CMPE 450
SOFTWARE ENGINEERING
DATABASE DESIGN DOCUMENT

Version 1.0

Project Name	:	Stock Follow Up System
Submitted to Instructor	:	Ayşe Başar Bener
Asistant	:	Gül Çalıklı, Onur Güngör
Submitted by	:	Database Design Group 3

Cengiz Bayram
Alptekin Uzel
Gülnur Derelioğlu

Boğaziçi University
Fall 2006

TABLE of CONTENTS

1. INTRODUCTION

In this document, as the Database Management group of the Stock Follow up System Project, we will first briefly explain our design principles. Then, our database design will be explained in detail including the properties, descriptions of the tables, key fields, the content of the fields, data types and the relations between the tables. Finally, we will end the document by an evaluation of our design from an objective perspective as much as possible.

1.1. Purpose

Our main purposes in this paper are;

- To define database tables according to the requirements of the Stock Follow up System v1.0.
- To define fields in tables, their properties and restrictions on them if any exists.
- To explain the design trade-offs while forming tables.
- To define primary keys in tables and foreign keys between them.
- To define standards when naming fields, tables and stored procedures.
- To define the ways for accessing tables that will be used by other teams. In other words, to define stored procedures.
- To give general information about system or software properties that is used in the implementation.
- To define important factors in design approach.

1.2. System Properties

In this project, we will use MySQL to implement tables and procedures. The MySQL® database has become the world's most popular open source database because of its consistent fast performance, high reliability and ease of use. It's used in more than 10 million installations ranging from large corporations to specialized embedded applications on every continent in the world. (Yes, even Antarctica!)

MySQL is owned and sponsored by a single for-profit firm, the Swedish company MySQL AB, which holds the copyright to most of the codebase. This is similar to the JBoss model and how the Free Software Foundation handles copyright in its projects, and dissimilar to how the Apache project does it, where the software is developed by a public community, and the copyright to the codebase is owned by its individual authors.

The company develops and maintains the system, selling support and service contracts, as well as proprietary-licensed copies of MySQL, and employing people all over the world who collaborate via the Internet. MySQL AB was founded by David Axmark, Allan Larsson, and Michael "Monty" Widenius.

Not only is MySQL the world's most popular open source database, it's also become the database of choice for a new generation of applications built on the LAMP stack (Linux, Apache, MySQL, PHP / Perl / Python.) MySQL runs on more than 20 platforms including Linux, Windows, OS/X, HP-UX, AIX, Netware, giving you the kind of flexibility that puts you in control.

1.3.Design Approach

1.3.1.Investigation

As we mentioned in the previous document, we have a XML-based standard for getting data and stored procedure specifications. Teams have sent the needed specifications and according to these specifications we formed a general basis for tables.

1.3.2.Scope

We tried to set realistic and affordable goals. Our main focus is functionality. In other words, we concentrate mainly on completing database operations on time and correctly. After that performance issues are considered.

1.3.3.Design

Our main design goals, although they are certainly conflicting, are to avoid space inefficiency and minimize join operations. We tried to take a balanced approach between those alternatives as much as possible. In order to improve space efficiency we sometimes divided the data into several tables, separating elements having different information. This enables us to save space as we do not have to keep null fields for elements of the system that do not bear the given information in a column. However, sometimes we decided that it would be better to have some space inefficiency rather than doing many join operations. These will be explained in more detail in the trade-offs and design evaluation section; overall, the main design issue for us was to balance space inefficiency versus the overhead of join operations.

1.3.4. Maintenance

Once the system was in place and stable, requests were implemented carefully in order not to disturb the operation of the system. Especially changes on the tables, addition of new ones or especially the modification of existing ones, required considerable amount of maintenance work on the SPs because of the inflexibility introduced by MySQL regarding returning multiple columns. Sometimes types had to be defined from scratch. Also as the number of SPs increased, it became harder to spot errors.

1.4. Trade-offs

As mentioned above, the most important design choice we considered was whether to allow some space inefficiency in exchange for a performance increase gained by fewer join operations. During design and implementation, we usually favoured preserving space efficiency. We implemented extra tables to preserve space efficiency. This caused maintenance to be a problem because there were lots of tables. However, performance increased due to less number of rows.

Second, giving access permissions to each group was a matter of trade-off between security and complexity. If we gave every group the same right of accessing everywhere in database then there would be problems of security. On the other hand, if we gave each person a different access permission and password then it would require an extra time and work to handle it. Despite the extra work, we chose to give each group different passwords, since security is a much more important concern.

A detailed evaluation of the database design is available in the design evaluation section.

1.5.Interface Design Standards

Database naming standards

- Table names must start with tbl ex: *tbl_user*
- Underscores will be used to separate meaningful words or phrases.
- If table names contain one word, field names must start with first three characters of the name of the table ex: *use_name*
- If table names contain more than one word, field names must start with initials of the table name. Ex: if the table name is *tbl_user_types* then field name will be *ut_operations.*
- Stored procedure names must begin with the name of the group that the request came from, i.e. *aa_get_transactions_by_orderid*. However, if a group requested a formerly implemented stored procedure, the group was just given permission to execute the existing stored procedure.

All standards mentioned above are defined by Test and Security Group. In this document and also in our database implementation we use these standards.

2. TABLES

2.1.TABLE_USER_TYPE

usertypeID	udescription
1	Administrator
2	Rector
3	Purchase Department Manager
4	Purchase Department Staff
5	Department Manager
6	Department Staff

2.1.1.Purpose

Currently, there are 6 types of user in the program. These are :

1. Pruchasing Department Staff
2. Purchasing Department Manager
3. Rector
4. Administrator
5. Department Staff (Other Deaprtment's staff except Purchasing Department)
6. Department Manager (Other Department's staff except Purchasing Department)

Our main purpose of creating this table is to define users' type numerical and seperated from other users' information thus we scan specify each user types.

2.1.2.Key Fields

“usertypeID” field is unique for each record and the primary key field of the USERTYPE table.

2.1.3.Referential Integrity Constraints

There is no referential integrity in the USERTYPE table.

2.2.TABLE_OPERATIONS

authorityID	adescription
1	Add New Department
2	Add New Item
3	Approve New User
4	Add New Transactions
5	Approve Transactions
6	View Transactions
7	View Related Transactions
..	...
..	...

2.2.1.Purpose

The main aim to create this table is to give user a chance to add new authorities to the system. To achieve this, we have created a new table called OPERATIONS to store all authorities defined by numerical id and their explanations.

2.2.2.Key Fields

authorityID is a unique field and the primary key of the table OPERATIONS.

2.2.3.Referential Integrity Constraints

There is no referential integrity in the USERTYPE table.

2.3.TABLE_USERTYPEOPERATIONS

usertypeID	**authorityID**
1	5
..	..

..	..
2	3
3	3
4	1
4	2
4	3
...	...
...	...
...	...
...	...

2.3.1.Purpose

Our main purpose in defigning the USERTYPEOPERATIONS table is to create relation between USERTYPE table and OPERATIONS table. Thus, we can define the authorities for each usertype. This model let us to change the authorities of the users when necessary.

2.3.2) Key Fields :

usertypeID and authorityID fields are composite primary key which are unique only together.

2.3.3) Referential Integrity Constraints :

usertypeID field is the foreign key referencing the usertype field of the USERTYPE table.

authorityID field is the other foreign key referenecing the authorityID field of the OPERATIONS table.

2.4.USERS

username	usertypeID	departmentID	name	surname	address	phone	email	password	statuID

2.4.1.Purpose

Our main purpose of creating this table is to hold the list of all users and their relevant information. Newcomers to the system will fill a form to register to the system and the data in this form will be hold in this table. Password field will be transferred via encryption to the database and statuID field will be used for to hold the status of a user. There will be 3 statuses:

- Active (1)
- Passive (0)
- Pending (2)

2.4.2.Key Fields

"username" field is unique for each record and the primary key field of the USERS table.

2.4.3.Referential Integrity Constraints

"departmentID" field is the foreign key referencing the "departmentID" field of the ALLOWED_DEPARTMENTS table.

"usertypeID" field is the foreign key referencing the "usertypeID" field of the USER_TYPE table.

"statuID" field is the foreign key referencing the "statuID" field of the STATUS table.

2.5.TRANSACTIONS

orderID	itemID	price	amount	orderdate	approvaldate	purchasedate	purchasedepID	seller	total	orderdepID	statu
67	8	50	5	1-1-2007	1-2-2007	1-3-2007	3	Koç	100	3	1
...	...	...	...	...	...	...	...	...	...	...	...

2.5.1.Purpose

The main aim to create this table is to hold the list of all transactions that are approved, rejected and pending. When a transaction is added to the system it's status is pending (0) as default. Later it can be approved or rejected. Also this table will be used to search and report transactions later on. Since the project is about Stock Follow Up this table is the heart of the database.

2.5.2.Key Fields

orderID and itemID fields are composite primary keys which are unique only together.

2.5.3.Referential Integrity Constraints

"itemID" field is the foreign key referencing the "itemID" field of the ALLOWED_ITEMS table.

"purchasedepID" field is the foreign key referencing the "departmentID" field of the ALLOWED_DEPARTMENTS table.

"orderdepID" field is the foreign key referencing the "departmentID" field of the ALLOWED_DEPARTMENTS table.

"statuID" field is the foreign key referencing the "statuID" field of the STATUS table.

2.6.ALLOWED_DEPARTMENTS

departmentID	departmentname	status
CMPE	Computer Engineering	1
IE	Industrial Engineering	1
PHIL	Philosophy	2
...	...	
...	...	

2.6.1.Purpose

Our main purpose in defining the ALLOWED_DEPARTMENTS table is to give control to the administrator of the system while registering users. In this way only the allowed departments can make transactions which are listed in this table. The department text field of the registration form will only allow to the users who are in the departments listed in this table.

2.6.2.Key Fields

"departmentID" field is unique for each record and the primary key field of the ALLOWED_DEPARTMENT table.

2.6.3.Referential Integrity Constraints

There is no referential integrity in the ALLOWED_DEPARTMENTS table.

2.7.ALLOWED_ITEMS

itemID	itemname	status	description
8	IKEA black steel desk	1	desk

2.7.1.Purpose

Our main purpose in defining ALLOWED_ITEMS table is to to give control to the purchasing department while approving transactions. In this way only the allowed items which are listed in this table can be stated in a transaction.

2.7.2.Key Fields

“itemID” field is unique for each record and the primary key field of the ALLOWED_ITEMS table.

2.7.3.Referential Integrity Constraints

There is no referential integrity in the ALLOWED_ITEMS table

3. DESIGN EVALUATION

This section aims to briefly evaluate our database design. We will try to be as objective as possible; we will talk about both weaknesses and strengths of our design.

First of all, we foresee that the main agents in this automation project are the users. Users have common information, which we store in user, as well as specific information according to their roles in the system such as manager, rector, staff etc. This specific information is kept in the appropriate table. This prevents data duplication and space inefficiency, however to retrieve all information about a user, join operations must be done. This is a small trade off compared to the inefficiency that would be introduced otherwise.

It is obvious that a user can have multiple roles, to allow such a design, role-user pairs are also kept in a separate table. This table tells with which table user is to be joined to retrieve info about that user. This system can also be extended to include a role based security model. At the beginning of the project, a user based security model was planned. However, it is difficult to determine each task in the system.

While we were trying to construct a structure for UserType and their related operations. In our approach we begin design an Usertype tables and an operations tables separately. And give each UserType different permission in a separate table UserTypeOperations.

usertypeID	Description
1	Administrator
2	Rector
3	Purchase Department Manager
4	Purchase Department Staff
5	Department Manager
6	Department Staff

Some parts of our design are not implemented by the other groups, although they were in their requests. We did not choose to remove these tables. Conversely, we had implemented some tables that were not included in the requests, which add more functionality to the project. These may be implemented in the future versions.

Suggestions:

Giving id to course-section-year-semester tuple will decrease data repetitions and increase the efficiency of the stored procedures. We propose an id template like year-department id-course code-course section-course semester, or some different pattern that can be easily understood by user, and will enable uniquely addressing the tuple.

As data kept in the database will never be deleted physically, there must be a inactive flag for every row in every table in the database. Actual deletion may be done by a database administrator, but like when a instructor retires, his information must still be kept.

SQL is not as easy to code as JAVA. The Java class that is used for database connection can also be used for processing the data. For example, loops, switch-case statements can be done within the class

There is also an XML generator in the class. This can be used to convert the result set into XML, which can act as a communication standard.

4. CONCLUSION

Our main goal was to implement an efficient and secure database system that satisfies all requests of the other groups. Since the some needs of groups coincide and some conflict with each other, we had to make a unified database design satisfying every group's requests. Moreover, we considered the space and time complexity trade-off. One main difficulty we faced is that we started this design from stretch; we did not have a template design to extend or model.

But the request-implement-response cycle did not work as planned. Altough at the beginning of the project, requests submitted to us was in XML format, later this procedure did not work. We directly received other groups requests, they simply told or mailed us their needs. We implemented the requests immediately. This was unavoidable, since there was limited amount of time and their first requests was far away from their actual needs. (After the first requests, we had about 45 tables and 120 procedures, now the database includes 70 tables and 460 procedures.Moreover, we believe that some of the procedures that are implemented in the first stage are not used at the end.) This was hard for us, since we had to spend most of our time in Grad Lab.

Overall, we were satisfied by the final design. Although it is a one term course project, it is not much different then a real application database. It is rather large, includes complex stored procedures and does not include data repetitions. We propose some enhancements to our design in the suggestion section.

5. GLOSSARY

Backup: This is an operation of storing current information and data of a database for further uses in a file.

Database: A database consists of some collection of persistent data that is used by the application systems of some given enterprise.

Database Management: This module includes the physical database design, and the necessary triggers and stored procedures to enforce data integrity. Since many different components will access the same database, it should be capable of meeting the needs and requirements of all these components without data duplication.

Database Management System (DBMS): is the software that handles all access to the database.

Foreign Key: A foreign key in one table is a set of one or more attributes whose values match the values, which exist in the primary key attributes of a related table.

Primary Key: The primary key of a table is the one or more attributes, which are used to distinguish every row of the table from every other row of that same table.

Normal Form: A relation is said to be in a particular normal form if it satisfies a certain prescribed set of conditions.

Determinant: A field, which functionally determines the other fields' values.

Stored Procedure: is a precompiled program that is stored at the server site. It is invoked from the client by a remote procedure call.

Functional Dependency: A functional dependency is a constraint between 2 sets of attributes from the database for each values of the first set there is a unique value of the second set.

Non-key Attribute: any attribute that does not participate in the primary key of the relation concerned.

Relational Database: A database consisting of more than one table. In a multi-table database, you not only need to define the structure of each table, you also need to define the relationships between each table in order to link those tables correctly.

Referential Integrity: Referential integrity is a concept that requires that a foreign key should be either wholly null or its value should match the value of the primary key in some tuple in the parent relation.

Restore: This is a recreation of a database from a backup file.

MySQL: MySQL is a multithreaded, multi-user, SQL Database Management System (DBMS) with more than six million installations. MySQL AB makes MySQL Server available as free software under the GNU General Public License (GPL), but they also offer the MySQL Enterprise subscription offering for business users and dual-license it under traditional proprietary licensing arrangements for cases where the intended use is incompatible with the GPL.
SQL: Structured Query Language (pronounced *sequel* in the US; *ess-queue-ell* elsewhere). A computer language designed to organize and simplify the process of getting information out of a database in a usable form, and also used to reorganize data within databases. SQL is most often used on larger databases on minicomputers, mainframes and corporate servers.
Stored Procedure: is a precompiled program that is stored at the server site. It is invoked from the client by a remote procedure call (RPC).
Trade-off: the exchange of one thing for another of more or less equal value, esp. to effect a compromise.
Trigger: A trigger is nothing more than a block of PL/SQL commands which is automatically fired when an insert, delete, update command is executed in a table.
XML: Extensible Markup Language, or XML for short, is a new technology for web applications. XML is a World Wide Web Consortium standard that lets you create your own tags

6. REFERENCES

- Cmpe 321 Introduction to Database Systems Lecture Notes
- http://www.en.wikipedia.org/wiki/MySQL
- www.thinkdesign.com/approach
- www.cs.pitt.edu/~chang/156/11funcdep.html
- www.faqs.org/docs/ppbook/c208.htm
- www.geocities.com/mnhashmi/triggers.html
- www.datawarehouse.com/iknowledge/articles/article.cfm?ContentID=1846
- "Standards Document" released by Test and Security Group
- MySql Official Documentation:

  http://www.mysql.com/