BUS 464 Term Project

Scenario & Background

1. An ideal scenario in businesses is that an analyst is given access to data from different sources, in different formats. One or more of such sources may give un-normalized data or maybe data stored using a different data model. The analyst then has to reconcile the data (normalize it) and establish relationships among the different pieces (using a data model such as an ERD).
2. Using the model, the analyst then merges the various pieces of data (and discards which is not needed). The data is more or less normalized (ideal for "TPSs"). But it is often voluminous (in tera or petabytes). It is possible to run some SQL queries on this data, for reporting to middle-level managers (say at the store level). Some TPSs are run on fairly outdated and slow hardware such as an in-store checkout system at a local Safeway. Some TPSs may not even be accessible real-time by the regional servers. Some TPSs may even store barely a month of data – and historical data is archived on (sometimes, dormant) hard drives.
3. However, executives need summarization at the company or business unit level and for long periods into time (for trends). This can sometimes be done by writing SQL queries on data in Step 2 – but because of the volume of the data and its low accessibility, each ad-hoc query run by an executive can take a very long time. This is not useful because managers need answers in real-time. So the analyst then interviews managers to understand approximately how they need the data summarized.
4. Next, using SQL, she then collects and converts the merged data in Step 2 into a data-mart (**Analytical DB**) that can answer the more targeted business questions very quickly. The SQL also updates this Analytical DB with fresh data regularly (depending on executive needs) with summarized data that can be queried by managers in real time.

Figure 1 shows the real-world scenario (all blue boxes) and the solid arrows. The steps are as follows:

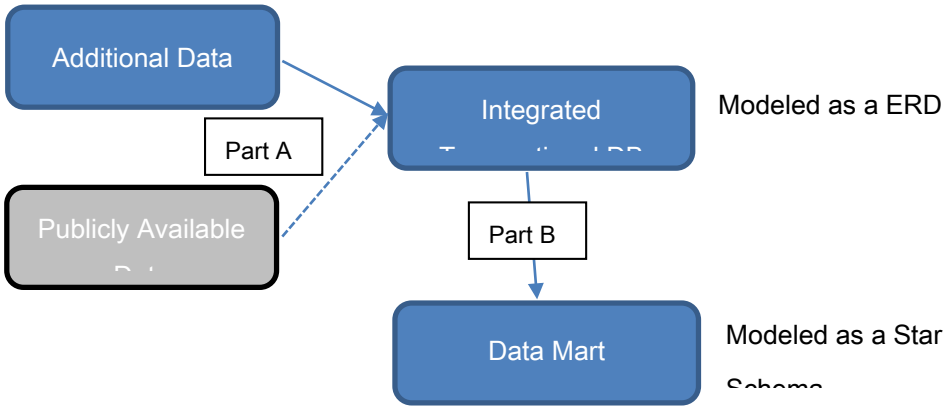
1. After examining the data provided (the input data), the students will develop an entity relationship diagram (ERD) to represent a typical business context. To the input data, students

will also be expected to some add additional data (partially, randomly generated, or real data from other sources or using http://dummydata.me/) – this will make the data model and the resulting transaction database in step 2 below, more interesting from a business point of view.

2. Using all the input data, they will develop SQL queries to create a database (called Transactional DB) and populate it with the input data. The process resembles the Extract, Transform and Load framework where SQL queries (Data Definition commands) are first created to create the database conforming to a normalized ERD.

3. Next, Data Manipulation Language (DML) commands of SQL are used to transform and load the data into a target Transactional DB. This is a very easy step if you are using WB or such tool to convert an ERD into a relational schema.

4. The final stage is conceptually transforming the Transactional DB into an Analytical DB (a Data Mart) using a Star Schema. The star schema is designed specifically to answer strategic level business questions you will be introduced to in the latter half of the semester. Star Schemas are created to condense large quantities of such data by aggregation specifically to get desired answers much faster.



BUS 464 Term Project

Initial raw data will be made available by instructor in class. It will pertain to a specific Client and will be referred to as Client DB.

Part A (10 points)

1. Create an ERD. Model the data underlying the Client DB dataset using your own understanding of the business context. Refer to the pdf file to help you understand the definitions of the various columns.

The rules and conventions discussed in class and class notes/readings must be followed. For example, the ERD must be clearly labeled, and should not contain any M:N relationships or multi-valued attributes. Indicate primary and foreign keys in the ERD. 1:1 relationships can be included but only with proper justification as to its purpose. Every syntax mistake will cost the team 1 point. If an obvious entity is absent in your solution it will cost 2 points. Use MySQL Workbench for drawing the ERD[1].

You need to extend your data model (ERD) by adding additional elements (Tables/Attributes/Relationships)[2] so that merging it with the data given to you will make it more valuable for Part B. Example: Customer names or household income will make the database more valuable[3].

Submit: i) ER Diagram of the transactional DB (using WB). ii) English description of the relationships between various entities in the ERD + assumptions (PDF, 2 pages maximum).

2. Convert a slimmer version of the ERD into an actual database and define the metadata. The trimmed ERD is to be conceptualized from the original one in step 1 above by keeping in

---

[1] Technically, the ERD is a conceptual diagram and what you can depict graphically in MySQL workbench as a relational model. For most practical purposes there is a one-to-one correspondence between the two. Therefore I am referring to them interchangeably as 'ERD' or the 'Relational Diagram'.

[2] As you may know, in an ERD we have Entities/Relationships/Attributes and these correspond to Tables/Constraints/Fields or Columns in the relational DB that matches the ERD.

[3] Even if you do not have customer ratings, you can generate this data randomly using SQL functions.

mind the requirements for Part B. Use MySQL Workbench for creating tables and setting appropriate referential integrity constraints (mostly, primary and foreign keys to enable linking tables).

Submit the MySQL code.

3. Write SQL to populate the transactional DB using the Client DB dataset + possible randomly generated data (or external spatial data from sources like Simply Map at the SFU library). SQL to extract, transform and load the data from the raw tables into the transactional DB. You can choose to organize your SQL nicely so that it is understandable to you and me. For example, you can create three subsections:

   i. **EXTRACT**: the data extraction queries that extracts data from the raw datasets and puts them in a staging area
   ii. **TRANSFORM**: the data transformation queries that transform (if needed) data in the staging area itself
   iii. **LOAD**: queries to load the new transactional DB.

New data from other sources should be uploaded – and any data that is to be randomly generated or added to existing should be done in MySQL (using SQL).

We will call step 3 as the ETL step (which populates the transactional DB)**.** Submit this SQL code for the ETL.

To grade Part A, I will simply copy/paste all your SQL code into my WB and execute it.

BUS 464 Term Project

Part B (15 points)

1. Identify three data verification and three data discovery questions for a datawarehouse based on this data. Explain giving more context to these questions in terms of how/when managers will need answers to these questions.

2. Develop a Star Schema (to be discussed after midterm) with the objective of answering the above sets of questions. When the Star Schema is converted to a "Data Mart", some of the tables will be different from those based in the Transactional DB. Implementing the Star Schema again involves writing SQL queries to transfer data from the Transactional Database to the Data Mart.

Submit:

4. **One page explanation for B.1 (3 points)**
5. **Star schema using WB (5 points)**
6. **SQL code to create the datawarehouse from the transactional DB in part A. Similar to Part A you can organize your SQL code as EXTRACT, TRANSFORM and LOAD segments. (However, you do NOT need to write SQL code to extract data *from* the star schema to answer the verification/discovery questions.)**
   (7 points)

Submitting data (if any)

Write very clearly how I can access the additional raw data you may have used and which I did not provide you. You do not need to submit the intermediate data you create since I will use your SQL queries to do so myself.

BUS 464 Term Project

FAQs

1. What is a practical approach to starting the ETL process?

First extract only a small piece of raw data, say 1000 records (using the LIMIT clause). While you may have an elaborate ERD, trim it down by eliminating all attributes but keeping the PKs and FKs, and few other important attributes such as Sales, Rooms, QtyOnHand (for example, depending on the context). Then, perform the ETL process on this small piece of data all the way through till you have a Transactional DB. Once the SQL queries are working, now involve all of the data by removing the LIMIT clause – and work the entire set of attributes in the ERD.

Working with a few thousand records for a start also lowers the processing load on the MySQL server and will speed up your query creation. You can try and run the queries on the full database at nights perhaps.

2. How do we develop the queries to perform the ETL process?

This is the responsibility of each group. The basic query commands are mentioned in the previous page. The groups are free to explore various sources including the textbook that will illustrate how SQL can be used to manipulate the data in various ways. A good source is the MySQL Essential Training on Lynda.com and Youtube. The instructor is available out of class to help with these commands.

3. What kind of help does MySQL Workbench offer in this process?

A useful help feature in MySQL is the menu that appears if you right click on a Table or Column in the left panel. Executing the desired command from the menu will lead you to an SQL query that executes the command. YOU NEED to store this query (ALTER or INSERT so that it can be included at the exact location in the submission as explained in the previous page).

4. What is a typical query to create random integer values?

BUS 464 Term Project

To create random integer between 1 and 2, run this command:

select floor(1+rand()*2);

To create several columns with random integers between 1 and 2 use:
select floor(1+rand()*2) as A, floor(1+rand()*2) as B, floor(1+rand()*2) as C, floor(1+rand()*2) as D

The code below can be used to populate a table with random values. Take a table, say Employees.Employees which already has 300K rows.

```
use db_nsaraf;

create table t1 as
select emp_no, floor(1+rand()*2) as A from employees.employees;

#check the distribution of random values

select count(*) from t1; ## 300K records;
select count(*) from t1 where a=1; #149K records;
select count(*) from t1 where a=2; #150K records;

#The second column of t1 has a random integer between 1 (floor) and 2 (ceiling).

#You can use UNION to create table with 300K X 2 values by appending a table with itself repeatedly.
```

5. What can be done to speed up querying considering the datasets can be large?

Explore the usage of INDEXES in database querying. Your MySQL account should have the privilege to create indexes that can speed up querying. The textbook has an explanation of indexes as well as the Internet resources. Use CREATE INDEX to create indexes for attributes which are typically used in WHERE statements.

6. What aspects of a team's submission can influence my subjective assessment of the work?

A few of them are:

BUS 464 Term Project

- Are all pieces of information (columns/fields) in the original dataset modeled in the ERD as attributes, and also populated in the transactional database?
- Have you used SQL as much as possible to do all data manipulations except creating and importing randomly generated or external data? E.g., if a SalesID and SalesDepartmentName is already in the original data given to you, then I would not like to see any Excel or TXT file that reformats this same piece of information and imports it again into the server.
- Are all FKs and PKs are properly set? Are they all of INT type?
- Housekeeping: Are all temporary tables deleted? Are all tables named appropriately?