Data Management for Big Data

July 1, 2019

2018–2019, 2nd summer session
Teachers: Angelo Montanari, Dario Della Monica, 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 film to be directed by a single filmmaker and each filmmaker to be univocally identified by his/her surname. Let us consider the possibility that distinct films have the same title (this is the case, for instance, with remakes). 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. For the sake of simplicity, let us assume that, in a given film, each actor may play one role only.

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 most two films directed by the filmmaker Kurosawa.

Exercise 2:

Let us sinthesize the ER conceptual schema of a database for the management of information about the organization of travels on the basis of the following set of requirements.

- Each person has a fiscal code, that univocally identifies him/her. We would like to keep track of the name, the surname, and the birth date of each person. If a person is married with another one recorded in the database, we would like to record the spouse and the wedding date.
- Each city has a name and is located in a country. Each country is univocally identified by its name and it is characterized by its population, capital, currency, and national language. Each city is univocally identified by its name within the country it belongs to, that is, there are no different cities with the same name in the same country, but we cannot exclude the possibility of having different cities with the same name in different countries. For each city, we keep track of its number of inhabitants, area, and mayor.
- Every year, persons can form groups in order to travel together to a city. Each group is univocally identified by its name, and it is characterized by the number of its members. A person may be part of the same or a different group in different years, but may be part of at most one group in any given year.
- A group travels to the same or different city in different years, but it travels to exactly one city in any given year.

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 5 tuples satisfying $A > X$, all of them being stored in the same block.)

   □ $h * (t_T + t_S) + t_S + t_T$

   □ $h * (t_T + t_S) + t_S + 5 * t_T$

   □ $h * (t_T + t_S) + 5 * t_S + 5 * t_T$

   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. Select the correct statement.

   □ The Catalog stores results of queries that are executed frequently

   □ The Catalog stores indices to be used to optimize query executions

   □ The Catalog stores statistical information useful for cost estimations

   Short explanation (optional): ______________________________

   ____

3. 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*. Briefly explain why the number of tuples of R_1 matching the condition τ of the *selection* affects the overall execution time of the whole expression

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. Which of the following statements fits the notion of fragmentation in the context of Distributed DB Systems?

   □ Fragmentation is to be avoided because causes waste of space

   □ Fragmentation can be used to improve the efficiency of query execution

   □ Fragmentation happens when a node of the system remains isolated from the rest of the nodes due to network failure

   Short explanation (optional):

6. Consider the 2 following transactions:

$$T_1: \begin{array}{l} \textit{Read}_1(x) \\ \textit{Write}_1(x) \\ \textit{Read}_1(y) \end{array} \qquad\qquad T_2: \begin{array}{l} \textit{Read}_2(y) \\ \textit{Write}_2(y) \end{array}$$

formalized through the 2 following linear orders, respectively:

$$T_1 = \{R_1(x) \prec W_1(x) \prec R_1(y)\} \qquad\qquad T_2 = \{R_2(y) \prec W_2(y)\}.$$

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.

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

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 describe the main architecture components.
2. Briefly describe the three steps of text indexing.
3. Describe the most suitable context for the usage of columnar databases.
4. Briefly explain the difference between *native* and *non native* graph databases.
5. Consider a time series from a single sensor which is transmitting a speed value every second. Using a document database you can store a single JSON document for each reading:

   ```
   {
     pressID: "S8008",
     speed: 85,
     ts: ISODate("2018-11-10T22:56:25.00-0500")
    }
   ```

   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 second):

   ```
    {
       pressID: "S8008",
         speed:{
            0: {0:63, ..., 59:45},
            ...,
            59:{0:65, ..., 59:65}
           },
       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 hour of readings?
6. List the four elements composing a MapReduce job.
7. Briefly describe the *Ranged Sharding* in MongoDB.