Project #2 Part 1

CSC 261/461 (Database Systems), Spring 2018,
University of Rochester
Due Date: 02/19/2018 (11:59 pm)
This is NOT a group project

Introduction

We provide you with a fairly large volume of data downloaded (over a decade ago!) from the eBay website stored as XML files. XML format is primarily used to store semi-structured data. We also provide you a parser, `parser.py`, which converts these xml files into four structured `.dat` files for loading into MySQL database.

Your task is to examine the `.xml` and `.dat` files, and provide an ER model and a schema for the database. Then, load the transformed data (.dat files) into MySQL database and test it by running some SQL queries over it.

Task A: Examine and transform the XML data files

We are providing an XML auction data set for you to use in your project. The data files are located in the directory `/home/tbiswas2/proj2/data/`. You can access this folder from `betaweb.csug.rochester.edu`. This AuctionBase database is consist of 40 files: `items-0.xml, items-1.xml, ..., items-39.xml`. You do not need to copy these files. But you should open a few of these files and look into the pattern or structure to familiarize yourself with the schema of the dataset.

There are a total of about 20,000 auctions. Note that the **ItemID** attribute is unique and involved in only one auction, while the **Name** attribute is not unique. One of the important aspects to understand about the data is that it represents a single point in time. (Specifically, it represents the following point in time: December 20th, 2001, 00:00:01.) It contains items that have been auctioned off in the past and items that are currently up for auction.

Here is a sample of the data:

```
<Item ItemID="1043402767">
<Name>Precious Moments Girl Stove Mini Tea Set</Name>
<Category>Collectibles</Category>
<Category>Decorative &amp;amp; Holiday</Category>
<Category>Decorative by Brand</Category>
<Category>Enesco</Category>
<Category>Precious Moments</Category>
<Currently>$4.00</Currently>
<First_Bid>$4.00</First_Bid>
<Number_of_Bids>1</Number_of_Bids>
<Bids>
<Bid>
<Bidder UserID="goldcoastvideo" Rating="2919">
<Location>Los Angeles,CA</Location>
<Country>USA</Country>
</Bidder>
<Time>Dec-06-01 06:44:54</Time>
<Amount>$4.00</Amount>
</Bid>
</Bids>
<Location>MIlwaukee Wi</Location>
```

```
        <Country>USA</Country>
        <Started>Dec-03-01 18:44:54</Started>
        <Ends>Dec-13-01 18:44:54</Ends>
        <Seller UserID="fallsantiques" Rating="952"/>
        <Description>PRECIOUS MOMENTS GIRL/STOVE MINI TEA SET:</Description>
        </Item>
```

You need to convert this unstructured data into a structure for loading into MySQL database. We are giving you the utility `parser.py` for that.

You can execute the following commands after you log in to Betaweb machine to copy `parser.py` and run it to get the structured .dat files.

```
mkdir proj2
cd proj2
cp /home/tbiswas2/proj2/parser.py .
python parser.py /home/tbiswas2/proj2/data/*.xml
```

This parsing would take some time and finally produce four `.dat` files. After the parsing, your directory should have four .dat files. `item.dat, user.dat, category.dat, and bid.dat`

Open these files to view their contents. Each of these files contains multiple fields separated by <>.

- `item.dat` contains nine fields (itemID, name, currently, buy_price, first_bid, started, ends, userID, and description)
- `user.dat` stores four fields (userID, rating, location, and country)
- `category.dat` stores only two fields (itemID, and category).
- `bid.dat` contains four fields (itemID, userID, time, and amount).

Task B: Design your relational schema

The four files that `parser.py` provides will eventually be loaded as four relations. Just to provide a basic understanding of this database to non-technical people, you need to provide an ER model of the data.

Your ER model should have a number of entities. Draw their attributes and how these entities are related to each other. Provide the cardinality and participation constraint (total vs partial), and (min, max) constraint. Your ER model should be in Chen Notation.

Provide your relational schema definitions in text form, including the attributes and their data types for each relation. Make sure to indicate your chosen keys (including primary and foreign).

Create a pdf file containing the ER model and the database schema and save it as `proj2p1report.pdf`.

While designing your relational schema, you may realize that two fields from the XML files are technically not necessary to have in your schema for it to represent the same information: Currently and Number of Bids. For example, the Number of Bids can be obtained by simply running a query to calculate the number of bids on a particular item. However, for the purposes of this project, ignore Number of bids and keep only 'currently' in your schema. (`user.dat` also ignores numberOfBids). For websites with large databases and many users, running a query to calculate an aggregation every time it needs to be viewed can be costly, so storing the aggregated result itself can help improve performance.

Task C: Load your data into MySQL

The next step is to create and populate your AuctionBase database. MySQL provides a facility for reading a set of commands from a file. You should use this facility for (re)building your database and running sample queries, and you must use it extensively in your submitted work.

Create a command file called `create.sql` that includes the SQL commands that create all of the necessary tables according to your schema design from Task B. Before creating the tables, you should also ensure that any

old tables of the same name have been deleted. (This makes it easier to test your submission.) This file will look something like:

```
drop table if exists User;
drop table if exists Item;
...
create table User ( ... );
create table Item ( .... );
...
```

Then, create a command file called `load.sql` that loads your data into your tables. This file will look something like:

```
...
LOAD DATA LOCAL INFILE "user.dat"
INTO TABLE User
FIELDS TERMINATED BY "<>";
...
```

If you are unsure what these commands do, refer to `https://dev.mysql.com/doc/refman/5.7/en/load-data.html`

It is possible to execute SQL statements from a text File. Refer to `https://dev.mysql.com/doc/refman/5.7/en/mysql-batch-commands.html` for details. In a nutshell, If you are already running MySQL, you can execute an SQL script file using the source command. For example,

```
mysql> source create.sql
```

Execute sql statements from `create.sql` and `load.sql` to create and populate these tables.

Task D: Test your MySQL database

The final step is to take your newly loaded database for a test drive by running a few SQL queries over it. As with database creation, first test your queries interactively using the MySQL command-line client, then store these queries in a file to execute them using source command. First, try some simple queries over one relation, then more complex queries involving joins and aggregation. Make sure the results look correct. When you are confident that everything is correct, write SQL queries for the following specific tasks:

1. Count the number of Items from category "Enesco"
2. Count the number of Items (same as auctions) belonging to exactly four categories.
3. Find the ID(s) of auction(s) with the maximum buy_price.
4. Count the number of sellers whose rating is higher than 1000.
5. Count the number of users who are both sellers and bidders.
6. Count the number of categories that include at least one item with a bid of more than $1000.
7. Count the number of pair of bidders, who bid for the same item.
8. Count the number of sellers who sell items in more than 10 categories

Your answers to the above eight queries over the large data set should be (in order):
`349, 8365, 1677348181, 3130, 6717, 4, 24668, and 234.`

Your queries should be fairly efficient – they should each take at most ten seconds to execute. If any of your queries is taking much longer than that, you've probably written them in an unnatural way; please try rewriting them, and see the course staff if you need any help. Also, this is a good time to attend the workshops and meet TAs. The TAs are always there to help you. Every day, there would be at least one TA who is holding office

hours for you. You can get the timing from: `http://www.cs.rochester.edu/courses/261/spring2018/calendar.html`. Also, you can make appointments with any TA.

Put each of the eight queries in its own command file: query1.sql, query2.sql, ..., query8.sql. Make sure that the naming of your files corresponds to the ordering above, as we will be checking correctness using an automated script.

Submission instructions

To submit Part 1 of the project, first gather the following files in a single submission directory `proj2p1`:

```
proj2p1report.pdf
create.sql
load.sql
query1.sql
query2.sql
query3.sql
query4.sql
query5.sql
query6.sql
query7.sql
query8.sql
```

Now, create an archive file (with .tar extension) for submission.

```
cd ..
tar -cvf proj2p1.tar proj2p1
ls
```

(Note: `cd ..` takes you to the parent directory and `ls` command should display all the files int the directory including the tar file.)

Submit the tar file using the command (Note: Don't forget the tilde (~) at the beginning):

```
~csc261/submit p2p1 proj2p1.tar
```

You should get a feedback whether the submission is successful or not.

You may resubmit as many times as you like; however, only the latest submission (along with the timestamp) will be saved, and we will use your latest submission for grading your work. Submissions via email will not be accepted! No late submissions allowed.

Grading

Total: 100 pts

Report (50 pts)

- (20 pts) Written schema
  You must include all the attributes and their datatypes. Also, include the primary key and the foreign key(s) (if any) for each relation. If you have a foreign key, you must show which field it references to.
- (30 pts) ER Diagram
  You must include all the entity sets, their attributes, and all the relations. Also, you must provide the cardinality, participation constraint (total vs partial), and(min, max) constraint.

  Note that though only providing (min, max) constraint is enough and providing cardinality and participation constraint is redundant, but for mastering the concepts you MUST provide all, cardinality, participation constraint (total vs partial), and(min, max) constraint.

SQL (50 pts)

5 pts for each of the following files:

```
create.sql
load.sql
query1.sql
query2.sql
query3.sql
query4.sql
query5.sql
query6.sql
query7.sql
query8.sql
```

Academic Honesty

You MUST NOT share (or get) any component of this project with (or from) any other students which includes your Project 1 team members. This project should be solely your own work. Once you master the material, you are encouraged to share your learning while collaborating in Project 1.

For more details, see: `https://www.rochester.edu/college/honesty/`

Acknowledgement

Adapted from a project given in CS145 offered by Stanford University.