

## Bases de Datos Avanzadas

### Informe etapa 1

Fabián Olivares, Erik Regla  
{olivares13, eregla09}@alumnos.utalca.cl

19 de Abril del 2016

## 1. Etapa 2: Indexado

### 1.1. Base de datos normalizadas

#### 1.1.1. Tiempo de carga en base de datos

Tabla 1: Tiempos de ejecución para cada script en milisegundos

script	non_indexed	btree	hash	both
01_table_generation.sql	9	11	4	9
02_roles.sql	69760	69299	69586	69760
03_problems.sql	1964690	1958507	1982939	1964690
04_countries.sql	65882	65776	65772	65882
05_sites.sql	672127	668997	671179	672127
06_contests.sql	918502	934399	917848	918502
07_contestants.sql	2021650	2069131	2077798	2021650
08_problem_sets.sql	11099786	11066399	11095732	11099786
09_contest_sites.sql	650900	650706	651507	650900
10_teams.sql	1846802	1837683	1877973	1846802
11_scoreboards.sql	3015905	3004977	3009461	3015905
12_team_members.sql	3350787	3336618	3403136	3350787
13_queries.sql	3350787	1455	498	1558

No se puede apreciar una diferencia notable entre los tiempos de carga para bases de datos con diferentes índices, siendo la diferencia más notoria con la tablas indexadas utilizando hash (ver Figura 5). Sin embargo, el tiempo total de ejecución para las instancias es claramente diferente (ver 13\_queries.sql en Tabla 1) .

De todas las tablas, la que más demoró en cargarse fue **problems** la cual dada su estructura es bastante más pesada debido a la columna **plain\_text\_content**, la cual en repetidas instancias contiene una cantidad importante de texto plano (ver Figuras 1, 2, 3, 4 ) .

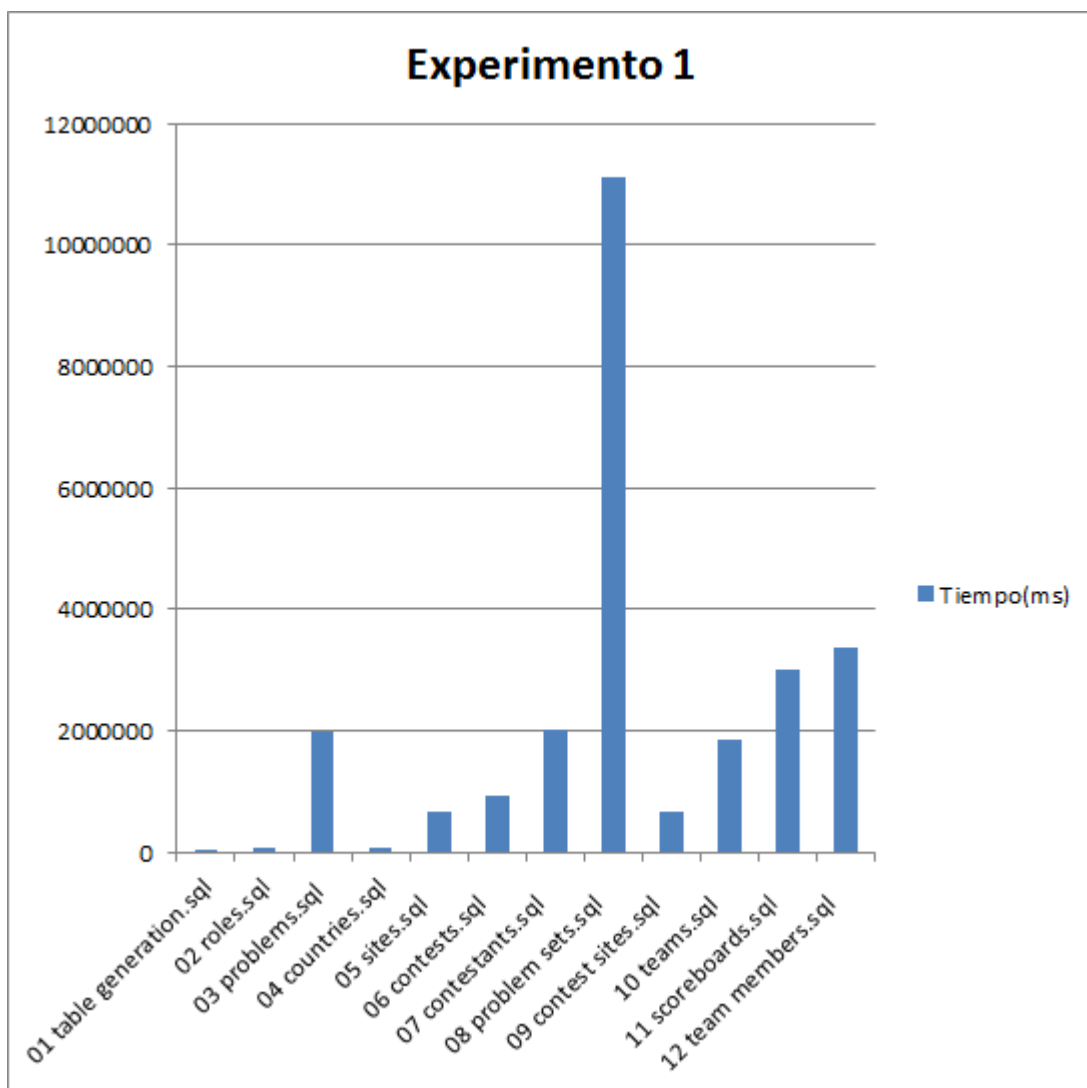


Figura 1: Tiempo de carga de experimento 1.

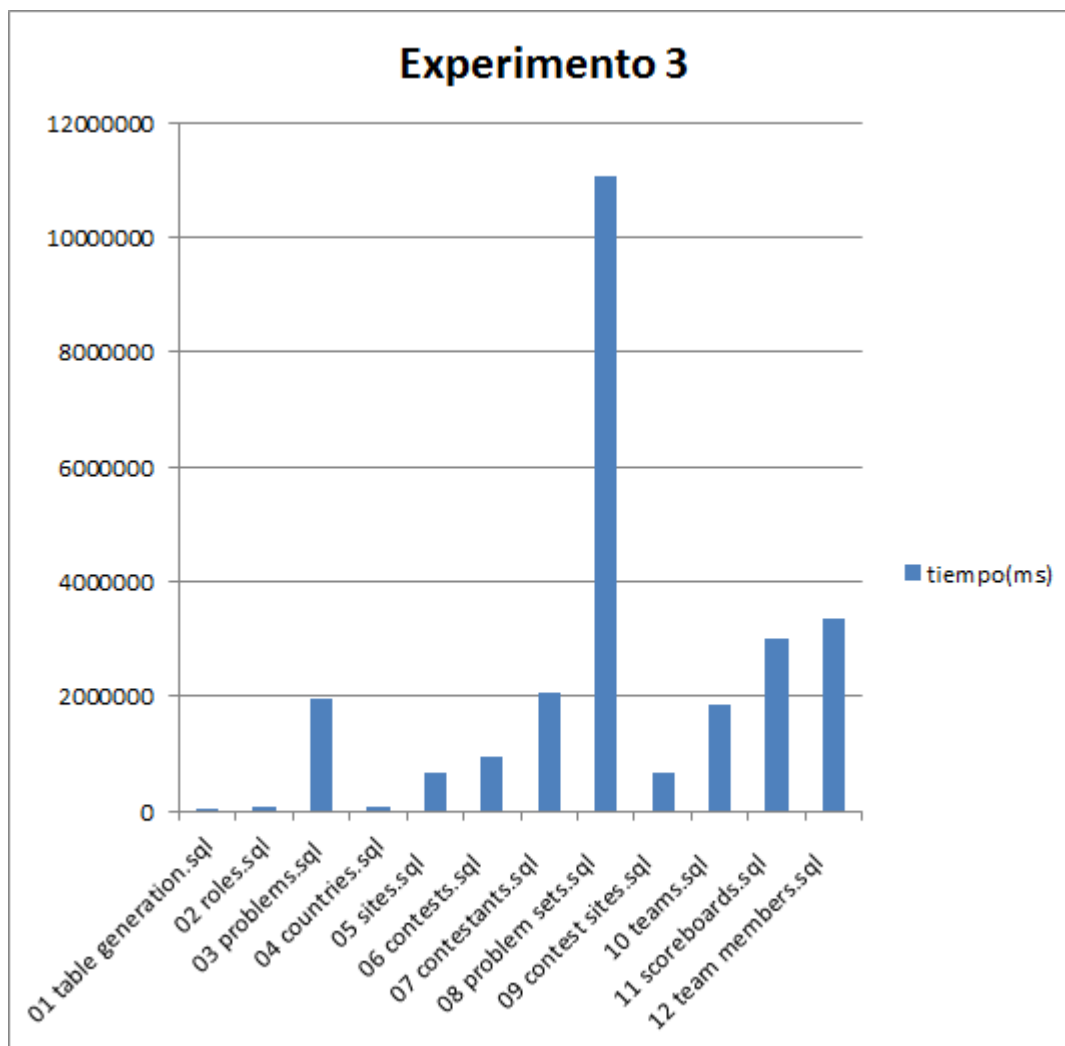


Figura 2: Tiempo de carga de experimento 3.

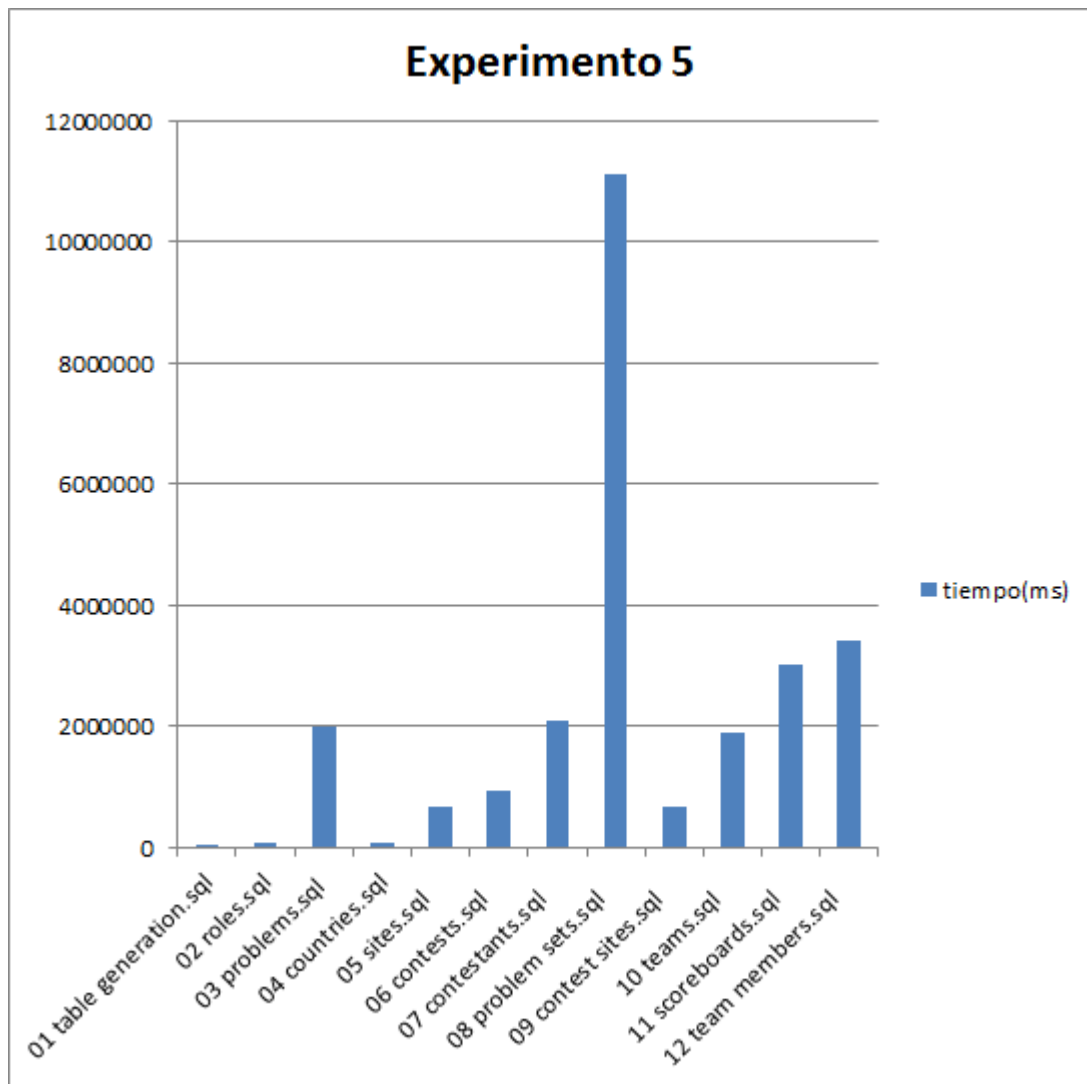


Figura 3: Tiempo de carga de experimento 5.

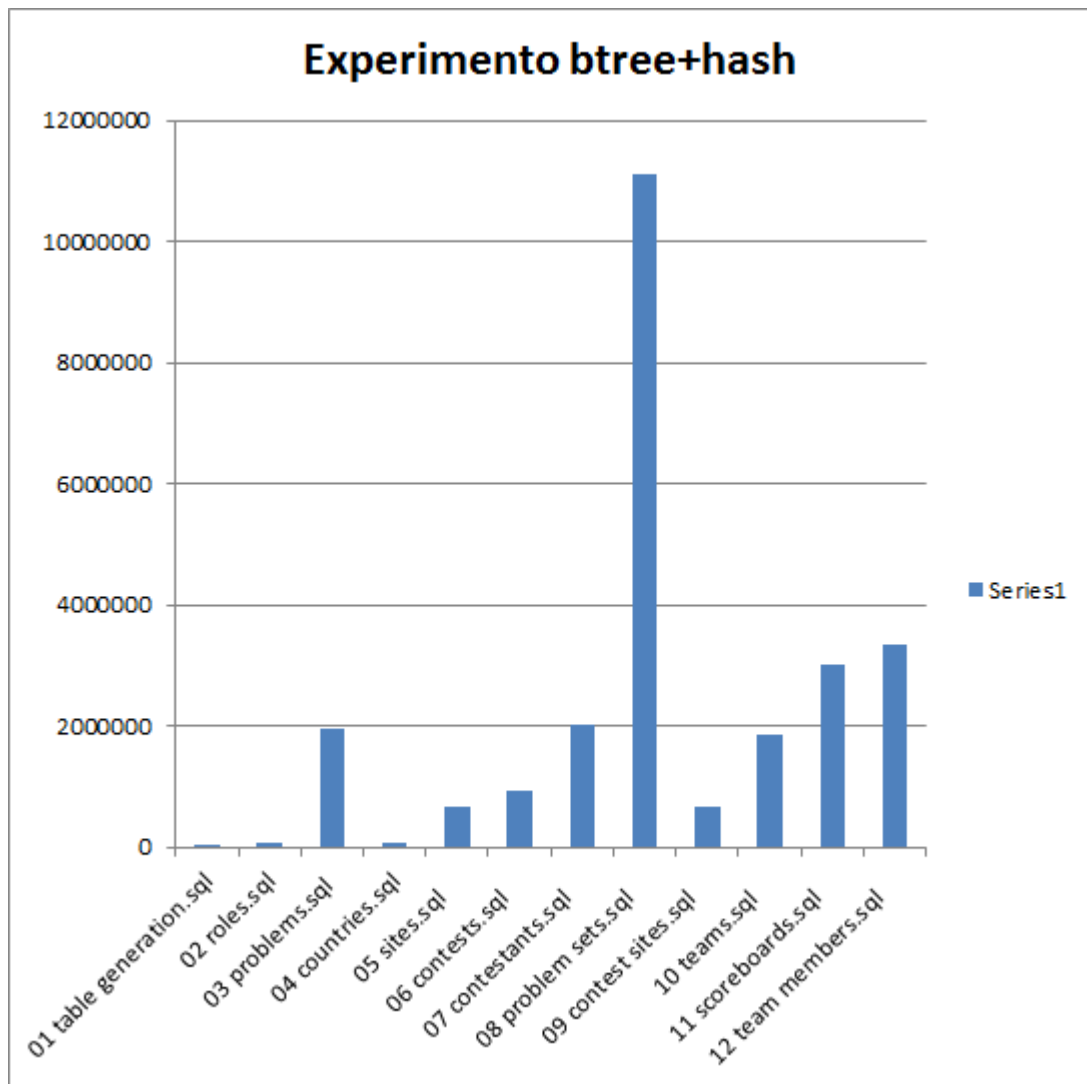


Figura 4: Tiempo de carga de experimento btree + hash.

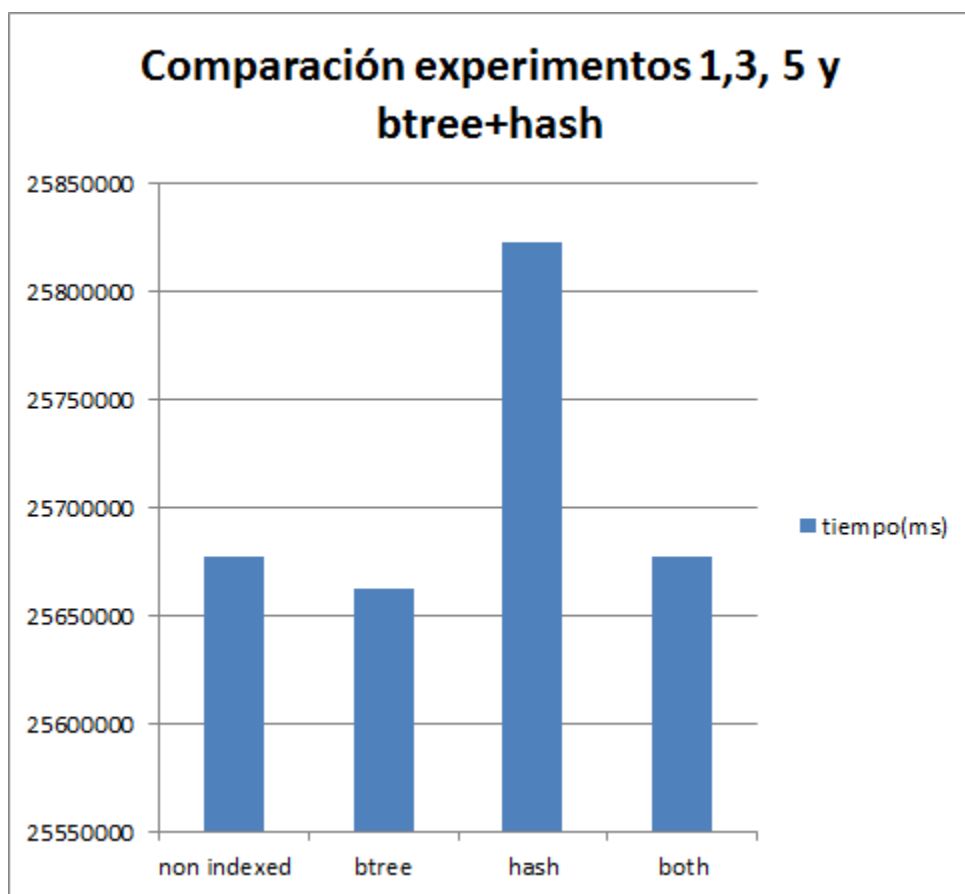


Figura 5: Comparación en tiempo de carga de experimentos 1, 3, 5 y btree+hash.

### 1.1.2. Ejecución de instancias

Tabla 2: Tiempos de ejecución por consulta en milisegundos

Query	non_indexed	btree	hash	both
INDEX_1	3072	3156	12	14
INDEX_2	925	872	3	2
INDEX_3	756	753	2	2
INDEX_4	304	407	2	2
INDEX_5	332	320	2	1
INDEX_6	1913	2136	6	7
INDEX_7	576	536	3	1
INDEX_8	360	629	2	3
INDEX_9	517	731	4	4
INDEX_10	500	396	5	4
INDEX_11	2490	1169	1497	1032
INDEX_12	1170	1159	704	454
INDEX_13	775	1469	520	379
INDEX_14	971	1375	712	397
INDEX_15	1400	1179	965	438
INDEX_16	1979	2012	583	306
INDEX_17	2068	1283	527	280
INDEX_18	1644	1631	524	302
INDEX_19	1462	1433	537	285
INDEX_20	1558	1455	498	287

Sin necesidad de mencionar figuras o tablas en especial, es claramente notorio como los diferentes tipos de índices afectan las consultas (de rango o equidad) de la manera esperada (ver Anexo), sin embargo, es curioso como en algunas instancias, las consultas de rango realizadas con índices hash muestran una clara mejora en comparación a su contraparte con índices btree. Finalmente, al utilizar en conjunto ambos tipos de índices, las consultas exhiben un mejor rendimiento que en sus versiones separadas.

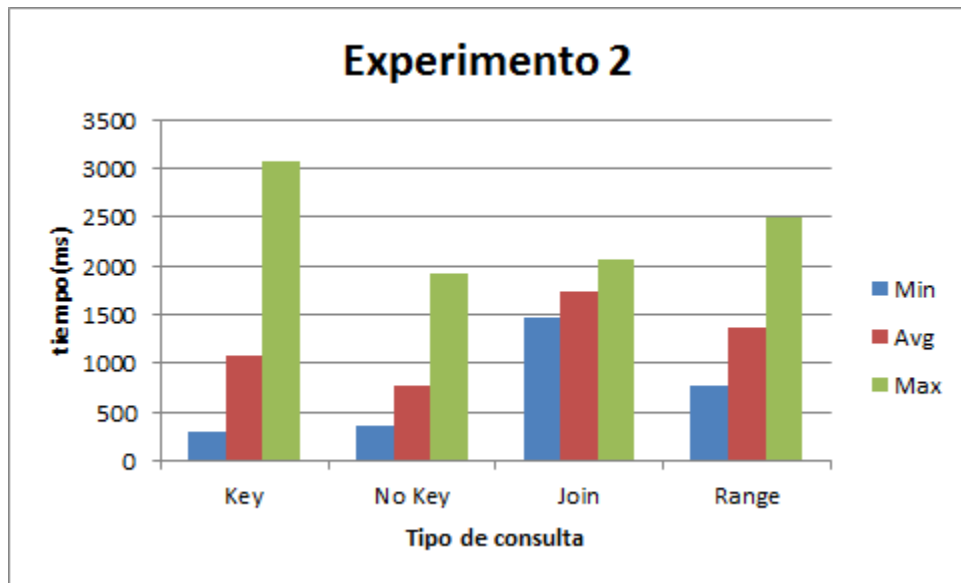


Figura 6: Tiempos de ejecución de consultas en experimento 2.

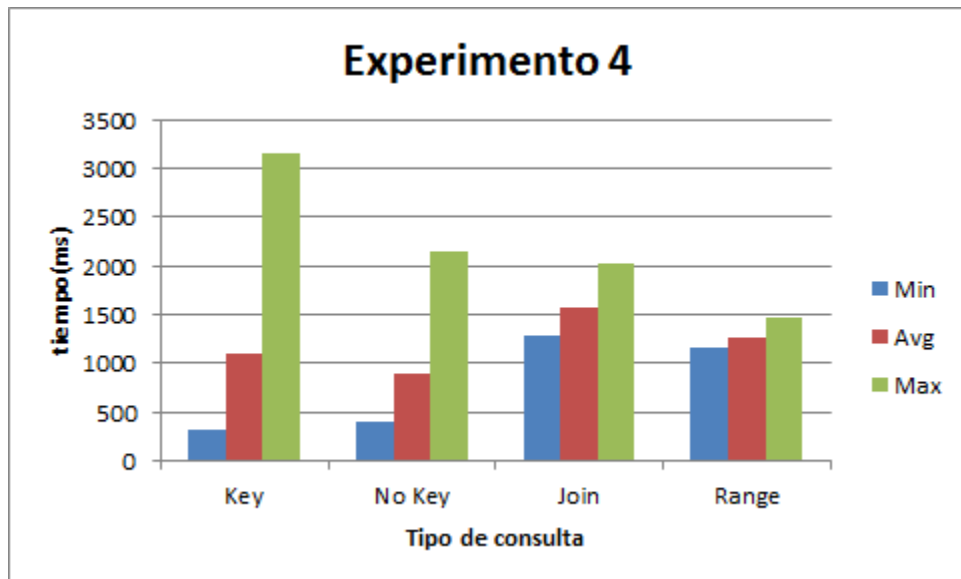


Figura 7: Tiempos de ejecución de consultas en experimento 4.



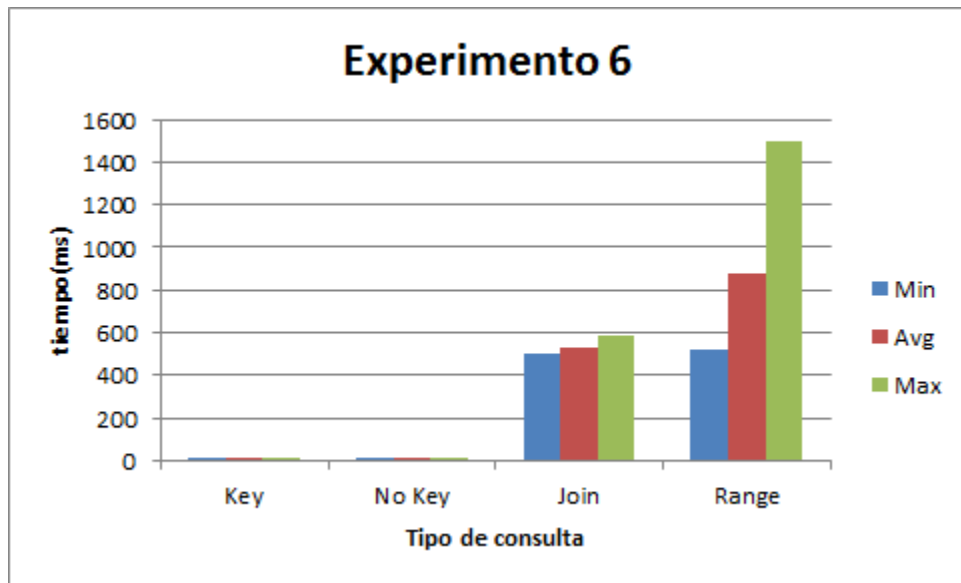


Figura 8: Tiempos de ejecución de consultas en experimento 6.

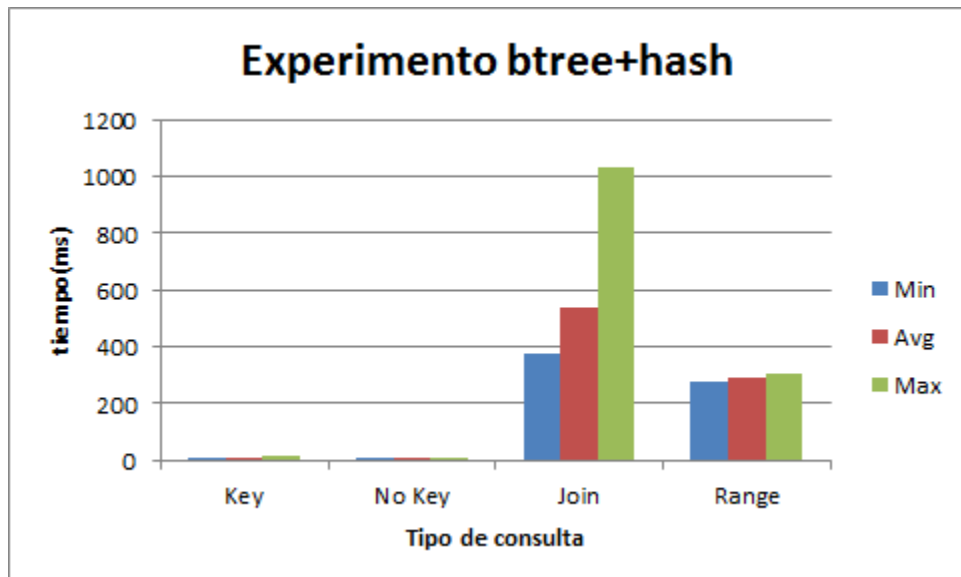


Figura 9: Tiempos de ejecución de consultas en experimento btree + hash.

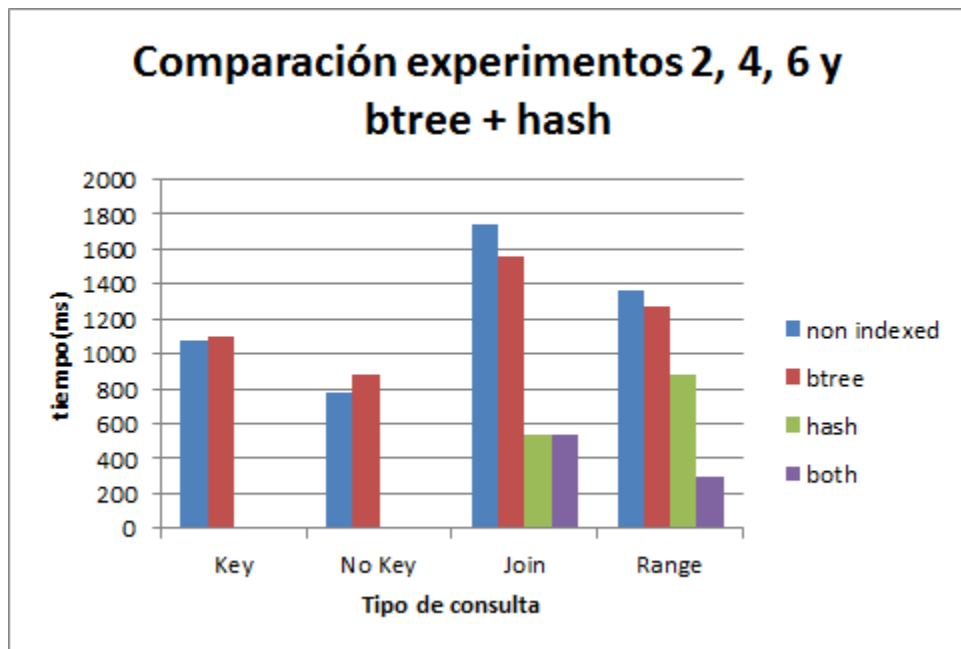


Figura 10: Comparación en tiempo de experimentos 2, 4 y 6.

## 2. Etapa 3: Desnormalización

### 2.1. Tablas a desnormalizar y motivo

1. *Relación 1-1*: Se separa columna `plain_text_content` de `problems` porque es muy grande y no se consulta frecuentemente.
2. *Relación 1-N no clave*: Nueva columna `contest_name` en tabla `problems` dado que consultar a que competencia pertenece un problema dado es bastante frecuente.
3. *Relación 1-N clave*: Columna `country_id` añadida a tabla `teams` dado que es muy frecuente consultar el país atribuido a un equipo.
4. *Relación N-M*: Nueva columna `team_name` a tabla `team_members` ya que es frecuente consultar los equipos en los que ha participado un competidor.
5. *Tabla de búsqueda*: Se separa al malescrito `ballon_colour` de la tabla `problems` ya que son 20 colores que se utilizan. De este modo, reducir espacio en la base de datos.

### 2.2. Experimento 1: Carga

Tabla 3: Estadísticas de migración

Query	Aff. Rows	Time(ms)	Prev. Size (MB)	Final size(MB)
C_1, C_2	1760088	7669	problems = 534	d1_problems_content = 335 d1_problems = 91
C_3	12520000	51977	problems = 534	d1_problems = 2698MB
C_4	2090000	3054	sites = 28 teams = 101	d3_teams = 107
C_5	4080000	4195	team_members = 80 team = 101	d4_team members = 137
C_6, C_7	1760020	7070	problems = 534	d5_colours = 0.08 d5_problems = 403

A modo de carga se utilizó una migración de datos a una tabla nueva a fin de poder comprar directamente el rendimiento de las consultas en diferentes tablas. Cabe destacar que en el enunciado nunca se especifica el método a usar para cargar los datos.

Las tablas con la nomenclatura `dX_` son tablas nuevas y adaptaciones para la base de datos original, a modo de poder diferenciar entre los tamaños de una u otra.

Como se puede observar en la Tabla 3 y en la Figura 11, la desnormalización 2 que involucra a la tabla `problems` fue la más costosa de todas, probablemente por el número de filas afectadas en la base de datos y tiempo de escritura en disco duro.

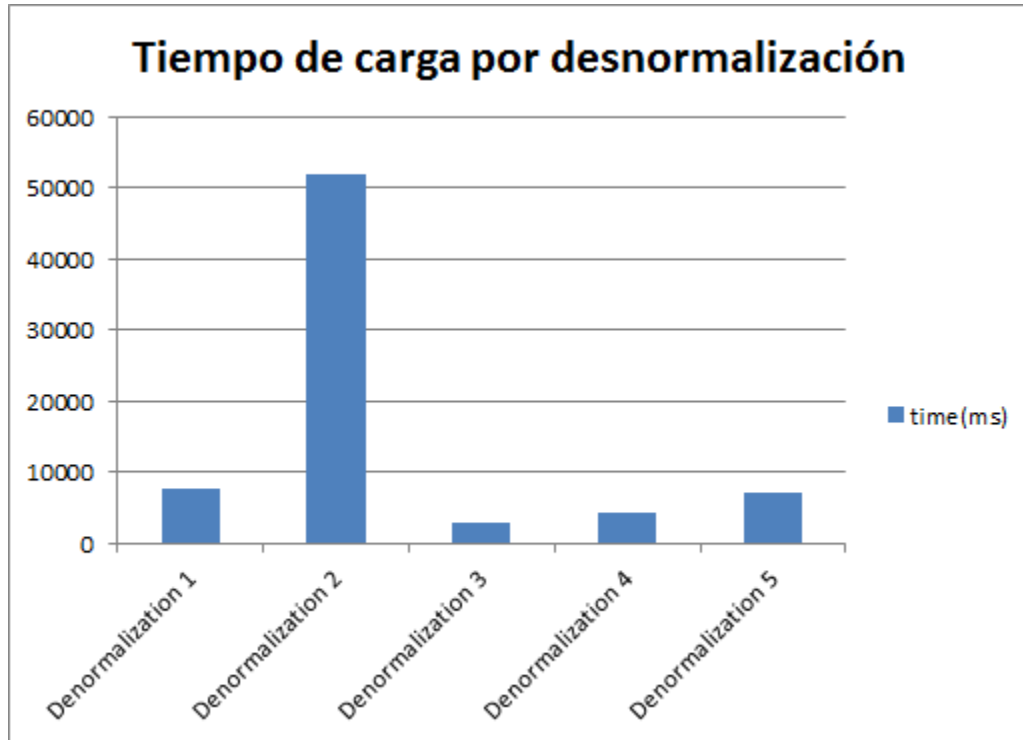


Figura 11: Tiempos de carga para BD desnormalizada.

### 2.3. Experimento 2: Instancias

Exceptuando el caso de la desnormalización 2, las consultas exhibieron un mejor rendimiento para todos los casos. La desnormalización 2, la cual se realiza vía la Consulta C\_3 fue la que más sorprendió con un incremento de 300% en el tamaño original de la tabla, el cual no es comparable con el resto de las modificaciones de la base de datos.

La normalización 5 que involucra las Consultas C\_6, C\_7 si bien mostró mejoras en la utilización del espacio, esta no fue tan diferente debido a que el texto almacenado para las consultas no excedía los 9 caracteres en peor caso.

Tabla 4: Tiempos de ejecución por consulta en milisegundos

<b>Original Query</b>	<b>Original Query Time (ms)</b>	<b>Denormalized Query Time (ms)</b>	<b>Denormalized Query</b>
NORM_1	559	227	DENORM_1
NORM_2	484	213	DENORM_2
NORM_3	530	236	DENORM_3
NORM_4	574	216	DENORM_4
NORM_5	485	216	DENORM_5
NORM_6	1855	2381	DENORM_6
NORM_7	1332	3175	DENORM_7
NORM_8	1357	2978	DENORM_8
NORM_9	2072	2331	DENORM_9
NORM_10	2563	2295	DENORM_10
NORM_11	581	250	DENORM_11
NORM_12	281	241	DENORM_12
NORM_13	303	238	DENORM_13
NORM_14	394	232	DENORM_14
NORM_15	524	341	DENORM_15
NORM_16	669	254	DENORM_16
NORM_17	541	251	DENORM_17
NORM_18	697	250	DENORM_18
NORM_19	436	264	DENORM_19
NORM_20	530	255	DENORM_20
NORM_21	986	561	DENORM_21
NORM_22	835	497	DENORM_22
NORM_23	783	465	DENORM_23
NORM_24	897	534	DENORM_24
NORM_25	764	570	DENORM_25

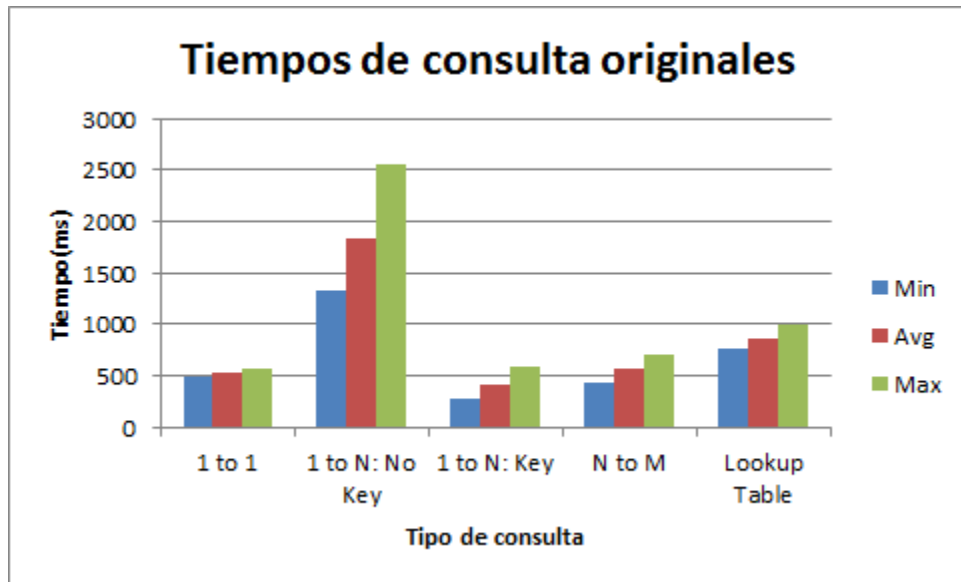


Figura 12: Tiempos de consultas para BD original.

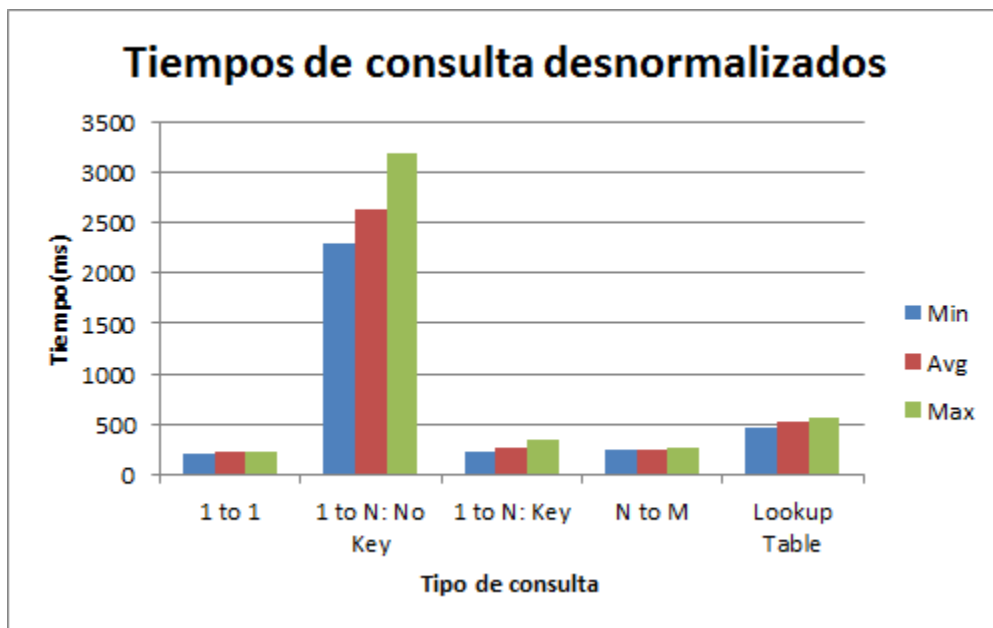


Figura 13: Tiempos de consultas para BD desnormalizada.

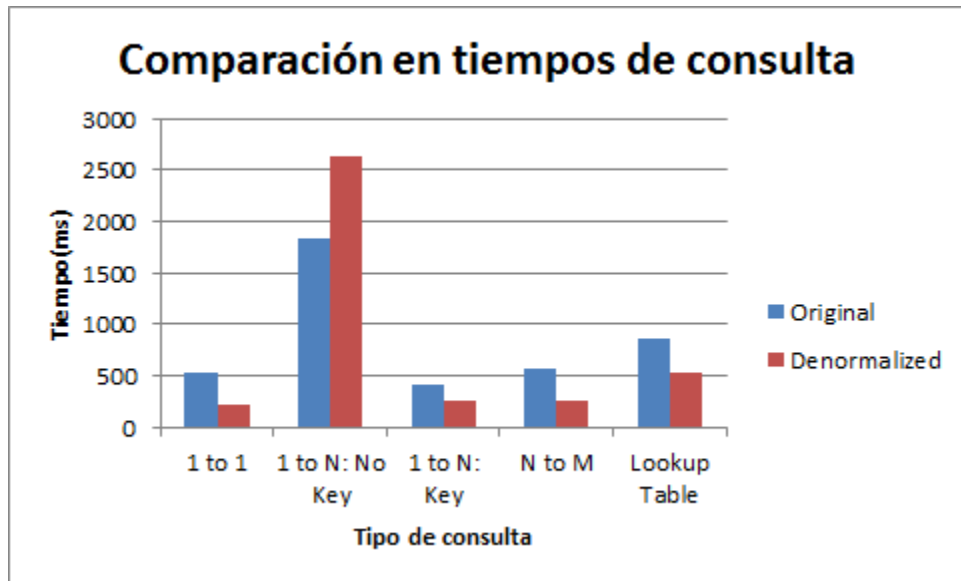


Figura 14: Comparación entre tiempos de consulta de BD original y desnormalizada.

### 3. Etapa 4: Vistas

Tabla 5: Tiempos de ejecución para PgAdmin3 y Conector en milisegundos

Query	PgAdmin3	Conector
FINAL_1	1600	1906
FINAL_2	1600	1821
FINAL_3	32	1
FINAL_4	1800	1722
FINAL_5	1800	1640

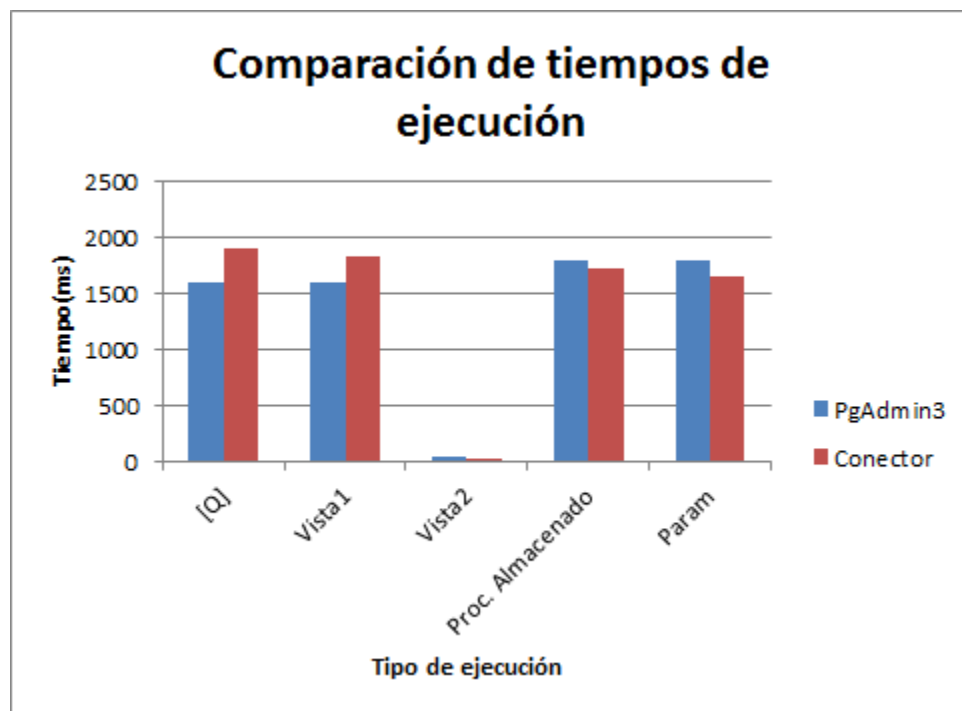


Figura 15: Comparación de tiempos de ejecución.

No se puede apreciar gran diferencia entre la velocidad de las consultas utilizando un conector o la “interfaz” de PostgreSQL (que en realidad interfaz no tiene, también es un conector). Fuera de los resultados esperados vistos en clase no hay mayor novedad. Cabe destacar que PgAdmin3 cambia la unidad de medición de tiempo, volviéndola imprecisa (cambia de msecs a seg después de un segundo).

Como fue visto en clases la ejecución de la consulta en forma de vista materializada fue considerablemente mas rápida que el resto, tanto en el caso de ejecución en pgAdmin como de conector, debido a que guarda la tabla resultado de la consulta con anterioridad.



## 4. Anexos

#### 4.1. Tabla de Consultas

---

Tabla 6: Anexo 1: Lista de consultas

[illegible]