Uso de bases de datos

Práctica 1: El lenguaje SQL I

Queremos disponer de una base de datos para registrar información sobre grupos musicales, canciones y músicos. A continuación se describe cada una de las relaciones.

Las relaciones con las que trabajamos son las siguientes (claves primarias subrayadas, claves foráneas en cursiva) y los atributos no pueden tener valor nulo a menos que se diga lo contrario.

MUSICIAN (id_musician, name, birth, death, gender, nationality)

La relación *MUSICIAN* contiene los datos generales sobre los músicos que aparecen en la BD. En concreto, para cada músico se guarda un número identificador (*id_musician*) que es clave primaria, el nombre (*name*), la fecha de nacimiento (*birth*), fecha de defunción (*death*) que puede ser *NULL*, el género (*gender*) y la nacionalidad (*nationality*).

El atributo *gender* solo puede tomar los valores {'*M*', '*F*'}. *M* para el masculino i *F* para el femenino.

BAND (id_band, name, year_formed, year_dissolution, style, origin)

La relación contiene información sobre los grupos musicales. En concreto, para cada grupo, se guarda el identificador (*id_band*) que es clave primaria, el nombre del grupo (*name*), el año de formación (*year_formed*), el año de disolución (*year_dissolution*) que puede ser *NULL*, el estilo musical (*style*) y el país de origen del grupo (*origin*).

El atributo *style* sólo puede tomar los valores {'*Blues*', '*Country*', '*Heavy*', '*Jazz*', '*Pop*', '*Punk*', '*Reggae*', '*Rock*', '*Soul*', '*Thrash*', '*Techno*'}.

ALBUM (id_album, title, year, *id_band*)

Información sobre los álbumes. Per a cada àlbum se almacena el seu identificador (*id_album*) que és clave primaria, el título (*title*), el año de publicación (*year*) i el identificador del grupo musical que ha grabado el álbum (*id_band*), que es clave foránea de *BAND*.

SONG (id_song, title, duration, *id_album*)

Guarda información sobre las canciones. Por cada canción guarda el identificador (*id_song*) que es clave primaria, el título (*title*), la duración en minutos y segundos (*duration*), que no puede ser negativa ni cero y el identificador del álbum al que pertenece la canción (*id_album*), que puede ser NULL y que es clave foránea de *ALBUM*, con política de anulación en caso de borrado.

MEMBER (*id_musician*, *id_band*, instrument)

Información sobre los músicos que forman parte de un grupo o grupos musicales. Se almacenan los valores (*id_musician, id_band, instrument*) que son clave primaria, donde *id_musician* es clave foránea de *MUSICIAN*, *id_band* es clave foránea de *BAND* y, finalmente, *instrument* es el instrumento que utiliza el músico en el grupo; la voz (*Vocals*) se considera un instrumento. Un músico puede tocar más de un instrumento en un grupo.

El atributo *instrument*, sólo puede tomar los valores {'*Bass*', '*Drums*', '*Guitar*', '*Keyboard'*, '*Vocals', 'Trumpet', 'Clarinet', 'Oboe', 'Flute'*}.

COMPOSER (*id_musician*, *id_song*, year)

Información sobre las canciones y los compositores de las mismas. Se almacenan los valores (*id_musician, id_song*) que son clave primaria, donde *id_musician* es clave foránea de *MUSICIAN, id_song* es clave foránea de *SONG* y el año en que se compuso la canción (*year*).

ACLARACIONES

En el fichero `create_db.sql` se proporcionan las sentencias SQL necesarias para crear la base de datos. En el fichero `inserts_db.sql` se proporcionan las sentencias de inserción de datos que hay que ejecutar para responder a las preguntas de la práctica.

Recordad que para poder trabajar sobre tablas que son de un esquema de base de datos concreto se debe utilizar el nombre del esquema como prefijo, o hay que tener actualizada la variable `search_path`. Para vuestra comodidad, al principio de cada sesión haced:

```
SET search_path TO "nombre_bd_que_utilicéis";
```

Así mismo, acordaos de eliminar de la solución que entreguéis aquellas sentencias auxiliares que utilicéis para vuestras pruebas, como `DROP`, inserciones de prueba, etc. que puedan alterar las salidas de los resultados esperados.

Nota importante: El SQL implementado en PostgreSQL puede aceptar diferentes variantes de sintaxis, que además pueden variar según la versión que instaléis, y que pueden ser o no SQL estándar. Evitad (a menos que se indique lo contrario) utilizar sentencias de este tipo, y concentraros en las que se explican en los módulos didácticos. Esto es especialmente relevante en el caso del módulo 4 (evaluado en esta primera parte de la práctica), donde se explica SQL estándar. Si utilizáis sentencias SQL estándar, vuestro código funcionará en cualquier SGBD.

Pregunta 1 (30 % puntuación)

Enunciado

En el fichero `create_db.sql` se proporcionan las sentencias SQL necesarias para crear las tablas *MUSICIAN*, *BAND*, *ALBUM, MEMBER* y *COMPOSER*.

Después de un último análisis de la base de datos, se ha llegado a la conclusión que se necesita introducir las mejoras siguientes:

1. Se pide dar las sentencias SQL para la **creación** de la tabla *SONG* según la definición dada.

También se pide dar las sentencias SQL de **alteración** de las tablas creadas, para aplicar los cambios siguientes:

2. En la tabla *MEMBER* queremos añadir las restricciones de las claves foráneas con política de borrado en cascada.
3. En la tabla *BAND,* queremos indicar que la columna *year_dissolution* puede tener un valor *NULL* y que en caso de tener un valor diferente a NULL, este no puede ser nunca inferior a *year formed*.
4. En la tabla *MUSICIAN* queremos añadir las restricciones siguientes:
    a. El atributo *gender* sólo puede tomar los valores {'*M*', '*F*'}. *M* para el masculino y *F* para el femenino.
    b. La columna *gender* puede ser *NULL*.
5. En la tabla *COMPOSER* queremos añadir las siguientes modificaciones:
    a. Añadir la restricción de la clave foránea *SONG* con política de actualización en cascada en caso de modificación.
    b. La columna *year* no hace falta y se debe suprimir.
    c. Queremos añadir una nueva columna *awards* (premios ganados por el compositor de la canción) que tendrá por defecto valor 0 y no puede ser negativo.

En el fichero `inserts_db.sql` encontraréis las sentencias de inserción que debéis lanzar sobre las tablas **una vez modificadas**. Tened en cuenta que el número de columnas que deben incluir las tablas es el que viene dado en el fichero `inserts_db.sql`, por lo cual no se aceptarán como soluciones válidas aquellas prácticas que contemplen otro tipo de columna no incluida en este fichero.

Nota importante: Para aplicar los cambios requeridos **NO** debéis utilizar los dominios explicados en el módulo teórico.

Nota: En el siguiente enlace encontraréis información sobre el orden de ejecución de las expresiones.

https://www.postgresql.org/docs/current/static/sql-expressions.html#SYNTAX-EXPRESS-EVAL

Criterios de evaluación

- *Todas las preguntas tienen el mismo peso.*
- *Las preguntas no contestadas no penalizan.*
- *Las sentencias SQL que no se puedan ejecutar (den error de sintaxis) no serán evaluadas.*
- *Las sentencias de creación de tablas que propongáis que den error al ejecutar el fichero inserts_db.sql que os proporcionamos no serán evaluadas (en definitiva, el número de columnas que han de incluir las tablas es el que viene dado en el fichero inserts_db.sql).*
- *No se evaluarán propuestas de solución que utilicen dominios (CREATE DOMAIN).*

Pregunta 2 (45 % puntuación)

Enunciado

a) Diseñad una consulta que dé las canciones interpretadas por bandas de Rock que tengan una duración inferior a 2 minutos ordenadas por su duración. En concreto queremos que se dé el nombre de la canción, su duración, el año en que se publicó en un álbum, el número de músicos que la han compuesto, (0 si no se conoce el compositor) ordenadas de mayor a menor duración y en caso de empate por orden alfabético del título.

b) Diseñad una consulta que dé las diferentes bandas españolas en las que algún músico ha nacido a partir de 1960 y que no han publicado ningún álbum entre los años 2000 y 2009. En concreto queremos que se dé el nombre de la banda, el nombre del músico nacido a partir de 1960 y el año de nacimiento del mismo.

c) Diseñad una vista (*top_bands*) que obtenga las 4 bandas que más álbumes han publicado con el nombre y el año de formación de la banda, así como el número de miembros y el número de álbumes publicados. Queremos el resultado ordenado por número de álbumes de mayor a menor, y en caso de empate por año de formación de la banda.

Nota: En el siguiente enlace encontraréis información sobre el uso de la cláusula COUNT.
https://www.postgresqltutorial.com/postgresql-count-function/
Nota: En el siguiente enlace encontraréis información sobre el uso de la cláusula LIMIT.
https://www.postgresql.org/docs/8.1/queries-limit.html

Criterios de evaluación

- *Todas las preguntas tienen el mismo peso.*
- *Las preguntas no contestadas no penalizan.*
- *Las sentencias SQL que no se puedan ejecutar (den error de sintaxis) no serán evaluadas.*
- *Se valorará positivamente el uso de sentencias SQL estándar (al margen de otros elementos indicados en el enunciado).*

- *Para obtener la máxima nota, la propuesta de solución de cada pregunta debe incluir el resultado (captura de pantalla o similar).*
- *Para obtener la máxima nota en cada pregunta, la propuesta de solución se tiene que ajustar estrictamente a lo que se pide en el enunciado (por ejemplo, tiene que incorporar todas las columnas que se esperan en el resultado, se deben hacer las ordenaciones pedidas …)*

Pregunta 3 (25 % puntuación)

Enunciado

Se ha descubierto que la composición de algunas canciones debe atribuirse al músico José Luis Perales. Esto afecta a las canciones con una duración inferior a 5 minutos que aparecen en álbumes durante la década de los 80, de grupos españoles donde el compositor tocaba más de un instrumento (incluyendo la voz como instrumento).

Proponed una **única sentencia SQL** para corregir los registros incorrectos (en caso de utilizar más de una, el ejercicio se considerará incorrecto). Por otro lado, mostrad el conjunto de las filas que se actualizan.

Nota importante: cuando probéis la implementación de esta actualización, tened en cuenta que siempre debéis partir de la misma base de datos, de otro modo podéis encontrar incoherencias en vuestros análisis.

Criterios de evaluación

- *Las sentencias SQL que no se puedan ejecutar (den error de sintaxis) no serán evaluadas.*
- *Para obtener la máxima nota en cada pregunta, la propuesta de solución se tiene que ajustar estrictamente a lo que se pide en el enunciado (hay que actualizar los datos estrictamente necesarios) y con una única sentencia de UPDATE*.
- Para obtener la máxima nota, la solución debe ser eficiente (por ejemplo, se valorará negativamente hacer más *joins* de las necesarias).
- Se debe argumentar a la respuesta proporcionada. Para obtener la máxima puntuación se debe mostrar la relación de filas que se actualizan.