1 Purpose of the project

You are to analyze the requirements for, design, implement, demonstrate and document a database system that could automate the administrative functions of an electronic movie store. This fictitious store is a cross between ‘amazon.com’ and ‘epinions.com’: It allows ordering of video tapes, as well as recording feedback and browsing of other people’s feedbacks.

2 Requirements

A description of the major functions and data items follows.

Data: The system should include information about the (registered) users, the tapes they have ordered and the warehouse availabilities. Also, information about opinions, ratings, and ‘trust-circles’ (see later).

- **Membership data**: The name of every registered member, his/her login name, major credit card number, address, phone number, tapes he/she has ordered.
- **Movie data**: A unique id for the movie (‘m-id’), its title, names of major actors in it, name of director, year of release, rating (G,PG, PG13, R), category (action, drama, comedy, mystery, horror), number of copies available in the warehouse, and the price of each tape.
- **Opinions**: Users can give ‘feedback’ for a movie, as a grade (1-10) along with optional short text; they can also rate other users’ feedback as ‘useless’, ‘useful’, ‘very useful’; finally, they are allowed to declare other users as ‘trusted’ or ‘not-trusted’.

Tasks: The following type of events should be handled by the target system:

1. *Registration*: a new user has to provide the appropriate information; he/she can pick a login-name and a password. The login name should be checked for uniqueness.
2. *Ordering*: After registration, a user can order one or more video tapes. The total amount of its order is reported to him/her. A user may order multiple copies of a movie, one or more times. (The charging of the credit card and the shipping of the tapes are outside the scope of this project)
3. **User record**: upon user demand, we should print the full record of a user:
   - all his/her personal data
   - the full history of sales (movie name, number of copies, date)
   - his/her full history of feedbacks
   - the list of all the feedbacks he/she ranked wrt usefulness
   - the logins of the 'trusted' and 'not-trusted' users, along with the corresponding dates.

4. **New movie**: The store manager records the details of a new movie, along with the number of tapes that have arrived in the warehouse.

5. **Arrival of more tapes**: The store manager increases the appropriate counts.

6. **Feedback recordings**: Users can record their feedback for a movie. We should record the date, the numerical score (0= terrible, 10= masterpiece), and an optional short text. No changes are allowed; only one feedback per user per movie is allowed.

7. **Usefulness ratings**: Users can assess a feedback record, giving it a numerical score 0,1, or 2 (‘useless’, ‘useful’, ‘very useful’ respectively). A user should not be allowed to provide a usefulness-rating for his/her own feedbacks.

8. **Trust recordings**: A user may declare zero or more other users as 'trusted' or 'not-trusted'.

9. **Movie Browsing**: Users may search for movies, by asking **conjunctive** queries on the actors, and/or director, and/or title-words, and/or rating (G, PG, etc). Your system should allow the user to specify that the results are to be sorted (a) by year, or (b) by the average numerical score of the feedbacks, or (c) by the average numerical score of the **trusted** user feedbacks.

10. **Useful feedbacks**: For a given movie, a user could ask for the top \( n \) most 'useful' feedbacks. The value of \( n \) is user-specified (say, 5, or 10). The 'usefulness' of a feedback is its average 'usefulness' score.

11. **Buying suggestions**: Like 'amazon.com', when a user orders a tape of movie 'A', your system should give a list of other suggested movies. Movie 'B' is suggested, if there exist a user 'X' that bought both 'A' and 'B'. The suggested movies should be sorted on decreasing sales count (i.e., most 'popular' first); count only sales to users like 'X'.

12. **'Six degrees of separation'**: Given two actor names, determine their 'degree of separation', defined as follows: Two actors 'A' and 'B' are 1-degree away if they played in at least one movie together; they are 2-degrees away if there exists an actor 'C' who is 1-degree away from each of 'A' and 'B'; and so on.  

13. **Statistics**: every semester, the store manager wants
   - the list of the \( m \) (say \( m=10 \)) most popular movies (in terms of tapes sold in this semester),
   - the list of \( m \) most popular directors and
   - the list of \( m \) most popular actors

14. **User awards**: At random points of time, the store manager also wants to give awards to the 'best' users; thus, the manager needs to know
   - the top \( m \) most 'trusted' users (the trust score of a user is the count of users 'trusting' him/her, minus the count of users 'not-trusting' him/her).

\[1\] A fun fact: most pairs of actors seem to have degrees less than six! see us.imdb.com/Games/slinks
the top \( m \) most ‘useful’ users (the usefulness score of a user is the average ‘usefulness’ of all of his/her feedbacks combined)

3 Phases

The three phases of the project cover the following work-processes from the \textit{Adaptable Methodology for Database Design} by Roussopoulos and Yeh [IEEE Computer, May 1984]:

- \textit{Phase I}: “Environment and Requirement Analysis” and “System Analysis and Specification”
- \textit{Phase II}: “Conceptual Modeling” and “Task Emulation”
- \textit{Phase III}: “Implementation” and “Testing”

3.1 Reports:

A \textbf{TYPED} report should be handed in for grading at the end of each phase. The final project report consists of three parts: the Phase I report, the Phase II report, and the Phase III report. Each report will be graded out of 100 points. Numbers in square brackets indicate the points (out of 100) for each part of the reports.

The Phase I report must contain

1. [2pt] a description of the purpose of the project and the purpose of this phase of the project.
2. [13pts] the assumptions that you have made about the enterprise.
3. all the documentation produced in this phase, i.e.
   - [30pts] the top-level information flow diagram, (very important - also, don’t forget the system boundary)
   - [5pts] the list of tasks and subtasks,
   - [15pts] the task forms,
   - [5pts] the list of documents, and
   - [30pts] the document forms (very important)

The Phase II report must contain

1. [1pt] a description of the purpose of this phase of the project.
2. [4pts] the top level information flow diagram from Phase I, as well as a list of any revisions that were made to the specification described in the Phase I report
3. the documentation produced in this phase, i.e.,
   - [20pts] the graphical schema using the E-R model,
   - [5pts] list of the attributes for each entity and relationship,
   - [5pts] explanations of the non-obvious entities and relationships,
   - [20pts] the schema in the relational model, in some appropriate Normal Form: BCNF, or \textbf{3NF at worst}.
   - [5pts] explanations (e.g., primary keys, additional functional dependencies, explanations why a table is not in BCNF e.t.c.),
   - [10pts] the DDL statements to create the above relational schema,
• [30pts] the code for each task: Pascal-like pseudo-code and the embedded DML code.

The Phase III report must contain

1. [20pts] a description of the purpose of this phase of the project, the relational schema definition from Phase II, and any revisions made to the specification described in the Phase II report.

2. the documentation produced in this phase, i.e.:
   • [20pts] a source program listing.
   • [10pts] a user’s manual for the system.
   • [40pts] your testing efforts: erroneous cases, that your system can detect and handle reasonably.
   • [10pts] a description of the system’s limitations and the possibilities for improvements.

3.2 Grading - due dates

The due dates are as announced in the schedule. The weights are as follows:

• 30% for the report of Phase I
• 30% for the report of Phase II
• 10% for the report of Phase III
• 30% for the Demo.

4 Clarifications in the R+Y methodology

This discussion clarifies or simplifies some points of the methodology by [Roussopoulos and Yeh, IEEE Computer, May 1984]. In your project you should use the above methodology, with the following exceptions/modifications:

1. Ignore queries, symbolized as ‘circles’ in the paper. Replace them with tasks (‘ovals’)
2. VERY IMPORTANT: Do not show consecutive tasks, nor consecutive documents in your top level diagram. That is, a task should be linked to documents only; a document should be linked to tasks only.
3. You may use the list of construct in your Document Forms. For example, a student record could be:

   Student record
   SSN
   Name
   Courses: list of
     course name
     grade
     ...
