

DATABASE DESIGN PROJECT - Netflix Database

1 Purpose of the project

You are to analyze the requirements, design, implement, document, and demonstrate a database system that could be used as the backend database of the Netflix DVD rental service.

2 Overview

The main goal of this project is to design a relational database, called NetflixDB, that can be used to automate the data management functions of Netflix. You will also be required to populate this database using XML and HTML data available on the Web, and create a web interface to interact with this database.

3 Application Requirements

Netflix is an online DVD rental service, that offers flat-rate rental-by-mail to customers in the United States. Netflix offers different subscription plans that limit how many *outstanding* movies a customer can have at any time (the most common plan allows 3 outstanding DVDs for \$17.99 per month). The DVDs are mailed back and forth between the customers and Netflix using USPS. For more information on Netflix, see <http://www.netflix.com> or <http://en.wikipedia.org/wiki/Netflix> (the Wikipedia entry for Netflix).

You are free to change the service model, subscription plans or anything else you might like if you feel your idea is better. In case of drastic changes, you might want to talk to the TA or me first.

3.1 Data

- **DVD Data:** Information about the DVDs available for rental. This should include all the pertinent information about the DVDs. For movies, this should include the actors, actresses, director, producer, release year, genre (Comedy, Mysteries) etc. For music DVDs, this will include the artist, record company, category (rock, jazz etc), release data, possibly the venue (for live recordings) etc. Netflix also has TV show DVDs, sports DVDs, and an extensive collection of foreign movies.

Along with the basic movie information, you should also store additional information that would be of help to customers. For example, you should include information about awards movies have won, and maybe critic reviews.

Finally, information about DVDs not offered by Netflix also needs to be stored, so that Netflix can at least inform the users that the DVD they want exists, but is not offered for rental by Netflix.

- **Customer Data:** Information about the customers. Along with the basic information, this should also include all the DVDs rented by the user, and the plan they are under. The database should also store each user's *DVD ratings*. Users are allowed to rate DVDs they haven't rented from Netflix, but

have already seen. This information will be used to recommend movies to the customers. Finally, the customer id, password, and credit card information should also be maintained.

3.2 Tasks & Queries

This section gives only a few of the tasks and queries. You may choose to add and/or substitute some other more interesting ones depending on the emphasis of your project.

- **Extract-Transform-Load (ETL)** Populating the database (data entry) is one of the most important tasks that you have to *design*. Much of the auxiliary information stored by this database is available on the web. For example, information about American movies (and many foreign movies) is available at IMDB; critic reviews are available at <http://www.rottentomatoes.com> among other sites; Academy Awards information is available on their website and so on.

Such Web data will most likely be in the form of HTML or XML documents. Therefore, parsing such documents is an important aspect of the project and will give you a unique experience that will be useful in the increasing number of databases that depend on data coming from the web. *Automatic* data entry of such documents is necessary.

You should break this task in several subtasks for dealing with different types of data, sources, and, other peculiarities that you will be confronted. You might need to do some “data cleansing” during the transformation and loading. One such cleansing would be the discovery and elimination of duplicate data coming from different sites. Another could be discrepancies in the form of the data. An example could be the variations of dates, European vs American, etc. All these reconciliations are done during ETL.

You do not need to load all the data that you obtain into the database (in fact, given the storage limits on your accounts, you won't be able to do so). Only insert a small portion of the data into the database.

Also, be careful in downloading data from websites. Be *very* careful if you use automated tools for downloading data; you should never leave such tools running without supervision. Most of the websites do not like it if you send them to many requests for data.

- **Customer Interaction Queries:**
 - **Browse/Search for DVD's:** A customer should be able to search the DVD collection by various search parameters such as the genre, actor name, director name, DVD name, year of release, oscar winners, oscar nominated movies etc. Additionally, the customer may specify that only the DVD's she hasn't seen before be returned.
 - **Display information about a DVD:** This would include all known information about the DVD, including for example, the critic and customer reviews. This should also include information about whether the DVD is currently available for renting.
 - **Rent a DVD:** If the movie is available, and the customer has less than 3 outstanding movies, then the DVD can be rented to the user. If the movie is not available, then the customer should be placed on a wait-list. The customer should be informed about at least the length of the wait-list before doing this (this information can be displayed as part of the above query).

- **Rate a DVD:** The customer should be able to rate DVDs that she has seen previously, whether through or not through Netflix.
- **Netflix Administrative Task Queries:**
 - **Mail movies to the customer:** At the end of the day (or at some other predetermined frequency), find out which movies need to be shipped to the customers. Collect information that can be used for creating the mailing labels (the DVD name/stock number, customer address etc).
 - **Movie returns:** A movie has been returned, update the inventory.
 - **Update inventory:** Add new DVDs purchased by Netflix. Allow for removal of DVDs from the inventory as well.
 - **Add/remove customers.**

There are many directions that you can explore to enhance your project for extra credit. Under no circumstances the extra credit can exceed 20% of the projects highest score or 6% of the total grade. Some suggestions:

- Recommend movies to the users based on their preferences/ratings, and other auxiliary information that you have stored. There are many ways this can be done. Ask me if you want ideas (e.g. *collaborative filtering*).
- Store the customer credit card information *securely*.
- In case of waiting lists, use the previous information about the rentals of a DVD to estimate the number of days a customer would have to wait.

Be creative. Remember Blockbluster has just started its own service along similar lines, and you need to be able to compete with it (and its name recognition).

4 Rules of the game

- **Groups:** The project is to be done in groups of 2 students. A roster for each group must be submitted to the TA by the date specified in the “Due Dates” section of class schedule. The groups are “self-policing” (e.g., each group is responsible for its own division of labor, scheduling, etc.). *Note: If an unreconcilable problem arises in your group, it is your responsibility to contact both the professor and the TA as soon as possible. After the project is due, it will be too late.*
- **Assumptions:** In cases where you have questions on the above description, it is acceptable to make assumptions about the application providing that: 1) they are explicitly stated in the report, 2) they don’t terribly conflict with any of the requirements specified above, and 3) they are “reasonable”. If you have a question about the acceptability of any of your assumptions, check with the TA or the professor.
- **Reports:** A report should be handed in for checking at the end of each phase. The report must be formatted in a reasonable manner (i.e., using a text processor and a decent printer). Reports are due during class on the date specified in the “Due Dates” section below.

- **Implementation:** The final phase of the project requires a working implementation of the system to be built, tested, and demonstrated. A large part of the project grade depends on the quality of this implementation. The implementation will be done as a client-server system in which a web server runs on your cluster unix account, accepts web queries, and connects to the Oracle DBMS to retrieve from the database. Details on the ORACLE system and accounts for both the cluster and ORACLE will be provided. Check at the TA's corner in the class home page and with the TA himself. *A small portion of the project grade will be based on the Web-based user interface. Use of tools such as JDBC or cgibin is mandatory. A bigger portion of the grade will depend on the quality of data entry which will assure robustness of the database.*

5 Project Phases

The three phases of the project cover the following work-processes from the paper *An Adaptable Methodology for Database Design* [1]:

| Phase | Phase Name | Due Date |
|------------|--|--------------|
| <i>0</i> | Group names to the TA- drop dead | Feb 17, 2005 |
| <i>I</i> | Environment and Requirement Analysis and System Analysis and Specification | Mar 3, 2005 |
| <i>II</i> | Conceptual Modeling and Task Emulation | Mar 31, 2005 |
| <i>III</i> | Implementation and Testing | Apr 28, 2005 |

6 Reports

The Phase I report must contain:

1. a *short* description of the purpose of the project and the purpose of this phase of the project.
2. a description of the scope of the project
3. a description of the technical/conceptual problems encountered in this phase and justification for the solutions.
4. the assumptions that you have made about the enterprise.
5. a description of the procedures in the enterprise, as you imagine they happen.
6. all the documentation produced in this phase, i.e.
 - the top-level information flow diagram, (*very* important)
 - the list of tasks, subtasks, and the task forms
 - the list of documents and their form (important)

The Phase II report must contain:

1. Phase I- with corrections addressing the TA's feedback.
2. a *short* description of the purpose of this phase of the project.

3. a description of the problems encountered in this phase and justification for the solutions.
4. the documentation produced in this phase, i.e.,
 - the graphical schema using the E-R model,
 - list of the attributes for each entity and relationship,
 - the relational schema obtained by mapping the E-R to relations, and their Boyce-Codd or 3rd Normal Form with keys.
 - the code for each task: Pascal-like pseudo-code and the embedded DML code.

The Phase III report must contain:

1. Phase I and Phase II reports with corrections addressing TA's feedback.
2. a description of the purpose of this phase of the project,
3. a description of the problems encountered in this phase and justification for the solutions.
4. any revisions made to the relational schema definition from Phase II,
5. documentation produced in this phase, i.e.:
 - a source program listing.
 - a users manual for the system.
 - your testing efforts: erroneous cases that your system can detect and handle reasonably.
 - a description of the system's limitations and the possibilities for improvements.
6. In addition, a demo of the system is required. All members of the group should attend this demo, to explain the aspects of the project for which they were responsible.

References

- [1] N. Roussopoulos and R. T. Yeh. An adaptable methodology for database design. *Computer*, May 1984.