INFSCI 2711: Advanced Topics in

Database Management

Spring 2017

PROJECT: Data Warehousing Strategies

1. **Overview**

Our system is designed for online book retailer, by which customer can access our website to browse different kinds of books, purchase their favorite books and check their transaction records.

For sales person, they have the permission to handle MySQL operating database to create, delete and update database.

Meanwhile, for business manager, they desire to make prediction by querying from database, so we provide data analytic operations to them by building responding data warehouse, including MongoDB, Neo4j and MySQL date warehouse. In this way, they can efficiently gain the querying result.

1. **Assumptions**

In our system, we complete all the functions that listed in the form.

(1) System works, including front-end & back-end;

(2) The system allows to create a new customer, and store the information in the database;

(3) Managers can delete the customer and create, update and delete new stores;

(4) Customers can buy products and select the quantity;

(5) Customers can review the past orders;

(6) Managers can add, update and delete a product on the website;

(7) When buying a product, customers can see the storage amount and select the store and amount;

(8) When creating a new customer, it will appear error message when customers put the wrong type of phone number and email address;

1. A description of the data that will be maintained in your system
2. A description of STAR schema design. Specification of FACT table and tables for the Data  Warehouse dimensions (including corresponding DDL statements). The SQL statements to  populate the STAR schema.

Fact Table:

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Transaction | Customer | Product | Store | Time | Amount | Sale | Cost | discount |

This table means that one Customer buy one Product in one Store on one Time slot in one Transaction. So, Dimension tables includes Transaction, Customer, Product, Store and Time. In addition, Amount, Sale and Cost are three measure.

Dimension Table:

Product Table

|  |  |  |  |
| --- | --- | --- | --- |
| Id | Name | Category | Price |

Store Table

|  |  |
| --- | --- |
| Id | Region |

Time

|  |  |  |  |
| --- | --- | --- | --- |
| Year | Month | Day | Week |

Customer Table

|  |  |
| --- | --- |
| Id | Category |

1. Specification of pre-aggregated summary tables (including corresponding DDL statements). The  SQL statements for creation and populating of the summary tables. Specification of nightly  scheduled batch job to summarize data.
2. A description of Data Warehouse queries and front-ends required for the warehouse.
3. Some example scenarios of how various types of users will interact with the system.
4. A description of alternative implementation of your DW system using two NoSQL platforms.
5. **Comparison of Relational and NoSQL Database**

* MongoDB  
  **Disadvantages**:

MongoDB doesn’t support join operation. In relational database, Fact table could store primary key, like Id of every dimension table. So, when some queries need the join of Fact table and Dimension table, relation database could easily execute join operation through primary key. However, in MongoDB, users need to execute more than one query operation to implement join operations. For example, the user wants to know which regions sell most products. This is aggregation operation, which requires Fact table joining Store table by Store Id, and then grouping by store region. In mongodb, firstly users have to aggregate total sales of each store in Fact table and record the results. Secondly, users have to union these results and Store Dimension table to get the total sales of each regions. This is very painstaking.

As a result, in case of joining operation, MongoDB support nested documents, which means dimension tables could be added to the Fact table as documents. So, Fact table contains all of information about database. However, this will definitely introduce redundancy, especially when database is large enough.

**Advantages**:

MongoDB has obvious advantages, as it supports that one key contains more values.This is very applicable for building materialized views. As we know, one product can have different kinds of tags. In Mongodb, we can store one product as a key, and its tags as values. So one products can have many keys. For example, when we want to query the ratio of home customer to business customer, we actually build one materialized view name “Type”, which just including two key, “business” and “customer”. Each key could contain many customers, like the following,

Type materialized views:

|  |  |
| --- | --- |
| home | Cusotmer1, Cusotmer2, Cusotmer6 |
| business | Cusotmer3, Cusotmer4, Cusotmer5 |

So, if users want to get the ratio of home customer to business customer, they can easily get the list of home customers and business customers by querying the key.