

# Databases

## Project Assignment

### Academic Year 2019-2020

Professor:  
Prof. Dr. Bas Ketsman ([Bas.Ketsman@vub.be](mailto:Bas.Ketsman@vub.be))

Assistants:  
Maxim Van de Wynckel ([Maxim.Van.de.Wynckel@vub.be](mailto:Maxim.Van.de.Wynckel@vub.be))  
Ruben Opdebeeck ([Ruben.Denzel.Opdebeeck@vub.be](mailto:Ruben.Denzel.Opdebeeck@vub.be))

## Project Description

In this document you can find the description for the database that you are tasked to design. The questions are similar to the exercises that were given during this semester.

Questions concerning the exercise can be asked to either assistants. However, we can only provide clarity on the description or questions. **This assignment has to be done individually!** Each project will be checked for plagiarism upon submitting.

We expect a project report of around **6 pages** (ER images not included) that includes answers on all of the questions. More in-depth descriptions on the required answers are given on each individual question. Your deliverable needs to be saved in a file "FirstnameLastname-studentnr" (e.g. if your name is John Doe with student number 1234, it would be "JohnDoe-1234").

During the oral exam in June, you will receive several questions concerning the project.

**Deadline:** Sunday 24th of May, 2020 23:59 (Brussels Time)

**Deliverables:** Project report (PDF) + database (SQL)

**Submitting:** Upload through Canvas (upload section will be opened before the deadline). When you have issues uploading, please send your project as an email to **both assistants**.

## Streaming Platform

A start-up wants to build a new video streaming service and is asking you to develop its database. The service has three different types of users: viewers (that just watch videos), content creators (that produce and upload videos) and moderators that manage the platform.

All users of the service have at least a user name, email address and date of birth. Content creators also have a first and last name, an address and bank account number. Similar to content creators, moderators have first and last name, an address and bank account number. Moderators can not produce videos, as this would cause a conflict of interest. However, both moderators and content creators can be normal viewers on the platform and have the same capabilities as normal viewers.

Content creators manage one or more channels. Each of these channels has a channel name, a description and category. A channel hosts a number of videos, uploaded by the channel manager. These videos have a title, a description, the length (in seconds) of the video, the upload date, the published date, an age restriction, a like and dislike counter and finally multiple tags that describe the video in broad terms.

When content creators upload a video, the video can have multiple different stages: during the upload and processing stages it is not visible to viewing users and moderators. Videos can also not be seen by other users when the video is pending approval (see later). When starting the upload, a unique URL is created (similar to other streaming platforms such as YouTube) that users can use to share the video.

Users can watch videos, potentially multiple times. The date at which the video is watched is stored in the database, as well as the time spent watching the video (their attentiveness). Users can also comment on videos. These comments contain the text and the date at which the comment is posted. Users can like any comment and even post comments on other comments (maximum one level). Users can subscribe to channels. In this case, the database also stores the date at which the subscription was made and the percentage of the channel's videos the user has watched.

Moderators moderate a number of channels to ensure their content follows the rules of the platform. To this end, they moderate the channel, the videos, and the comments posted on each video. Before videos can be published, they need to be approved by the moderator. Moreover, moderators need to review each comment after it is posted, so the system records which comments have been reviewed by which moderator. Note that reviewing comments is a shared task, and not specific to the channel the moderator moderates.

## Potential Extensions

*Part of this project assignment is to add your own personal ideas into the project. Below are some potential extensions that you can add to the project. You are free to create your own (non-trivial) extension, but make sure to create at least two queries related to your extension.*

- **Advertisements:** The platform offers companies the possibility to show advertisements before videos. To this end, it stores advertisement company profiles, containing the name of the company, its address (street, number, postal code, city, country), an email address, a telephone number, and a bank account number for billing purposes.

Advertisers can upload advertisement videos in a similar manner to content creators. They additionally provide tags for their ads, and the amount of money they are willing to pay for each ad view. The platform will automatically match videos and ads so that each video has a number of related, on-topic ads, and stores these matches in the database. When a user watches a video, there exists a chance that he/she is shown an ad at the start. The specific ad that is shown when a user views a video, if any, is stored in the database.

To perform the billing, the start-up asks you to write a query that returns, for a given advertiser, the cost of each of their ads in the past 30 days. This query should return the number of times the ad was shown in the past 30 days, the cost for a single ad view, and the aggregate cost for this advertisement over all views of the past 30 days.

To further improve their short-term revenue, the start-up wants to promote “influential”

content creators who get lots of ad views. Write a query to find the 10 most influential content creators, i.e., the creators whose total ad view count across all their videos is highest.

- **Copyright:** The start-up fears its platform may be used for illicit purposes, such as sharing copyrighted content. To prevent this, they enrol the help of copyright agencies. Copyright agencies have a unique company ID, a name, an address (street, number, postal code, city, country), an email address, and a telephone number.

Copyright agencies upload videos containing copyrighted information, which is kept internal to the platform. Each of these videos have a copyright ID and a title. The system periodically scans the content creators' videos to find videos which closely match the copyrighted content. Such matches are stored in the database, containing the video, the copyrighted content, and whether or not the match is confirmed. A moderator reviews these matches and confirms or rejects matches.

To notify the appropriate content moderator of a potential copyright infringement, the start-up asks you to write a query. For each moderator, return all matches which have not yet been reviewed, and for which the moderator is responsible (i.e., the video is posted on a channel which is moderated by the moderator).

In case copyright infringement becomes rampant, the startup wants to know which content creators are to blame. Write a query that returns the names and addresses of the top 10 content creators uploading infringing material, i.e., the creators with the largest numbers of videos with confirmed copyright matches.

- **Video quality:** Different viewers have different Internet connections, and the start-up wants to keep video buffering to a minimum. Hence, they define a number of quality profiles for the videos on the platform. These quality profiles have an ID, a name, a video codec, a video resolution, a video bit rate, an audio codec, and an audio bit-rate.

Each video is available in at least one quality profile. Note that not all videos are available in all quality profiles (e.g., a 720p source video will not be up scaled to 1080p). Each user has a preferred quality profile, which the platform will attempt to provide whenever possible. The platform thus records for each view, the quality profile that has been selected while viewing the video.

The start-up wants to periodically clean up its file storage and remove rarely used video versions to save storage space. Write a query that finds the video IDs of every video that is available in all quality profiles.

Moreover, write a query that returns, for a given channel ID, all the channels videos IDs and for each of these videos, the quality profile that has been viewed the least over the past 14 days. To prevent deleting too many versions, the startup wants your query to ignore a video if it is only available in 3 or less quality profiles, or if the least popular quality profile has more than 10.000 views in the past 14 days.

- **Subtitles:** To accommodate users speaking different languages, the platform wants to add subtitles to videos. It enrolls the help of its viewers, who can volunteer to be translators. Translators are normal viewers and additionally have a number of languages they speak. Translators create subtitles for videos, each subtitle contains the contents of the subtitle, the language into which it is translated, and the date at which the subtitle is added to the video.

Any viewer can create a request for translation for a video. Such a request contains the language into which the video should be translated. Requests are filled by a translator, and the system records which subtitle was used to fill the request.

The start-up wants to monitor which languages are most in need of additional translators, so it can focus its recruiting efforts on certain demographics. To aid them in making these decisions, write a query that returns all languages for which there exist open requests, along with the number of open requests. Make sure to order the results by the number of open requests.

To entice the translators to translate more videos, the platform should provide a leader board. In order to populate this leader board, the start-up asks you to write a query which returns the user names of the top 10 weekly translators, i.e., the translators that have created the most subtitles over the last 7 days.

## Question 1: (E)ER Modelling

For the first question you have to create a complete (E)ER-model for the database with the above description. Additionally, we expect a small description (1 paragraph) for each entity and relation inside this model.

*Tip: Check question 5 for the queries that you will have to perform on this database. This can help you to determine the functional requirements.*

### Instructions

You can use any drawing utility for the model. However, we do request a digitally created copy (not a scanned picture) creating using a tool such as <https://draw.io>. When including the image in your report, make sure it can be zoomed in. To do this in draw.io, you can export the image as an SVG.

## Question 2: (E)ER Reducing

Reduce the above (E)ER-model to schemas. If you made changes to the names of entities, attributes or relations - add a small sentence where you explain your reasoning.

## Question 3: Functional Dependencies

Using the schemas from above, provide the functional dependencies for each schema. If not already, put the schemas in 6NF.

*Note: Make sure to clearly describe any changes that you make when converting it to 6NF*

## Question 4: SQL (Creating)

Provide SQL queries to create the schemas reduced in question 2. You can also provide an SQLite database file. Adding some test data can be useful for question 5.

### Instructions

You can either provide the SQL CREATE statements to create the database or the database file. When you include the database file, make sure that it is an SQLite database.

## Question 5: Relational Algebra and SQL

For this question we expect both **SQL queries** and **Relational Algebra** formulas to get the required information.

1. Find a list of all comments that still need reviewing.
2. Find a list of all new videos from channels that the user is subscribed to (only the first 20).
3. Find how many times a specific user has watched a specific video.
4. The start-up wants to show popular videos on the front page of their website. Write a query that finds the 20 most popular videos of the past 24 hours, i.e., the videos with the most views recorded in the past 24 hours.
5. Similar to the previous question, find the 20 most popular videos of the past 24 hours without counting multiple views from the same user (e.g. unique views).
6. The start-up noticed that top videos is good, but not all viewers prefer the same type of videos. Based on the categories of the channels that the user is subscribed on, get the all-time top 20 most popular videos in those categories.
7. Find all pairs of user names that share at least one channel in their subscriptions.
8. For each channel, calculate the average viewing time of its videos, in percent, that have been viewed by its non-subscribed viewers.
9. Create a search query where users can search videos based on words that appear in either the channel description, keywords, content creator name or video description
  - Results should be sorted by the probability of the match. E.g. matches in the video description have a higher probability weight than matches in the channel description
10. *As mentioned in the project description, create two useful queries for your extension. Describe the use case where these queries would make sense.*

### Instructions

SQL queries have to be included in your report. Relational Algebra formulas also need to be included in your report. Preferably these formulas are created using LaTeX or the equation function in MS Word ([click here](#)). In case there is still an issue including the formulas, you can write them on paper and include a scanned copy of those formulas in your report.

## Question 6: Indexing

Given the queries of question 5 (only the given queries, not the two queries that you had to choose yourself), where would you put indices to speed up the queries. Give the type of index and a small paragraph on why you chose these indices.

### Instructions

For each of the 9 queries in question 5, give the indices you would place, the attribute and schema you would place them on, the type of index and a short motivation.