Data Warehousing Assignment

This problem set consists of two data modeling scenarios. You will be asked to analyze the strengths and weak- nesses of some design alternatives for each scenario. Short answers are fine – one or two paragraphs per question would be an appropriate length.

**Scenario I** In this scenario, we are interested in modeling student enrollment in Stanford courses. We would like to answer questions such as:

* Which courses are most popular? Which instructors are most popular?
* Which courses are most popular among graduate students? Undergraduates? • Are there courses for which the assigned classrooms is too large or too small?

We are planning to have a course enrollment fact table with the grain of one row per student per course enrollment. In other words, if a student enrolls in 5 courses there will be 5 rows for that student in the fact table. We will use the following dimensions: Course, Department, Student, Term, Classroom, and Instructor. There will be a single fact measurement column, EnrollmentCount. Its value will always be equal to 1.

We are considering several options for dealing with the Instructor dimension. Interesting attributes of instructors include FirstName, LastName, Title (e.g. Assistant Professor), Department, and TenuredFlag. The difficulty is that a few courses (less than 5%) have multiple instructors. Thus it appears we cannot include the Instructor dimension in the fact table because it doesn’t match the intended grain. Here are the options under consideration:

# OptionA

ModifytheInstructordimensionbyaddingspecialrowsrepresentinginstructorteams.Forexample,CS276ais taught by Manning and Raghavan, so there will be an Instructor row representing “Manning/Raghavan” (as well as separate rows for Manning and Raghavan, assuming that they sometimes teach courses as sole instructors). In this way, the Instructor dimension becomes true to the grain and we can include it in the fact table.

**Option B**

Change the grain of the fact table to be one row per student enrollment per course per instructor. For example, there will be two fact rows for each student enrolled in CS 276a, one that points to Manning as an instructor and one that points to Raghavan. However, each of the two rows will have a value of 0.5 in the EnrollmentCount field instead of a value of 1, in order to allow the fact to aggregate properly. (Enrollments are “allocated” equally among the multiple instructors.)

**Option C**

Create two fact tables. The first has the grain of one row per student enrollment per course and doesn’t include the Instructor dimension. The second has the grain of one row per student enrollment per course per instructor and includes the Instructor dimension (as well as all the other dimensions). Unlike Option B, the value of

EnrollmentCount will be 1 for all rows in the second fact. Tell warehouse users to use the second fact table for queries involving attributes of the instructor dimension and the first fact table for all other queries.

Please answer the following questions.

**Question 1. What are the strengths and weaknesses of each option?**

1. Using Option A will be difficult when doing aggregation or if we want to find all the courses taught by single instructors. Option A can be performed when we don’t have any fixed queries related to the instructor depending up on business.
2. Option B looks reasonable choice, since it is splitting the enrollment variable in half (i.e 0.5). this way when we fire a query we see that enrolment variable has 0.5 and we search for the other half and find another instructor.

But this might be problem when **we have multiple instructors for multiple course.**

1. Option c looks better, but it does not solve the problem, since we again keeping the enrollment variable as 1 and adding all the other dimensions and keeping the instructor variable same.

**Question 2. Which option would you choose and why?**

I would choose option B since it looks reasonable choice. But again as i mentioned this could lead us to problem if we have multiple courses taught by multiple instructors. To conclude, option B looks better for this scenario.

**Question 3. Would your answer to Question 2 be different if the majority of classes had multiple instructors? How about if only one or two classes had multiple instructors? (Explain your answer.)**

When we have multiple instructors (i.e 2 members) for a single course, option 2 would be better choice.

When we have multiple instructors for multiple courses, we need to consider the change of design (i.e adding new messument to fact as number of instructors.)

**Question 4. [OPTIONAL] Can you think of another reasonable alternative design besides Options A, B, and C? If so, what are the advantages and disadvantages of your alternative design?**

I would redesign the model by adding new measurement to the fact table as **number of instructors.**

Although, we will have 1 instructor for 1 course, but we are not sure for future courses.

The disadvantage of this design would be, when all the courses are taught by single instructor, there wont be any use of this measurement in the fact table and it occupies space doing nothing.

**Scenario II** In this scenario, we are building a data warehouse for an online brokerage company. The company makes money by charging commissions when customers buy and sell stocks. We are planning to have a Trades fact table with the grain of one row per stock trade. We will use the following dimensions: Date, Customer, Account, Security (i.e. which stock was traded), and TradeType.

The company’s data analysts have told us that they have developed two customer scoring techniques that are used extensively in their analyses.

* Each customer is placed into one of nine Customer Activity Segments based on their frequency of transactions, average transaction size, and recency of transactions.
* Each customer is assigned a Customer Profitability Score based on the profits earned as a result of that customer’s trades. The score can be either 1,2,3,4, or 5, with 5 being the most profitable.

These two scores are frequently used as filters or grouping attributes in queries. For example:

* How many trades were placed in July by customers in each customer activity segment?
* What was the total commission earned in each quarter of 2003 on trades of IBM stock by customers with a profitability score of 4 or 5?

There are a total of 100,000 customers, and scores are recalculated every three months. The activity level or profitability level of some customers changes over time, and users are very interested in understanding how and why this occurs.

We are considering several options for dealing with the customer scores:

# OptionA

The scores are attributes of the Customer dimension.When scores change, the old score is overwritten with the new score (Type 1 Slowly Changing Dimension).

**Option B**

The scores are attributes of the Customer dimension. When scores change, new Customer dimension rows are created using the updated scores (Type 2 Slowly Changing Dimension).

**Option C**

The scores are stored in a separate CustomerScores dimension which contains 45 rows, one for each combination of activity and profitability scores. The Trades fact table includes a foreign key to the CustomerScores dimension.

**Option D**

The scores are stored in a CustomerScores outrigger table which contains 45 rows. The Customer dimension includes a foreign key to the outrigger table (but the fact table does not). When scores change, the foreign key column in the Customer table is updated to point to the correct outrigger row.

Please answer the following questions.

**Question 5. What are the strengths and weaknesses of each option?**

1. Option A is not preferred in this business use case since SCD1 will replace the row values and we will loose historical data. It was also mentioned that there includes a data analysis team that work on historical data.

This could be better option when we don’t care much about historical data

1. Option B works fine since it uses SCD 2. We can keep track of historical data for analysis.

But this leads us to problem since this increase the table size. Using option B we might also consider the storage and performance and this also complicates ETL process.

1. Option C is not preferable in this business use case since it is not connected to customers dimention and this could lead us to problems to join three table when ever we need to find customer activity and score (i.e customer dimension, fact table, and customer score dimention ). This can be used when we don’t use relate scores with customer dimension
2. Option D works better when we have **very frequent changes** (i.e consider the example in currency exchange or bitcoin price rate exchange). But here in this business problem they mentioned that for **every three months scores changes,** so I don’t think this could be effective one

**Question 6. Which option would you choose and why?**

I would go with option B, since it captures historical data that is very important in performing analysis and bring profit to company. But we need to compromise for the storage and performance

**Question 7. Would your answer to Question 6 be different if the number of customers and/or the time interval between score recalculations was much larger or much smaller? (Explain your answer.)**

1. If the number of customers are increasing we should consider opting big storage or big data storage.
2. Even if the time interval between score recalculations was much larger, **I still choose the option B** since we can track the changes in the form of historical data and this also benefits for not worry about storage.
3. if the time interval between score recalculations was much smaller, **I’d choose to go with option D**

**Question 8. [OPTIONAL] Can you think of another reasonable alternative design besides Options A, B, C, and D? If so, what are the advantages and disadvantages of your alternative design?**

**I would choose option D (i.e.,** CustomerScores dimension is only connected to customers dimentions and we store the changes in the CustomerScores dimention) and redesign this model by also apply SCD2 (i.e., option B) to CustomerScores dimention since we can keep track of changes. This keeps CustomerScores out of customers dimension table and does not make mess in the customers table. By applying SCD2 on top of it allows use to track the . the only problem would be performance and storage is this design.