MEIC 2018/2019

Data Analysis and Integration

1st semester

Exam – January 16, 2019

Part I. Virtual Data Integration and Data Cleaning

1. Given the following relational database:

   insurance_policy(policy_id, type, description)
   person(person_id, name, address, policy_id) *policy_id: FK(insurance_policy)*

   a) Write a query expression in Datalog that returns the names of people and the description of the insurance policies that they have.

   Q(N,D) :- person(_, N, _, P), insurance_policy(P, _, D).

   b) Give an example of another query expression over this database, whose results are contained in the results of the previous query.

   Same query as above, but with a bound variable or an additional predicate, e.g.:
   Q(N,D) :- person(_, N, _, P), insurance_policy(P, 'health', D).

   c) Why is it useful to write query expressions in Datalog rather than SQL? Justify, based on the kind of manipulations that can be done with query expressions.

   Datalog captures the essential logic of SQL queries as query expressions and allows such query expressions to be manipulated in a logic framework where it is possible to prove, for example, whether two SQL queries are equivalent or are contained in one another.

2. Answer the following questions:

   a) Suppose you have the query expression: Q1(X) :- Q2(X,Y), Q3(Y, 'abc'). Now suppose you want to answer to this query: Q4(X) :- Q1(X), Q5(X,Y), Y>2. What is the unfolded expression for Q4?

   Y in both queries must become different variables when unfolding, because they are independent:
   Q4(X) :- Q2(X,Y), Q3(Y, 'abc'), Q5(X,Z), Z>2.

   b) Suppose Q1 is a Global-as-View schema mapping, written as Q1(X) ⊇ Q2(X,Y), Q3(Y, 'abc'). What is the meaning of the symbol ⊇ in this expression? In practice, what does it mean?

   The symbol means that the left-hand side contains what is on the right-hand side, and it may contain more. In practice, this means that the global view Q1 contains the results of the query expression in the right-hand side, and may also contain the results of other query expressions / data sources.

3. The table *medical_examinations(name)* contains the names of all medical examinations available in a hospital. However, there may be slightly different names for the same medical examination. The idea is to use the Jaro measure to compute the similarity between names.

   a) For the two names below, indicate the values of $|x|$, $|y|$, c and t that should be placed in the formula. Show how you arrived at those values.

   amniocentesis
   aminocintese

$$|i-j| \leq \frac{min\{|x|,|y|\}}{2} \qquad jaro(x,y) = \frac{1}{3}\left(\frac{c}{|x|} + \frac{c}{|y|} + \frac{c-\frac{t}{2}}{c}\right)$$

```
1234567890123
AMNIOCENTESIS    |x|=13     |i-j|<=6
AMINOCINTESE     |y|=12
ccccccccccc       c =12
  tt  t    t      t =4
```

   b) If some names are very short and others are very long, do you think that the value returned by the edit distance represents by itself a measure of how close the two names are? Justify. (Note: consider the edit distance between "blood" and "urine"; and between "amniocentesis" and "colonoscopy").

In the presence of very short/long strings, the edit distance is not a good measure by itself, because for short strings the distance would always be small (suggesting high similarity) even if the strings to be compared are completely different, but for very long strings it could be large (suggesting low similarity) even if the strings are mostly similar. The length of the strings should also be taken into account.

   c) Indicate the sequence of steps (as in a PDI transformation) to detect approximate duplicates and return the pairs of names with Jaro measure above 0.8. Briefly explain what each step does.

1. **Table input** step to read the *medical_examinations(name)* table.
2. Two **Select values** steps, one renames *name* to *name1* and the other renames *name* to *name2*.
3. A **Join rows** step that has the two **Select values** as input (there should be a way to reduce the number of comparisons, but this would require the input data to have another column, such as *id*).
4. A **Calculator** step to compute the Jaro measure for *name1* and *name2* and put the result in a new column called *jaro*.
5. A **Filter rows** step with the condition *jaro* > 0.8.

4. The table *person(name, city, country)* may store records that refer to the same person.

   a) Suppose that there is a PDI transformation to detect approximate duplicates based on two attributes (e.g. *name* and *city*). Now, if someone tells you to use three attributes, what needs to be changed in the transformation? Indicate all the changes and identify the steps that need to be changed.

In the **Calculator** step, add a new metric for the additional attribute.
In the **Formula** step, insert a new term and weight for this new metric.

b) Suppose that table *person* contains millions of records, so that it becomes unfeasible to compare every record with every other record. In this case, which method could you use? Explain the method. (Note: the method does not have to be supported by PDI.)

I would use the *sorted neighborhood method*. This allows to compare records only within a sliding window. This method consists on: (i) the computation of a key for each record; (ii) sorting the records according to that key; and (iii) comparing only the records that are within a fixed size window.

➔ Note: answer the questions in the next page in a separate sheet of paper.

Part II. Data Warehousing and OLAP

For the questions below, consider a data warehouse for health insurance policies with three dimensions (dim_client, dim_policy, dim_time) and a fact table (fact_claim) that stores the amount claimed by a given client for a given policy at a given time. In each dimension, we want to store the following attributes:

dim_client: name, street, city, country (with hierarchy: name -> street -> city -> country)
dim_policy: type, description
dim_time: day, month, year (with hierarchy: day -> month -> year)

5. Answer the following questions:

a) In general, what is the difference between star, snowflake, starflake and constellation schemas? Explain these concepts.

The difference between star and snowflake schemas is that snowflake schemas have normalized dimensions. Starflake schemas are a mix between star and snowflake schemas, in the sense that some dimensions may be normalized, and others not. Constellation schemas have multiple fact tables.

b) Present the relational schema (star schema) for the data warehouse above using the following notation, where FK means foreign key:
table1(primary_key, attribute1, attribute2) *attribute2: FK(table2)*
Use surrogate keys in all dimension tables.

dim_client(client_id, name, street, city, country)
dim_policy(policy_id, type, description)
dim_time(time_id, date, day, month, year)
fact_claim(client_id, policy_id, time_id, amount)
client_id: FK(dim_client)
policy_id: FK(dim_policy)
time_id: FK(dim_time)

c) Write a single query in SQL/OLAP, using GROUPING SETS, ROLLUP or CUBE, that returns the union of all the following results:
- The total amount claimed per client.
- The total amount claimed per year.
- The total amount claimed per client and per year.

```
SELECT c.name, t.year, SUM(f.amount)
FROM fact_claim AS f, dim_client AS c, dim_time AS t
WHERE f.client_id = c.client_id
  AND f.time_id = t.time_id
GROUP BY GROUPING SETS ((c.name), (t.year), (c.name, t.year))
```

Also possible with c.client_id instead of c.name. In that case, no need to join with dim_client .

6. Answer the following questions:

   a) Suppose you have several insurance policies for a particular client in a text file. If you use PDI to read data from the file and load it into the dim_client table, which step(s) can be used to avoid inserting duplicate records (with the same name, street, city, country) in that table? Briefly explain how those steps work.

   Insert/Update, Dimension Lookup/Update, Combination Lookup/Update are all steps that check if a record with the specified attributes already exists in the table. If such record does not exist, it is inserted; if it exists, it is updated. In some cases, the update operation is actually an insert with a different version number (as in the Dimension Lookup/Update).

   b) Which aggregation functions would make sense to use for aggregating the claimed amounts through the dimensions dim_client and dim_time? Does it make sense to use SUM, MIN, MAX, AVG? Justify.

   The measure is additive across the client and time dimensions, so it makes sense to use SUM.
   MIN, MAX and AVG could also be used if we want to know the minimum, maximum and average amount by client, time, or both.

   c) Suppose that besides aggregating claims by total amount, we want to have another measure to count the number of claims by client, time, etc. From the software you used in the labs, which tool would you use to define this new measure and how would you define it using that tool? Explain.

   This can be done in Pentaho Schema Workbench. Under the tree structure that defines the OLAP cube, we would have to define a new measure based on the COUNT aggregation function.

7. Answer the following questions:

   a) Consider the dim_client dimension. What kind of hierarchy should be used if we want to make it possible to aggregate claims directly from street to country? Justify.

   The hierarchy should be ragged thus allowing to skip the city level.

   b) If one street can belong to two different cities, what kind of problem could occur when aggregating claims by client? How could you handle this problem? Explain.

When aggregating claims, the same claim could be counted twice because it appears in two cities. This is the double-counting problem. It could be solved by assigning percentages to how much the street belongs to each city, for example 50% to one city and 50% to the other. In this case, the claim amount would be divided equally between the two cities.

8. Answer the following questions:

   a) Suppose the data cube is called Claims. How would you write the following query in MDX?
   Show the amount of claims by client city, but only for the year 2018.

```
SELECT Measures.Amount ON COLUMNS,
       Client.City.Members ON ROWS
FROM Claims
WHERE Time.Year.[2018]
```

   b) If both OLAP tools (such as Saiku Analytics) and reporting tools (such as Pentaho Report Designer) can run MDX queries, what is the purpose of using these separate tools? Why not just use one of them? Justify.

OLAP tools such as Saiku are used for exploratory analysis where the MDX query is being changed as the analysis proceeds. Reporting tools such as PRD allow to generate pre-defined reports from queries which are not supposed to change (if the MDX query changes, then it is probably a new report).