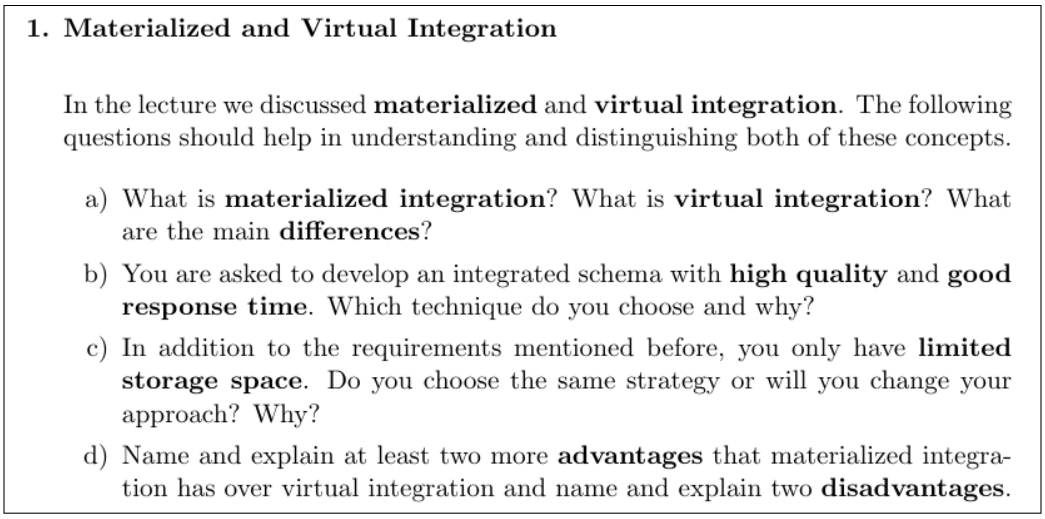
Information Integration – Exercise 3 – Gabriel Glaser

Task 1: Materialized and Virtual Integration



1. Materialized and Virtual Integration

In the lecture we discussed **materialized** and **virtual integration**. The following questions should help in understanding and distinguishing both of these concepts.

a) What is **materialized integration**? What is **virtual integration**? What are the main **differences**?

b) You are asked to develop an integrated schema with **high quality** and **good response time**. Which technique do you choose and why?

c) In addition to the requirements mentioned before, you only have **limited storage space**. Do you choose the same strategy or will you change your approach? Why?

d) Name and explain at least two more **advantages** that materialized integration has over virtual integration and name and explain two **disadvantages**.

a)

- *Materialized integration* means performing the integration before any user can query the integrated system. Therefore, all data (from the sources) has to be extracted at once and is stored using the (global) schema. Also, needs further techniques like periodic new data import and various data maintenance processes (e.g., deletion of old data).
- In contrast, *virtual integration* means performing the integration part as soon as a user queries the "integrated" system. Thus, only relevant (with respect to the query) data is retrieved from the sources (using the query language of each source). Also, the retrieved data (in source schema) is dropped as soon as the query handling is finished. Finally, needs further techniques like handling of source unavailability.

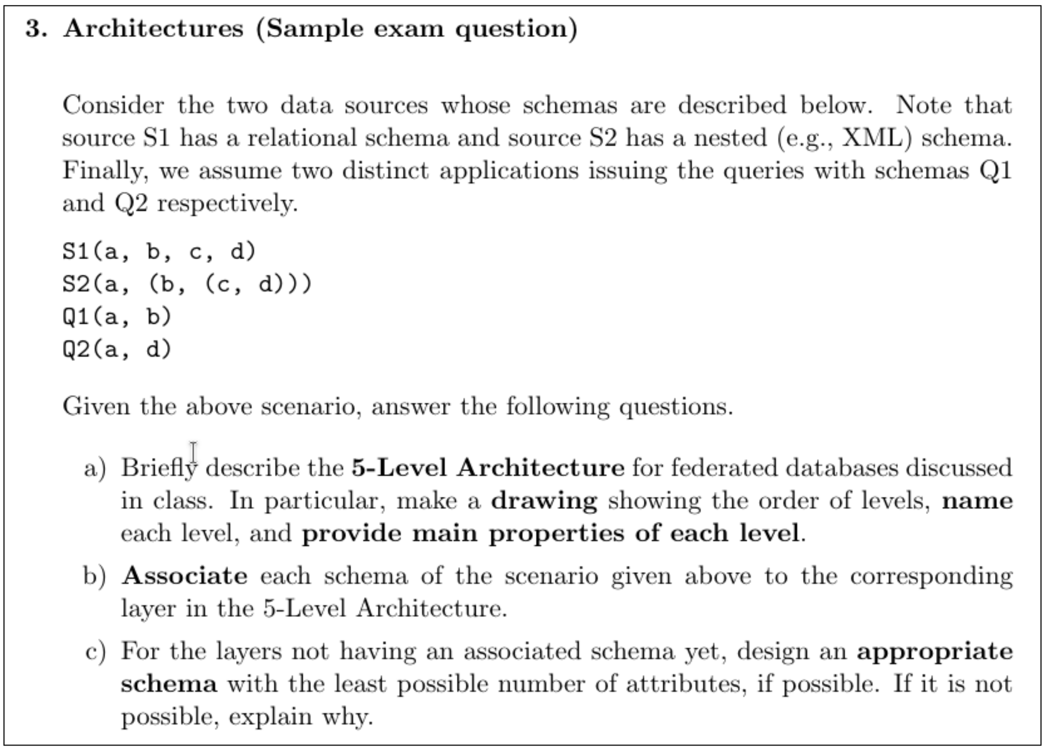
b) In this case, materialized integration is a good choice. On the one hand, it is capable of producing high quality results, because it is possible to apply many cleaning steps and consider many sources without a user waiting during all computations. On the other hand, it can be queried with low response times, because all computations have already been done before the system can be queried.

c) This requirement may change the choice, because using materialized integration consists of the step of copying all data of all sources into an own storage. If the needed storage is higher than the given storage space limit, virtual integration has to be used instead.

d)

- An advantage of materialized integration is that (if the building of the system has finished) it doesn't rely on the availability of the sources anymore. Also, it materialized integration is usually more complete, because extracting all data (relevant to a query) for each query is expensive.
- However, data of materialized integration may not be fresh and changes to the global schema requires transformations on the whole integrated data base.

Task 3: Architectures



3. Architectures (Sample exam question)

Consider the two data sources whose schemas are described below. Note that source S1 has a relational schema and source S2 has a nested (e.g., XML) schema. Finally, we assume two distinct applications issuing the queries with schemas Q1 and Q2 respectively.

```
S1(a, b, c, d)
S2(a, (b, (c, d)))
Q1(a, b)
Q2(a, d)
```

Given the above scenario, answer the following questions.

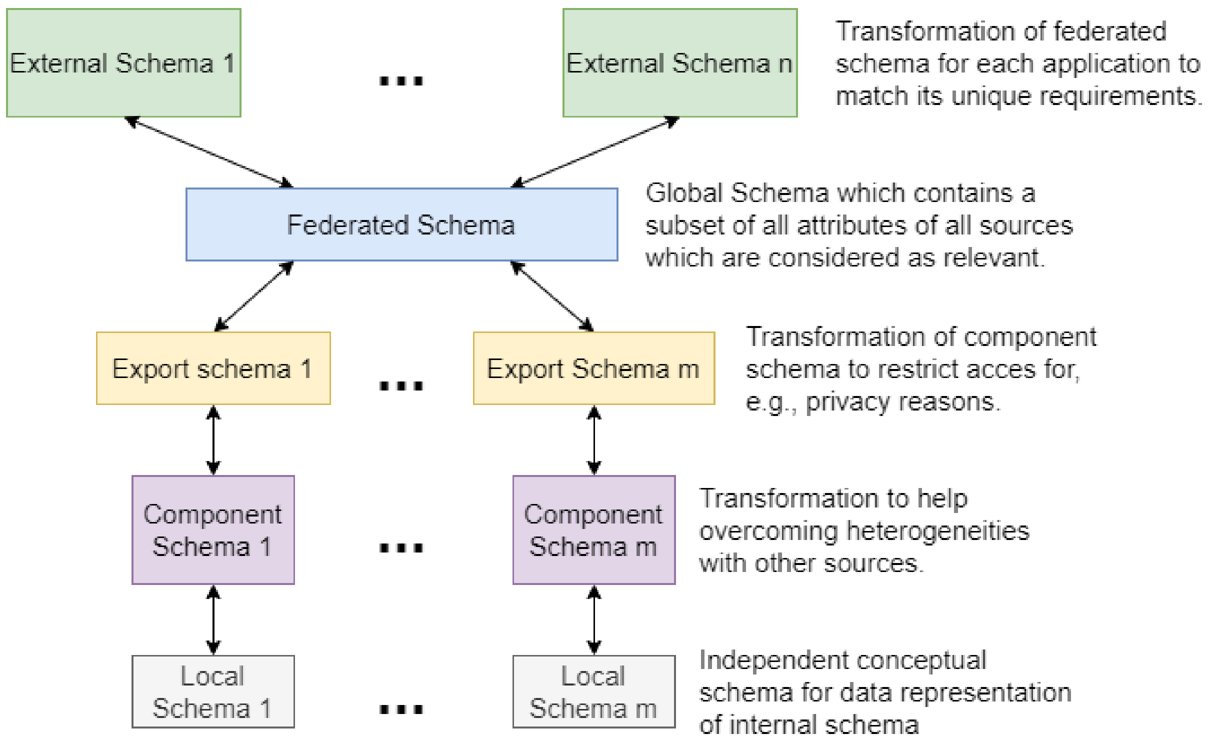
a) Briefly describe the **5-Level Architecture** for federated databases discussed in class. In particular, make a **drawing** showing the order of levels, **name** each level, and **provide main properties of each level**.

b) **Associate** each schema of the scenario given above to the corresponding layer in the 5-Level Architecture.

c) For the layers not having an associated schema yet, design an **appropriate schema** with the least possible number of attributes, if possible. If it is not possible, explain why.

a) The 5-Level architecture extends the 4-Level architecture by the *component schema* which is a transformation of the *local schema*. It is needed to help overcoming heterogeneities. The remaining parts of this architecture are...

- *external schemas* of federated schema to be able to offer different schemas to different applications.
- *federated schema* which consists of a subset of all attributes of all sources. (in the form of a different schema)
- *export schemas* to implement access restrictions or leave out unnecessary data. (for each source)

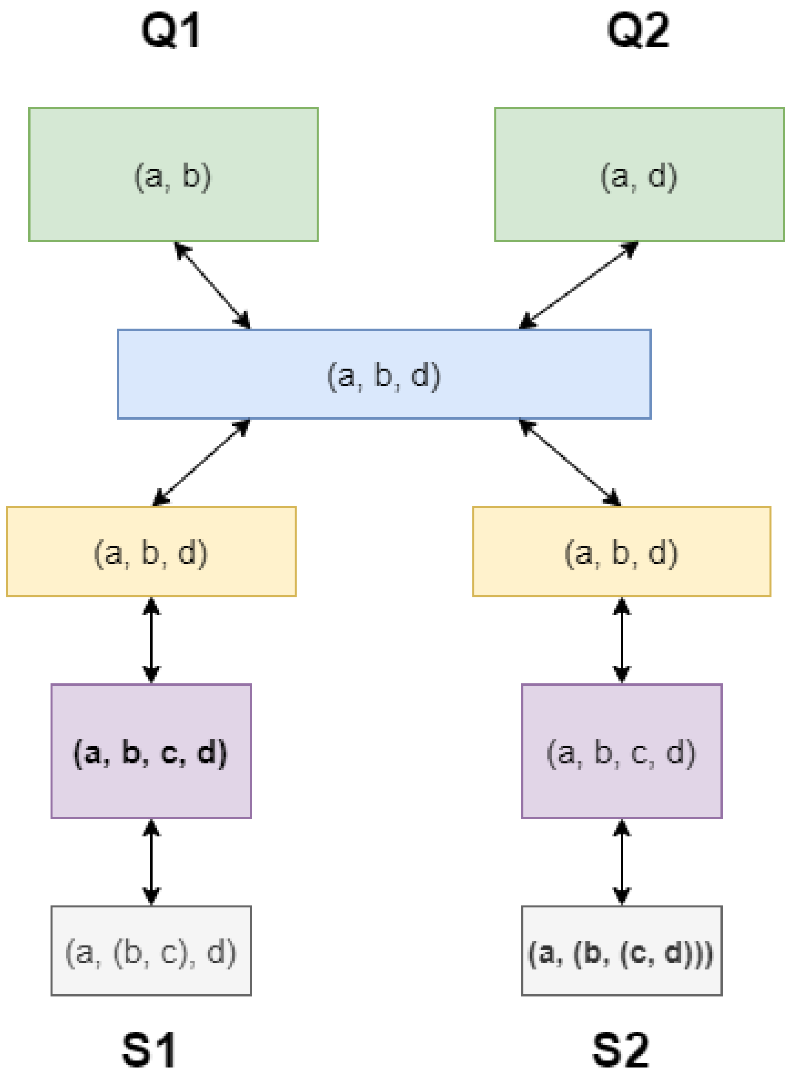

External Schema 1
...
External Schema n
Transformation of federated schema for each application to match its unique requirements.
Federated Schema
Global Schema which contains a subset of all attributes of all sources which are considered as relevant.
Export schema 1
...
Export Schema m
Transformation of component schema to restrict acces for, e.g., privacy reasons.
Component Schema 1
...
Component Schema m
Transformation to help overcoming heterogeneities with other sources.
Local Schema 1
...
Local Schema m
Independent conceptual schema for data representation of internal schema

b)

- *S1* = (a, b, c, d): This scheme likely is a **component schema**, because it is relational. Therefore, heterogeneities can be handled easier when considering the step of federation. (However, it could also be a local schema, in case a local source already used a relational schema or even an export schema, if it already contains access restrictions of a relational component schema)
- *S2* = (a, (b, (c, d))): This scheme likely is a **local schema**, because hierarchical schemas are (often) difficult to handle during the process of overcoming heterogeneities.

c) Schemas for remaining layers:

- *Local Schema* for *S1*: (a, (b, c), d).
- *Component Schema* for *S2*: (a, b, c, d)
- *Export Schema* for both *S1*, *S2*: (a, b, d)
- *Federated Schema*: (a, b, d)
- *External* Schema for *Q1*: (a, b); for *Q2*: (a, d)



No problems.