**PECL2 – Documentation**

Manuel Urbano Schere Rodríguez & Sergio de la Mata Moratilla

2ºGIC E & 2ºGII E

**Introduction**

This practice was made using as base the requirements provided at PECL1 and the previous practice made. For that reason, it was needed to make a set of changes to the previous practice to get the request results from the new one.

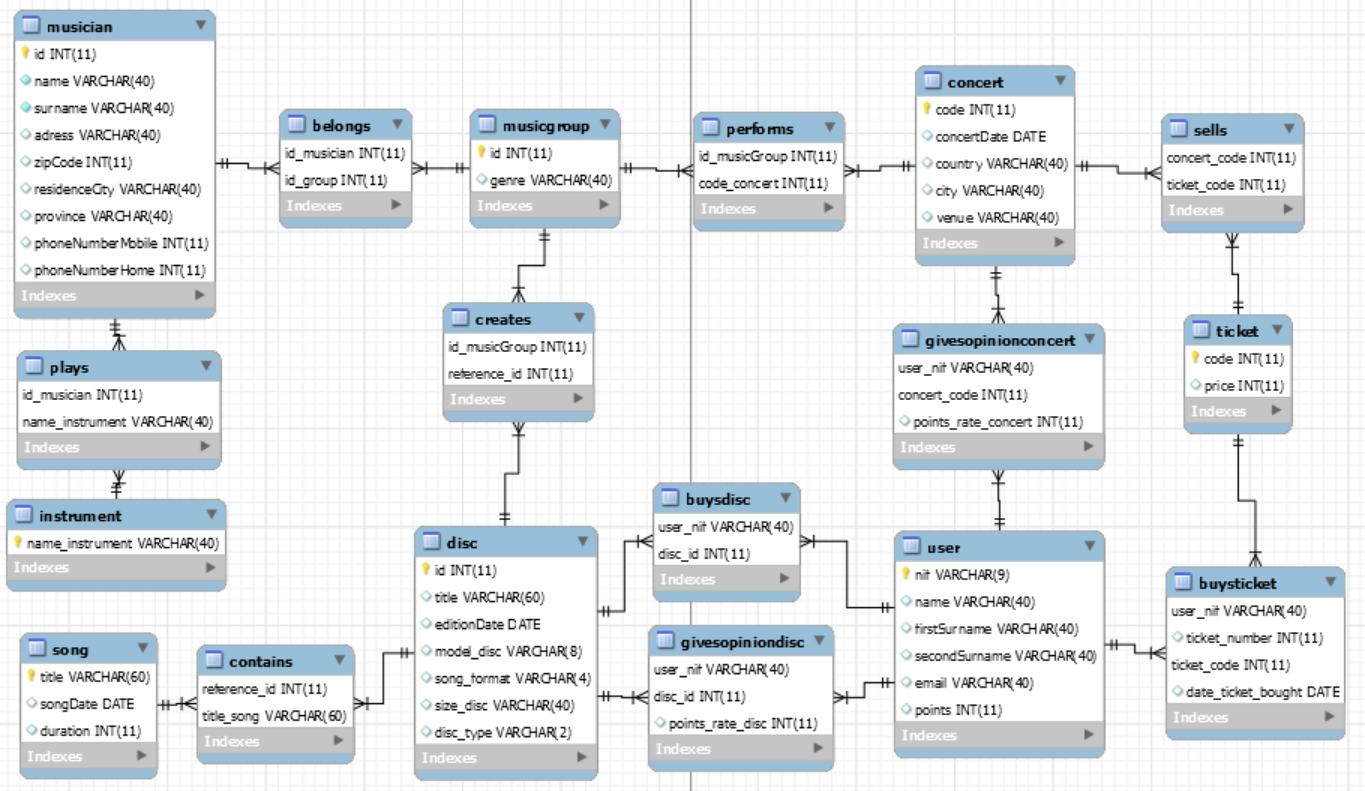
**Some changes which have been made from PECL1. The changelog is as follows:**

* The relation “composes” has been removed due to a relational redundancy with “contains” and to the difficulty implementing it.
* “buysTicket” has a new field called “date\_ticket\_bought” which symbolizes the day in which the user purchased the ticket. It also has a “ticket\_number” field that indicates the number of the tickets from the ones he/she has bought for a same concert with a maximum number of tickets that a user can buy for it.
* “physical” and “digital” tables along with the “is a” relation have been removed due to the difficulty of implementing inheritance with MySQL. Instead, the disc table now has a “model\_disc” field which indicates if it is a physical or a digital disc. If it is a physical disc, the “disc\_type” field will have either CD or LP as an indicator of the disc´s physical format. If it´s a digital disc, the “song\_format” field will indicate if it has a mp3, wav or other digital format, and the size\_disc field will contain the disc´s size in megabytes. Whenever one of the options is taken the fields related with the other will be directed to null.
* The field “phoneNumbers” in musician has been substituted by the “phoneNumberMobile” and “phoneNumberHome” fields. This has been done due to the difficulty implementing multivalued attributes and to simplify the program.
* “givesOpinionConcert” and “givesOpinionDisc” both have new fields “points\_rate\_concert” and “points\_rate\_disc” respectively, which indicate the score that a user has given to a concert or disc, being the lowest score a 0 and the highest one a 10. The field “opinion” has been removed as it was unnecessary.
* The “duration” field of song has been changed from a varchar to an int which indicates the duration in seconds. This allows us to compare song durations in an easy manner.
* Cardinality constraints are mostly gone due to the difficulty of correctly implementing it using MySQL. Instead, it was used the field “ticket\_number” in the case of the relation “buyTicket” and in the rest of cases to represented it, it was used foreign keys in the relations tables like “plays”, “belongs” and “contains” letting us to have different number of relations between the entities.
* Some minor renames have been made, which should seem obvious.
* The entity “Instrument” isn’t anymore a weak-entity to avoid more complexity in the database.

**Advantages from this new model**

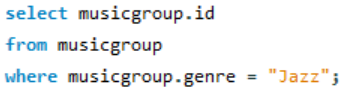
* This new model lets the database be more organized and reduces the complexity from the weak entities which were created in the previous model.
* If a new insertion is made regarding the entity “disc” is made, the user doesn’t need to indicate to which kind of disc do we want to introduce it.
* If a new insertion is made regarding the entities “instrument” and “song” it is not need to specify the id from the musician, in the case of the “instrument”, or else have problems regarding having as discriminator the primary key of two other entities plus their attribute consider part of the discriminator, in the case of the “song”.

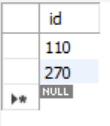
**The following diagram was obtained using reverse engineering from MySQL and accurately illustrates the current database structure:**



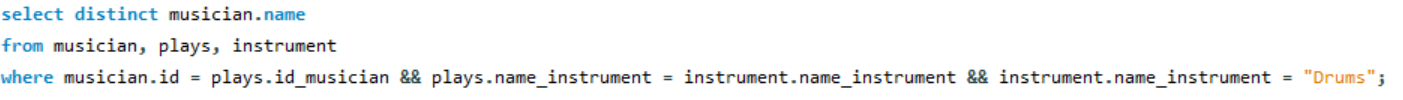
**Queries and results:**

**1.Show all Jazz groups**



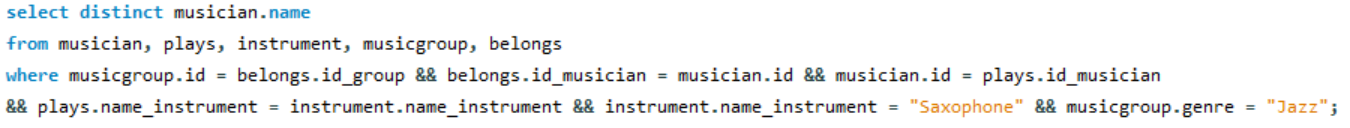


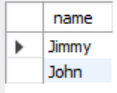
**2. Show the name of all musicians playing drums**



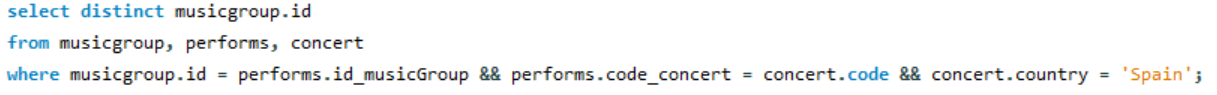


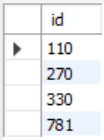
**3. Show the name of all Jazz saxophonists**



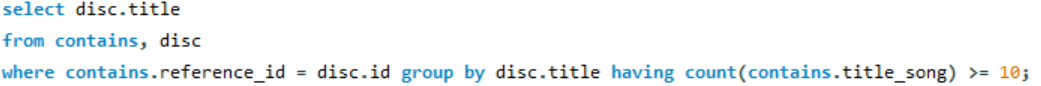
****

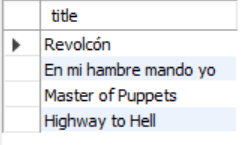
**4. Show all the groups that have given concerts in Spain**

****

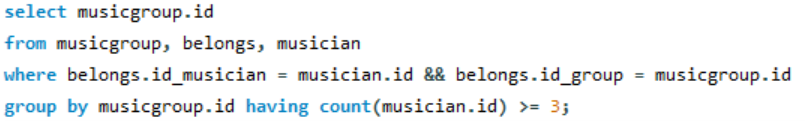
****

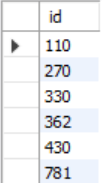
**5. Show all discs with more than 10 songs**

****

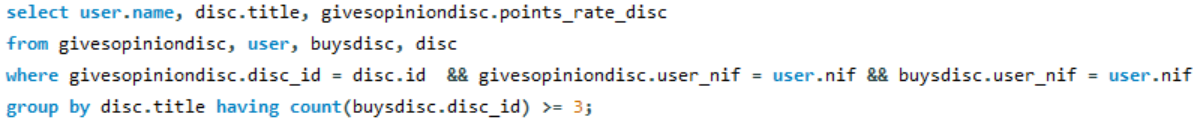
****

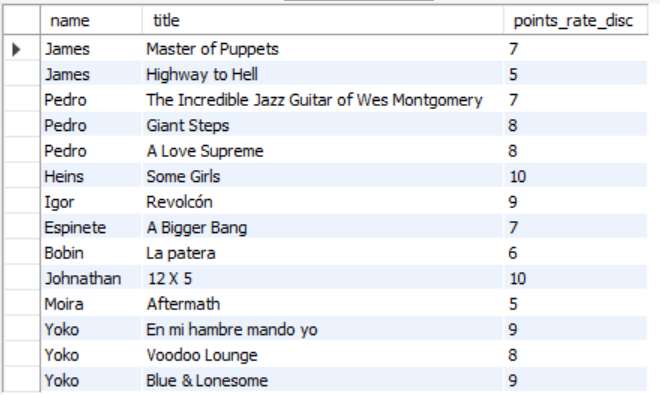
**6. Show all groups with more than 3 members**

****

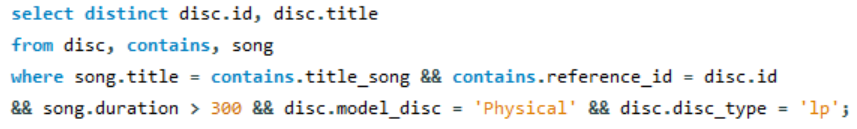
****

**7. Show all reviews of discs issued by users who have purchased at least three discs**

****

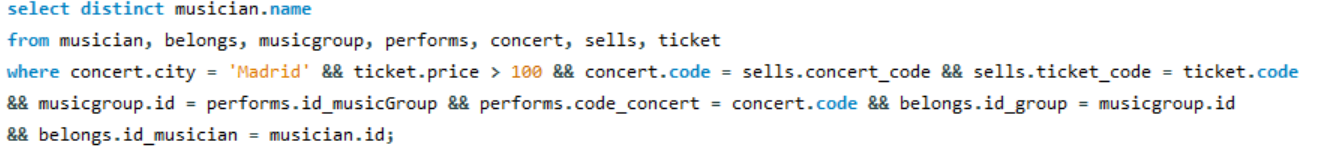
****

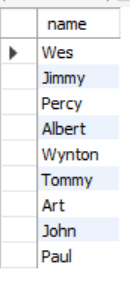
**8. Show all vinyl records that have at least one song longer than 5 minutes and have been recorded by a Rock group.**

****

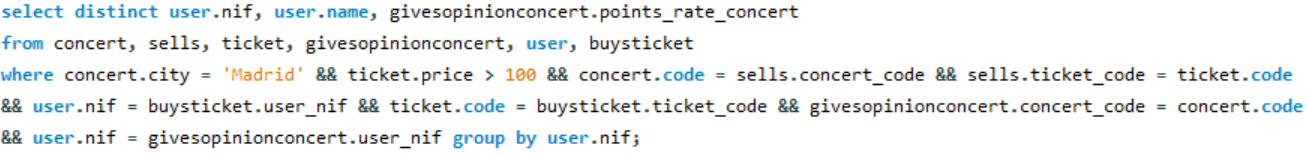
****

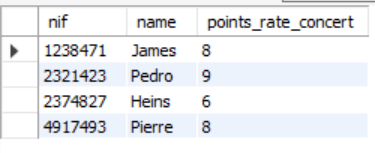
**9. Show the name of the musicians of groups that have given concerts at Madrid and have put on sale tickets with a price higher than 100 euros.**

****

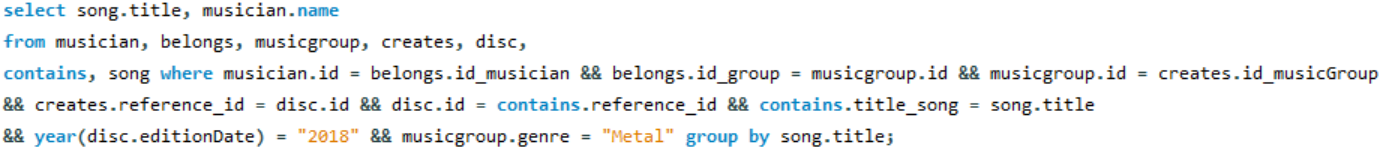
****

**10. Show the opinions of the users attending the concerts shown form the previous query.**

****

****

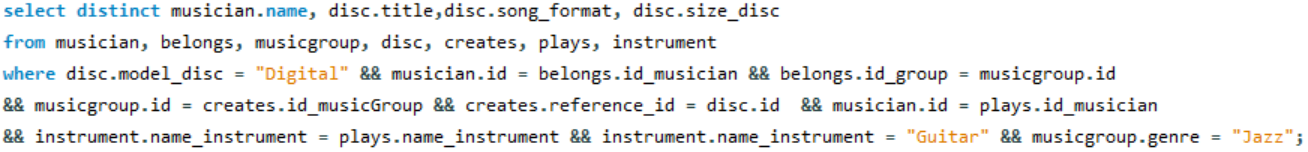
**11. Find the songs and composer names for Heavy Metal albums published during 2018.**

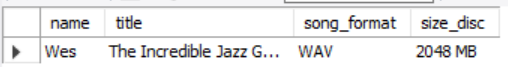
****

****

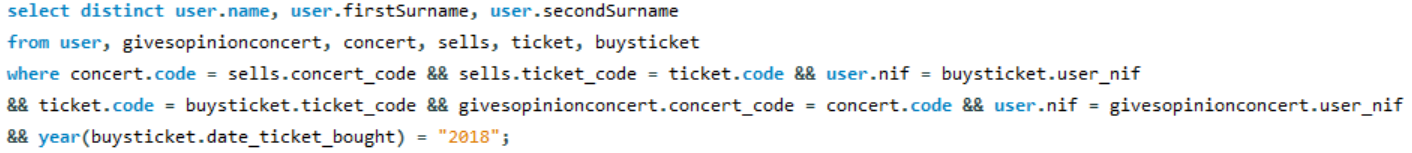
(This query was a bit odd because if we show all the musician names, the same songs are repeated again for each of the members. We are considering that the whole group composes a song, not just one member).

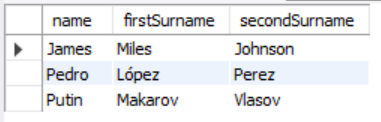
**12. Show name of the Jazz group guitarists having digital discs together with the name and format of such discs and their size in MB.**

****

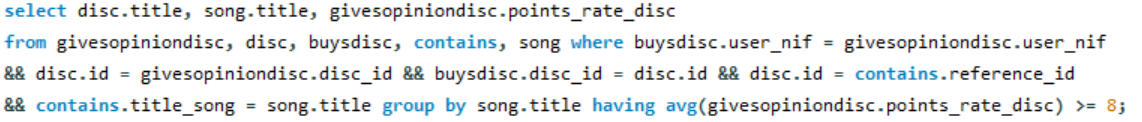
****

**13. Show name and surname of the users who bought tickets during 2018 for jazz groups and published opinions about those concerts.**

****

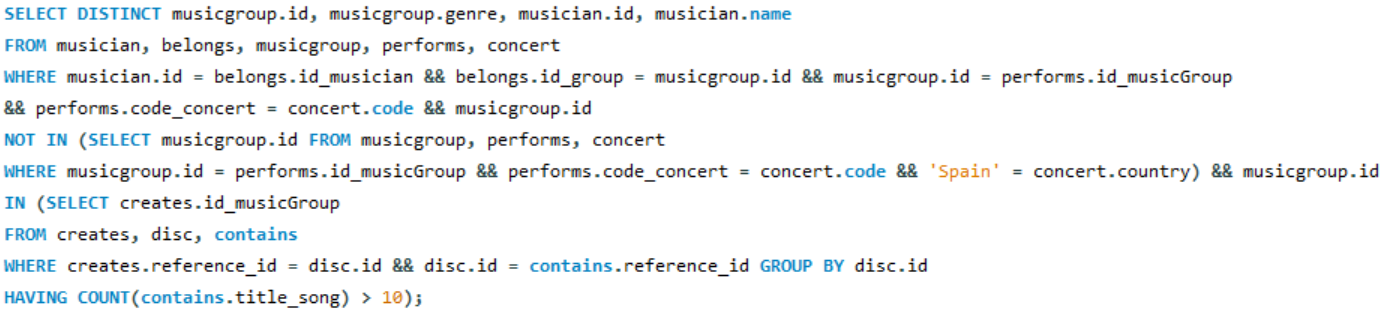
****

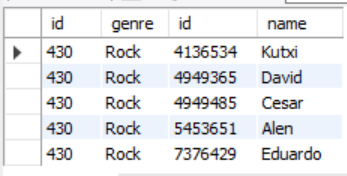
**14. Show the title, and the songs for any disc bought by users whose ratings on that disc were 8 or more points.**

****

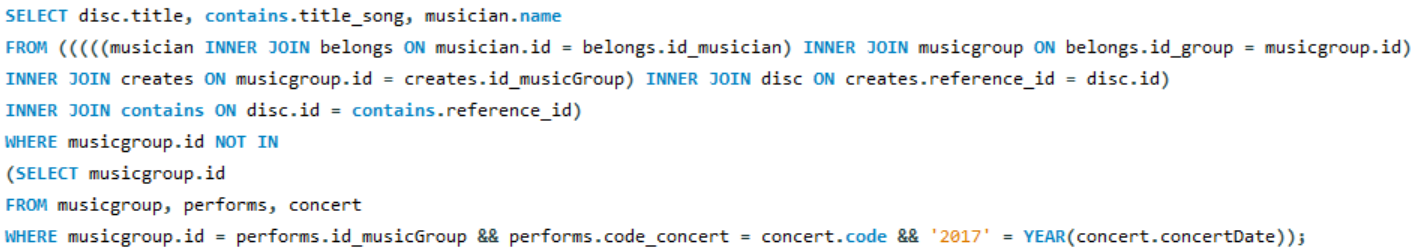
****

**15. Find the groups and their members that never performed in Spain and have recorded discs with more than 10 songs.**

****

****

**16. Show song names and composer names for groups not performing any concert during 2017.**

****

****

(The same situation happens as in query 11).

**Difficulties:**

At first, we had some problems regarding the creation of tables with foreign keys as we wanted to put as primary keys from other entities, but which have more than one field as primary key. We learnt how it worked, but it wasn’t used at the final version from the database.

There were also during a time some complications regarding limited cardinality, regarding the number of tickets from a specific concert bought by a user; inheritance, in the case of the types of discs, and considering the order of how the different insertions needed to be made. Even though we had those problems, finally we could solve them.

Finally, we had a few problems with the command not in at the begging, making us to see that we need to compare with something to get the correct result instead of nothing.