Data Management for Big Data

January 13, 2020

2018–2019, 1st winter session
Teachers: Angelo Montanari, Dario Della Monica, and Paolo Gallo

Surname and name: _____

Student ID (matricola): _____ email: _____

Each part of the exam will contribute equally (one third) to determine the final score. Thus, total score of each part might be normalized so that you get at most 10 points from each part. Teachers can assign extra bonus points at their own discretion.

Part I: Fundamentals of database systems

Exercise 1:

Let us consider the following relational schema about actors and films:

$FILM(FilmCode, Title, Filmmaker, Year)$;

$ACTOR(ActorCode, Surname, Name, Sex, BirthDate, Nationality)$;

$INTERPREATATION(Film, Actor, Role)$

Let every film be univocally identified by a code and characterized by a title, a filmmaker, and the year when it has been released. For the sake of simplicity, let us assume each filmmaker to be univocally identified by his/her surname and each film to be directed by a single filmmaker. Let us consider the possibility that distinct films have the same title (this is the case, for instance, with remakes), but exclude the possibility for two films with the same title to be released the same year. Let every actor be univocally identified by a code and characterized by a name, a surname, a sex, a birth date, and a nationality. Let us assume that more than one actor may act in a film and that the same actor may act in more than one film. Finally, let us assume that, in a given film, each actor may play more than one role.

Define preliminarily primary keys, other candidate keys (if any), and foreign keys (if any). Then, formulate an SQL query to compute the following data (exploiting aggregate functions only if they are strictly necessary):

- the actors that acted in at least two films directed by the filmmaker Ken Loach, but not in any film directed by Quentin Tarantino.

Exercise 2:

Let us sinthesize the ER conceptual schema of a database of cinematographic data recording information about film makers and actors on the basis of the following set of requirements.

- Let us assume that both film makers and actors are univocally identified by their name and surname, and are characterized by their birth date and their nationality. The sets of actors and film makers are not necessarily disjoint. We would like to keep track of the relationships wife/husband and parent/child over the union of the set of actors and the set of film makers.
- Each film is characterized by the title, the year during which it has been released, the production country, the producer, the film maker, the actors, and the actresses. The set of films may include animated films, which have neither actors nor actresses.

  We assume that there is only one production country, one producer, and one film maker per film. We do not exclude the possibility that distinct films have the same title (this is the case, for instance, with remakes), but we exclude the possibility for two films with the same title to be released the same year. We would like to keep track of the countries (more than one in general), where a given film has been recorded.
- We would like to record the number of films produced every year by every country.

Build an ER schema that describes the above requirements, clearly explaining any assumption you made. In particular, for each entity, identify its possible keys, and carefully specify the constraints associated with each relation.

Part II: Advanced database models, languages, and systems

Instructions for multiple-choice questions.

- A right answer is worth +1.
- A wrong answer is worth -0.33.
- Short explanations should be **very** short (no more than 2 lines) and should be written in a neat handwriting. They are optional and used **at teacher's discretion**, mainly to increase the score of a wrong answer; seldom, to lower the score of a right one.

Instructions for open questions.

- The score assigned to an open question can range from 0 to 3 points.
- No negative score is assigned to open questions.
- Be careful: use a neat handwriting.
- Try to be succinct also for open questions.
- You can use additional sheets for open questions.

1. Let t_S = "time for one seek", t_T = "time for one-block transfer", and h be the height of a primary index (a tree) over attribute A of relation R. Which is the estimated cost for accessing all tuples where $A > X$? (Assume that A is a key and there are n tuples satisfying $A > X$, spread over b blocks.)

   ☐ $(h + n) * (t_T + t_S)$

   ☐ $h * (t_T + t_S) + t_S + b * t_T$

   ☐ $h * (t_T + t_S) + b * (t_S + t_T)$

   Short explanation (optional): _____

   _____

   Hint: recall that
   - *a primary index is defined over the attribute(s) used to physically order the file in the filesystem;*
   - *a secondary index is defined over any (subset) of the other attributes.*

2. Consider a relational algebra expression of the form

   $$\sigma_\tau(R_1) \bowtie R_2,$$

   i.e., where a *selection* occurs as sub-expression of a *natural join*. Which of the following is correct.

   ☐ The number of tuples of R_1 matching the condition τ of the *selection* does not affect the overall execution time of the whole expression.

   ☐ The number of tuples of R_1 matching the condition τ of the *selection* affects the overall execution time of the whole expression.

   ☐ Executing the *selection* before the *join* makes the execution faster.

   Short explanation (optional): _____

   _____

3. Distributed DB Systems are a necessity for several reasons. Name three such reasons (independently from their benefits and drawbacks).

4. Which of the following statements applies to the context of Distributed DB Systems?

   ☐ Information is distributed among the nodes and data replication is allowed for efficiency purposes

   ☐ Some of the nodes are devoted to information storage and other ones to query management

   ☐ Information is distributed among the nodes and no data replications is allowed to avoid data inconsistency issues

   Short explanation (optional):

5. Let R be a relation whose primary key is the attribute *key*, M be a set of minterms over attributes of R, and S be a relation such that there is a link L with *owner(L)* $= R$ and *member(L)* $= S$. In other word, one of the attributes of S is a foreign key referring to attribute *key* of relation R.

   Which is the horizontal fragmentation over relation R induced by M?

   ☐ $\{R_i \mid R_i = \sigma_{m_i}(R), m_i \in M\}$

   ☐ $\{R_i \mid R_i = \Pi_{m_i \cup \{key\}}(R), m_i \in M\}$

   ☐ $\{R_i \mid R_i = R \ltimes S_{m_i}, m_i \in M\}$

   Short explanation (optional):

6. Consider the 2 transactions T_1 (over operations $W_2(x), W_2(y)$) and T_2 (over operations $R_1(y), R_1(x), W_1(x)$) formalized through the 2 following partial orders, respectively:

   $$T_1 = \{W_2(x) \prec W_2(y)\}$$
   $$T_2 = \{R_1(x) \prec W_1(x), R_1(y) \prec W_1(x)\}.$$

   Is there a history over $\{T_1, T_2\}$ that is neither serializable nor serial? If yes, write down one such history. If not, write down a history that is both serializable and serial.

Is there a history over $\{T_1, T_2\}$ that is serializable but not serial? If yes, write down one such history. If not, write down a history that is both serializable and serial.

Hint: recall that serial histories are always concurrent transaction executions, that is, they are formalized through linear orders.

7. Explain the difference between checking that an XML document is well-formed and validating it.

Hint: which is the difference between an XML parser and an XML validator?

Part III: Data analysis and big data

1. With respect to the Data Warehouse context, briefly define the ETL process phase.

2. List two common operations that can be performed over an OLAP cube.

3. List the four main families of NoSQL databases.

4. Briefly explain the difference between *replication* and *sharding*.

Hint: With respect to MongoDB

5. Consider a time series from a single sensor which is transmitting a **pressure value every minute**. Using a document database you can store a single JSON document for each reading:

```
{
  pressID: "I8008",
  pressure: 1024,
  ts: ISODate("2018-11-10T22:56:00.00-0500")
 }
```

alternatively, you can store a JSON document containing a nested object ready to store one hour of observations and update it as soon as a new measure comes (**every minute**):

```
{
  pressID: "I8008",
  pressure: {0:1003, 1:1023, 2:1012, ..., 59:1002},
  ts: ISODate("2018-11-10T22:00:00.00-0500")
}
```

For each of the two models above write down:

- How many document writes will typically occur **in one hour**?
- How many documents updates will typically occur **in one hour**?
- Suppose that each document will include a 100 bytes long field for indexing purposes, what will be the space used only by those fields for **storing an entire day** of readings?

6. Give an example of a computation that can be performed using the MapReduce paradigm on data split across HDFS nodes (not the code, only the computation).

Hint: Some computations are suitable because they can be split ... and then the final result ...

7. Explain why dbRef(s) (also named references) across documents in MongoDB differ from foreign keys used by relational databases.

Hint: Relational model guarantees ... on tables by default ... while document DB ...