempo promedio por solicitud en función del número de usuarios.

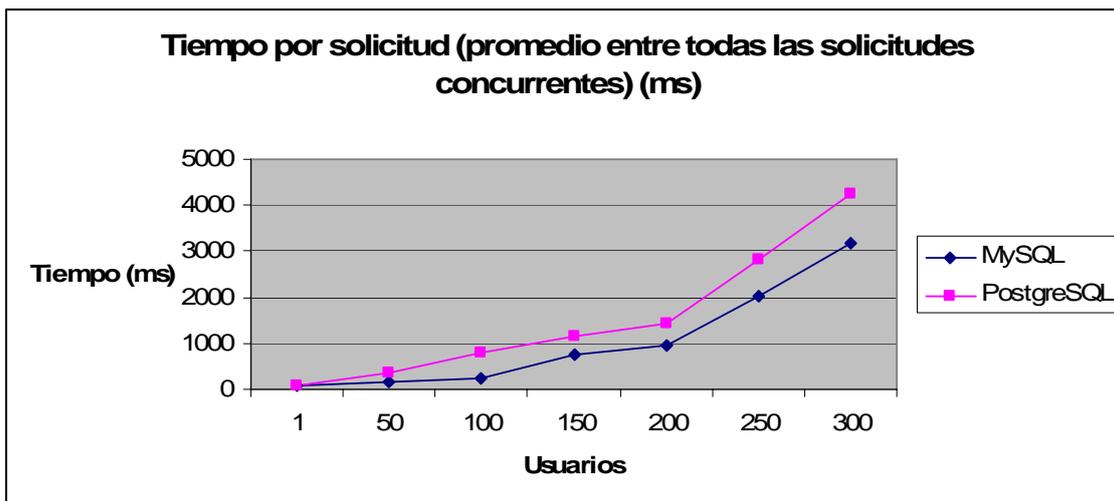


Figura 7.28. Tiempo por solicitud (promedio entre todas las solicitudes concurrentes) en función del número de usuarios.

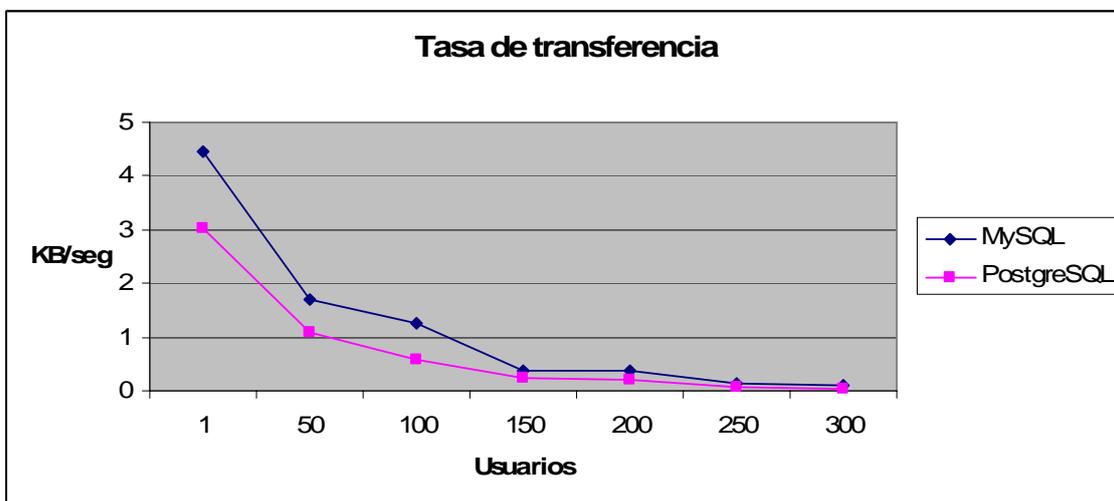


Figura 7.29. Tasa de transferencia en función del número de usuarios.

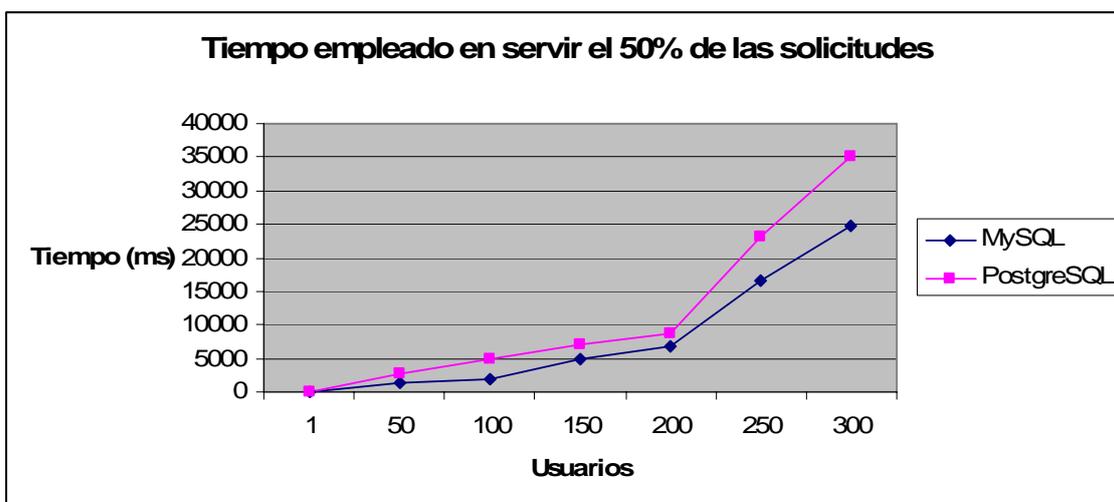


Figura 7.30. Tiempo empleado en servir el 50% de las solicitudes en función del número de usuarios.

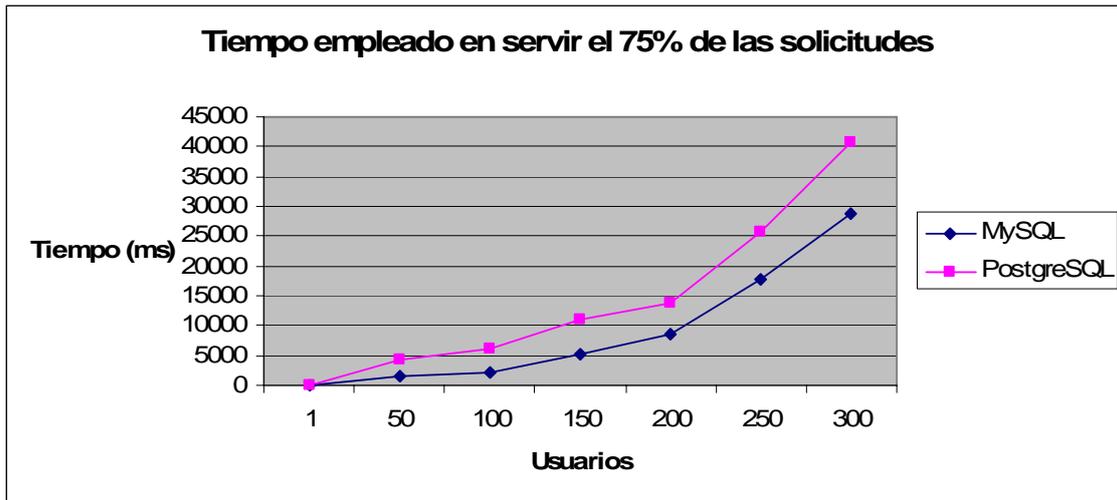


Figura 7.31. Tiempo empleado en servir el 75% de las solicitudes en función del número de usuarios.

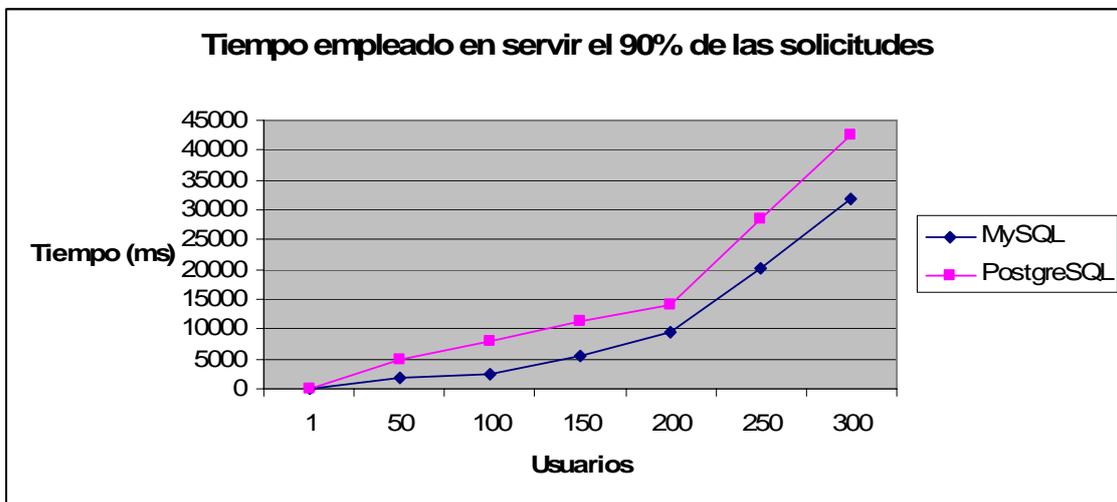


Figura 7.32. Tiempo empleado en servir el 90% de las solicitudes en función del número de usuarios.

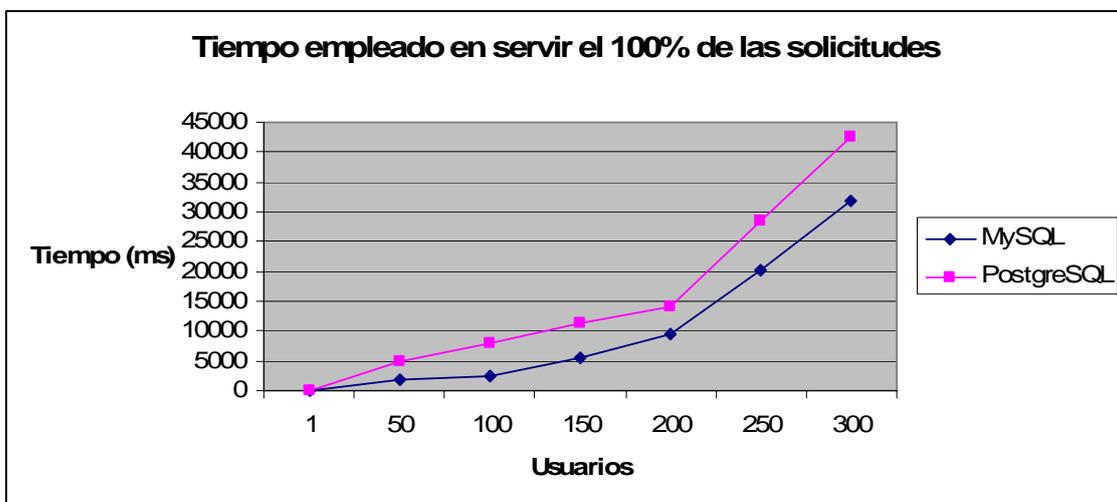


Figura 7.33. Tiempo empleado en servir el 100% de las solicitudes en función del número de usuarios.



7.3.2.5. Comentarios.

A continuación se van a comentar los resultados obtenidos a partir de las gráficas del apartado anterior. En la gráfica correspondiente al tiempo de ejecución de los test, se ve que dicho tiempo aumenta a medida que se incrementa la carga de usuarios, siendo siempre inferior el correspondiente a MySQL.

La segunda gráfica demuestra que en término medio se pueden atender más solicitudes por segundo empleando MySQL como *backend* de la aplicación. Este número de solicitudes atendidas por segundo disminuye con la carga de usuarios, pero siempre es mayor para el caso de MySQL, con independencia del número de usuarios.

Los tiempos promedio por solicitud son mayores en el caso de emplear PostgreSQL como *backend*, lo cual indica que las solicitudes se cursan más lentamente empleando este sistema gestor de bases de datos. Este hecho ocurre para cualquier número de usuarios.

La tasa de transferencia de datos, que decrece siempre a medida que se aumenta el nivel de concurrencia, es siempre mayor para el caso de que se utilice la aplicación con MySQL.

Por último, los tiempos promedio que se tardan en servir el 50%, el 75%, el 90% y el 100% de las solicitudes son menores en el caso de que se emplee MySQL con la aplicación, para cualquier valor de carga de usuarios.

A la vista de los resultados anteriores, se puede concluir que el uso de MySQL como *backend* de la aplicación proporciona un mayor rendimiento a la misma. Dado que la aplicación se puede implementar sin ningún tipo de problema o restricción con ambos sistemas gestores de bases de datos, resultaría más apropiado emplear MySQL, que proporciona unos tiempos de respuesta inferiores y un mejor comportamiento cuando la carga de usuarios es alta.