SACRAMENTO
STATE

CSc 177

Assignment 2

10/13/2013

Prof. Meiliu Lu

VIPIN BATCHU

Question 1) Multiple Choice Questions

Lecture 1 – Fundamentals of Data Warehouse

1) B
2) C
3) A
4) B
5) A
6) B
7) C
8) D
9) D
10) A

Lecture 2 – Data Warehouse Design

1) E
2) B
3) C
4) B

Lecture 3 – Building Enterprise Data Warehouse

1) C
2) A
3) D
4) E
5) B
6) C
7) B
8) D
9) C
10) A

Lecture 4 – Interviewing

1) A
2) B
3) A
4) F
5) E
6) B
7) E

Lecture 5 – Fundamentals of Aggregates

1) B
2) F
3) B
4) C
5) A
6) B
7) A
8) B
9) A
10) F

Lecture 6 – Designing Aggregates

1) A
2) D
3) B
4) B
5) A
6) B
7) A
8) D
9) B

10) A

Lecture 7 – Using Aggregates

1) E
2) C
3) C
4) A
5) C
6) B
7) B
8) D

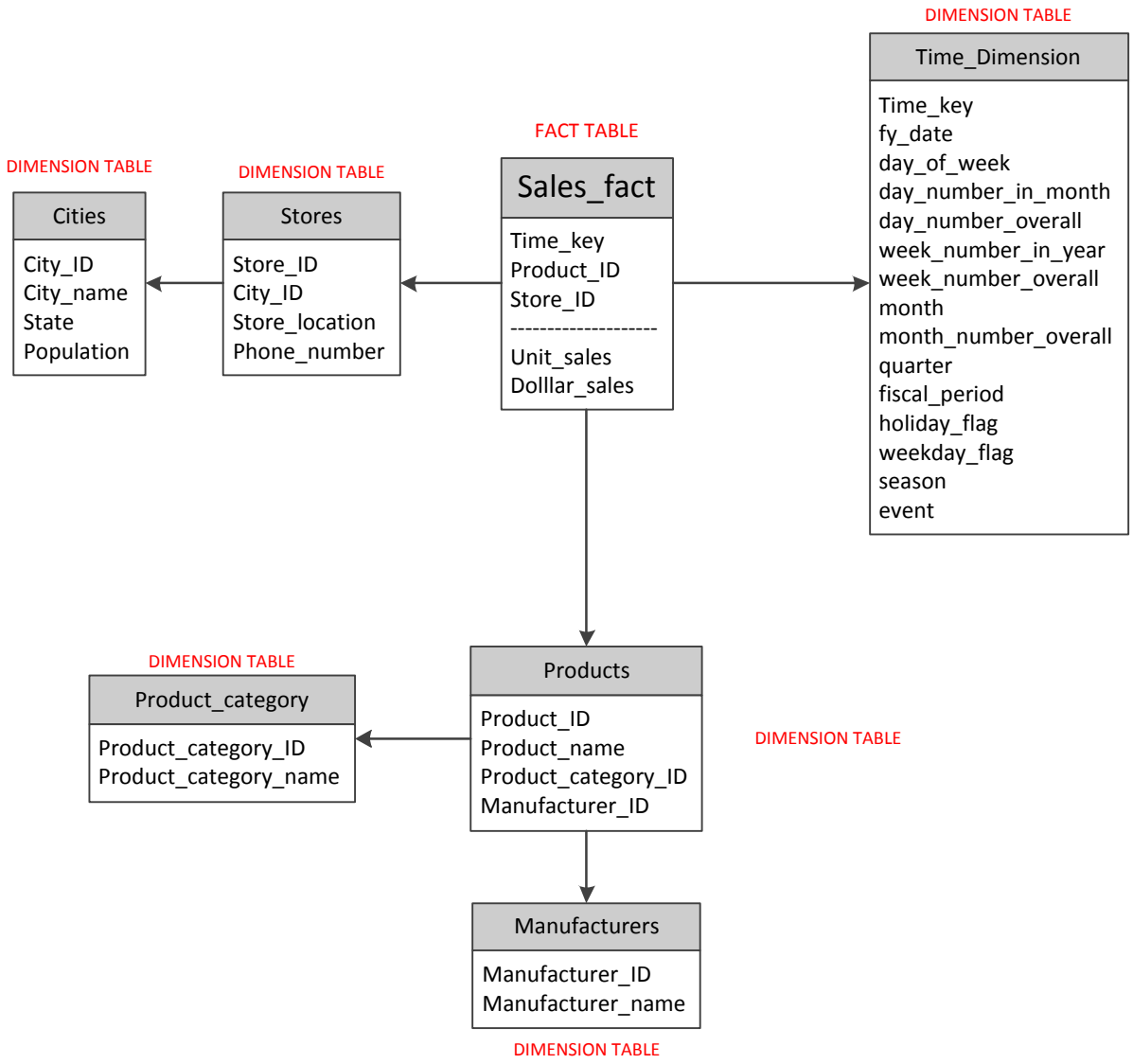
Lecture 8 – ETL

1) D
2) A
3) E
4) B
5) A
6) B
7) G
8) F

Question 2) Design Problems

(a) As a Reverse Engineering process, it is required to build the schema implemented in the data mart application given.

From the SQL text files, the fact table and also the corresponding dimension tables can be found out easily. The following is the Snowflake Schema design of the application.



(b)

Problem 4.3

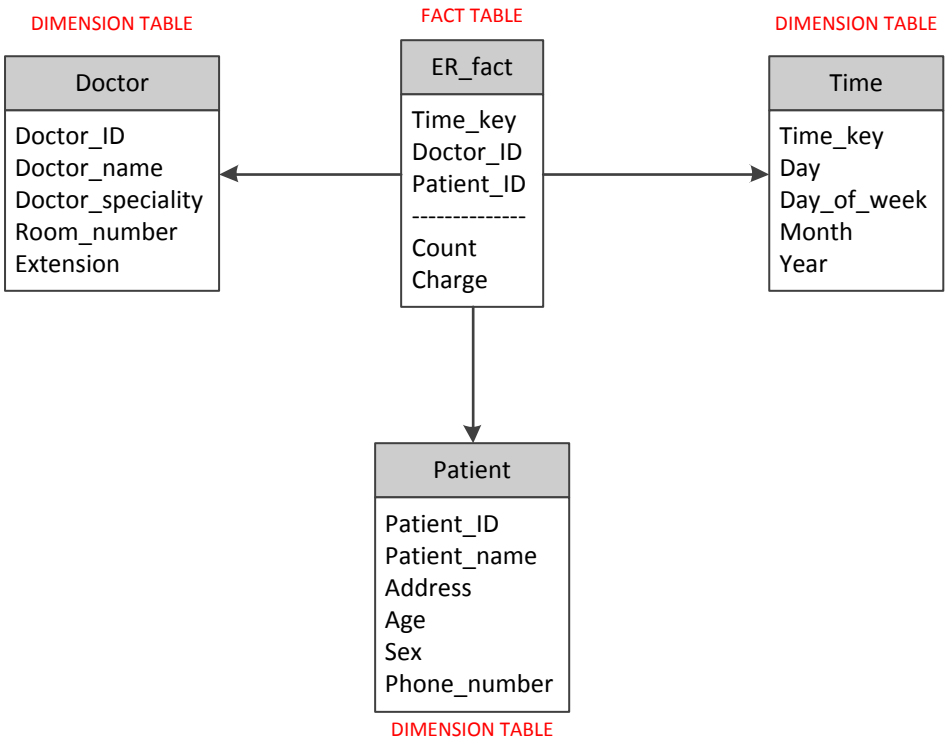
Suppose that a data warehouse consists of the three dimensions time, doctor, and patient, and the two measures count and charge, where charge is the fee that a doctor charges a patient for a visit.

i. Enumerate three classes of schemas that are popularly used for modeling data warehouses.

The most popular data model for a data warehouse is the multidimensional model and it can have 3 schemas

- **Star Schema:** The most common modeling paradigm is the Star schema, where the database has a large central table (Fact table) containing the bulk of data and several smaller attendant tables (Dimension tables), one for each dimension.
- **Snowflake Schema:** The snowflake schema is a variant of the star schema model, where some dimension tables are normalized, thereby further splitting the data into additional tables. The resulting schema graph forms a shape similar to a snowflake.
- **Fact Constellation:** Sophisticated applications may require multiple fact tables to share dimension tables. This kind of schema can be viewed as a collection of stars, and hence is called a galaxy schema or a fact constellation.

ii. Draw a schema diagram for the above data warehouse using one of the schema classes listed in (i)



iii. Starting with the base cuboid [day, doctor, patient], what specific OLAP operations should be performed in order to list the total fee collected by each doctor in 2010?

Consider a central data cube with the dimensions [day, doctor, patient]. Each entry in the cuboid has the fee collected by each doctor from each patient on each day.

➔ **Roll-Up**: First we roll-up the cuboid on '**Day**' dimension according to the concept hierarchy,

Day<Week<Month<Year

The resulting cuboid has the [Doctor, Patient, Year] dimensions.

➔ **Slice:** Now that we have the cuboid in place we slice it using the criterion **Year = 2010**. The resulting sub cube will now have the fee collected by each doctor from each patient in the year 2010.

➔ **Roll-Up:** Finally the cuboid is rolled up on the '**Patient**' dimension to all. The resulting cuboid will have the measures of total fee collected by each doctor in the year 2010.

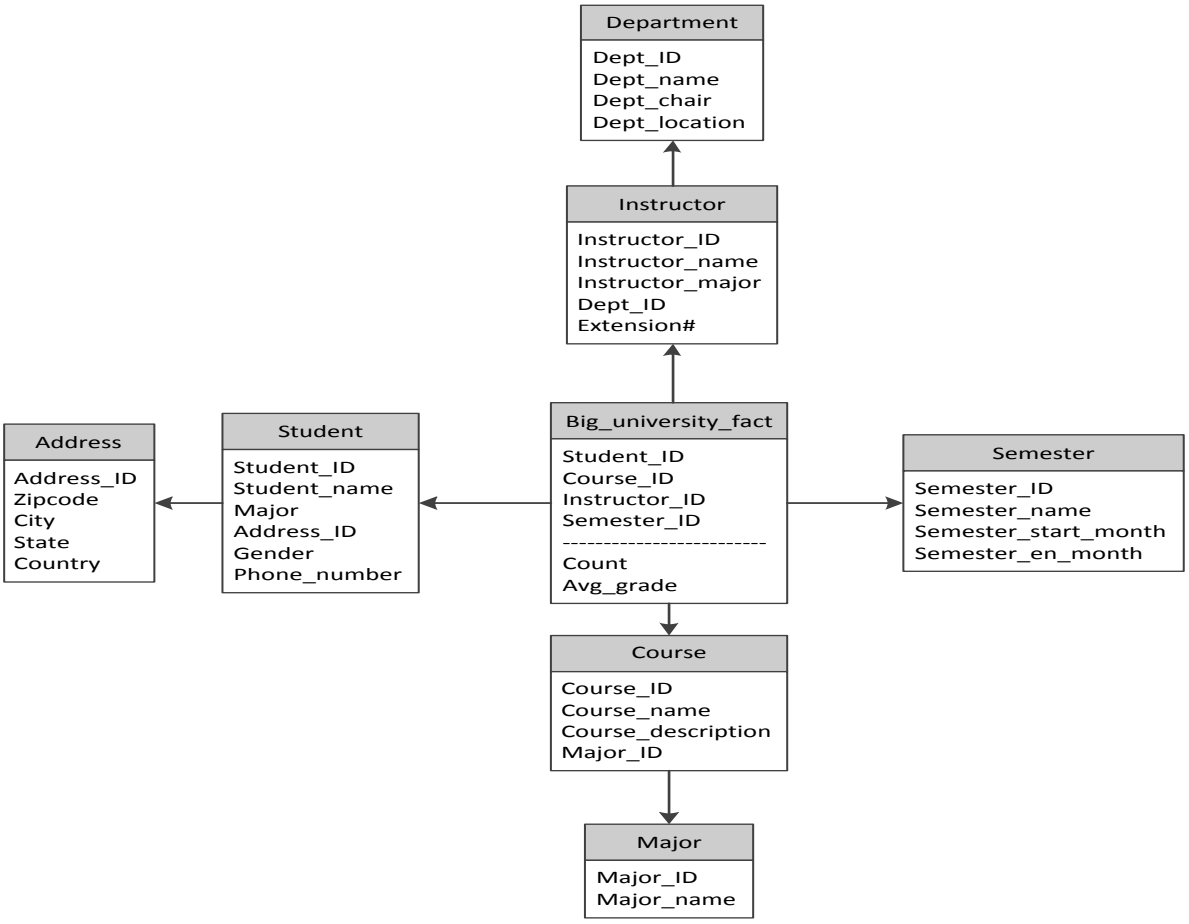
iv. To obtain the same list, write an SQL query assuming the data are stored in a relational database with the schema fee (day, month, year, doctor, hospital, patient, count, charge).

➔ Select doctor, SUM (charge) from fee where year=2010 group by doctor

Problem 4.4

Suppose that a data warehouse for Big_University consists of the four dimensions student, course, semester, and instructor, and two measures count and avg_grade. At the lowest conceptual level (e.g., for a given student, course, semester, and instructor combination), the avg_grade measure stores the actual course grade of the student. At higher conceptual levels, avg_grade stores the average grade for the given combination.

i. Draw a snowflake schema diagram for the data warehouse



ii. Starting with the base cuboid [student, course, semester, instructor], what specific OLAP operations (e.g., roll-up from semester to year) should you perform in order to list the average grade of CS courses for each Big_University student.

To solve such problems, the concept hierarchies have to be developed first before going any further. The following are the sample concept hierarchies for the given four dimensions

Student<major<status<university

Instructor<department<university

Semester<half-year (two semesters combined like summer and fall) <year

Course<concentration<major<branch (arts or science)

The specific OLAP operations to be performed are:

- ➔ Roll-up on course from Course_ID to major
  The resulting sub cube will have [major, semester, student, instructor]
- ➔ Roll-up on student from Student_ID to university.
  The resulting sub cube will have [major, semester, university, instructor]
- ➔ Dice on course, student with Major=”CS" and university = “Big University".
  The resulting sub cube will have [major, university]
- ➔ Drill-down on student from university to student name.
  This final sub cube will give the average grade of CS courses taken by each Big_University student.

iii. If each dimension has five levels (including all), such as “student <major <status <university <all”, how many cuboids will this cube contain (including the base and apex cuboids)?

Since there are 5 levels and 4 dimensions, it should accommodate,

5^4 = 625 cuboids.

Question 3) Helpful courseware

a) The courseware has given me an overall picture of the Data warehouse including all the important topics. It gave an elaborate and nice description on the topics with diagrams and clear examples, which were easier to read than what was given in the text. The students can assess their understanding of data warehousing via interactive quizzes provided at the end of each demonstration. Also, it provides an approach of how to design a data warehouse. This courseware would be very useful to users who have no knowledge about

- ➔ What is data warehouse?
- ➔ How it is implemented?
- ➔ What schemas are used?
- ➔ What is aggregation?
- ➔ How aggregate tables will help to speed up the execution time?
- ➔ What is an Enterprise Data Warehouse?

All these topics are discussed in simple language. The ETL courseware was very impressive as it was simple to use, user-friendly. ETL tools are very expensive, so this free tool has got abundant importance.

b) Possible Improvement: Some changes can be made to the courseware to cover more concepts related to aggregation. Just an overview of the concepts at a higher level is given. We could have considered more examples. Working principle of aggregate navigator can be explained in a better way by creating a web-application and explaining how it behaves. The aggregation of dimension tables and fact tables can be discussed individually.

Many of the links are broken and some tabs say a thing and do something else. Some of them lead to the same demo. The web interface needs a few tweaks here and there.

c) If possible, it would be helpful if the aggregation part of DW is explained in a much more clear and concise way. A lot of concepts can be put in to explain,

- The importance of the rule of conformance
- How the aggregate navigator works, with a flow-chart of a sequence diagram

Also, if the courseware had some information on the data mining and data pre-processing tasks, it would have been more helpful. Not the whole part, just highlighting the main procedures.

Question 4) Big Data without a Database

In the video, the speaker Kate Matsudaira speaks about creating Big data applications without the actually pinging the back-end database. She starts by giving the types of data that are present over the network.

The first type is the user data which deals with the dynamic data pertaining to a single user. This includes anything from user passwords, account information to items in the shopping cart. This data is relevantly slow changing.

The other type of data is the reference data. This data is returned by the search results. The different mapping of the results to the key words is pretty complex. Also the data set returned is pretty huge (search results for backpack on amazon). The speaker points the importance of speed and accuracy in these kinds of operations. For this to happen, all the data should be in memory. She then gave the idea of Caching all these data sets on to the service layer to get high performance.

The general problems that exist when dealing with large databases are availability, performance bottleneck and scalability. The availability problem can easily be taken care of creating replicas. But these induce operational overhead to keep all the copies consistent. The solution is, as mentioned earlier, move everything in to the memory. She mentioned about some caching tools like Oracle Coherence and ElastiCache. The data is stored in the form of key value pairs. They scale horizontally and dynamically balance the data. The drawback with these Big Data Cache tools is the operational overhead and cost.

The speaker then talks about the huge Δ associated with the use of remote data marts and access over the network. This point also directs the talk toward the cache-loading issue. She talked about the Redis technology which basically is an in-memory persistent database. This technology trades accuracy and the size of datasets that could be stored with higher access speeds.

The development model the speaker mentioned that will be useful for deploying web applications with large amount of reference data views the entire application as one single deployment cell.

The speaker talked about getting this done by keeping the data immutable, understanding the hierarchies and keeping them independent, optimize the numeric and string data by compressing and also optimizing the collection strategy. For example, the 'Trove' collection offered by Java can be used instead of regular HashMaps or TreeMaps.

The numeric data can be compressed using run length encoding which can drastically change the space required. The character data can be converted to ByteArray which occupies less space than compared to regular UTF-16 format. Then these ByteArrays can be compressed using GZip or BZip2.

Then she talked about the strategies to load the datasets into the cache. They include, compressing the data, keeping the format simple (CSV.JSON) and polling for updates. The data can also be swapped quickly, but this applies to infrequent data, she said. Some of the optimizations included periodically generating serialized data and validating them with CRC checks.

Overall, the speaker talked about the importance of memory caching of big datasets to provide data at faster rates. The video also focuses on the importance of scalability and performance issues when the data size keeps growing. The speaker mainly asks the developers to 'Know your Data more intimately' to enjoy the fruits of in-memory caching.