DIAS: Data-Intensive Applications and Systems Laboratory
School of Computer and Communication Sciences
Ecole Polytechnique Fédérale de Lausanne
Building BC, Station 14
CH-1015 Lausanne
URL: http://dias.epfl.ch/

Lab 1 Description

CS-300 Data-Intensive Systems
Spring 2024

Table of Contents

Summary

This document describes the *lab 1*.

We provide the ER model, the tables available in the database, as well as the queries that you should write. This document explains the elements we provide as the common starting point for every student. This document should also be considered educational and instructional, and we hope you find it useful.

READ THE DOCUMENT FULLY AND CAREFULLY.

Introduction

This document presents the ER model, and reasoning for the model simplifications. The purpose of this document is to serve as a starting point for **lab 1** and to be instructive and educative regarding some common practices that we will explain.

In this document, we provide the following:

1. ER model,
2. Tables available in our DBMS,
3. Queries for you to write and run.

NOTE: This is not the only, best, or ideal ER model. Modeling often requires simplifications and adapting to a particular use case of existing entities and relationships that can be captured in a best effort by the data related to some real-world process or informed by domain experts.

ER Model

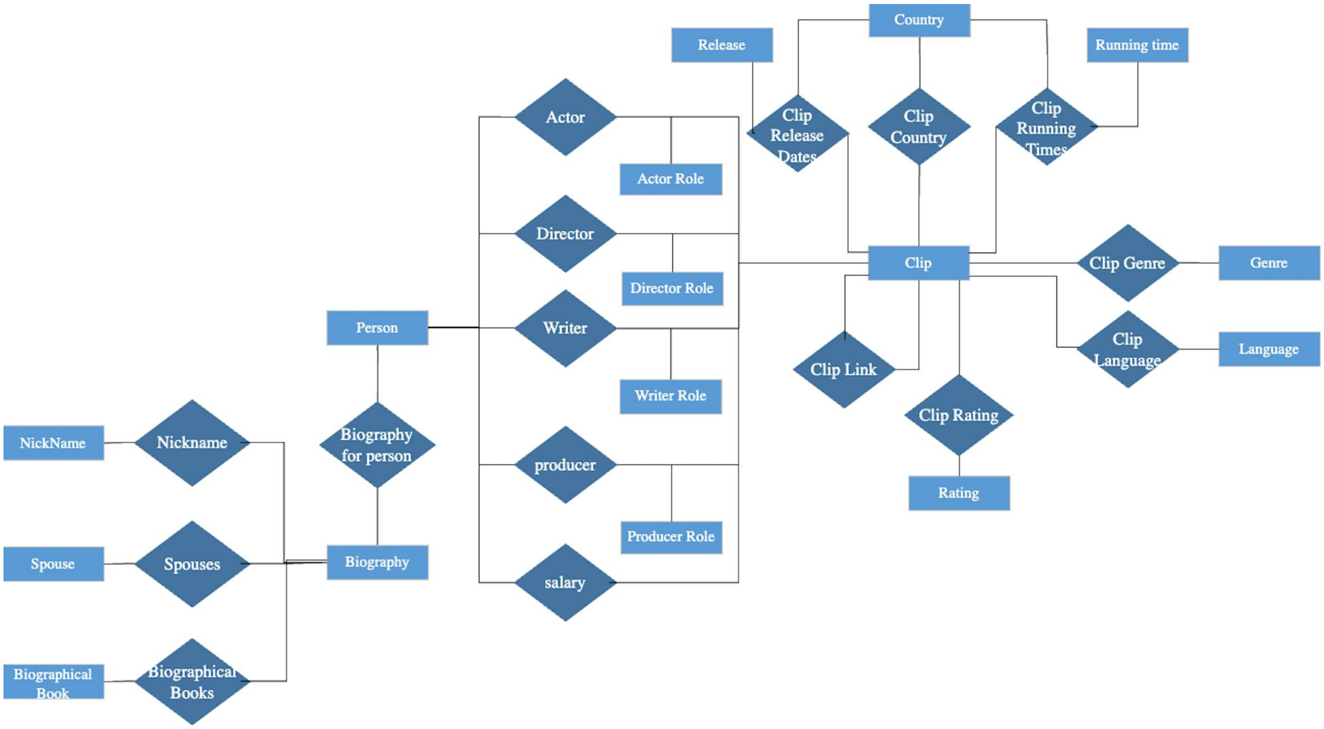


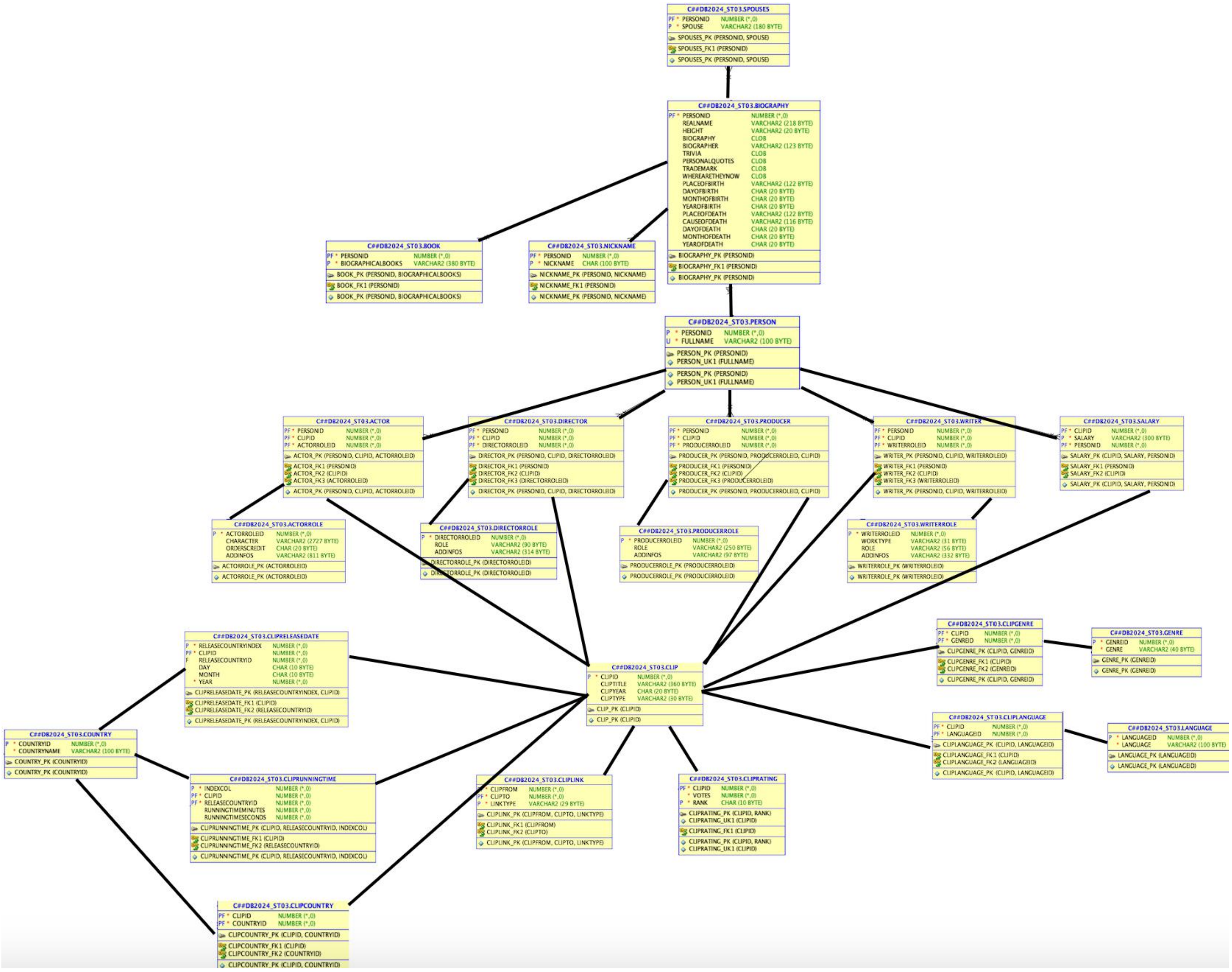
Figure 1. ER Model overview, simplified, without all the attributes and the keys

Different relationship types(1-to-1, many-to-many, etc) between entities and constraints are not represented, and weak entities are not marked.

One common design for ER model schema is a snowflake schema – where fact tables (that contain and connect the information) and dimension tables (attributes or sets of attributes grouped as entities further describing the data) are normalized and organized so they resemble a snowflake. The effect of such design is normalization – where data is split up to avoid redundancy but introducing the need for joins. Formally, you can check out normalization and normal forms. 1st normal form is the basis of the relational model: each table column must have a single value, and sets of values or nested records are not allowed.

Database Tables

The tables in the database, containing information about the attributes per table, their primary keys, their constraints and with which tables they are connected, are provided bellow. **This is not the ER model.**


C##DB2024_ST03.SPOUSES
PF * PERSONID NUMBER (*,0)
P * SPOUSE VARCHAR2 (180 BYTE)
SPOUSES_PK (PERSONID, SPOUSE)
SPOUSES_FK1 (PERSONID)
SPOUSES_PK (PERSONID, SPOUSE)
C##DB2024_ST03.BIOGRAPHY
PF * PERSONID NUMBER (*,0)
REALNAME VARCHAR2 (218 BYTE)
HEIGHT VARCHAR2 (20 BYTE)
BIOGRAPHY CLOB
BIOGRAPHER VARCHAR2 (123 BYTE)
TRIVIA CLOB
PERSONALQUOTES CLOB
TRADEMARK CLOB
WHEREARETHEYNOW CLOB
PLACEOFBIRTH VARCHAR2 (122 BYTE)
DAYOFBIRTH CHAR (20 BYTE)
MONTHOFBIRTH CHAR (20 BYTE)
YEAROFBIRTH CHAR (20 BYTE)
PLACEOFDEATH VARCHAR2 (122 BYTE)
CAUSEOFDEATH VARCHAR2 (116 BYTE)
DAYOFDEATH CHAR (20 BYTE)
MONTHOFDEATH CHAR (20 BYTE)
YEAROFDEATH CHAR (20 BYTE)
BIOGRAPHY_PK (PERSONID)
BIOGRAPHY_FK1 (PERSONID)
BIOGRAPHY_PK (PERSONID)
C##DB2024_ST03.BOOK
PF * PERSONID NUMBER (*,0)
P * BIOGRAPHICALBOOKS VARCHAR2 (380 BYTE)
BOOK_PK (PERSONID, BIOGRAPHICALBOOKS)
BOOK_FK1 (PERSONID)
BOOK_PK (PERSONID, BIOGRAPHICALBOOKS)
C##DB2024_ST03.NICKNAME
PF * PERSONID NUMBER (*,0)
P * NICKNAME CHAR (100 BYTE)
NICKNAME_PK (PERSONID, NICKNAME)
NICKNAME_FK1 (PERSONID)
NICKNAME_PK (PERSONID, NICKNAME)
C##DB2024_ST03.PERSON
P * PERSONID NUMBER (*,0)
U * FULLNAME VARCHAR2 (100 BYTE)
PERSON_PK (PERSONID)
PERSON_UK1 (FULLNAME)
PERSON_PK (PERSONID)
PERSON_UK1 (FULLNAME)
C##DB2024_ST03.ACTOR
PF * PERSONID NUMBER (*,0)
PF * CLIPID NUMBER (*,0)
PF * ACTORROLEID NUMBER (*,0)
ACTOR_PK (PERSONID, CLIPID, ACTORROLEID)
ACTOR_FK1 (PERSONID)
ACTOR_FK2 (CLIPID)
ACTOR_FK3 (ACTORROLEID)
ACTOR_PK (PERSONID, CLIPID, ACTORROLEID)
C##DB2024_ST03.DIRECTOR
PF * PERSONID NUMBER (*,0)
PF * CLIPID NUMBER (*,0)
PF * DIRECTORROLEID NUMBER (*,0)
DIRECTOR_PK (PERSONID, CLIPID, DIRECTORROLEID)
DIRECTOR_FK1 (PERSONID)
DIRECTOR_FK2 (CLIPID)
DIRECTOR_FK3 (DIRECTORROLEID)
DIRECTOR_PK (PERSONID, CLIPID, DIRECTORROLEID)
C##DB2024_ST03.PRODUCER
PF * PERSONID NUMBER (*,0)
PF * CLIPID NUMBER (*,0)
PF * PRODUCERROLEID NUMBER (*,0)
PRODUCER_PK (PERSONID, PRODUCERROLEID, CLIPID)
PRODUCER_FK1 (PERSONID)
PRODUCER_FK2 (CLIPID)
PRODUCER_FK3 (PRODUCERROLEID)
PRODUCER_PK (PERSONID, PRODUCERROLEID, CLIPID)
C##DB2024_ST03.WRITER
PF * PERSONID NUMBER (*,0)
PF * CLIPID NUMBER (*,0)
PF * WRITERROLEID NUMBER (*,0)
WRITER_PK (PERSONID, CLIPID, WRITERROLEID)
WRITER_FK1 (PERSONID)
WRITER_FK2 (CLIPID)
WRITER_FK3 (WRITERROLEID)
WRITER_PK (PERSONID, CLIPID, WRITERROLEID)
C##DB2024_ST03.SALARY
PF * CLIPID NUMBER (*,0)
P * SALARY VARCHAR2 (300 BYTE)
PF * PERSONID NUMBER (*,0)
SALARY_PK (CLIPID, SALARY, PERSONID)
SALARY_FK1 (PERSONID)
SALARY_FK2 (CLIPID)
SALARY_PK (CLIPID, SALARY, PERSONID)
C##DB2024_ST03.ACTORROLE
P * ACTORROLEID NUMBER (*,0)
CHARACTER VARCHAR2 (2727 BYTE)
ORDERSCREDIT CHAR (20 BYTE)
ADDINFOS VARCHAR2 (811 BYTE)
ACTORROLE_PK (ACTORROLEID)
ACTORROLE_PK (ACTORROLEID)
C##DB2024_ST03.DIRECTORROLE
P * DIRECTORROLEID NUMBER (*,0)
ROLE VARCHAR2 (90 BYTE)
ADDINFOS VARCHAR2 (314 BYTE)
DIRECTORROLE_PK (DIRECTORROLEID)
DIRECTORROLE_PK (DIRECTORROLEID)
C##DB2024_ST03.PRODUCERROLE
P * PRODUCERROLEID NUMBER (*,0)
ROLE VARCHAR2 (250 BYTE)
ADDINFOS VARCHAR2 (97 BYTE)
PRODUCERROLE_PK (PRODUCERROLEID)
PRODUCERROLE_PK (PRODUCERROLEID)
C##DB2024_ST03.WRITERROLE
P * WRITERROLEID NUMBER (*,0)
WORKTYPE VARCHAR2 (31 BYTE)
ROLE VARCHAR2 (56 BYTE)
ADDINFOS VARCHAR2 (332 BYTE)
WRITERROLE_PK (WRITERROLEID)
WRITERROLE_PK (WRITERROLEID)
C##DB2024_ST03.CLIPGENRE
PF * CLIPID NUMBER (*,0)
PF * GENREID NUMBER (*,0)
CLIPGENRE_PK (CLIPID, GENREID)
CLIPGENRE_FK1 (CLIPID)
CLIPGENRE_FK2 (GENREID)
CLIPGENRE_PK (CLIPID, GENREID)
C##DB2024_ST03.GENRE
P * GENREID NUMBER (*,0)
* GENRE VARCHAR2 (40 BYTE)
GENRE_PK (GENREID)
GENRE_PK (GENREID)
C##DB2024_ST03.CLIPRELEASEDATE
P * RELEASECOUNTRYINDEX NUMBER (*,0)
PF * CLIPID NUMBER (*,0)
F RELEASECOUNTRYID NUMBER (*,0)
DAY CHAR (10 BYTE)
MONTH CHAR (10 BYTE)
* YEAR NUMBER (*,0)
CLIPRELEASEDATE_PK (RELEASECOUNTRYINDEX, CLIPID)
CLIPRELEASEDATE_FK1 (CLIPID)
CLIPRELEASEDATE_FK2 (RELEASECOUNTRYID)
CLIPRELEASEDATE_PK (RELEASECOUNTRYINDEX, CLIPID)
C##DB2024_ST03.CLIP
P * CLIPID NUMBER (*,0)
CLIPTITLE VARCHAR2 (360 BYTE)
CLIPYEAR CHAR (20 BYTE)
CLIPTYPE VARCHAR2 (30 BYTE)
CLIP_PK (CLIPID)
CLIP_PK (CLIPID)
C##DB2024_ST03.CLIPLANGUAGE
PF * CLIPID NUMBER (*,0)
PF * LANGUAGEID NUMBER (*,0)
CLIPLANGUAGE_PK (CLIPID, LANGUAGEID)
CLIPLANGUAGE_FK1 (CLIPID)
CLIPLANGUAGE_FK2 (LANGUAGEID)
CLIPLANGUAGE_PK (CLIPID, LANGUAGEID)
C##DB2024_ST03.LANGUAGE
P * LANGUAGEID NUMBER (*,0)
* LANGUAGE VARCHAR2 (100 BYTE)
LANGUAGE_PK (LANGUAGEID)
LANGUAGE_PK (LANGUAGEID)
C##DB2024_ST03.COUNTRY
P * COUNTRYID NUMBER (*,0)
* COUNTRYNAME VARCHAR2 (100 BYTE)
COUNTRY_PK (COUNTRYID)
COUNTRY_PK (COUNTRYID)
C##DB2024_ST03.CLIPRUNNINGTIME
P * INDEXCOL NUMBER (*,0)
PF * CLIPID NUMBER (*,0)
PF * RELEASECOUNTRYID NUMBER (*,0)
RUNNINGTIMEMINUTES NUMBER (*,0)
RUNNINGTIMESECONDS NUMBER (*,0)
CLIPRUNNINGTIME_PK (CLIPID, RELEASECOUNTRYID, INDEXCOL)
CLIPRUNNINGTIME_FK1 (CLIPID)
CLIPRUNNINGTIME_FK2 (RELEASECOUNTRYID)
CLIPRUNNINGTIME_PK (CLIPID, RELEASECOUNTRYID, INDEXCOL)
C##DB2024_ST03.CLIPLINK
PF * CLIPFROM NUMBER (*,0)
PF * CLIPTO NUMBER (*,0)
P * LINKTYPE VARCHAR2 (29 BYTE)
CLIPLINK_PK (CLIPFROM, CLIPTO, LINKTYPE)
CLIPLINK_FK1 (CLIPFROM)
CLIPLINK_FK2 (CLIPTO)
CLIPLINK_PK (CLIPFROM, CLIPTO, LINKTYPE)
C##DB2024_ST03.CLIPRATING
PF * CLIPID NUMBER (*,0)
* VOTES NUMBER (*,0)
P * RANK CHAR (10 BYTE)
CLIPRATING_PK (CLIPID, RANK)
CLIPRATING_UK1 (CLIPID)
CLIPRATING_FK1 (CLIPID)
CLIPRATING_PK (CLIPID, RANK)
CLIPRATING_UK1 (CLIPID)
C##DB2024_ST03.CLIPCOUNTRY
PF * CLIPID NUMBER (*,0)
PF * COUNTRYID NUMBER (*,0)
CLIPCOUNTRY_PK (CLIPID, COUNTRYID)
CLIPCOUNTRY_FK1 (CLIPID)
CLIPCOUNTRY_FK2 (COUNTRYID)
CLIPCOUNTRY_PK (CLIPID, COUNTRYID)

Database Connection

For connecting, we will provide the username and password - do not forget to use EPFL VPN if not on EPFL network.

Every student has access with the same username and password.

hostname: cs322-db.epfl.ch, **port:** 1521, **SID:** ORCLCDB
username: C##DB2024_ST03
password: DB2024Lab1

NOTE: You cannot modify the provided tables.

Queries (100 points)

Since the data are already loaded, you can proceed with the queries (10 points per query).

Make sure to follow the query specification carefully with the attributes, order, and number of the results. Write each query in a separate .sql file. For each query, use the prefix Q, for example, Q_1.sql is the 1st query. Then, save these files in a folder named S[sciper], e.g. S359679. <u>You must follow this file naming convention, or else you will get penalized.</u>

1. Calculate the maximum number of clips any director has directed. Group by PERSONID from table DIRECTOR and print the PERSONID from table DIRECTOR of the director with the maximum number of clips that you calculated, and this maximum number of clips. Output should be in the format PERSONID, NUMBER_OF_CLIPS as the example bellow (this is not the correct answer):

```
 PERSONID                                   NUMBER_OF_CLIPS
---------------------------------------------------------------------------------------------------- ----------
  274566                                  231
```

2. Print the 10 most common clip languages. Order by LANGUAGE in a descending order. Output should be in the format LANGUAGE as the example bellow (this is not the correct answer):

```
LANGUAGE
----------------------------------------------------------------------------------------------------
Sweedish
.... (10 in total)
```

3. Find the FULLNAME from table ACTOR of the actor/actress who has acted in more clips than anyone else. Calculate the number of clips he/she has acted in. Group by FULLNAME and print the FULLNAME from table ACTOR and the calculation. Output should be in the format FULLNAME, YOUR_CALCULATION as the example bellow (this is not the correct answer):

```
FULLNAME                                                                   CALCULATION
---------------------------------------------------------------------------------------------------- ----------
Anna Lousera                                                               344
```

4. Print the CLIPTITLE from table CLIP, RUNNINGTIMEMINUTES from table CLIPRUNNINGTIME, and RUNNINGTIMESECONDS from table CLIPRUNNINGTIME, of the 10 clips with the longest duration (consider both the minutes and seconds of the clip) that were released in France. Use the RELEASECOUNTRYID from CLIPRUNNINGTIME table. Order by descending order according to duration. Output should be in the format CLIPTITLE,

RUNNINGTIMEMINUTES, RUNNINGTIMESECONDS as the example bellow (this is not the correct answer):

```
CLIPTITLE    RUNNINGTIMEMINUTES     RUNNINGTIMESECONDS
--------------------------------------------------------------------------------------------------------------------
Comics        2013                                    0
.... (10 in total)
```

5. Compute the number of clips released per country. Print the 10 first rows grouped by countryname from table COUNTRY. Printed output should include the COUNTRYNAME from table COUNTRY, and NUMBEROFCLIPS that you will calculate. Order by COUNTRYNAME from table COUNTRY in descending order. Output should be in the format COUNTRYNAME, NUMBEROFCLIPS as the example bellow (this is not the correct answer):

```
COUNTRYNAME                                                                          NUMBEROFCLIPS
---------------------------------------------------------------------------------------------------- -------------
Sweeden                                                                              213
.... (10 in total)
```

6. Compute the numbers of clips per genre released in the USA. Print the 10 first rows grouped by genre from table GENRE. Order by GENRE from table GENRE. Output should be in the format GENRE, YOUR_CALCULATION as the example bellow (this is not the correct answer):

```
GENRE                            CALCULATION
--------------------------------------- -------------
Horror                           33451
.... (10 in total)
```

7. Compute the average number of votes of an actor's clips (for each actor) when she/he has a leading role (actorroleid is strictly smaller than 4). Group by PERSONID, print the CLIPID and the average votes for the first 10 rows. Order by PERSONID in a descending order. Output should be in the format PERSONID, AVERAGE_VOTES as the example bellow (this is not the correct answer):

```
PERSONID AVERAGE_VOTES
---------- --------------------- --------------------- --------------------- --------------------- ------------
     120      45,8
 .... (10 in total)
```

8. Print the FULLNAME from table PERSON of screenplay story writers who have worked with more than 2 producers. Group by FULLNAME and print the first 10 rows. Order by FULLNAME in a descending order. There should be no duplicates in the output. Output should be in the format FULLNAME as the example bellow (this is not the correct answer):

```
FULLNAME
----------------------------------------------------------------------------------------------------
Mark Volgel
```

.... (10 in total)

9. Find the FULLNAME from table PERSON of the actors that are not married or have not been married in the past and have participated in more than 2 clips as "co-director". Calculate the number of films they have participated in as "co-director". Group by FULLNAME and print the fullname and the calculation for the first 10 rows. Order by FULLNAME in a descending order. Output should be in the format FULLNAME, YOUR_CALCULATION as the example bellow (this is not the correct answer):

```
FULLNAME                                                                                         CALCULATION
---------------------------------------------------------------------------------------------------- ----------
Maria Antoinette                                                                                  9
.... (10 in total)
```

10. Compute the average votes from table CLIPRATING for the clips whose genre is the most popular genre. Print the CLIPID and the average votes for the first 10 rows. Order by CLIPID in a descending order. Output should be in the format CLIPID, AVERAGE_VOTES as the example bellow (this is not the correct answer):

```
CLIPID    AVERAGE_VOTES
---------- --------------------- --------------------- --------------------- --------------------- ------------
      33         1023
.... (10 in total)
```

Submission

The deadline for the submission is on Monday 25 March at 11.00 am. Each student needs to submit on Moodle a folder named S[sciper], e.g. S359679, which contains 10 sql files, one for each query. For each file, use the prefix Q, for example, Q_1.sql is the 1st query. If you dont use the abovementioned submission format, you will get penalized. Additionally, make sure to follow the query specifications carefully with the attributes, order, and number of the output rows requested.

Frequently Asked Questions

Can I collaborate?

The lab assignment is individual. The SQL code you write should be your own and should not be copied from someone else, or the internet. We will run plagiarism checks once the deadline expires to catch any offenders. Any violation of the honesty policy will be met with strict action.

What if I submit the deliverable late?

As soon as the deadline passes, your submission is marked **LATE** and we will not evaluate it.

When can I ask questions about the lab 1?

The weekly lab session is the intended place for questions. Otherwise, please use the forum for questions that are of interest to your colleagues too.

GOOD LUCK!