Introduction to Database Systems
Homework 4

Eleni Tzirita Zacharatou

Fall 2022

Do as many of the following exercises as you have time for. Focus on the exercises you feel will benefit your preparation the most. To pass the homework, you only need to have *5/80* points.

1 Hardware and DBMS Design [10 points]

Question 1.A Select the correct statements below:

(a) SSDs are especially well suited to improve the performance of retrieving large result sets using unclustered indexes.

(b) Before a transaction can be committed, all the disk pages it has updated must be written to secondary storage to ensure durability.

(c) In a main memory system, it is not necessary to include the "old" values when logging changes made by transactions to secondary storage.

(d) As discussed in a lecture, Google claims to have built a distributed system that offers both consistency and availability in the face of network partitions; this proves that the CAP theorem is wrong.

Question 1.B Reflect upon why ACID transactions are rarely used in distributed systems.

2 Data Systems for Analytics [10 points]

Question 2.A Select the correct statements below:

(a) Today's key-value stores implement all the functionality required to support *all* needs of analytics applications.

(b) In big data, “velocity” means that it is necessary to react quickly to the large amounts of data that are being added to the system.

(c) Emails are an example of semi-structured data.

(d) Spark supports low-latency big data analytics.

Question 2.B Consider a scenario in which you have 1PB of raw data files that have just been produced as a result of a scientific experiment. You are trying to make scientific discoveries using complex computations over this data, and also determine the experiments you would like to perform in the future. You have a cluster of machines in your lab (around 100 nodes), which have 128GB main memory and 16 cores each. Explain (with convincing arguments) what type of data management/processing system you would choose for this scenario.

3 Normalisation [20 points]

Consider the SQL script and the associated description in Part 2 of Homework 3. In Homework 3, you normalised the Rentals relation. Now, you are asked to normalise the Projects relation. For the Projects relation, take the following steps:

1. Find all the FDs in the relations, given the constraints and assumptions from Homework 3.
2. Decompose the relation until each subrelation is in BCNF/3NF, while preserving all non-redundant FDs. Write down the resulting schema description in a simple Relation(columns) format.
3. Write the detailed SQL commands to create the resulting tables (with primary keys and foreign keys) and populate them, by extracting the relevant data from the original relations.
4. Select the correct normal form for the decomposed schema.

Please note that the details of these steps, as well as constraints of the relation, are found in the description of Homework 3. Please refer to the description there!

4 Godly strife (by Johan von Tangen Sivertsen) [20 points]

Zeus is tired of all the infighting among the gods. He has brought you to mount Olympus to help deal with the problem. To avoid conflicts, Zeus wants a simple database that contains a description of each promise that the gods have made to humans. That way, a god can quickly check for conflicts before making a new promise. In case conflicts arise anyway, the database should track the various sacrifices made by humans to the gods. The time, place, and items sacrificed should all be recorded. Sacrifices are divisible into one of three categories, flesh, wine, and valuables (e.g., gold, gemstones, etc.). Each category has different attributes that describe them, but they all have a value. It should be possible to sum up the total value of sacrifices a human has offered to a God to help resolve any conflicting promises.

Furthermore, some humans are appointed priests. A priest is always promised protection by his associated God, and if a sacrifice is conducted in a ceremony presided over by a priest, it is twice as valuable as normally. Any priest serves a single God. Finally, Poseidon has requested that the database also tracks if a human attacks any sea-monster or cyclops, and in that case a hecatomb should be deducted from their total sacrifice value for all gods.

1. Draw an ER-diagram that supports the requirements. Feel free to add identifiers and other attributes.
2. Write the DDL for a database according to your design.

Remember that Zeus will likely strike you down with lightning if he is not pleased with your work!

5 SQL [20 points]

Note: The following queries are relatively simple. In the exam, there will also be some more advanced queries, including division queries. Please, go through the old exams for some examples of more challenging queries.

In this part you will work with a music database. To start working with the database, import/run `HW4.sql` found in LearnIT using the PostgreSQL DBMS on your laptop. The database has the following relations:

```
Artists(ArtistId, Artist, ArtistImageUrl)
Songs(SongId, Title, ArtistId, Duration, IsExplicit, ImageUrl, ReleaseDate)
Genres(GenreId, Genre)
Albums(AlbumId, Album, AlbumImageUrl, AlbumReleaseDate)
AlbumArtists(AlbumId, ArtistId)
AlbumGenres(AlbumId, GenreId)
AlbumSongs(AlbumId, SongId)
SongGenres(SongId, GenreId)
```

Primary and foreign key attributes have names that end with `Id`. The meaning of other attributes should be self-explanatory. The first four relations have their first attribute as primary key. In the last four relations there is a composite primary key consisting of both attributes, and each attribute separately is a foreign key reference to one of the first four relations. Secondary indexes exist on `AlbumArtists(ArtistId)`, `AlbumGenres(GenreId)`, `AlbumSongs(SongId)`, `SongGenres(GenreId)`, and `Songs(ArtistId)`.

You will need to work with the `ReleaseDate` and `Duration` attributes, which have the type `date` and `time`, respectively. The expression `interval '1 minute'` can be used to generate a duration value to compare to, the expression `extract(year from ReleaseDate)` can be used to get a year from a date, and `extract(epoch from Duration)` can be used to get the total number of seconds in an interval.[1]

Answer each of the following questions using a single SQL query on the examination database. Enter the result of each query into the quiz on LearnIT. As before, queries should adhere to the detailed guidelines given in Homework 1.

(a) In the database, 372 songs have a duration of at most 1 minute. How many songs have a duration of over 1 hour?

(b) What is the total duration, in seconds, of all songs in the database?

(c) The database contains just 5 songs released in 1953. What is the *largest* number of songs released in a single year?

Note: This is a very simple query. Try also to answer which year had the largest number of songs. Observe how much harder this query is!

(d) The database contains 12 albums by the artist `Queen`. How many albums by the artist `Tom Waits` are in the database?

(e) The database contains 187 different albums with a genre whose name starts with `Ele` (for example, some of these have the genre `Electronica`). How many different albums have a genre whose name starts with `Alt`?

(f) For how many songs does there exist another different song in the database with the same title?

Note: Which join method is used by PostgreSQL to evaluate this query? Does the join method change if you have an index on `Songs(Title, SongId)`*?*

(g) The average number of `albumIds` per `genreId` in `albumGenres` is 26.5246. An album can have multiple genres. What is the average number of `genreIds` per `albumId`?

(h) An album can have multiple genres. There are 1215 albums in the database that do *not* have the genre `Rock`. How many albums do *not* have the genre `HipHop`?

[1]For more details, see: `https://www.postgresql.org/docs/current/functions-datetime.html`.