



UNIVERSIDAD DE VALPARAÍSO
Facultad de Ingeniería
Escuela de Ingeniería Civil en Informática
Ingeniería Civil Informática

TAREA 2

Camilo Antonio Ravelo Durán

Ian Franco Quiroga Farías

camilo.ravelod@alumnos.uv.cl

ian.quirogafd@alumnos.uv.cl

29 de mayo de 2015

Profesor: Pedro Hernández

Índice

1. Introducción	3
2. Consultas a realizar	4
3. Descripción de la solución	7
4. Pruebas de Carga y Comparación	8
5. Tabla de Comparación de las pruebas	15
6. Conclusión	16
Referencias	17

1. Introducción

En el presente trabajo se presenta una aplicación realizada en Php y Mysql utilizando el framework de desarrollo Codeigniter, el cual es un entorno de desarrollo para crear sitios web dinámicos en PHP y funciona de manera mas rápida que muchos otros entornos.

La aplicación cuenta con 3 consultas a una Base de Datos integrada, las cuales van subiendo en grado de dificultad o complejidad.

Posteriormente se realizan pruebas de carga usando la herramienta Jmeter, la cual es una herramienta de testing.

Finalmente se realizan 5 iteraciones a cada consulta realizada con el fin de optimizar la solución

2. Consultas a realizar

Consulta 1: La consulta 1 muestra el resultado de los nombres de los departamentos de la tabla "departments":

```
$query = $this->db->query('SELECT dept_name FROM departments');
```

Figura 1: Consulta 1.



nombre_departamento
Customer Service
Development
Finance
Human Resources
Marketing
Production
Quality Management
Research
Sales

Figura 2: Resultado Consulta 1.

Consulta 2: La consulta 2 muestra el nombre concatenado con el apellido, además el cumpleaños y finalmente el genero filtrado por masculino. Todos los campos pertenecientes a la tabla .employees”

```
$query = $this->db->query("SELECT concat_ws(' ',first_name,last_name) as nombre_apellido, birth_date, gender from employees where gender='M' AND birth_date='1954-01-01'");
```

Figura 3: Consulta 2.



nombre_apellido	birth_date	gender
Georgi Facello	1953-09-02	M
Lillian Haddadi	1953-01-23	M
Mayuko Warwick	1952-12-24	M
Shahaf Famili	1952-07-08	M
Yongqiao Berztiss	1953-04-03	M
Alain Chappellet	1953-02-08	M
Zvonko Nyanchama	1952-06-29	M
Hidefumi Caine	1953-07-28	M
Kwee Schusler	1952-11-13	M
Claudi Stavenow	1953-01-07	M
Remzi Waschkowski	1952-02-27	M
Akemi Birch	1953-11-26	M
Eben Aingworth	1952-08-29	M
Lunjin Giveon	1952-04-07	M
Magdalena Eldridge	1952-02-19	M
Diederik Siprelle	1953-04-15	M
Debatosh Khasidashvili	1953-10-18	M
Karsten Szmurlo	1952-09-17	M
Shigehito Kropatsch	1953-07-16	M
Pranav Furedi	1953-05-28	M
Genta Kolvik	1952-05-02	M
Ulises Takanami	1952-02-10	M
Remko Maccarone	1952-04-01	M
Moty Kusakari	1953-05-13	M
Hercules Benzmueller	1953-11-27	M
Kauko Birjandi	1953-07-08	M

Figura 4: Resultado Consulta 2.

Consulta 3: En la consulta 3 se establece una relación de 4 tablas en las cuales las tablas involucradas son EMPLOYEES del cual se muestra el campo emp_no, first_name. La segunda corresponde a SALARY de la cual se muestra el campo salary que corresponde al salario del empleado. La tercera tabla utilizada es DEPT_EMP del cual se rescata el dept_no que es el numero del departamento. Por último la cuarta tabla involucrada es DEPARTMENTS donde se obtiene el campo dept_name que corresponde al nombre del departamento. Todo esto es filtrado por emp_no < 23000 y agrupado por el campo emp_no.

```
$query = $this->db->query(
    "SELECT e.emp_no as emp_no, e.first_name as first_name, s.salary as salary, d.dept_no as dept_no, dep.dept_name as dept_name
    FROM employees e, salaries s, dept_emp d, departments dep
    WHERE e.emp_no=s.emp_no AND s.emp_no < 23000 AND e.emp_no=d.emp_no AND d.dept_no=dep.dept_no GROUP BY e.emp_no");
```

Figura 5: Consulta 3.

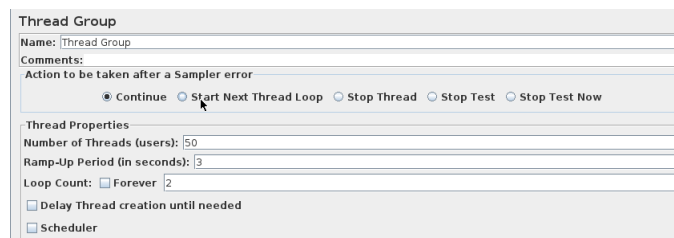
Nº Empleado	Nombre	Salario	Nº Departamento	Nombre Departamento
10001	Georgi	60117	d005	Development
10002	Bezael	65828	d007	Sales
10003	Parto	40006	d004	Production
10004	Chrstian	40054	d004	Production
10005	Kyoichi	78228	d003	Human Resources
10006	Anneke	40000	d005	Development
10007	Tzvetan	56724	d008	Research
10008	Saniya	46671	d005	Development
10009	Sumant	60929	d006	Quality Management
10010	Duangkaew	72488	d004	Production
10011	Mary	42365	d009	Customer Service
10012	Patricio	40000	d005	Development
10013	Eberhardt	40000	d003	Human Resources
10014	Berni	46168	d005	Development
10015	Guoxiang	40000	d008	Research
10016	Kazuhito	70889	d007	Sales
10017	Cristinel	71380	d001	Marketing
10018	Kazuhide	55881	d004	Production
10019	Lillian	44276	d008	Research
10020	Mayuko	40000	d004	Production
10021	Ramzi	55025	d005	Development
10022	Shahaf	40000	d005	Development
10023	Bojan	47883	d005	Development
10024	Suzette	83733	d004	Production
10025	Prasadram	40000	d005	Development
10026	Yongqiao	47585	d004	Production

Figura 6: Resultado Consulta 3.

3. Descripción de la solución

En la siguiente apartado se muestran los resultados de las pruebas de carga realizadas con la herramienta Jmeter, para realizar estas pruebas se utilizó el servidor apache proporcionado por la herramienta Wamp el cual fue accedido remotamente desde una Máquina Virtual.

La configuración de las Pruebas se aprecia en la Figura 8



The image shows the 'Thread Group' configuration window in Apache JMeter. It contains the following fields and options:

- Name:** Thread Group
- Comments:** (empty text area)
- Action to be taken after a Sampler error:** A row of radio buttons with the following options:
 - ☒ Continue
 - ☐ Start Next Thread Loop
 - ☐ Stop Thread
 - ☐ Stop Test
 - ☐ Stop Test Now
- Thread Properties:** A section containing:
 - Number of Threads (users):** 50
 - Ramp-Up Period (in seconds):** 3
 - Loop Count:** ☐ Forever, ☒ 2
 - ☐ Delay Thread creation until needed
 - ☐ Scheduler

Figura 7: Configuración de las Pruebas.

4. Pruebas de Carga y Comparación

Prueba 1 sin modificar, utilizando las 3 consultas:

```
PID USER      PR  NI  VIRT  RES  SHR  S  %CPU  %MEM    TIME+  COMMAND
2674 root        20   0 704m 252m  21m  S   15.8  12.5   32:54.58 java
```

Figura 8: Prueba 1- Consumo CPU-MEMORIA .

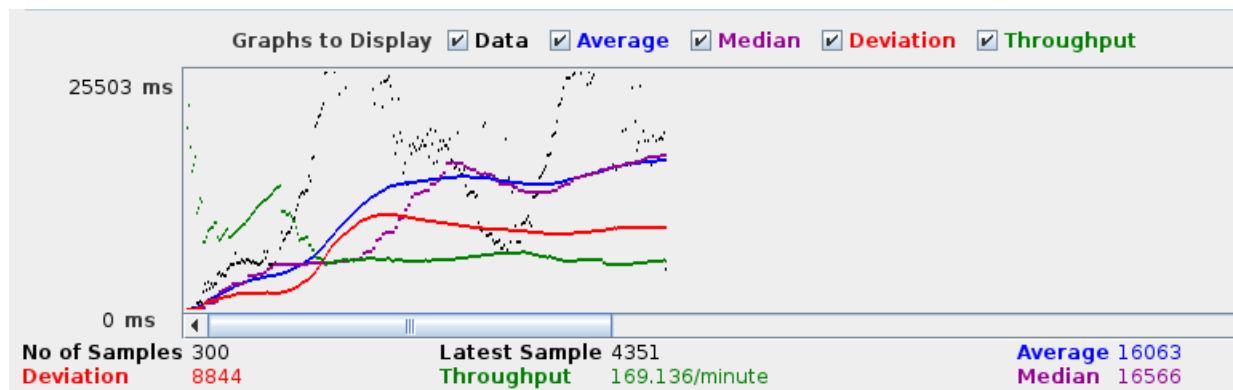


Figura 9: Prueba 1- Gráfico.

Label	# Samples	Average	Min	Max	Std. Dev.	Error %	Throug...	KB/sec
167 /tarea2/index.php/consulta1/hola	100	8177	96	26117	5956.73	0.00%	1.5/sec	1.07
170 /tarea2/index.php/consulta1	100	18825	2861	33961	7862.51	2.00%	59.6/min	1594.29
166 /tarea2/index.php/consulta1/chao	100	21187	4351	34866	6425.51	8.00%	58.2/min	978.67
TOTAL	300	16063	96	34866	8844.01	3.33%	2.8/sec	2457.23

Figura 10: Prueba 1- Tabla de Resultado.

En la tabla de resultados Figura 10 la primera fila es la consulta 1, la segunda fila es la consulta 2 y por último la consulta 3 en la tercera fila.

Prueba 2: Para la iteración 2 se ocupó la sentencia INNER JOIN en MySQL la cual combina cada fila una tabla con cada fila de la otra tabla, seleccionando aquellas filas que cumplan una determinada condición.


```
" SELECT e.emp_no as emp_no, e.first_name as first_name, s.salary as salary, d.dept_no as dept_no,
dep.dept_name as dept_name |FROM employees e INNER JOIN  salaries s on e.emp_no=s.emp_no
INNER JOIN dept_emp d on e.emp_no=d.emp_no
INNER JOIN departments dep on d.dept_no=dep.dept_no
WHERE e.emp_no < 20000 AND s.salary >109000  GROUP BY e.emp_no ");
```

Figura 11: Prueba 2- Consulta.

```
PID USER      PR  NI  VIRT  RES  SHR S %CPU %MEM    TIME+  COMMAND
2561 root        20   0 701m 186m 19m S 30.0  9.2   1:10.27 java
678F      .         0   0 877k  84k  0m S   0.0  0.3    0:00.54 java
```

Figura 12: Prueba 2- Consumo CPU - MEMORIA.

Label	# Samples	Average	Min	Max	Std. Dev.	Error %	Throughput	KB/sec	Avg. Bytes
5 /tarea2/i...	100	6851	96	12066	3785.02	0.00%	1.8/sec	1.28	729.5
6 /tarea2/i...	100	17148	10332	22575	3577.57	8.00%	1.4/sec	2067.07	1543718.4
7 /tarea2/i...	100	9201	795	17740	4773.24	0.00%	1.6/sec	36.29	23494.1
TOTAL	300	11067	96	22575	6004.25	2.67%	4.1/sec	2070.22	522647.3

Figura 13: Prueba 2- Tabla de Resultado.

Prueba 3: Se modifico la Base de Datos utilizando la indexación de los datos almacenados en las tablas, quedando la sentencia de la siguiente forma respectivamente.

- **Consulta 1:** ALTER TABLE employees ADD INDEX indice_names (gender, birth_date)
- **Consulta 2:** ALTER TABLE departments ADD INDEX indice_depto_name
- **Consulta 3:** ALTER TABLE salaries ADD INDEX indice_salaries (salary)

PID	USER	PR	NI	VIRT	RES	SHR	S	%CPU	%MEM	TIME+	COMMAND
2674	root	20	0	705m	304m	21m	S	34.4	15.0	37:49.91	java

Figura 14: Prueba 3- Consumo CPU-MEMORIA.

Label	# Samples	Average	Min	Max	Std. Dev.	Error %	Throug...	KB/sec	Avg. Bytes
167 /tarea2/index.php/consulta1/hola	100	14071	197	27207	8019.15	0.00%	1.0/sec	0.74	729.5
170 /tarea2/index.php/consulta1	100	23187	6780	32861	7040.30	7.00%	48.0/min	1220.30	1560469.8
166 /tarea2/index.php/consulta1/chao	100	20402	5111	28079	4776.19	0.00%	46.2/min	843.57	1122395.1
TOTAL	300	19220	197	32861	7753.14	2.33%	2.2/sec	1903.98	894531.5

Figura 15: Prueba 3- Tabla de Resultado.

En la tabla de resultados Figura 15 la primera fila es la consulta 1, la segunda fila es la consulta 2 y por último la consulta 3 en la tercera fila.

Prueba 4: El método para mejorar el rendimiento de las consultas fue la habilitación de la función Caché de Codeigniter:

```
'cache_on' => TRUE,
```

Figura 16: Prueba 4- Habilidad función caché.

PID	USER	PR	NI	VIRT	RES	SHR	S	%CPU	%MEM	TIME+	COMMAND
2674	root	20	0	705m	304m	21m	S	59.3	15.0	41:40.22	java

Figura 17: Prueba 4- Consumo CPU-MEMORIA.

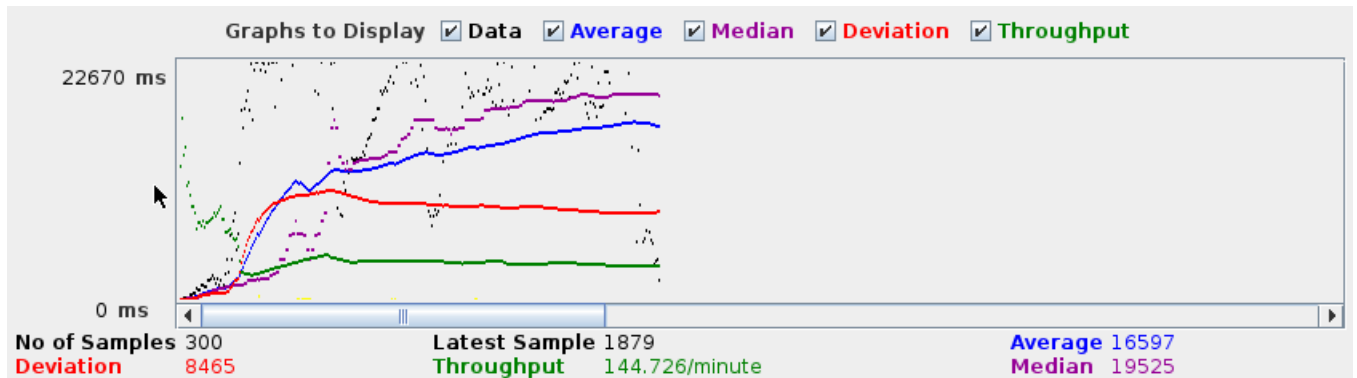


Figura 18: Prueba 4- Gráfico de Resultados.

Label	# Samples	Average	Min	Max	Std. Dev.	Error %	Throug...	KB/sec	Avg. Bytes
167 /tarea2/index.php/consulta1/hola	100	11500	1	26112	8596.24	2.00%	1.1/sec	0.78	756.4
170 /tarea2/index.php/consulta1	100	22960	6236	29725	4179.31	0.00%	49.1/min	1340.84	1677698.0
166 /tarea2/index.php/consulta1/chao	100	15329	1	24473	7452.04	11.00%	51.8/min	842.52	999159.0
TOTAL	300	16597	1	29725	8465.02	4.33%	2.4/sec	2102.43	892537.8

Figura 19: Prueba 4- Tabla de Resultados.

En la tabla de resultados Figura 19 la primera fila es la consulta 1, la segunda fila es la consulta 2 y por último la consulta 3 en la tercera fila.

Prueba 5: El método a utilizar para esta prueba es la paginación, que consiste en limitar el rango de datos que entrega la consulta así podemos mejorar la carga de los datos en la interfaz. A modo de ejemplo los resultados se aprecian en la figura 20 y figura 21

Consulta:

[123>Last >](#)

nombre_departamento

Customer Service

Development

Finance

Human Resources

Marketing

Production

Quality Management

Research

Sales

Figura 20: Prueba 5- Resultado Consulta 1.

Consulta:

[<1234>Last >](#)

Nº Empleado	Nombre	Salario	Nº Departamento	Nombre Departamento
10068	Charlene	111623	d007	Sales
10151	Itzhak	109501	d007	Sales
10173	Shrikanth	109964	d002	Finance
10237	Yannis	112892	d007	Sales
10256	Irene	109222	d007	Sales
10258	Basil	110212	d005	Development
10304	Bernt	110731	d007	Sales
10454	Wonhee	110048	d007	Sales
10455	Sverrir	109531	d007	Sales
10521	Kwangsub	111936	d007	Sales
10532	Mary	109069	d007	Sales
10539	Yucel	109935	d007	Sales
10548	Ramalingam	110477	d001	Marketing
10583	Prasadram	109161	d004	Production
10603	Wayne	109772	d007	Sales
10637	Heejo	109892	d007	Sales
10666	Zhonghua	109858	d002	Finance
10688	Xiaopeng	110338	d002	Finance
10897	Arno	109227	d007	Sales
10955	Urs	110243	d007	Sales
11029	Georgi	110964	d007	Sales
11099	Woody	109280	d007	Sales
11120	Masato	109966	d007	Sales

Figura 21: Prueba 5- Resultado Consulta 3.

PID	USER	PR	NI	VIRT	RES	SHR	S	%CPU	%MEM	TIME+	COMMAND
2564	root	20	0	702m	92m	18m	S	17.8	4.6	0:46.00	java

Figura 22: Prueba 5- Consumo CPU-MEMORIA.

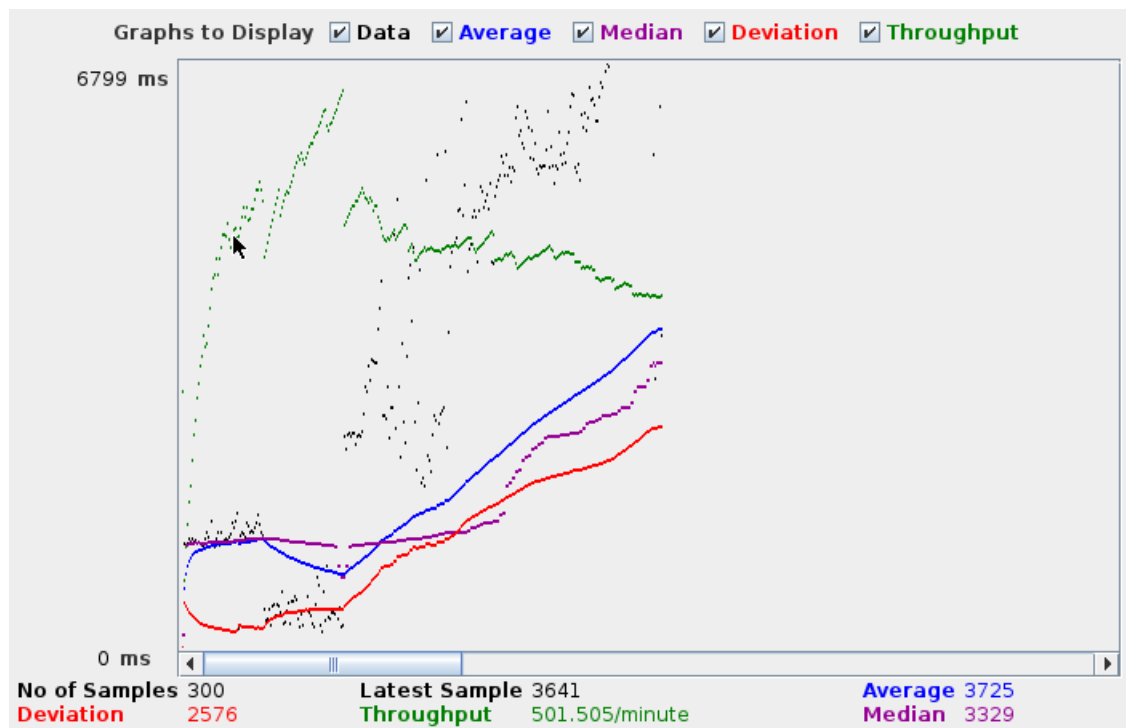


Figura 23: Prueba 5- Gráfico de Resultados.

5. Tabla de Comparación de las pruebas

A continuación se presenta la tabla de comparación de las pruebas:

Nombre	Promedio	Error	Throughput
Prueba 1 Normal			
Consulta 1	8177	0 %	1.5 /sec
Consulta 2	18825	2 %	59.6 /min
Consulta 3	21187	8 %	58.2 /min
Prueba 2 InnerJoin			
Consulta 1	6851	0 %	1.8 /sec
Consulta 2	17148	8 %	1.4 /sec
Consulta 3	9201	0 %	1.6 /sec
Prueba 3 Índices			
Consulta 1	14071	0 %	1.0 /sec
Consulta 2	23187	7 %	48 /min
Consulta 3	20402	0 %	46.2 /min
Prueba 4 Caché			
Consulta 1	11500	2 %	1.1 /sec
Consulta 2	22960	0 %	49.1 /min
Consulta 3	15329	11 %	51.8 /min
Prueba 5 Paginación			
Consulta 1	2689	0 %	4.2 /sec
Consulta 2	3137	0 %	3.5 /sec
Consulta 3	5349	0 %	3.4 /sec

Tabla 1: Tabla comparativa de los resultados de las pruebas

6. Conclusión

Durante la realización de este trabajo se han aplicados conocimientos de programación web y base de datos.

A través del framework seleccionado (codeigniter), aprendimos el comportamiento del patrón de diseño utilizado por este framework, el patrón Modelo Vista Controlador (MVC).

Codeigniter nos permitió separar la lógica de la capa de datos y de la capa de interfaz, reflejando las buenas prácticas de desarrollo. También nos entregó herramientas para la optimización de las pruebas realizadas, como la función de caché, que nos permitió almacenar datos de las consultas realizadas a la base de datos, así optimizando la cantidad de consultas ejecutadas por el usuario.

Al ejecutar las pruebas, observamos el comportamiento de la aplicación bajo a ciertas cantidades de peticiones inmediatas.

Para las pruebas realizadas, nos dimos cuenta que el tiempo de respuesta varía según la cantidad de datos y la complejidad de la consulta.

Cada optimización realizada, pudimos observar que para las pruebas de índices y utilización de caché, nuestro sistema empeoró, en cambio, en el prueba de paginación, entregó los mejores resultados durante las consultas.

Para finalizar comprendimos que para optimizar el sistema, depende en demasía la utilización del framework, de la base de datos, el diseño y la lógica del sistema, presentando dificultades en mejorarlo, según las pruebas obtenidas en este trabajo.