Laboratorio 1 - El Modelo Entidad/Relación

Se le pide construir una base de datos para el juego EntreNos, para la cual le han encargado realizar primero un modelo Entidad/Relación (E/R). A continuación se presentan los requerimientos de EntreNos; se le recomienda leerlos por completo antes de comenzar a realizar el diagrama E/R. Su diagrama debe contener sólo la información aquí pedida; otros atributos no son permitidos. Debe modelar las restricciones de multiplicidad; sin información particular, se puede asumir que la multiplicidad es cero-o-más.

P1. [40 PUNTOS] Su diagrama E/R debe capturar lo siguiente:

- Cada jugador de EntreNos se identifica por su nick. Cuenta también con un nombre.
- Los jugadores pueden ser jugadores de PC o de Android.
- Los jugadores de PC registran si es jugador de Steam o no.
- Los jugadores pueden pertenecer a lo más a una sala. Se debe registrar la fecha de ingreso a la sala.
- Una sala se identifica por su código, y además cuenta con la cantidad máxima de jugadores.
- Una sala es creada por exactamente un jugador.
- Los juegos existen solo en el contexto de (exactamente) una sala y se distinguen de otros juegos de la misma sala a través de su fecha.
- Los juegos indican también en qué mapa ocurrió y si ganaron los impostores o no.
- Un mapa está identificado únicamente por un nombre, y también cuentan con una descripción.
- Un jugador puede jugar en 0 o más juegos.
- Un juego tiene que tener al menos un jugador.
- Un jugador debe escoger un único color para cada juego, además se debe registrar si es que el jugador está vivo, y si es que es secretamente un impostor.
- Durante el juego los jugadores pueden dejar mensajes en un chat grupal.
- Cada mensaje existe en el contexto de un jugador que juega un juego, y se distinguen de otros mensajes del mismo juego y mismo jugador a través de su fecha.

P2. [20 PUNTOS] Intente transformar su diagrama E/R (de la P1) en un modelo relacional. No es necesario describir los tipos, pero es necesario indicar las llaves primarias y las llaves foráneas. En caso de haber varias opciones posibles, debe explicar su elección.

Laboratorio 2 - Álgebra Relacional

Profesores: Claudio Gutiérrez
Matías Toro
Auxiliares: Scarlett Plaza
Daniel Radrigán
Cristian Salazar
Fran Zautzik

Suponga que usted dispone del siguiente modelo relacional para el juego EntreNos presentado en el Laboratorio 1:

- **Jugador**(**nick**, **nombre**)
- **Sala**(**código**, **max_jugadores**, **nick_creador**)
  nick_creador REF **Jugador**(**nick**)
- **Mapa**(**nombre**, **descripción**)
- **JugadorPc**(**nick**, **jugador_steam**)
  nick REF **Jugador**(**nick**)
- **JugadorAndroid**(**nick**)
  nick REF **Jugador**(**nick**)
- **Juego**(**código**, **fecha**, **mapa**, **gana_impostor**)
  código REF **Sala**(**código**), **mapa** REF **Mapa**(**nombre**)
- **Partida**(**nick**, **código**, **fecha**, **es_impostor**, **color**, **vivo**)
  nick REF **Jugador**(**nick**), (**código**, **fecha**) REF **Juego**(**código**, **fecha**)
- **Mensaje**(**nick**, **código**, **fecha**, **fecha-mensaje**, **contenido**)
  (**nick**, **código**, **fecha**) REF **Partida**(**nick**, **código**, **fecha**)
- **Pertenece**(**nick**, **código**, **fecha_ingreso**)
  nick REF **Jugador**(**nick**), **código** REF **Sala**(**código**)

P1. La empresa desarrolladora del juego desea evaluar su diseño y tener un recuento más específico de los dato que maneja. Considerando este modelo relacional, escriba usando álgebra relacional las siguientes consultas. Recuerde usar solo los operadores que fueron vistos en clases.

(a) Los nombres de los jugadores

(b) La descripción del mapa “Nave espacial”

(c) Los mensajes que hayan sido enviados por el jugador “Lxopato” o el jugador “Rauwul”.

(d) Los nicks de jugadores que hayan jugado alguna vez con colores “rojo” y “azul”

(e) El nombre de los jugadores que hayan jugado alguna vez en el mapa “Nave espacial”

(f) Los nicks de jugadores que nunca han creado una sala.

(g) Los nombres de jugadores de steam que hayan creado o pertenecido a una sala

(h) Los nicks de jugadores que han jugado con todos los colores.

(i) Todos los nicks de jugadores que nunca han perdido una partida al jugar como impostor.

(j) El o los juegos mas antiguos.

Laboratorio 3 - S¿qué?L

Profesores: Claudio Gutiérrez
Matías Toro
Auxiliares: Scarlett Plaza
Daniel Radrigán
Cristian Salazar
Fran Zautzik

Nota: Solo puede utilizar los conceptos vistos en la clase de SQL I. La entrega *debe* ser un archivo txt.

En este laboratorio usted deberá realizar consultas SQL en un servidor PostgreSQL que contiene datos de películas extraídos de IMDb. El esquema de los datos es el siguiente:[1]

- **pelicula**(**nombre**, **anho**,**calificacion**)
- **actor**(**nombre**,**genero**)
- **personaje**(**p_nombre**,**p_anho**, **a_nombre**, **personaje**)

La tabla **personaje** usa llaves foráneas que hacen referencia a las tablas de **actor** (**a_nombre**) y **pelicula** (**p_nombre**,**p_anho**).

Para conectarse al servidor, usted puede usar una terminal en Linux/Mac o Putty en Windows. El servidor está hosteado en `cc3201.dcc.uchile.cl` y escucha en el puerto 240. Su usuario es `cc3201` y la contraseña se publicará en u-cursos en Material Docente. El comando que deberían ejecutar en el terminal es:

```
ssh -p 240 cc3201@cc3201.dcc.uchile.cl
```

Para conectarse a la base de datos basta con ejecutar el comando `psql cc3201`. Una vez dentro de la base de datos, usted puede explorar las tablas usando `\dt`, puede ver los atributos y tipos de una tabla usando `\d+ tabla`. Para cerrar la sesión utilice el comando `\q`.

Ejecute la siguiente consulta para probar que todo ande bien:

```
SELECT * FROM topimdb.pelicula;
```

Tenga en cuenta que la consulta termina en ';'. Para ejecutar la consulta presione Enter. Para terminar de navegar a través de los resultados presione q. Para matar una consulta que demora mucho tiempo (más de 5 segundos) presione ctrl + c.

Ahora, debe diseñar las consultas que resuelvan las siguientes preguntas usando los operadores vistos en clases. Debe entregar un documento de texto con las instrucciones SQL así que no olvide copiar sus respuestas en un archivo aparte.

[1] Aunque PostgreSQL no tiene problema con los acentos y las eñes, algunas terminales que uds. pueden usar para ingresar al servidor, sí, pueden tener problemas, así que vamos a evitar su uso, al menos en el esquema.

P1. [6 PUNTOS] Las películas de los 80s, ordenadas por calificación de mayor a menor.

P2. [6 PUNTOS] Los nombres de los personajes que ha interpretado su actriz/actor favorito en los datos, ordenados por año.

P3. [6 PUNTOS] Las películas en las que participó su actriz/actor favorito en los datos, ordenadas por calificación de mayor a menor.

P4. [6 PUNTOS] Los nombres de los personajes interpretados por mujeres, en películas de los 90s, con calificación mayor or igual a 8,5.

P5. [6 PUNTOS] Las películas de la saga "The Lord of The Rings" (usando el prefijo de su nombre) ordenadas por calificación y por año.

P6. [6 PUNTOS] Los nombres de los actores que interpretan más de un personaje en la misma película.

P7. [6 PUNTOS] Las películas en que actúan juntos Uma Thurman y Samuel L. Jackson.

P8. [6 PUNTOS] Las películas en que actúa Uma Thurman y *no* Samuel L. Jackson.

P9. [6 PUNTOS] Los pares de actores que aparecen juntos en más de una película. Cada par debe aparecer una sola vez, es decir, si (A, B) aparece, no debe aparecer (B, A), pues es el mismo par. Tampoco se deben incluir pares de la forma (A, A).

P10. [6 PUNTOS] La(s) película(s) con calificación más alta.

Laboratorio 4 - S¿qué?L II

Profesores: Claudio Gutiérrez
Matías Toro
Auxiliares: Scarlett Plaza
Daniel Radrigán
Cristian Salazar
Fran Zautzik

En este laboratorio usted deberá realizar consultas SQL en un servidor PostgreSQL que contiene datos de películas extraídos de IMDb. El esquema de los datos es el siguiente:

- **pelicula**(**nombre**, **anho**,**calificacion**)
- **actor**(**nombre**,**genero**)
- **personaje**(**p_nombre**,**p_anho**, **a_nombre**, **personaje**)

La tabla **personaje** usa llaves foráneas que hacen referencia a las tablas de **actor** (**a_nombre**) y **pelicula** (**p_nombre**,**p_anho**).

Para conectarse al servidor, usted puede usar una terminal en Linux/Mac o Putty en Windows. El servidor está hosteado en `cc3201.dcc.uchile.cl` y escucha en el puerto 240. Su usuario es `cc3201` y la contraseña es la misma publicada en u-cursos en Material Docente. El comando que deberían ejecutar en la terminal para conectarse al servidor desde Linux/Mac es:

```
ssh -p 240 cc3201@cc3201.dcc.uchile.cl
```

Para conectarse a la base de datos basta con ejecutar el comando `psql cc3201`. Una vez dentro de la base de datos, usted puede explorar las tablas usando `\dt`, puede ver los atributos y tipos de una tabla usando `\d+ tabla`. Para cerrar la sesión utilice el comando `\q`.

A continuación se presentan las consultas que usted debe formular a la base de datos. Debe entregar un archivo con las respectivas consultas escritas en SQL.

P1. [4 PUNTOS] La cantidad de películas.

P2. [4 PUNTOS] La cantidad de años distintos de estreno de las películas.

P3. [6 PUNTOS] Los nombres, años y calificaciones de las 10 películas mejor calificadas, ordenadas por calificación descendente, luego por nombre ascendente y luego por año ascendente (solo hay que devolver 10 tuplas, incluso si hay empates de calificación).

P4. [6 PUNTOS] Los nombres de los actores y las actrices que aparecen en las películas de la **P3**. Cada actor/actriz solo debe aparecer en los resultados una vez.

P5. 6 PUNTOS Los nombres de solo las *actrices* de las películas de la **P4**. Cada actriz debe aparecer en los resultados solo una vez.

P6. 6 PUNTOS Las distintas décadas de las películas en las cuales apareció Steve Buscemi. (Tiene dos resultados; uno es 1990)

P7. 6 PUNTOS El conteo de películas distintas por año (para cada año en la tabla **pelicula**, devolver su conteo de películas).

P8. 6 PUNTOS Devuelva los mismos resultados como en la **P7**, pero ordenados por conteo descendente y borrando los años cuyo conteo sea menor que 2.

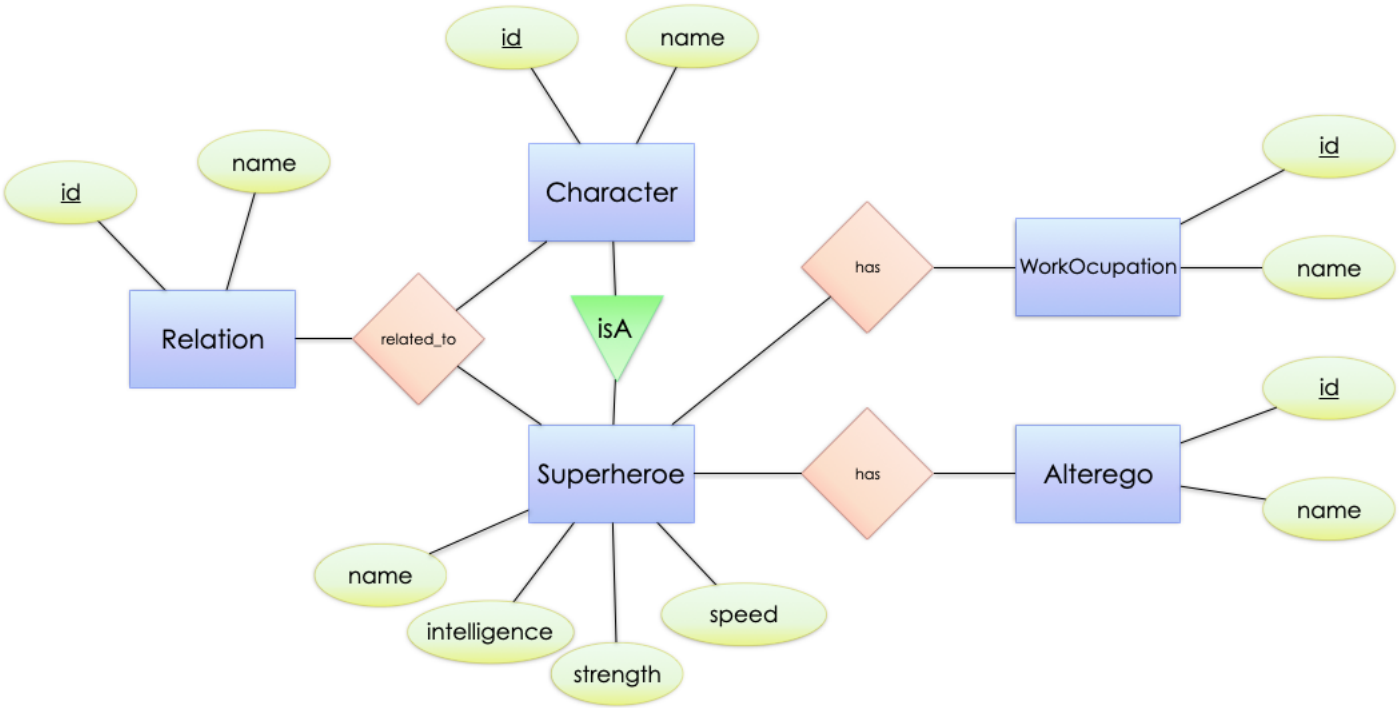
P9. 8 PUNTOS Sea una buena película una película con una calificación $\geq 8{,}6$. Para cada actor/actriz, cuente sus distintas buenas películas. Si un actor/una actriz no tiene una buena película, se puede omitirlo/la de los resultados. Ordene los resultados por conteo descendente.

P10. 8 PUNTOS Cuente el número de actrices distintas por película. De haber una película sin actrices, deje un conteo de 0 (en vez de omitir la película).

Laboratorio 5 - LDD y Acceso Programático

Profesores: Claudio Gutiérrez
Matías Toro
Auxiliares: Scarlett Plaza
Daniel Radrigán
Cristian Salazar
Fran Zautzik

Considere el siguiente modelo entidad relacion que corresponde a una base de datos de superheroes:



En este laboratorio usted deberá crear las tablas necesarias correspodientes al modelo entidad relación y luego importar los datos a partir de un archivo csv.

La creación de tablas e importación debera ser hecha en el servidor del curso en el esquema `superheroes`.

- Para conectarse al servidor directamente, usted puede usar una terminal en Linux/Mac/Windows o Putty en Windows. El servidor está hosteado en `cc3201.dcc.uchile.cl` y escucha en el puerto 240. Su usuario es `cc3201` y la contraseña se publicó en u-cursos en Material Docente. El comando que deberían ejecutar en la terminal para conectarse al servidor desde Linux/Mac es:

```
ssh -p 240 cc3201@cc3201.dcc.uchile.cl
```

Para conectarse a la base de datos basta con ejecutar el comando `psql cc3201`. Para cerrar la sesión utilice el comando `\q`.

- Para conectarse al servidor programáticamente, puede crear una conexión en python como se describe a continuación:

```
conn = psycopg2.connect(host="cc3201.dcc.uchile.cl",
    database="cc3201",
    user="cc3201",
    password="j'<3_cc3201", port="5440")
```

Usted deberá entregar un archivo de texto (`.txt`) y un archivo python (`.py`) con el desarrollo de los ejercicios que siguen.

P1. [30 PUNTOS] Cree las tablas necesarias de modo que satisfagan el modelo entidad relación en el schema `superheroes`, agregando las restricciones de llave primaria y foránea según corresponda. Tenga en cuenta que el nombre de cada tabla debe tener un prefijo único para cada grupo, como por ejemplo `equiporocket_tablita`.
Hints:

- El campo id lo puede crear de tipo `serial`.
- Sugerimos traducir la herencia (isA) usando dos tablas (para `character` y `superheroe`).

P2. [60 PUNTOS] Crea y corra un archivo python para poblar la base de datos creada en el punto anterior a partir del archivo de datos csv provisto en ucursos. Este archivo tiene varios errores e inconsistencias, por lo que le sugerimos leer las siguientes consideraciones:

- Le recomendamos hacer la tarea en python. Para instalar la librería necesaria para conectarse a la base de datos utilice `pip3` ejecutando `pip3 install psycopg2-binary`
- A veces el nombre del pariente es el nombre de superhéroe y otras veces el nombre real, por lo que se sugiere procesar el archivo csv dos veces: una para ingresar los superhéroes y otra para procesar las relaciones (y crear personajes que no sean superhéroes). Esto significa que al buscar parientes debe buscar en ambas tablas (puede escoger el primer valor encontrado).
- Algunos strings pueden venir con comillas dobles extras (e.g. `adventurer, "agent", mechanic`)

Por lo tanto le sugerimos seguir los siguientes pasos.

(a) **Creación del superhéroe**. Lea el archivo. Por cada personaje (fila):

i. Seleccione el id del personaje por nombre en caso de ya exista. De lo contrario insertelo y recupere su id. Puede asumir por simplicidad que no hay dos personajes (characters) que se llamen igual. Si hay personajes sin nombre, por simplicidad puede usar el de superhéroe.

ii. Seleccione el id del superhéroe dado el id del personaje del punto anterior. En caso de no existir, crealo.

iii. Para cada alter ego (asuma que están separados por "," o ";")
   A. Elimine espacios blancos al comienzo y al final, y comillas dobles.
   B. Busque si ya existe el alter ego para ese superhéroe. Si no existe, insertelo.

iv. Para cada ocupación/oficio (asuma que estan separadas por "," o ";")
   A. Elimine espacios blancos al comienzo y al final, y comillas dobles.
   B. Seleccione el id de la ocupación dado el nombre de esta. Si no existe, créala. Note que un mismo oficio puede venir en distintas combinaciones de mayúsculas y/o minúsculas en distintas filas.
   C. Busque si ya existe un elemento en la tabla intermedia entre superhéroe y ocupación. Si no existe insertela.

(b) **Creación de los parientes**. Lea de nuevo el archivo. Por cada personaje:

i. Seleccione el id del personaje (ya debería existir según lo de arriba)

ii. Ignore filas donde la relación es el string "-".

iii. Para cada relación (asuma que están separadas por "," o ";")
   A. Elimine espacios blancos al comienzo y al final, y comillas dobles.
   B. Separe el pariente de su parentesco. Ignore el estado del pariente (deceased). Para validar un pariente y separar el nombre de su parentesco (e.g. en `Ashley Zolomon (wife, deceased)`), use el comando `m = re.search("([^(]+)[ ]*\(([^)]+)\)", texto)`. Luego, si `m` está definido, `m.group(1)` apunta al nombre (e.g. `Ashley Zolomon`), y `m.group(2)` al tipo de relación y estado (e.g. "wife, deceased").
   C. Obtenga el id del personaje correspondiente al pariente, ya sea por el nombre de superheroe o de personaje. Si no encuentra al personaje, cree uno nuevo.
   D. Busque si existe la relación de parentesco. Si no existe, cree una nueva.

(c) Haga "commit" de sus cambios.

P3. 10 PUNTOS Entregue las consultas sql que responden las siguientes preguntas

(a) **Pregunta:** Calcule los nombres de los 3 superheroes con más parientes (note que hay nombres de superheroes que se repiten, e.g. Batman (Bruce Wayne), Batman (Terry McGinnis) y Batman (Dick Grayson)).

(b) **Pregunta:** Calcule los nombres de los 3 personajes no superheroes con más parientes.

(c) **Pregunta:** Calcule los nombres de los 5 superheroes con más parientes superheroes.

(d) **Pregunta:** Calcule el nombre de relación más común

(e) **Pregunta:** Calcule los 3 trabajos más populares

Laboratorio 6 - Acceso Programático e Inyecciones SQL

Profesores: Claudio Gutiérrez
Matías Toro
Auxiliares: Scarlett Plaza
Daniel Radrigán
Cristian Salazar
Fran Zautzik

En este laboratorio usted hackeará el sistema de bases de datos del curso a través de una página web. Usted debe entregar un archivo de texto con las consultas de **P1** y las inyecciones de **P2** junto a alguna evidencia de su hackeo (foto).

P1. [15 PUNTOS] Conéctese vía SSH al servidor, donde encontrará las dos tablas que usaremos. Puedes revisar los detalles de las dos tablas usando `\d+ TABLA`.

(a) [5 PUNTOS] En `uchile.transparencia` encontrará las remuneraciones brutas de todos los empleados de la Universidad desde enero de 2015. Escriba una consulta SQL para obtener los datos de todos los empleados con un apellido paterno elegido por usted (puede devolver las tuplas enteras) ordenados por saldo (la columna `total`).

(b) [5 PUNTOS] En `nota.cc3201` encontrará las notas finales de CC3201 de usted y sus compañeros (obviamente no tienen nada que ver con la realidad... ¿o sí?). Escriba una consulta SQL que obtenga solo **su** nota final del ramo (puede devolver la tupla entera; puede usar una condición sobre `id` o `nombre` o lo que le parece conveniente).

(c) [5 PUNTOS] Si le parece injusta la nota, puede intentar cambiarla. Escriba una instrucción SQL que modifique **solamente su nota**. Como es de esperar, la base de datos está preparada para este tipo de "ataques". Escriba el resultado de su instrucción.

(d) [5 PUNTOS] Tenga en cuenta que en Postgres se pueden hacer consultas de la forma siguiente:

- `SELECT table_name, table_schema FROM information_schema.tables;`
- `SELECT column_name, data_type FROM information_schema.columns WHERE table_name='TABLA' AND table_schema='ESQUEMA';`

Ejecuta la primera consulta para ver todas las tablas y sus esquemas. Después ejecute la segunda consulta para ver solo las columnas de la tabla `nota.cc3201` y sus tipos. (Serán útiles estas consultas.)

P2. [45 PUNTOS] Adicionalmente usted se entera de que existe una página web (`http://cc3201.dcc.uchile.cl`) que se conecta a la misma base de datos, pero que extrae la información de transparencia de funcionarios de la universidad. En dicha página, usted puede ingresar

algún apellido paterno, y se entregarán los 250 sueldos mensuales más altos de empleados con **exactamente** ese apellido.

¿Es está página segura ante inyecciones SQL? (*Spoiler*: No.) Es momento de ponerlo a prueba. Todo su poder se basa en la capacidad de escribir en el campo de texto "apellido". Su objetivo es realizar inyecciones SQL a través del campo de texto para intentar cambiar su nota. *Hints*:

- La base de datos requiere que la nota esté entre 1.0 y 7.0.
- Puede inyectar la consulta de P1 (d), para saber los nombre de las columnas.
- Se sabe que el sistema de base de datos es Postgres (hay formas de adivinar el sistema usado; p.ej. se puede probar con consultas que solo funcionan con un sistema particular).
- Parece que el programador dejo accidentalmente en el código fuente de la página web un link a github que podría serle útil.
- Si la página arroja algún error, no *siempre* significa que su ataque fue infructuoso. ¿Acaso esperaba un mensaje de felicitaciones por hackear la base de datos?

Tendrá que ingresar inyecciones SQL para:

(a) [7 PUNTOS] devolver todas las tablas en la base de datos;

(b) [7 PUNTOS] devolver las columnas de la tabla `nota.cc3201` y sus tipos;

(c) [7 PUNTOS] devolver su nota en la tabla `nota.cc3201`;

(d) [7 PUNTOS] cambiar su nota en la tabla `nota.cc3201`;

(e) [7 PUNTOS] cambiar su comentario en la tabla `nota.cc3201`.

(f) [10 PUNTOS] Escriba una propuesta de código python que arregla esta vulnerabilidad. Indique que linea debe cambiarse por cual. Hint: vea el codigo fuente del html para ver donde podría encontrar el código fuente de la aplicación.

Laboratorio 7 - Índices

Profesores: Claudio Gutiérrez
Matías Toro
Auxiliares: Scarlett Plaza
Daniel Radrigán
Cristian Salazar
Fran Zautzik

La entrega de este laboratorio deberá ser un informe en pdf, el cual está descrito al final de este archivo./

P1. Sea la relacion $R(a, b, c, d)$ cuyo tamaño es de 1 millón de tuplas, y cada página/bloque contiene $B > 2$ tuplas. Las tuplas de R están ordenados de manera aleatoria (pila). El atributo a es llave, cuyos valores son enteros que van del 0 al 999.999 (distribuídos uniformemente). Para cada una de las consultas a continuación, diga el número de I/O (lecturas/escrituras) que se harán en cada uno de los siguientes casos:

- Analizar R sin ningún índice.
- Usar un *B+Tree unclustered* sobre el atributo a. El árbol es de altura h y cada página contiene P punteros ($P > B$).
- Usar un *B+Tree clustered* sobre el atributo a (asuma que el archivo está ordenado por a). El árbol es de altura h y cada página de hoja está ocupada al 60%.
- Usar un *Hash Index unclustered*. Cada página del indice contiene P punteros ($P > B$).
- Usar un *Hash Index clustered*.

Las consultas son:

(a) Encontrar todas las tuplas de R.
(b) Encontrar todas las tuplas de R tal que $a < 50$.
(c) Encontrar todas las tuplas de R tal que $a = 50$.

P2. Se cuenta con varias tablas que son instancias del siguiente esquema abstracto:

- **pelicula**(**nombre**, **anho**,**calificacion**, **votos**)
- **actor**(**nombre**,**genero**)
- **personaje**(**a_nombre**, **p_nombre**,**p_anho**, **personaje**)

En la base de datos hay dos esquemas: uno con datos indexados (`opti`) y otro sin índices (`opt`). En cada esquema está la misma estructura tres veces con los datos para películas con más de 10.000 votos, más de 1.000 votos y más de 100 votos. Note que las tablas de más de 10.000 votos tienen **menos** tuplas que las 1.000 y muchas **menos** que las 100. En esta pregunta, usted medirá el efecto de utilizar índices en los tiempos de consultas, entregando un informe (en PDF) con sus respuestas.

Use el siguiente comando para obtener los planes de consulta y tiempos de ejecución de alguna consulta:

```
EXPLAIN ANALYZE CONSULTA SQL;
```

Por ejemplo:

```
EXPLAIN ANALYZE SELECT * FROM opti.personaje100 WHERE p_nombre='Up' AND p_anho=2009;
```

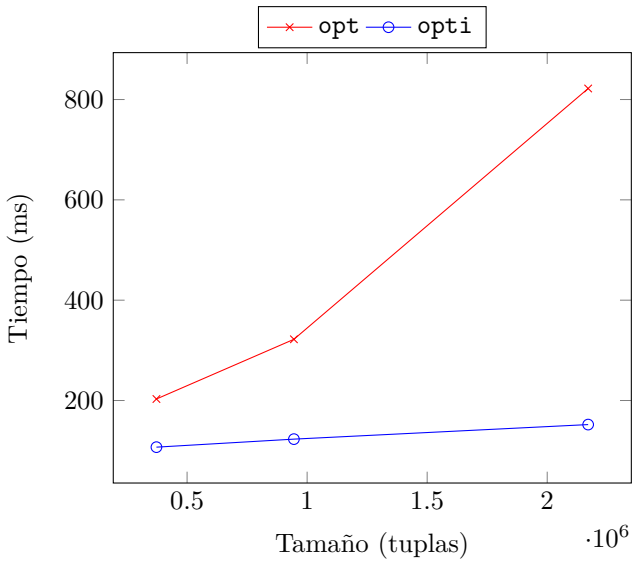


Figure 1: Gráfico de ejemplo: se muestran las curvas para la misma consulta con y sin índices y cómo varía el tiempo de ejecución respecto al tamaño (número de tuplas) de la tabla `personaje*`.

(a) Indique que índices existen para las tres tablas con el prefijo `10000` en ambos esquemas usando los dos comandos `\d+ opt.`TABLA`10000` y `\d+ opti.`TABLA`10000`. Recuerde que Postgres agrega un índice para la llave primaria por defecto, entonces `opt` solo tiene esos índices. (Se configuran los índices de las tablas `1000` y `100` de la misma forma.)

(b) Consulte por las **otras** películas (nombre y año) en las cuales los actores/actrices de PELÍCULA FAVORITA han participado (que la película aparezca en la tabla `pelicula10000`).
Para ello, escriba dos versiones equivalentes de la misma consulta: una versión sin anidación y otra versión con anidación (usando `IN`).

Para las dos consultas resultantes, debe:

i. Ejecutar las consultas en el esquema `opt` usando las tablas terminadas en `100`, `1000` y `10000` usando `EXPLAIN ANALYZE` y registrar los tiempos totales (planificación más ejecución).

ii. Ejecutar las consultas en el esquema `opti` usando las tablas terminadas en `100`, `1000` y `10000` usando `EXPLAIN ANALYZE` y registrar los tiempos totales (planificación más ejecución).

iii. Mostrar gráficamente (con la herramienta que estime conveniente) cómo varía el tiempo tomado (la suma del tiempo de ejecución y el tiempo de planificación) respecto al tamaño de las tablas, tanto en la versión sin índices, como en la versión indexada, como se ejemplifica en la Figura 1. Debe incluir un gráfico por consulta. Considere que el tamaño de la tabla es el número de tuplas en la tabla `personaje*`.

Informe

En su informe debe incluir las respuestas de **P1** junto a su justificación y una descripción de los índices de la **P2**. Finalmente, para cada consulta en la **P2**, debe incluir:

- La consulta misma (*solo* para la escala `100` en `opt`).
- La planificación de la consulta con y sin índices (con `EXPLAIN ANALYZE`, solo para la escala `100`).
- Los gráficos de comparación debidamente detallado (mirar ejemplo de Figura 1).

Universidad de Chile
Departamento de Ciencias de la Computación
CC3201 - Bases de Datos

Laboratorio 8 - Evaluación de consultas

Profesores: Claudio Gutiérrez
Matías Toro
Auxiliares: Scarlett Plaza
Daniel Radrigán
Cristian Salazar
Fran Zautzik

La entrega de este laboratorio deberá ser un informe en pdf, el cual está descrito al final de este archivo.

P1. [30 PUNTOS] Al igual que el lab anterior, se cuenta con varias tablas que son instancias del siguiente esquema abstracto:

- **pelicula**(**nombre**, **anho**,**calificacion**, **votos**)
- **actor**(**nombre**,**genero**)
- **personaje**(**a_nombre**, **p_nombre**,**p_anho**, **personaje**)

En la base de datos del curso hay dos esquemas: uno con datos indexados (`opti`) y otro sin índices (`opt`). En cada esquema está la misma estructura tres veces con los datos para películas con más de 10.000 votos, más de 1.000 votos y más de 100 votos. Note que las tablas de más de 10.000 votos tienen **menos** tuplas que las 1.000 y muchas **menos** que las 100.

(a) [10 PUNTOS] Usando el esquema `opt` cuente las tuplas en las nueve tablas presentes. Debe notar que las tablas `10000` tienen **menos** tuplas que las `1000` y muchas **menos** que las `100`. Registre las consultas y sus resultados. Usando la siguiente consulta, contar cuántos bloques (`relpages`: se llaman páginas en Postgres, cada uno de 8 kb. típicamente) hay en cada tabla:

```
SELECT DISTINCT relname, relpages FROM pg_class WHERE relname = '[TABLA-NOMBRE]';
```

Calcule el número promedio de tuplas por bloque para cada tabla. (Observe que estos conteos no cambian en el caso de `opti`, así que no hay que contar las tuplas y los bloques dos veces.)

(b) [10 PUNTOS] Use el siguiente comando para obtener los planes de consulta y tiempos de ejecución de la consulta indicada. Ejecútela en los esquemas `opt` y `opti`.

```
EXPLAIN ANALYZE SELECT * FROM [ESQUEMA].personaje100 WHERE p_nombre='Up' AND p_anho=2009;
```

Registre el plan de consulta y el tiempo de ejecución. Calcule y registre la cantidad de consultas por segundo (según la suma del tiempo de planificación y ejecución) que pueden realizarse.

(c) [10 PUNTOS] De la pregunta anterior, calcule y registre una estimación de la cantidad de bloques leídos (± 1) por las dos consultas. Para esto considere el plan, el número de tuplas en el resultado, el número de tuplas en la tabla, y el número promedio de tuplas por bloque, en cada uno de los dos esquemas.

P2. [30 PUNTOS] Considere el siguiente esquema relacional y consulta SQL. El esquema captura información sobre empleados, departamentos y finanzas de la empresa (organizadas por departamento).

- **Empleado**(**eid**, **did**, **sueldo**, **hobby**)
- **Departamento**(**did**,**nombre**, **piso**, **telefono**)

- **Finanzas**(**did**, **presupuesto**,**ventas**, **gastos**)

```
SELECT D.nombre, F.presupuesto
FROM Empleado E, Departamento D, Finanzas F
WHERE E.did = D.did AND D.did = F.did AND D.piso = 1
AND E.sueldo >= 59000 AND E.hobby = 'magic'
```

(a) [10 PUNTOS] Identifique los posibles árboles de álgebra relacional que reflejen el orden de las operaciones que un optimizador podría considerar.

(b) Suponga que la siguiente información adicional está disponible: existen índices de árbol B+ no agrupados en **Empleado.did**, **Departamento.piso**, **Deptartmanto.did** y **Finanzas.did**. Las estadísticas/catálogo del sistema indican que los salarios de los empleados oscilan entre 10.000 y 60.000, los empleados disfrutan de 200 hobbies diferentes y la empresa posee 2 pisos en el edificio. Hay un total de 50.000 empleados y 5.000 departamentos (cada uno con información financiera correspondiente) en la base de datos. Suponga una distribución uniforme. El sistema de gestión de bases de datos utilizado por la empresa tiene solo un método de join disponible, que es el loop anidado con índice.

   i. [10 PUNTOS] Para cada una de las relaciones base de la consulta (Empleado, Departamento y Finanzas), estime la cantidad de tuplas que se seleccionarían, si se aplican las selecciones correspondientes (filtros), antes de hacer los joins.

   ii. [10 PUNTOS] Construya dos árboles distintos, y argumente cuál cree usted que será más eficiente de evaluar.