Validation and Implementation of an Enzyme Activity Mapping Database

Longchang Fu

A Thesis

in

The Department

of

Computer Science

Presented in Partial Fulfillment of the Requirements
for the Degree of Master of Computer Science at
Concordia University
Montreal, Quebec, Canada

December 2003

Your file Votre référence
ISBN: 0-612-91031-8
Our file Notre référence
ISBN: 0-612-91031-8

ABSTRACT

Validation and Implementation of an Enzyme Activity Mapping Database

Longchang Fu

An enzyme activity database (EAMDB) is required to support a fungal genomics project currently underway at Concordia. EAMDB supports high-throughput assays to determine enzyme activities, kinetic constants, optimal temperatures and pH values. This thesis implements and validates the design of EAMDB done by Ronghua Shu, a previous Masters student at Concordia. By working closely with the wet lab scientists, we have validated it through several techniques and implemented a MySQL relational database for EAMDB. We also have created the user interfaces on the web for EAMDB and links between protocol files and relations in this database by using PHP Hypertext Preprocessor (PHP).

Acknowledgements

I would like to express my warmest gratitude to my supervisor, Dr. Gregory Butler, for his patience and invaluable guidance. His profound knowledge in computer science and bioinformatics is highly appreciated.

Dr. Justin Powlowski and Dr. Régis-Olivier Benech are gratefully acknowledged for their introduction to enzyme assays and for their discussions. Special thanks go to Vineet Dua for proofreading my thesis. I would like to thank Dr. Adrian Tsang and people working in fungal genomics project for providing opportunity of participating in this project.

I would like to thank all graduate students in Dr. Butler's group for their helpful discussions and friendship, and especially Nie Bin, Sun Jian whom I discuss often with.

I would like to express my deepest gratitude to my parents, my wife as well my children. Without their love, encouragement and support, I would never have accomplished my study.

Contents

List of Figures

List of Tables

Chapter 1 Introduction

Research in biochemistry and biology is evolving through its ability to generate and interpret large volumes of data from automated laboratory processes. There is a considerable potential to improve the benefits of these data by organizing better access.

1.1 The Problem

The fungal genomics project currently underway at the Centre for Structural and Functional Genomics (CSFG) at Concordia University is about discovering new and interesting enzymes produced by fungi [35]. Researchers in this project are working on finding fungal enzymes that can be harnessed in pulp and paper processing and in the synthesis of fine chemicals. The use of such enzymes permits these industrial processes to be performed under milder conditions, requiring less energy and producing fewer toxic byproducts. In addition, many fungal enzymes are capable of breaking down a broad range of complex compounds, making them potentially useful for removal of persistent pollutants.

In this project, the researchers are divided into several groups according to research objectives. After obtaining cultured enzymes from the gene expression group, researchers in the enzyme activity assay group attempt to determine optimum pH, temperature and substrate concentration for these enzymes by conducting high throughput experiments. From this, they can assign numerical values to enzyme catalytic variables, such as K_m, K_{cat} and substrate concentrations. They are planning to assay

several hundreds of enzymes produced by 14 fungi. Consequently, there is a large volume of data, which needs to be stored into a database and retrieved in a scientific way.

The method of data storage at the moment is unsatisfactory for numerous reasons. First, experimental results are being written to CDs or file systems. This renders quick access to data impossible. Another challenge is the fact that most of the data associated with the experiment are not stored with the results, but in the scientist's own personal lab book [1]. This creates three problems: a) the data will not be fully completed with necessary description; b) the data will never be accessible to others, unless that particular scientist is available at all times to explain their data; c) even if the lab book is accessible, there is often difficulty in interpreting another person's set of results.

1.2 Public Enzymatic Reaction Databases

There are a few databases available on the web providing information about enzymatic reactions. BRENDA (Braunschweig Enzyme Database) [31], UM-BBD (University of Minnersota Biocatalysis/Biodegradation Database) [12], LIGAND [17], and ExPASy (Expert Protein Analysis System) [14] are the main examples. BRENDA is a comprehensive relational database on functional and molecular information of enzymes, based on the primary literature. The database contains information extracted and evaluated from approximately 46,000 references, holding data of at least 40,000 different enzymes from more than 6900 different organisms, classified in approximately 3900 EC numbers. BRENDA is an important tool for biochemical and medical research covering information on properties of all classified enzymes, including data on the occurrence, catalyzed reactions, kinetics, substrates/products, inhibitors, cofactors, activators,

structures, and stability. The data and information provide a fundamental tool for research of enzyme mechanisms, metabolic pathways, the evolution of metabolism and for medicinal diagnostics and pharmaceutical research. The database is a resource for data of enzymes, classified according to the EC system developed by the International Union of Biochemistry and Molecular Biology (IUBMB) Enzyme Nomenclature Committee. The entries are cross-referenced to other databases, namely, organism classification, protein sequence, protein structure and literature references.

UM-BBD contains information on reactions related to biodegradation pathways of xenobiotic compounds in microorganisms. It provides information on pathways, single reactions, chemical compounds including intermediate states of the reactions, organisms, enzymes and genes. The system is connected to other resources such as GenBank, ChemFinder and Medline. A search function allows queries for specific compounds or enzymes by defining names or substrings, chemical formula, CAS number or EC number, respectively. The search page also has links to lists of pathways, compounds, enzymes, reactions and organisms that can all be selected. UM-BBD gives no general pathway overview. Most of the pathway and reaction are disconnected [37].

LIGAND is a composite database comprising three sections: COMPOUND for the information about metabolites and other chemical compounds, REACTION for the collection of substrate-product relations representing metabolic and other reactions and ENZYME for the information about enzyme molecules. It includes 7298 compounds, 5166 reactions and 3829 enzymes. In addition to the keyword search provided by the DBGET/LinkDB system, a substructure search to the COMPOUND, REACTION, and ENZYME sections is also available.

ExPASy of the Swiss Institute of Bioinformatics consists of two databases related to biochemical pathway information—ENZYME and Biochemical Pathways [14]. Querying for compounds, enzymes or pathways by names or substrings results in a list of related positions (squares) on the graphical map. If a reaction is not complete within that resulting square, it is removed and the corresponding information is lost. One of the four adjacent squares can be selected to expand the view.

However, in most of databases meta-information, the source of origin of the data (i.e., experiments, homology search) is missing. A user is not provided with information whether the data is experimentally proven or not.

1.3 Proposed Solution

Unlike the databases discussed above, our proposed database provides more experimental information, where experimental details are recorded along with the characteristics of the enzymes, chemical compounds, and reactions. If the enzyme assays reveal new and useful properties, these enzymes could potentially be applied to the pulp and paper industry, where these new findings would be eventually patented. The information stored in the database also will be a useful resource for the researchers in related scientific fields.

Following the design of Enzyme Activity Mapping Database (EAMDB) [32] done by the Master student, Ronghua Shu, we have reached a feasible solution to the above problem. The first step is to establish a database, using a MySQL database management system, capable of managing the stored relational data. It was decided that the best option

would be to alter an existing database system to achieve this objective. Furthermore, a HTML file system would be set up to handle graphs, protocols and data files.

The main objective of this thesis is to validate and implement the existing design for storing the data for current application and use in the future. Future use of the database should entail its compatibility with other similar databases and data mining.

1.4 Contribution of the Thesis

Based on the requirements of enzyme catalytic assay research, this thesis presents validation of the existing design and implementation of all necessary information about enzyme experimental details and conclusions into EAMDB within MySQL. Specifically, detailed requirements are captured by discussion with researchers informally, analyzing some data files, using use cases approach, and utilizing a query approach. Afterwards, the database design is refined by splitting, adding and deleting tables according to detailed requirements. The next step is to normalize the relations to 4NF. Furthermore, to populate a certain amount of data into the EAMDB, and to query it using a set of queries finalizing the schemas. Moreover, to create HTML files for all current protocols, graphs and data files and links between those files and information stored in MySQL. Finally, create a user interface for operating the EAMDB by using PHP facilities.

1.5 Organization of the Thesis

There are six chapters and an appendix in the thesis. Chapter 1 presents the introduction about the thesis. Chapter 2 gives background information required to understand the context of the thesis. Chapter 3 presents the detailed requirements.

Chapter 4 discusses validation of the EAMDB design. Chapter 5 presents the implementation of the database. Chapter 6 concludes the thesis.

Chapter 2 Background

To understand fully the context of this project and thesis, one may be required to have the moderate background knowledge of biology and biochemistry. This chapter is intended to give the reader a brief and suitable background reading for both the biological context and biochemical assay. First, it gives definitions of scientific terms, for example, a definition of the enzyme and its roles in real life are presented. Secondly, enzyme catalysis and kinetics are discussed. A high throughput enzyme assay and its protocol are briefly described. The chapter is concluded by an overview of Ronghua Shu's design.

2.1 Bioinformatics

Bioinformatics [10] is the application of computer technology to the management of biological information. It evolves with the emergence of new molecular biology techniques that produce vast collections of biological data, namely, determining the nucleotide sequence of an organism's genome. Without any computational tools, searching a genome for a particular gene would be a difficult and lengthy process. However, bioinformatics tools are available to accomplish this more efficiently. Bioinformatics plays an essential role in the discovery of new drug targets, understanding diseases, etc. unfortunately, bioinformatics is also full of pitfalls for those without a complete understanding of the generation and meaning of biological data [27].

More specifically, bioinformatics encompasses: a) the information processing needs of biological data; b) the acquisition of knowledge from the data; c) the mathematical and computer modeling of phenomena in a cell; and d) visualization of

modeling and information. This is a fertile domain for both biologists and computer scientists. It requires genuine advances in algorithms, database and knowledgebase systems, artificial intelligence, software for supporting the experimental processes, data analysis and the relative aspects of biology and biochemistry [6].

2.2 Overview of Enzyme Activity

2.2.1 What is an Enzyme?

First of all, what is a catalyst? A catalyst is a substance that increases the rate of a chemical or biochemical reaction by reducing the activation energy, but which is left unchanged by the reaction. An enzyme is a protein that works like a catalyst in biological cells to prompt chemical changes in other substances. Breaking down food into energy is an example. Thus, enzymes are biological catalysts [9].

2.2.2 Enzyme Structure

Amino acid sequences of an enzyme ultimately determine its structure and function. Usually, enzymes consist of specific sequences of amino acid residues linked to each other by peptide bonds. The sequence of residues in the polypeptide chain is called the primary structure. The chemical properties of the amino acid functional groups (i.e., hydroxyl, carboxylic acid, amino, guanido, phenolic, and sulfhydryl) are largely responsible for the chemical activities, binding specificities, and electrical properties of enzymes [9]. The non-polar hydrocarbon groups are important in maintaining the overall structure of an enzyme and creating the appropriate chemical environments within the enzyme. The backbone of the polypeptide chain consists of peptide bonds. The folding

path of the backbone through space is called the secondary structure. The patterns are complex, having bends, twists and spirals. Secondary structure is mainly determined by hydrogen bonds between backbones. The well-known examples of secondary structure include α-helices and β-pleated sheets. Secondary structure is influenced by the type of amino acids presented in that part of the polypeptide chain. The complete three-dimensional folding pattern of a polypeptide chain, including all the positions of the amino acid functional groups is called the tertiary structure. The tertiary structure creates the crevices and pockets, which enable the protein to bind and react with other molecules.

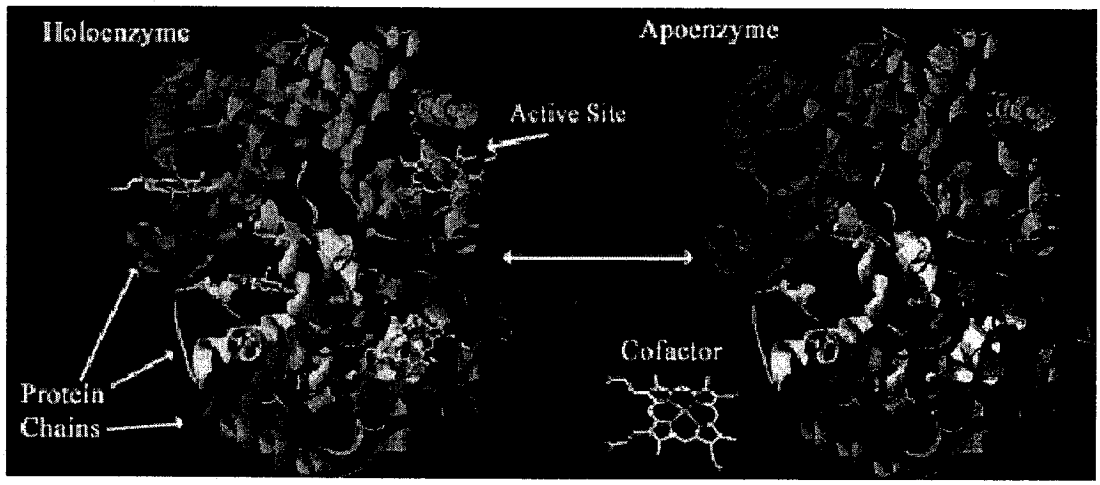


Figure 2.1 Enzyme Structure
(An active site is where catalysis takes place) [4]

A precise tertiary structure is absolutely necessary for the protein's biological activity. Many enzymes consist of several polypeptide chains that are specifically associated with each other by non-covalent and covalent bonds. Please see figure 2.1. The three dimensional arrangement of polypeptide chains to each other in a protein is called the quaternary structure. The individual polypeptide chains that comprise an enzyme are often called subunits. The subunits of an enzyme can be identical, similar, or completely

different from each other. Different subunits can be responsible for different functions within an enzyme [28].

Certain enzymes also contain, as integral parts of their structure, chemical groups that are independent from the amino acid residues, but are absolutely required for biological activity. These groups include small organic molecules, such as certain vitamin derivatives and certain metal ions.

2.2.3 Enzyme Catalysis

Enzymes are a specific group of proteins that are synthesized by living cells to function as catalysts for the many thousands of biochemical reactions that constitute the metabolism of a cell. More than 2000 different enzymes are known, and it is likely that many more are being researched. Enzymes are required in metabolism because at physiological temperature and pH, uncatalyzed reactions would proceed at too slow a rate for the vital processes necessary to sustain life [3].

In the enzyme-catalyzed case, the activation energy barrier has been greatly reduced (See Figure 2.2) [8]. This reduction in activation barrier results in a significant acceleration of reaction velocity in the presence of the enzyme. This is the common strategy for rate acceleration utilized by all enzymes [9]. There are two distinct ways enzymes lower activation energy: a) rearrangement of covalent bonds and b) noncovalent interactions between the enzyme and the substrate.

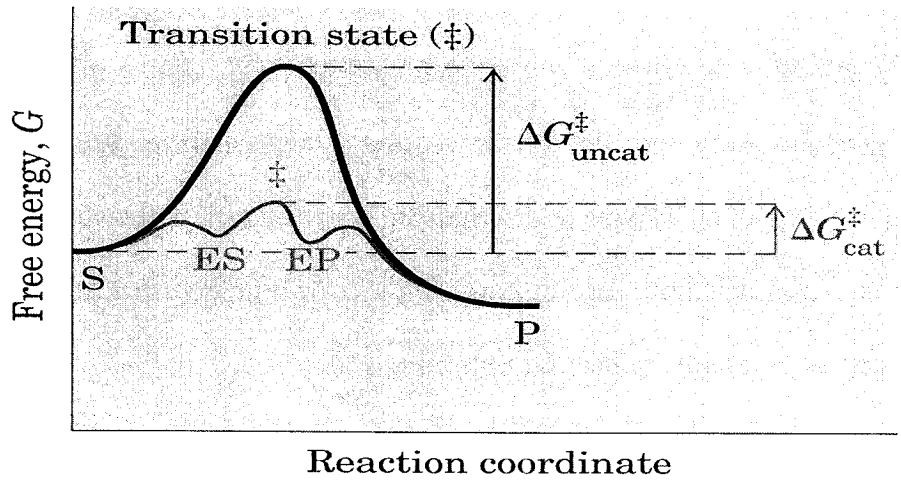


Figure 2.2 The Comparison Between Enzyme-catalyzed and Uncatalyzed Reactions [8]

Some Rate Enhancements Produced by Enzymes	
Cyclophilin	10^5
Carbonic anhydrase	10^7
Triose phosphate isomerase	10^9
Carboxypeptidase A	10^{11}
Phosphoglucomutase	10^{12}
Succinyl-CoA transferase	10^{13}
Urease	10^{14}
Orotidine monophosphate decarboxylase	10^{17}

Table 2.1 Different Rate Values Increased by Enzymes in the Reactions [8]

Two major properties of enzyme are rate enhancement and specificity. Enzymes can accelerate reactions 10^5 to 10^{17} times, which is far greater than any artificial catalysts. **Table 2.1** shows some rate enhancements produced by enzymes. Enzymes generally

produce these accelerations under comparatively mild physiological conditions of neutral pH (around 7), atmospheric pressure and temperatures of 37^0C. Unlike most catalysts, enzymes are generally very specific for the reactions they catalyze. See Figure 2.3. Specificity is the ability of an enzyme to discriminate between a substrate and structurally similar molecules. Consequently, each enzyme, with its own unique structure, catalyzes only a single reaction or closely related set of reactions.

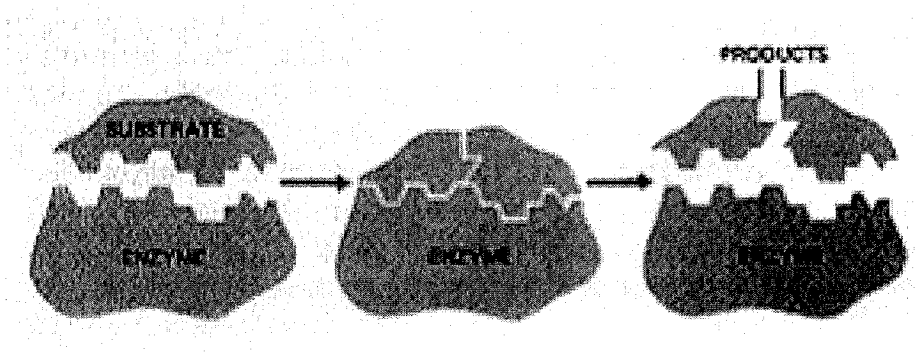


Figure 2.3 The Specificity between Enzyme and Substrate [20]

Various substances can reduce enzyme catalytic activity. These substances are referred to as inhibitors. Generally, they slow down the rate of enzyme-catalyzed reactions by interacting specifically with the enzyme's active site in such a way as to reduce access to the active site by the substrate. Inhibition is a major research subject for enzymology. Inhibitors are important factors in maintaining normal metabolism in cells. They also may be effective drugs to cure some diseases.

2. 2. 4 Enzyme Denaturation

An enzyme that contains all its natural structural elements and possesses biological activity is called native. When an enzyme has been unfolded, it no longer possesses biological activity despite the backbone and the amino acid groups remaining intact. Unfolding also causes subunit dissociation if there are no inter-subunit covalent links between them. Unfolded or inactive enzymes are called denatured.

Any agents or conditions that denature an enzyme will destroy its activity. Enzyme denaturation can be caused by excessive heat (temperatures greater than 45^0C, but there are exceptions), extremes in pH, organic solvents, and cycles of freezing. Ionic detergents, such as sodium, are potent enzyme denaturants. Other agents disrupt enzyme structure by direct chemical reaction with the amino acid residues. Examples include heavy metals and free radicals.

2.2.5 Enzyme Kinetics

A great deal of information can be learned about enzymes by studying reaction rates. To study the rate equation for a simple enzyme-catalyzed reaction, one must consider many conditions, such as temperature, pH and substrate concentration. Generally, in enzyme kinetics study, the Michaelis-Menten model is used as a mathematical model describing enzyme activity. This model is a general explanation of the kinetics and gross mechanism of enzyme-catalyzed reactions.

The substrate binds to the enzyme to form the enzyme-substrate complex (ES). While it is part of this complex, the substrate is converted to product; in other words, the

enzyme-substrate complex is converted to an enzyme-product complex (EP). This then dissociates to enzyme and free product.

$$E + S \rightleftharpoons ES \rightarrow EP \rightarrow E + P \qquad \text{Michaelis-Menten Equation [15]}$$

The rate of the reaction (v) is the rate of appearance of product (P). This rate is known to depend upon the concentration of substrate (S) in the reaction. The Michaelis-Menten model leads to the conclusion that the relationship between v and the concentration of S in the reaction depends on the following equation.

$$v = V_{max}[S]/(K_m + [S])$$

where V_{max} is the value of v when all of enzyme (E) has been converted to ES (i.e., at saturation). It is a measure of the actual rate of conversion of ES to EP. K_m is a constant, which is a measure of the tightness of binding of E to S. It is the value of [S] when $v = V_{max}/2$.

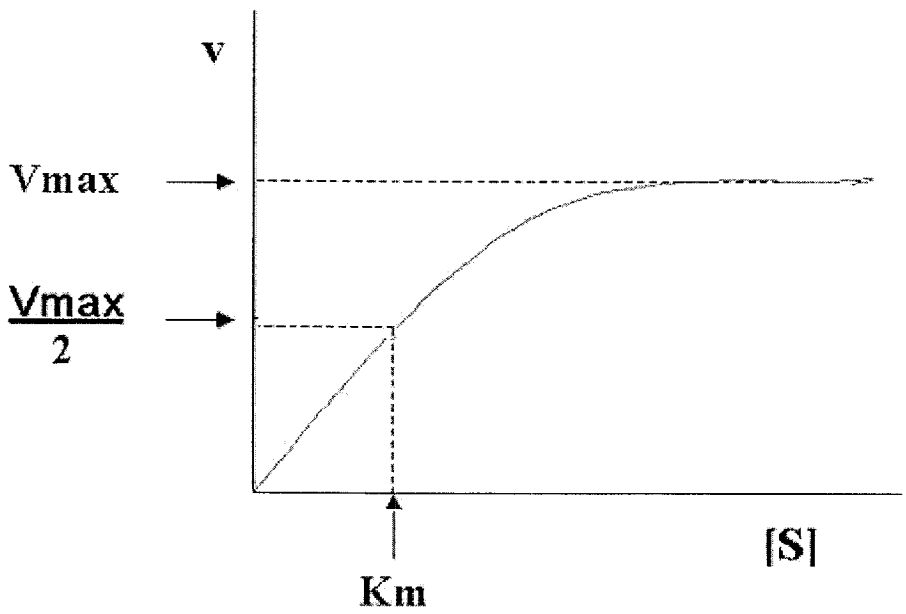


Figure 2.4 Velocity vs. Concentration of Substrate of an Enzyme Reaction

The Michaelis-Menten Equation can be algebraically transformed (by inverting both sides and rearranging) to an equation in which 1/v is a function of 1/[S]:

$1/v = (Km/Vmax)(1/[S]) + (1/Vmax)$ Lineweaver-Burke (inverse) Form.[16]

Note that this equation is of the form y=mx+b, where x = 1/[S] and y = 1/v. Thus a plot of 1/v vs. 1/[S] will yield a straight line whose slope = Km/Vmax, and whose y-intercept = 1/Vmax. It can also be shown that the x-intercept = -1/Km

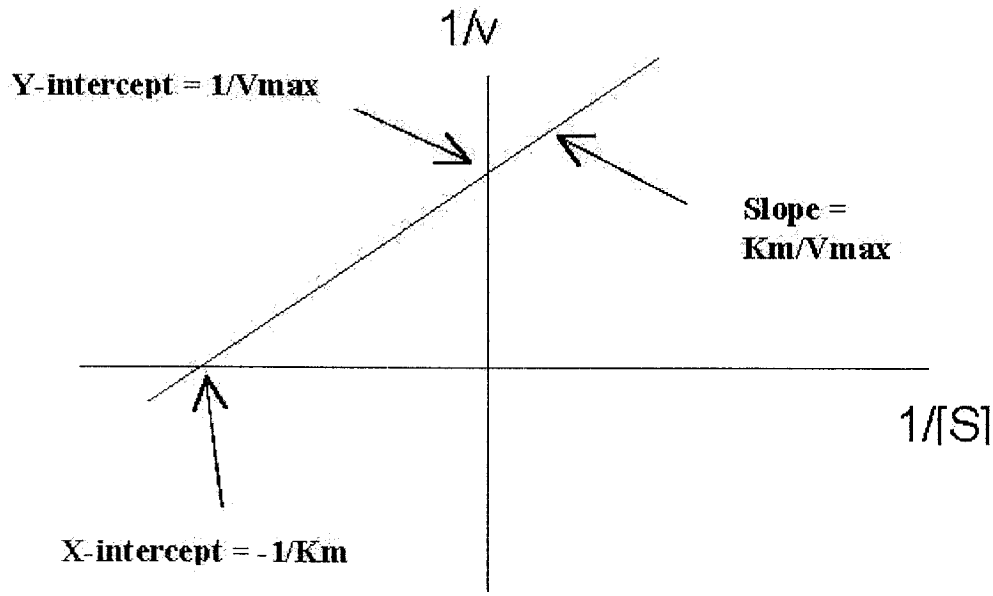


Figure2.5 A Lineweaver-Burk Double-Reciprocal Plot

Michaelis-Menten constants (K_m) have been determined for many commonly used enzymes. The value of K_m tells us some properties about a particular enzyme: a) a small K_m indicates the enzyme binds the substrate tightly. Thus, maximum velocity is reached at relatively low substrate concentrations; b) a large K_m indicates weak binding and the need for high substrate concentration to achieve maximum reaction velocity. Furthermore, catalytic efficiency and turnover number (K_{cat}) are the two useful kinetic

parameters to characterize the catalytic activity of enzymes [34]. K_{cat} is the number of substrate molecules converted to product per enzyme molecule per unit of time, when E is saturated with substrate. $K_{cat} = V_{max}/Et$. Values of K_{cat} range from less than 1 per second to many millions per second [26]. The catalytic efficiency, K_{cat}/K_m, is an estimate of "how perfect" the enzyme is.

2.3 High Throughput Enzyme Assay Principle

High throughput enzyme assay is the process of screening large numbers of compounds for binding activity or catalytic activity against target enzymes [29]. The assay focuses on finding whether enzymes have activities or not and the determination of optimal reaction conditions. High throughput assay based on a simple chromogenic or fluorogenic test can analyze several hundreds or thousands of samples each day.

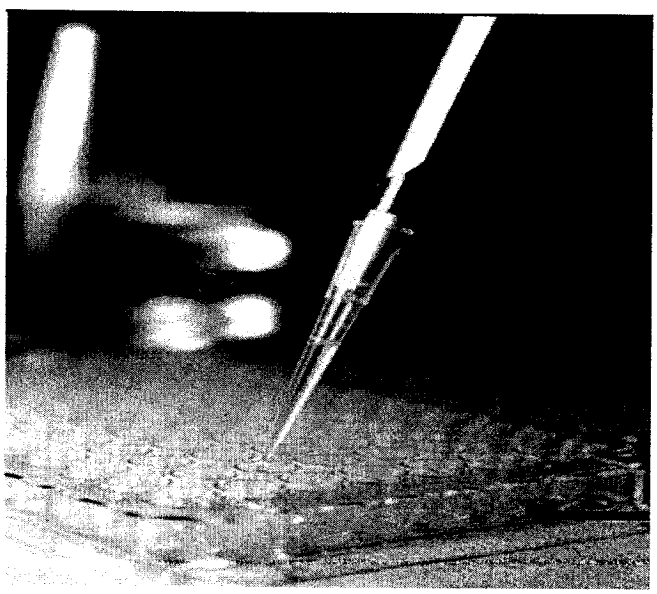


Figure 2.6 A 96-well Microtitre Plate of Enzyme Assay [25]

Among the widespread high throughput techniques for enzyme assays, microtitre plate-based assays are among the most applied ones. In a typical procedure, reaction samples are added to the wells of a 96-well microtitre plate (Figure 2.6). The reaction samples are prepared according to the specific experimental purpose. For instance, if an experiment is designed to determine an optimum temperature on reaction velocity, the reaction samples should be prepared with different temperatures while keeping the other reaction conditions the same. The microtitre plate with samples is then placed in a sample chamber of a plate reader. The samples are automatically mixed and scanned periodically for changes in absorbance. The readings are recorded and exported in various formats.

2.4 EnzChek Protease Assay Protocol

The EnzChek Protease Assay Kits provide fast, simple and direct fluorescence-based assays for detecting a wide variety of metallo-, serine, acid and thiol proteases. In the assay, one simply incubates the protease with the quenched substrate. Since the assay is continuous, researchers can easily obtain kinetic information by measuring the fluorescence increase over time.

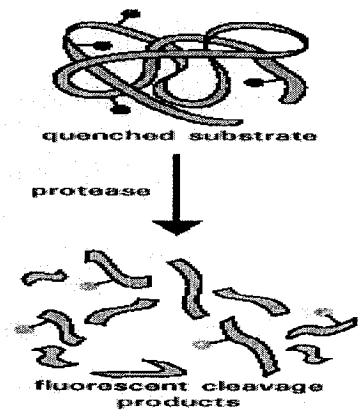


Figure 2.7 Principle of Protease Detection

The kits contain a casein substrate that is heavily labeled with green fluorescent BODIPY FL, resulting in almost total quenching of the conjugate fluorescence. Protease-catalyzed hydrolysis relieves this quenching, yielding bright green fluorescent peptides respectively (Figure 2.7). The increase in fluorescence, which can be measured with a fluorescence microplate reader, is proportional to protease activity [25].

2.5 Ronghua Shu's Design

2.5.1 Use Cases Analysis

Mr. Shu's design described four groups of users in this database: database administrator, internal scientists, external scientists, and bioinformaticians. See Figure 2.8. A database administrator has full control over the database. His activities include database maintenance and data manipulation based on the requirements of users. He may add new parts to the database and restructure it in order to query related information efficiently. Internal scientists are the data source and major users of the database. They have full access to the database, and their activities include information collection and retrieval. They can perform the following queries: 1) choose an enzyme as a catalyst for a given substrate or application, 2) choose a substrate for a given enzyme, 3) find out the optimal reaction conditions for a given enzyme and substrate, 4) check the enzyme stability in various chemical environments, 5) check the structural information of a substrate, 6) check the original data and details of an experiment, 7) check the technical protocol of the experiment, 8) obtain a detailed experimental procedure for a given enzyme and substrate, and 9) check available protocols for the assay of an enzyme. Activities of external scientists include accessing the enzyme and chemical information and some

research results. The bioinformaticians can make use of the enzyme catalytic assay results to predict and derive unknown enzyme properties.

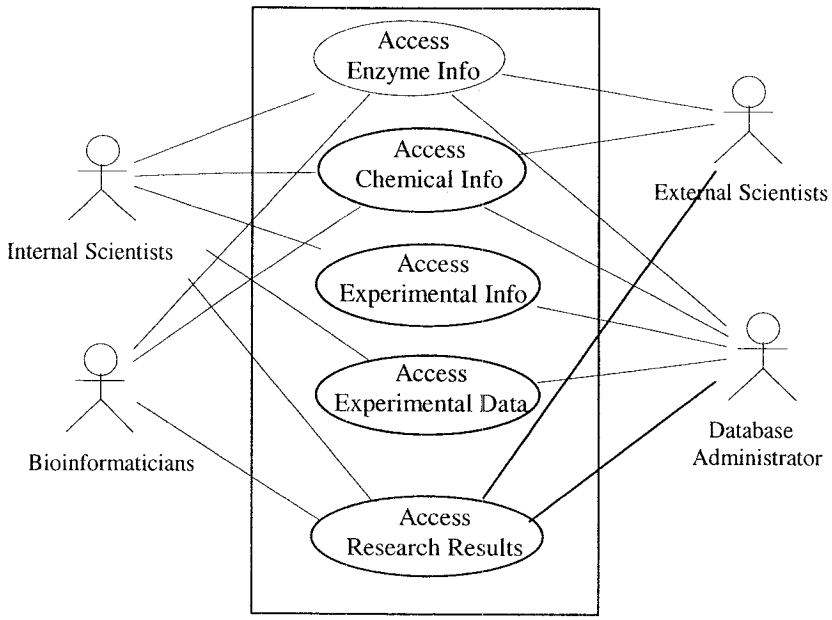


Figure 2.8 Use Cases of EAMDB of Ronghua's Design

2.5.2 Main Modules of the Database

Mr. Shu presented six modules: 1) enzyme activity mapping module, 2) enzyme module, 3) chemical compound module, 4) experiment module, 5) data module and 6) reference module. See Figure 2.9. Each module is composed of several tables that contain related information. In total, there are 17 tables. Among these modules, the enzyme activity-mapping module is a central part. It has two tables, which store the final results of the research work. The enzyme module contains three tables handling the information related to enzyme catalytic assay such as enzyme solution and enzyme stability. The chemical

compound module includes two tables storing information about substrates and products. The experiment module has four tables holding information about experiments. The data module contains four tables storing the information obtained from the enzyme catalytic assay experiments. Finally, the reference module has one table storing the literature sources cited in all other modules.

2.6 MySQL overview

MySQL developed by MySQL AB is one of the most popular SQL databases. The SQL part of "MySQL" stands for "structured query language" [23]. SQL is the most common standardized language used to access databases and is defined by the ANSI/ISO. The MySQL database server is very fast, reliable and easy to use and was developed to handle large databases. However, there are some faults. For example, it is not very good for visualizing large amounts of data. Version 3.23.49 that we are using does not support the subquery [36].

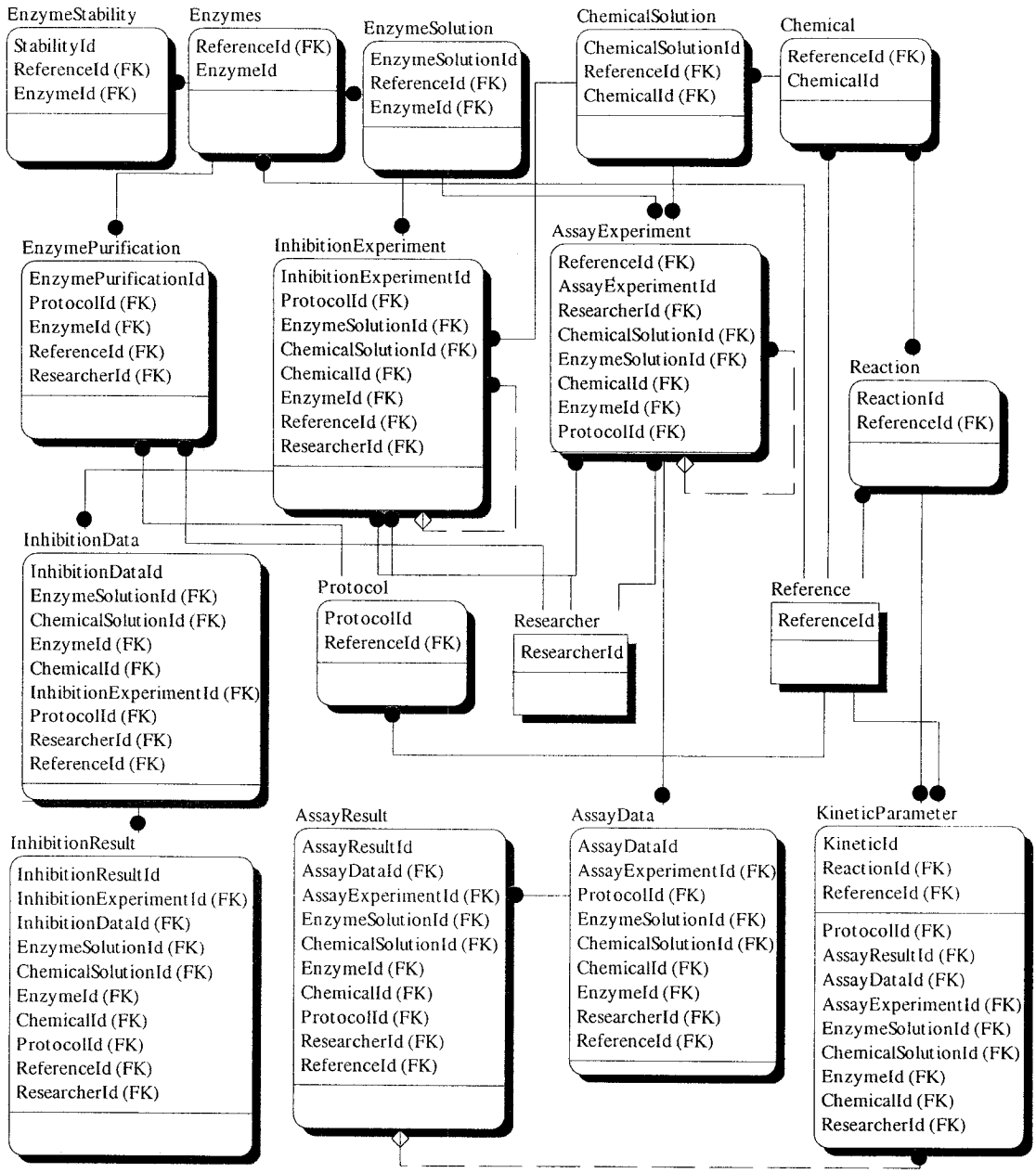


Figure 2.9 ER Model of EAMDB of Shu's Design

(For clarity, all the non-key attributes are omitted)

Chapter 3 The Detailed Requirements

3.1 Requirement Evolution

The requirements for database systems are always changing. One reason for this is that these systems are usually developed to address complex problems. Since the problem cannot be fully defined, the software requirements are bound to be incomplete. During the software process, the developer's understanding of the problem is constantly changing and these changes feed back to the requirements [33]. Please see Figure 3.1.

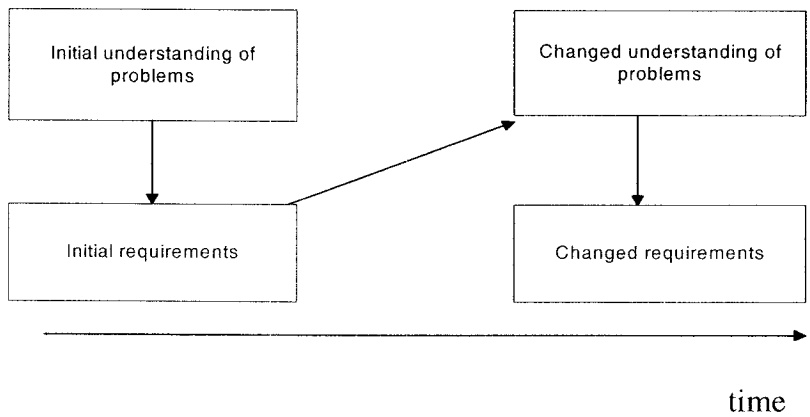


Figure 3.1 Requirements Evolution

The detailed requirements capture is the acquisition of the systems requirements in the endeavor to create a superior system. Most models of design stress the need for re-evaluation to ensure the product is evolving into the intended end product and verification that the functionality of the new system is being accurately realized [33]. System requirements can be grouped into: 1) functional requirements that specify what the system must do and 2) non-functional requirements specifying constraints on the

system. Thus, several techniques obtaining the detailed requirements are presented in the rest of this chapter.

3.2 Discuss with Internal Scientists

The first method of capturing detailed requirements is through discussion with internal scientists. This method allows the potential users to get their point of view across to the designer, while enabling the designer to query any unclear points and vice versa. This allows a greater insight into the enzyme activity assay experiments and their software. Moreover, this allows familiarization with the situation and problem, and potentially may allow some thoughts on a solution. As an example, researchers in the gene expression group go through passport file systems created by the internal bioinformatics group in the same project and select gene sequences of target enzymes according to the annotation of each sequences. Then, researchers amplify the DNA sequences of genes through PCR and culture hosts carrying and expressing these genes on different media (Term definitions are in Glossary). The researchers in the enzyme group get enzyme supernatants (samples) at different conditions. For instance, they may dilute the sample in digestion buffer, add BODIPY casein working solution, incubate samples protected from light, put each 96-well microplate containing different samples into a fluorometer and record fluorescence counts. Each experiment includes a positive control and two negative controls to account for background. All values of fluorescence count output are recorded in Excel files (Figure 3.2). We have created the relations in the database through understanding the experiment processes.

Another benefit of discussing with internal scientists is that one can get feedback about potential database schemas. After having applied techniques of capturing detailed requirements, one may document what is thought to be the necessary database schemas and ask the internal scientists to comment on any omissions, superfluous data or ambiguities. The feedback gained from this is invaluable.

3.3 Analyzing Data Files

An invaluable understanding of the problem also was gained through analyzing some data files, which show exactly what should be stored in the database. First, one looks at what information internal scientists have. Please see an example of an Excel file in Figure 3.2. In order to find target enzymes with proposed properties, they collect data on a variety of enzyme experiments, such as tests of enzyme activity, optimum temperature and pH. Eventually this information is stored into the database.

There are a number of things to note from this example file. First, it is a result of an experiment consisting of two cultures of the same gene sequence, Ccin3957. The experiment was carried out in duplicate. It also includes one positive control and two negative controls. The positive control uses the commercial enzyme trypsin as a source of comparison, and the resulting fluorescence count it produces is 287954 +/- 1312.5. The first one of two negative controls is cultures transformed with empty vector (ANEP2) to show what background values are. The second negative control is boiled cultured enzyme as proof that the observed activity disappears when the enzyme is denatured. A graph derived from data of Figure 3.2 indicates clearly significant activity of enzymes, Ccin3957a and Ccin3957b. See Figure 3.3.

27-08-2003

Detection of protease activity

Ccin 3957 (5 days culture of *A. niger* provide by Storm's Lab)
(Supernatants from Culture 1: 15 ml for each day)

Average	Time (day)				
Clones	1	2	3	4	5
Ccin3957a	56568	498354	352902	147848	213270
Boiled3957a	0	121895	91737	71679	69646
Ccin3957b	117232	537629	312351	172972	184558
Boiled3957b	24810	71250	87954	49251	43836
ANEP2b	4215	80466	65048	144262	215463
BoiledANEP2	2904	14575	24453	40073	41893

Clones	SD				
Ccin3957a	3612.6	2230.2	2255.7	585.5	585.5
Boiled3957a	40.3	4068.0	12788.7	2119.9	2119.9
Ccin3957b	207.2	7164.4	1539.4	636.4	636.4
Boiled3957b	173.2	4940.6	10888.0	2272.6	2272.6
ANEP2b	975.8	5268.7	10224.1	4681.0	108.2
Boiled ANEP2	978.6	353.6	1510.4	1656.8	651.2

Conditions:
Experiment was made in duplicate
Incubation at room temperature, 1h
Buffer used: 1X buffer (pH 7.8)
Final pH of the samples : pH 6.7
Positive control :
Trypsin (100 micro-g/ml MM-medium): 287 954 ± 1312.5 counts (Fluorescence)
Trypsin (10 mg/ml MM-medium): 419 839 ± 279.5 counts (Fluorescence)
Boiled samples were boiled 5 min at 100 degree C, using heat block

Comments:
The higher activity was obtained at day 2 for the 2 clones Ccin3957 a and b (6.2 and 6.7 times higher than the vector transformant only ANEP2, respectively).
Next Step:
Optimum pH determination

Figure 3.2 An Example of the Experimental Data

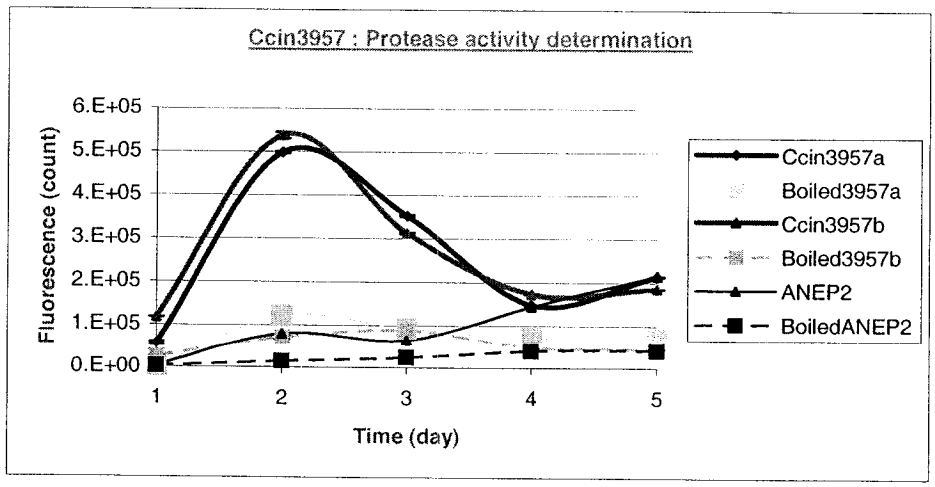


Figure 3.3 A Graph of Protease (Ccin3957) Activity Experiment

3.4 Use Case Driven Approach

3.4.1 Overview of Use Cases

A use case model consists of actors, use cases, and relations among them [33]. Actors represent everything that must exchange information with the system, including what are typically called users. When an actor uses the system, the system performs a use case. A good use case is a sequence of transactions that yields a measurable result of value for an actor. The collection of use cases is the system's complete functionality. The use case model is about describing what our system will do at a high-level.

3.4.2 Use Cases of the EAMDB

After analyzing requirements, we can see that use cases presented in Shu's design are not sufficient (Chapter 2). It lacked some cases such as loading the data although he described it in his thesis. Thus, based on his use cases model, we have modified and drawn the system use cases (Figure 3.4). We have modeled use cases from a high level [19]. For example, internal scientists load the experimental data information into the database. There are four groups of potential users of the database system: 1) internal scientists, 2) internal bioinformaticians, 3) a database administrator, and 4) public users. Nine use cases in the system include: 1) load SOP (standard operation procedures) data, 2) load experiment data, 3) load graph data, 4) generate reports, 5) view reports, 6) modify experiment data, 7) modify graph data, 8) retrieve experiment data, and 9) retrieve graph data. Each use case is described in following sections [30].

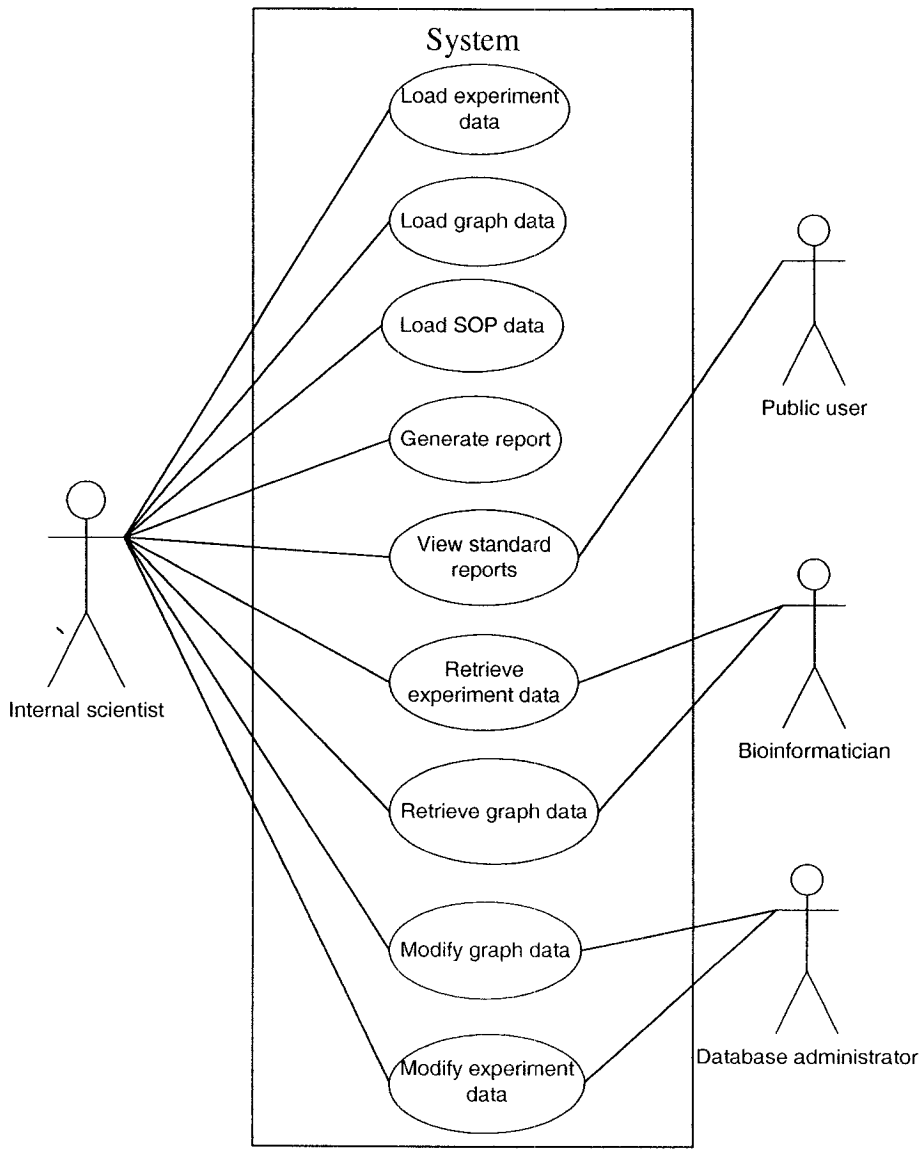


Figure 3.4 Use Case of EAMDB

3.4.2.1 Load SOP Data

This use case describes the process by which users inputs all the SOP data into the database.

Priority

This use case has high priority

Actors

Internal scientists

Precondition

A valid user logs in the system. The database is modifiable

Flow of events

Basic path

1. The use case starts when the user selects "Load SOP Data" on the screen.
2. The user types the data into related fields.
3. The user must submit by clicking the insert button.
4. The successful message will be displayed on the screen.
5. The user chooses other operations on the screen and use case ends.

Alternative path

None

Postcondition

Data has been inserted into the database.

The input window is closed.

3.4.2.2 Load Experiment Data

This use case describes the process by which the user inputs all the experiment data into the database.

Priority

This use case has high priority.

Actors

Internal scientists

Precondition

A valid user logs in the system. The database is modifiable.

Flow of events

Basic path

1. The use case starts when the user selects "Load Experiment Data" on the screen.
2. The user types the data into related fields.
3. The user must submit by clicking the insert button.
4. The successful message will be displayed on the screen.
5. The user chooses other operations on the screen and the use case ends.

Alternative path

None

Postcondition

Experiment data has been inserted into the database.

The input window is closed.

3.4.2.3 Load Graph Data

This use case describes the process by which the user inputs all graph data into the file system and the URI of a file into the database.

Priority

This use case has high priority

Actors

Internal scientists

Precondition

A valid user logs in the system. The database and File directory are accessible.

Flow of events

Basic path

1. The use case starts when the user converts a graph file into a HTML file.
2. The user saves the html file into a file directory for graphs.
3. The user selects the load "graph data" on the screen.
4. The user types the related URI into related fields.
5. The user must submit by clicking the insert button.
6. The successful message will be displayed on the screen.
7. The user chooses other operations on the screen and use case ends.

Alternative path

The user can select Cancel at any time before selecting the submit button. Data is not saved and use case ends.

Postcondition

A graph file has been saved and its URI is inserted into the database.

The input window is closed.

3.4.2.4 Generate Reports

This use case describes the process which standard result reports are generated by the users. Generally, the reports are experimental results.

Priority

This use case has middle priority among all of the use cases.

Actors

Internal scientists

Precondition

A valid user logs in the system.

The user has retrieved enough information from the database.

Flow of events

Basic path

1. The use case starts when the user writes reports according information getting through Retrieve Experiment Data use cases.
2. The user uploads the reports into the file system.
3. The user creates links between the report files and view report screen.
4. The use case ends when the user selects other operations.

Alternative path

None

Postcondition

A new report is added to the system.

3.4.2.5 View Reports

This use case describes the process by which the public user visits the system.

Priority

This use case has lower priority among all of the use cases.

Actors

Public users, internal scientists and bioinformaticians

Precondition

The user logins to the system successfully

Flow of events

Basic path

1. The use case starts when the user selects "View the reports" on the main screen.
2. The information is displayed on the screen.
3. The use case ends when the user chooses other operations.

Alternative path

None

Postcondition

Nothing has been changed in the database.

3.4.2.6 Retrieve Experiment Data

This use case describes the process through which the user queries information from the database.

Priority

This use case has middle priority.

Actors

Internal scientists and bioinformaticians

Precondition

The user logins successfully and the database is accessible.

Flow of events

Basic path

1. The use case starts when the user selects "Retrieve Experiment Data" on the screen.
2. The system display several options to be queried.
3. The user types specific criteria for selecting information
4. The information is displayed on the screen.
5. The use case ends when the user chooses another operation.

Alternative path

Step 4. If the records do not exist in the database go back to step 3.

Postcondition

Nothing is changed in the database

3.4.2.7 Retrieve Graph Data

This use case describes the process by which the user views the graph in the system

Priority

The use case has middle priority

Actors

Internal scientists and bioinformaticians

Precondition

The user logs in successfully and the database is accessible.

Flow of events

Basic path

1. The use case starts when the user selects "Retrieve Graph Data" on the screen.
2. The user inputs the graph path retrieved by the Retrieve Experiment Data use case.
3. The graph is displayed in the HTML file.
4. The use case ends when the user selects other operation.

Alternative path

If the user inputs a wrong file path, the system will display an error message. The user may go back to step2.

Postcondition

Nothing is changed in the database.

3.4.2.8 Modify Experiment Data

This use case describes the process by which the user updates or deletes the records in the database.

Priority

This use case has high priority.

Actors

Internal scientists and a database administrator

Precondition

The user has permission and the database is accessible.

Flow of events

Basic path

1. The use case starts when the user selects "Modify Experiment Data" on the screen.
2. The user needs to input information to specify which records are being modified.
3. The system displays the operation is successful.
4. The use case ends when the user selects other operation.

Alternative path

Step3. There are no records under this condition to be modified and go back to step 2.

Postcondition

The database has been changed.

3.4.2.9 Modify Graph Data

This use case describes the process by which the user changes or deletes the existing graph file.

Priority

The use case has high priority.

Actors

Internal scientists and a database administrator

Precondition

The user logs in successfully and has modification permission. The file system is accessible.

Flow of events

Basic path

1. The user case starts when the user selects "Modify Graph Data" on the screen.

2. The user accesses the file path getting from Retrieve Experiment Data use case.
3. The user opens the file.
4. The user replaces the graph or deletes the whole file.
5. The user saves the changes.
6. The use case ends when the user selects other operations.

Alterative path

If the file path is wrong and no file is found, The user goes back to step 2 after verifying the correct file path.

Postcondition

The file has been changed.

3.5 Query Approach

3.5.1 Internal Queries

Internal scientists are researchers who are working in the fungal genomics project. They are both the data source and users. As a result, they have full access right to the database. Their tasks are to load all of the necessary information into the database and to make use of information from the database guiding their research activities.

Internal bioinformaticians, working at CSFG, are adept at using computer software packages, such as Lucy, Phred, Phrap and BASE. These programs help with gene finding, gene annotation and managing vast quantities of data produced by the high-throughput experimental system. They also help sort through the vast mount of data to produce meaningful biological information. Internal bioinformaticians may query the EAMDB to find catalytic properties of enzymes and use them to verify the predicted 2 dimensional and 3 dimensional structures since structure relates to enzyme activity.

Moreover, bioinformaticians can query experimental data to obtain information on gene expression and enzyme activity conditions. They can use this information to update their annotation database. In short, they can query as much information as possible in order to understand the biology underlying the experimental processes and use the data in developing various software tools. After making use of information in the database, they can provide some feedback to the database administrator about modifying and maintaining the database.

A database administrator (DBA) has full control of the database. The DBA's activities include developing and implementing maintenance procedures. Thus, the DBA needs to ensure that the database is updated accurately and regularly. Also, the DBA should collaborate with internal scientists to meet new user needs and respond to technological innovations. Some other important aspects of the work are data integrity, backup and security. For example, a database administrator can add new tables or some attributes into existing tables based on requirements of users, such as internal scientists. Furthermore, The DBA can retrieve experimental data and view reports as well as internal scientists do. However, the DBA does not have responsibility concerning what data information should be stored in the database.

Some of the possible queries, which internal scientists, a database administrator and internal bioinformaticians might query the EAMDB, are listed below.

Query 1. What is used to link the protocol files?

Query 2. What experimental procedures are used in protease assay experiments?

Query 3. What are EC numbers and names for enzymes stored in Luc Varin lab?

Query 4. What are the catalytic constants of a given enzyme produced by a specific fungus?

Query 5. What are fluorescence counts for a two-day culture and the sequence ID of CcinSEQ3957 in the protease activity experiment?

Query 6. What is the culture information of the gene sequence PchrSEQ0108?

Query 7. What are the temperature and solvent stability of a given enzyme?

Query 8. What experiments have been done for enzymes produced by *Coprinus cinereus* in the laboratory?

Query 9. What are all of the enzymes produced by *Phanerochaete chrysosporium.*

Query 10. How does the graph look in pH determination for the enzyme of gene sequence CcinSEQ3957?

Query 11. Who has conducted the optimum pH determination experiment?

Query 12. What are chemical synonyms for N-Acetyl-D-glucosamine?

Query 13. Which articles reference trypsin from bovine pancreas in the reference relation?

Query 14. What are maximum and minimum temperatures for cellulase from *Aspergillus niger*?

Query 15. How many reagents are used in the lignin peroxidase protocol?

Query 16. What is the positive control in the inhibition of CcinSEQ3957 experiment?

Query 17. What are chemical names and category number for chemicals coming from the company Sigma?

Query 18. Which protocol is used in protease experiments?

Query 19. What is concentration of enzyme solution for chitinase from *Streptomyces griseus*?

Query 20. What is the inhibitor for the enzyme manganese peroxidase?

Query 21. Which media are used to culture the enzyme lipase?

Query 22. What sequence IDs have been transformed into hosts and cultured in the EAMDB?

Query 23. What are the enzyme names and EC and category numbers for enzymes coming from the Molecular Probes?

Query 24. What is the comment written in the experiment set of molecular characterization?

Query 25. What is the pH value in the enzyme solution for a-amylase?

Query 26. How many units of Beta-Glucosidase left in all laboratories?

Query 27. What method is used for purifying endoglucanase?

Query 28. What is the result for the enzyme of CcinSEQ3957 in the temperature experiment?

Query 29. What is the reaction equation for the degradation of sodium carboxymethyl cellulose?

Query 30. What enzymes have been tested in experiments?

3.5.2 Public views

Public users normally include external scientists, external bioinformaticians and general users. They can query the standard reports about the project through web interfaces when

the fungal genomics project is completed. For instance, they can view how many enzymes have activities with target substrates. However, some information about detailed experimental data is excluded from public view because experimental details may be patent-related. Generally, public users could view standard reports through user interfaces. More explanation is given at the section 5.7.

3.6 Iterative Process

An important issue in gathering detailed requirements is that they are often not all gathered at once. We need to repeat the processes sometimes for gathering as much detailed requirements as possible. Further functional requirements also may be uncovered throughout the developing process.

To summarize, in order to obtain detailed requirements, we have adopted several methods: 1) informal discussion with researchers; 2) analyzing some data files produced in the experiments; 3) using use cases approach and 4) analyzing possible queries. We analyze use cases of EAMDB from a high level. There are four types of users: internal scientists, internal bioinformaticians, the database administrator and public users. Nine use cases are Load SOP Data, Load Experiment Data, Load Graph Data, Generate Reports, View Reports, Query Experiment Data, Query Graph Data, Modify Experiment Data, and Modify Graph Data. Next, we listed 30 possible queries, which users would query the EAMDB. Based on obtained detailed requirements, we trace its impact at later steps of the database validation process.

Chapter 4 Validation of EAMDB Design

4.1 Impact of Detailed Requirements

From the above discussion, the type of information internal scientists are recording is known. Thus, enough relations (a relation and a table are not necessarily the same things, but we assume that they are in this thesis) should be designed to store these data according to the detailed requirements. In this section, tables are split, added, and deleted and attributes are modified in existing tables based on Shu's design.

4.1.1 Split Tables

In the experiment module, the assayExperiment table would not be sufficient because experiments can be grouped into experiment sets in actual lab processes. It suggests that we need to split the assayExperiment table into two tables: experimentSet and experiment to avoid data redundancy.

Biochemists currently conduct experiments using enzymes in the form of cultured supernatants instead of enzyme solutions. As a result, enzyme activity varies according to culture days and host. In order to account for this, four attributes: valueUnit, samplePH, positivControl and description are added into the experimentSet table instead of enzymeSolutionID, chemicalSolutionID. Similarly, the four attributes: sequenceID, result, numberSample and comment are inserted into the experiment table. SequenceID is used to trace back the DNA sequence and other properties of the enzyme. A result column stores data about whether the expected goal is achieved in an experiment (Table 4.1, 4.2 and 4.4). In order to verify the design, A record table is listed for the

experimentSet and experiment tables, respectively (Table 4.3 and 4.5). In addition, loading data into the field defined as a primary key in each table is mandatory, and any one of other fields can be optional field in inserting data. However, user should input as much data as possible in order to store complete information.

Biochemists also like to store graphs derived from each experiment. Graphs can intuitively show how significant the enzyme activities are. See Figure 3.4. Thus, we add an attribute fileURI to link with a graph file in the experiment table. Similarly, we add fileURI in experimentSet table to link with experiment data file. FileURI consists of a restricted set of characters, which consists of digits, letters and a few graphic symbols. For example, this URI, http://www.w3.org/Icons/WWW/w3c_main.gif, identifies a file that can be accessed using the web protocol application, Hypertext Transfer Protocol ("http://") that is housed on a computer named"www.w3.org"(which can be mapped to a unique internet addresses). In the computer's directory structure, the file is located at the pathname of "/icons/WWW/w3c_main.gif" [2] [28]. The graph and data files are stored in the separate file system. The whole system layout will be presented in Chapter 5.

Table 4.1 List of Fields in the AssayExperiment Table (Shu's Design)

Field Name	Data type	Description
assayExperimentId	VARCHAR(20)	primary key
experimentTitle	VARCHAR (100)	description of the goal of the experiment
date	DATE	experiment date
researcherId	VARCHAR (20)	foreign key from People table
protocolId	VARCHAR (20)	foreign key in the Protocol table
enzymeSolutionId	VARCHAR (20)	foreign key in the EnzymeSolution table
chemicalSolutionId	VARCHAR (20)	foreign key in the ChemicalSolution table
layoutFileName	VARCHAR (20)	the file name of the layout file
layoutFilePath	VARCHAR (20)	the file path of the file
dataFileId	VARCHAR (20)	foreign key in the DataFile table
relatedExperimentId	VARCHAR (30)	list of ExperimentId related experiments.

Table 4.2 List of Fields in the ExperimentSet Table

Field Name	Data Type	Description
experimentSetID	INT(7)	primary key, unsigned integer and not zero
experimentSetTile	VARCHAR(45)	description of specific experiment set
host	VARCHAR(20)	name of host
procedureID	INT(7)	foreign key from the procedure table
researcherID	INT(3)	foreign key from the researcher table
description	TEXT	brief description of experiment
controlEnzymeID	INT(7)	commercial enzyme used as comparison
fileURI	VARCHAR(80)	identify a file in a unique internet address

Table 4.3 A Record in the ExperimentSet Table

Field Name	Data
experimentSetID	1
experimentSetTile	detection of protease activity
host	Aspergillus niger
procedureID	1
researcherID	1
description	step 1, next step is optimum pH determination
controlEnzymeID	EN10
fileURI	http://www.cs.concordia.ca/~l_fu/experiment/Protease03-9-27.htm

Table 4.4 List of Fields in Experiment Table

Field Name	Data Type	Description
experimentID	INT (7)	primary key, unsigned integer not null and zero
experimentTitle	VARCHAR(30)	objective of experiment, i.e., looking for optimum pH
experimentSetID	INT(7)	foreign key from the experiment set table
date	DATE	experiment date
sequenceID	VARCHAR(15)	foreign key from the sequence table
result	VARCHAR(20)	general result of experiment, i.e., significant activity
numberSample	INT(1)	how many repeats, not null, not zero
description	TEXT	describing experiment conditions
comment	TEXT	any special observation from an experiment
positiveControl	TEXT	description result of positive control enzyme under same conditions
fileURI	VARCHAR(80)	unique string to identify a file in internet

Table 4.5 A Record in the Experiment Table

Field Name	Data
experimentID	1
experimentTitle	protease activity of CcinSEQ3957
experimentSetID	1
date	2003-08-27
sequenceID	CcinSEQ3957
result	obtain high activity at day 2
numberSample	2
description	Incubate at room temperature 1 hour. buffer used:1Xbuffer (pH7.8), final pH of the samples: pH6.7. Boiled samples were boiled 5 minutes at 100 degree Celsius, using heat block.
comment	5 days cultures, the high activity was obtained at day 2 for the 2 clones Ccin3957a and b (6.2 and 6.7 times higher than the vector transformant only ANEP2 respectively.
positiveControl	Trypsin (100 micro-g/ml MM-medium):287954+/-1312.5 fluorescence counts, Trypsin (10 mg/ml MM-medium):419839+/-279.5 fluorescence counts
fileURI	http://www.cs.concordia.ca/~l_fu/experiment/Ccin3957activity.htm

In the protocol module, users can store the metadata of protocols to the protocol table while real protocol files will be stored in a separate file system. The metadata is information about the data. Also, the protocol files are converted to HTML files to enable users to get them through the web site. Thus, in the protocol table, we add fileURI instead of protocol description column. Moreover, in protocol table (Shu's design), we eliminated some attributes, such as targetchemical and characterizationMethod, because we could not find the information in the protocol documents provided by users. Nonetheless, we added source, effectiveDate, numberReagent and comments to describe protocols (Table 4.6 and 4.7).

Moreover, procedures from laboratories suggest that we need to set up another two tables: labProcedure and procedureOrder to handle the needs in experiments. The labProcedure table includes labProcedureID, type, content and protocolID. Type can be

any one of procedure, child procedure (subprocedure) and step (Table 4.9). The protocol and labProcedure record tables are also given (Table 4.8 and 4.10).

Table 4.6 List of Fields in the Protocol Table (Shu's Design)

Field Name	Data Type	Description
ProtocolId	VARCHAR (20)	primary key
ProtocolName	VARCHAR (50)	name of the protocol
TargetEnzyme	VARCHAR (50)	target enzyme name
TargetChemical	VARCHAR (50)	target chemical name
Cofactor	VARCHAR (50)	cofactor used in the assay
Coenzyme	VARCHAR (50)	coenzyme used in the assay
Activator	VARCHAR (50)	activator used in the assay
CharacterizationMethod	VARCHAR (50)	measurement method for the kinetic data
ProtocolDescription	TEXT	detailed procedure of the protocol
ReferencesId	VARCHAR (20)	original source of the protocol

Table 4.7 List of Fields in the Protocol Table

Field Name	Data Type	Description
protocolID	VARCHAR(3)	primary key, unsigned integer, not null, not zero
protocolName	VARCHAR(50)	name of each protocol
source	VARCHAR(15)	where the protocol comes from
effectiveDate	DATE	last update
numberReagent	INT(2)	how many reagents used in this protocol
comment	TEXT	general description of protocol
fileURI	VARCHAR(70)	unique string to identify a file in internet

Table 4.8 A Record in the Protocol Table

Field Name	Data
protocolID	1
protocolName	EnzChek Protease Assay Kits
source	Molecular Probes
effectiveDate	2003-01-14
numberReagent	4
comment	Ensure that the substrate is fully reconstituted before use. Avoid freeze-thaw cycles after reconstituting. Storage upon receipt: -20 ^{0}C, desiccate, protect from light
fileURI	http://www.cs.concordia.ca/~l_fu/protocol/protocolprotease.htm

Table 4.9 List of Fields in the LabProcedure Table

Field Name	Data Type	Description
procedureID	INT(5)	primary key, id type, unsigned integer, no zero
type	VARCHAR(10)	one of procedure, sub-procedure, step
content	TEXT	description of each biochemical step
protocolID	VARCHAR(3)	foreign key from the protocol table

Table 4.10 A Record in the LabProcedure Table

Field Name	Data
procedureID	1
type	procedure
content	Using the EnzChek Protease Assay Kit for green fluorescence (E-6638)
protocolID	1

The procedureOrder table includes procedureID, childProcedureID and orderOf (Table 4.11). The attribute orderOf stores a number which a child procedure is listed under its parent procedure. For example, there are two records: the procedure 1 (using the EnzChek Protease Assay Kit for green fluorescence) and procedure 7 (to incubate the samples for one hour). However, procedure 7 is the 6^{th} step in procedure 1, so field orderOf should be 6 (Table 4.12). Thus, labProcedure and procedureOrder tables could provide sufficient information when users like to retrieve each step of a specific procedure.

Table 4.11 List of Fields in the ProcedureOrder Table

Field Name	Data Type	Description
procedureID	INT(5)	foreign key from procedure table
childprocedureID	INT(5)	ID of subprocedure or step
orderOf	INT(2)	order of each sub-procedure or step

Table 4.12 A Record in the ProcdureOrder Table

Field Name	Data
procedureID	1
childprocedureID	7
orderOf	6

4.1.2 Add Tables

In data module, the type of data internal scientists are recording from the above data file is known. For example, one value of fluorescence counts is related to two different parameters: the condition and the clone. Based on this observation and assayData table of Shu's design, two addition tables should be added: condition table and clone table in order to store complete information about each value. The condition table stores information about different experimental conditions and the clone table stores data on different clone samples. Their ids (conditionID and cloneID) are foreign keys in the experimentData table. In the experimentData table, an attribute experimentID is used to link back to the experiment table. Otherwise, the value without the related information is meaningless (Table 4.13, 4.14, 4.16 and 4.18). The experimentData, clone and condition record tables are given (Table 4.15, 4.17 and 4.19).

Table 4.13 List of Fields in the AssayData Table (Shu's Design)

Field Name	Data type	Description
AssayDataId	VARCHAR (20)	primary key
ExperimentId	VARCHAR (20)	foreign key from experiment table in the experiment module
DataFileName	VARCHAR (30)	file name of the data file
DataFilePath	VARCHAR (50)	directory path of the data file

Table 4.14 List of Fields in the ExperimentData Table

Field Name	Data Type	Description
experimentID	INT(7)	foreign key from the experiment table
conditionID	INT(4)	foreign key from the condition table
cloneID	INT(7)	foreign key from the clone table
value	FLOAT	an average of fluorescence counts

Table 4.15 A Record in the ExperimentData Table

Field Name	Data
experimentID	1
conditionID	1
cloneID	1
value	56568

Table 4.16 List of Fields in the Clone Table

Field Name	Data Type	Description
cloneID	INT(7)	primary key, not null, unsigned int, not zero
cloneName	VARCHAR(20)	name of clone from the culture process
cultureID	INT(7)	foreign key from the culture table

Table 4.17 A Record in the Clone Table

Field Name	Data
cloneID	1
cloneName	CcinSEQ3957a
cultureID	1

Table 4.18 List of Fields in the Condition Table

Field Name	Data Type	Description
conditionID	INT(4)	primary key, unsigned int, not null, not zero
name	VARCHAR(25)	name indicating which type test
condition	VARCHAR(10)	different samples

Table 4.19 A Record in the Condition Table

Field Name	Data
conditionID	1
name	average counts
condition	Day 1

In the experiment module, because internal scientists would query the database in numerous ways, we must provide the information to achieve query abilities. Enzymes being tested in the experiments are coming from culturing. That is, internal scientists clone target genes into a particular plasmids (i.e., ANEP2), transform plasmids into appropriate hosts (i.e., *Aspergillus niger*), and culture the host in a certain medium. Internal scientists get a certain amount of enzyme produced by the host. So, we should add culture and media tables, which cover the above information. (Table 4.20 and 4.22)

In the culture table, culture data and harvest data are important because at different days, target enzymes produced by a host could show diverse characteristics. Moreover, sequenceID and procedurelID are used to link back to the sequence table and procedure table, respectively. In the media table, the component stores each compound and its concentration in a given media (Table 4.21 and 4.23).

Table 4.20 List of Fields in the Media Table

Field Name	Data Type	Description
cultureID	INT(7)	foreign key from the culture table
name	VARCHAR(30)	name of media
component	VARCHAR(40)	media component

Table 4.21 A Record in the Media Table

Field Name	Data
cultureID	50
name	Glycerol-complex Medium(GCM)
component	1% yeast extrac

Table 4.22 List Fields in the Culture Table

Field Name	Data Type	Description
cultureID	INT(7)	primary key, not null unsigned integer, not zero
name	VARCHAR(35)	name of culture give information about which gene is inserted into host
sequenceID	VARCHAR(15)	trace back gene sequence, letter and number
procedureID	INT(5)	foreign key from the procedure table
cultureVolume	VARCHAR(10)	description of culture volume
startVolume	VARCHAR(10)	amount of volume when it start
startConidia	FLOAT	amount of conidia
host	VARCHAR(20)	name of host
stockConcentration	FLOAT	concentration of stock
harvestDate	DATE	date harvest
cultureDate	DATE	start date
endDate	DATE	end date
researcherID	INT(3)	foreign key from the researcher table

Table 4.23 A Record in the Culture Table

Field Name	Data
cultureID	1
name	one cultured host with CcinSEQ3957
sequenceID	CcinSEQ3957
procedureID	8
cultureVolume	200 ml
startVolume	2000 ul
startConidia	2e+08
host	Aspergillus niger
stockConcentration	1e+08
harvestDate	2003-08-08
cultureDate	2003-09-02
endDate	2003-09-05
researcherID	11

In order to store all the information of users who will access the EAMDB, the user table is added (Table 4.24). The user should provide a legal e-mail address since the login name and password are e-mailed to the user by the DBA when they registered into the database. The user also should indicate that the user type is an internal user or public user because the DBA can set up access permissions according to it (Table 4.25). The user can log in after receiving the login name and password by an e-mail.

Table 4.24 List of Fields in the User Table

Field Name	Data Type	Description
loginName	VARCHAR(8)	name that the user logs in
firstName	VARCHAR(20)	first name of the user
lastName	VARCHAR(20)	last name of the user
e-mail	VARCHAR(40)	e-mail of the user
password	VARCHAR(8)	password when the user logs in
comfirmPassword	VARCHAR(8)	same as password
workPlace	VARCHAR(25)	where the user works
address	VARCHAR(50)	address of the user
city	VARCHAR(20)	city name
postCode	VARCHAR(10)	post code
academic	Boolean	academic users (1) or commercial users (0)
internal	Boolean	internal users (1) or public users (0)

Table 4.25 A Record in the User Table

Field Name	Data
loginName	l_fu
firstName	Longchang
lastName	Fu
e-mail	l_fu@cs.concordia.ca
password	Addrrse5
comfirmPassword	Addrrse5
workPlace	Concordia University
address	1455 de Maisonneuve Blvd West
city	Montreal
postCode	H3G 1M8
academic	1
internal	1

4.1.3 Delete Tables

The significant changes made based on impact of requirements are that inhibitionExperiment, inhibitionData and inhibitionResult tables were deleted. The inhibition experiment table is redundant because all information stored in this table can be placed into experimentSet and experiment tables without changing any attributes. Compare Table 4.1 with Table 4.26.

Table 4.26 List of Fields in the InhibitionExperiment Table (Shu's Design)

Field Name	Data type	Description
InhibitionExperimentId	VARCHAR(20)	primary key
ExperimentTitle	VARCHAR (100)	brief description of the goal of the experiment
Date	DATE	experiment date
ResearcherId	VARCHAR (20)	foreign key from Researcher table
ProtocolId	VARCHAR (20)	foreign key in the Protocol table
EnzymeSolutionId	VARCHAR (20)	foreign key in the EnzymeSolution table
SubstrateSolutionId	VARCHAR (20)	referenced to ChemicalSolutionId in ChemicalSolution table
InhibitorSolutionId	VARCHAR (20)	referenced to ChemicalSolutionId in ChemicalSolution table
LayoutFileName	VARCHAR (20)	the file name of the layout file
LayoutFilePath	VARCHAR (20)	the file path of the file
RelatedExperimentId	VARCHAR (30)	list of ExperimentId of the related experiments.

In short, we have added 8 tables and deleted 3 tables in Shu's design in order to store the information of enzyme assay sufficiently. The eight tables are lab procedure, procedure order, condition, clone, culture, experiment set, media and user. The deleted tables are inhibitionExperiment, inhibitionData and inhibitionResult.

4.1.4 Modify Attributes

4.1.4.1 Chemical Compound Module

Several attributes in the chemical table are deleted from Shu's design since the focus of the project is searching the specific properties of certain enzymes produced by fungi. Attributes deleted are detailed descriptions of chemicals so that it is not necessary to store them in the EAMDB. These attributes are IUPACName, synonym, structureFomula, class, stereCenter, functionalGroup, boilingPoint, meltingPoint nmr, and ir. However, source, category number, and storageTempID were added into the table (Table 4.27 and 4.28). We also added two attributes, procedureID and concentrationUnit, in the chemicalSolution table since chemical solutions are prepared differently according to procedures used. However, there is not any record in the table (Table 4.29).

Table 4.27 List of Fields in the Chemical Table

Field Name	Data Type	description
ChemicalID	VARCHAR(10)	primary key, not null the number of characters is not more than 10.
chemicalName	VARCHAR(60)	name of chemical
molecularFormula	VARCHAR(20)	molecular formular of chemical
molecularWeight	FLOAT	weight of molecular
CASNumber	VARCHAR(15)	register number of compound
source	VARCHAR(20)	where the chemical is from
categoryNo	VARCHAR(10)	category number
storageTempID	INT(3)	foreign key from storage temp table

Table 4.28 A Record in the Chemical Table

Field Name	Data
ChemicalID	CH1
chemicalName	amplex red reagent
molecularFormula	C14H11NO4
molecularWeight	257.25
CASNumber	119171-73-2
source	Molecular Probes
categoryNo	A-12222
storageTempID	6

Table 4.29 List of Fields in the ChemicalSolution Table

Field Name	Data Type	Description
chemicalSolutionID	INT(7)	primary key, not null, unsigned integer, not zero
chemicalID	VARCHAR(10)	foreign key form the chemical table
procedureID	INT(5)	foreign key from labProcedure table
solvent	VARCHAR(20)	solvent used for solution
pH	FLOAT	pH value in the solution
buffer	VARCHAR(20)	buffer used to prepared solution
concentration	FLOAT	concentration of chemical
concentrationUnit	VARCHAR(10)	unit of concentration

4.1.4.2 Enzyme Module

The attributes of three tables are modified according to the detailed requirements. First, we add three attributes: researcherID, date and comment; and change protocolID to procedureID in enzymePurification table. The reasons are that the wet lab scientists usually like to write a conclusion about an experiment following a certain procedure and date the experiment. Please see Table 4.30. Secondly, in the enzymeSolution table, procedureID is added since the enzyme solution is prepared by following certain

procedures (Table 4.31). Lastly, we do not change any attribute for the enzymeStability table. However, these three tables do not contain records yet (Table 4.32).

Table 4.30 List of Fields in the EnzymePurification Table

Field Name	Data Type	Description
EnzymePurification ID	INT(5)	primary key, not null, unsigned integer, not zero
enzymeID	VARCHAR(10)	foreign key from the enzyme table
method	TEXT	method used in the purification
procedureID	INT(5)	foreign key from the labProcedure table
comment	TEXT	comment about experiment
yield	FLOAT	Amount of enzyme purified
researcherID	INT(3)	who does this experiment, foreign key
date	DATE	date the experiment is done

Table 4.31 List of Fields in the EnzymeSolution Table

Field Name	Data Type	Description
enzymeSolutionID	INT(5)	primary key, not null, unsigned int, not zero
enzymeID	VARCHAR(10)	foreign key from the enzyme table
procedureID	INT(5)	foreign key from the procedure table
pH	FLOAT	pH of enzyme solution
buffer	VARCHAR(30)	buffer used in the solution
stabilizer	VARCHAR(30)	stabilizer used in the solution
solvent	VARCHAR(30)	solvent used in the solution
concentration	FLOAT	how much enzyme per unit
ConcentrationUnit	VARCHAR(10)	unit chosen to express concentration

Table 4.32 List of Fields in the enzymeStability Table

Field Name	Data Type	Description
enzymeStabilityID	INT(7)	primary key, not null, unsigned int, not zero
enzymeID	VARCHAR(10)	foreign key from the enzyme table
pHStability	FLOAT	suitable PH value for reaction
tempStability	FLOAT	suitable temperature for catalysis
generalStability	TEXT	information about stability
solventStability	TEXT	suitable solvent to stabilize the enzyme
oxidationStability	TEXT	stability in the presence of oxidizing agents
storageStability	TEXT	condition for storage

4.1.4.3 Enzyme Mapping Module

In the reaction table, referenceID is replaced by enzymeID for two reasons: 1) all object ids having references are stored in the reference table it avoids that one reaction has two or more reference problems and 2) enzymeID is added as a foreign key to join enzyme table with reaction table easily. Two tables in this module are kineticParameter and reaction tables. The attributes deleted in the kineticParameter table are pHHigh, pHLow, tempHigh, tempLow, K_mRoom, K_{cat}Room and referenceID since lab scientists conduct only experiments for finding optimum pH, and temperatures, and the reference table contains objectID to which a record is referring. Table 4.33 and Table 4.34 do not show a data example since records are not yet available.

Table 4.33 List of Fields in the KineticParameter Table

Field Name	Data Type	Description
kineticID	INT(7)	primary key, not null, unsigned integer, not zero
enzymeID	VARCHAR(10)	Foreign key from the enzyme table
chemicalID	VARCHAR(10)	Foreign key from the chemical table
pHOptimum	FLOAT	Optimum pH value for catalysis
tempOptimum	FLOAT	Optimum temperature for the enzyme catalysis
inhibitor	VARCHAR(30)	Inhibitor used to stop enzyme catalysis
cofactor	VARCHAR(30)	Cofactor used in the reaction
activator	VARCHAR(30)	Activator used in the reaction
KmOptimum	FLOAT	Km under optimum condition
KcatOptimum	FLOAT	Kcat under optimum condition
specificActivity	TEXT	Unit to express the amount of enzyme
reactionID	INT(7)	Foreign key from the reaction table
resultID	INT(7)	Foreign key from the result table

Table 4.34 List of Fields in the Reaction Table

Field Name	Data Type	Description
reactionID	VARCHAR(7)	primary key, not null, unsigned int, not zero
reactionType	VARCHAR(20)	classification of the reaction
substrateID	VARCHAR(10)	foreign key, main substrate in the reaction
productID	VARCHAR(10)	foreign key, main product in the reaction
enzymeID	VARCHAR(10)	foreign key, enzyme catalysis
reactionEquation	VARCHAR(250)	equation of the reaction
comment	TEXT	indicate some information about the reaction

4.1.4.4 Reference Module

The referedID column added into the reference table is used to store an object ID among enzymeID, chemicalID and reactionID (Table 4.35). A record is displayed in the Table 4.36. The researcher table has not been changed (Table 4.37 and 4.38).

Table 4.35 List of Fields in the Reference Table

Field Name	Data Type	Description
referenceID	INT(7)	primary key not null, unsigned int
referedID	VARCHAR(10)	foreign key, any object referenced
title	TEXT	title of the literature
author	VARCHAR(100)	list of authors
publishDate	DATE	date published
source	VARCHAR(60)	title of journal or book
volume	VARCHAR(10)	journal volume
page	VARCHAR(10)	which pages in the literature

Table 4.36 A Record in the Reference Table

Field Name	Data
referenceID	2
referedID	EN10
title	Molecular Kinetic Properties of Crystalline Disopropyl Phosphoryl Trypsin
author	Cunningham, et al
publishDate	1954
source	The Journal of Biological Chemistry
volume	211
page	13-19

Table 4.37 List of Fields in the Researcher Table

Field Name	Data Type	Description
researcherID	INT(3)	Primary key, not null, unsigned int, not zero
name	VARCHAR(25)	Name of researcher
groupName	VARCHAR(25)	goup name researcher belongs to

Table 4.38 A Record in the Researcher Table

Field Name	Data
researcherID	1
name	Regis-Olivier Benech
groupName	enzyme group

4.1.4.5 Data Module

GraphFileName and graphFilePath are replaced by fileURI for reasons explained above. Two attributes, experimentID and sequenceID, are added into this table because the result is coming from an experiment exploring a gene sequenceID. The data type are changed from varchar(10) to float for Km, Kcat, and Vmax. The data type of ResultID is changed to INT(7) because the maximum number of research result is less than 10 million. Please see Table 4.39.

Table 4.39 List of Fields in the Result Table

Field Name	Data Type	Description
resultID	INT(7)	Primary key, not null, unsigned int, not zero
experimentID	INT(7)	Foreign key, result comes from
sequenceID	VARCHAR(15)	Foreign key, gene sequence of enzyme
Km	FLOAT	Km value
Kcat	FLOAT	Kcat value
Vmax	FLOAT	Vmax value
fileURI	VARCHAR(80)	The path to retrieve the result file

4.1.5 Data Type

Varchar(N) is used as a data type very often in our database design. What is the varchar(N)? Varchar is one kind of nominal data type that indicates the value of the attribute is not a number but a name [7]. There is no implied ordering of attribute values or magnitude associated with the values. Attribute values of varchar type cannot serve as operands in any arithmetic operation. On the other hand, one can use the attributes values in equality tests, such as, if enzymeName = "Trypsin", then, ... and inequality tests. In other words, varchar is a domain constraint, which restricts the values of an attribute to be drawn from the set of possible variable-length strings. The appropriate operators, such as "||"(concatenate) and SUBSTR (substring), can legally be applied to values of varchar type. For example, one can perform character string operation on chemical name, but on chemical id defined as integer type. Moreover, varchar and other data types can be defined depending on the intended meaning or semantics of attributes, not on the way values of that type happen to be physically represented [11].

In MySQL, one can declare a varchar column to be any length between 1 and 255. Varchar values are stored using only as many characters as needed, plus one byte to record the length. Values are not padded, since trailing spaces are removed when values are stored. No case conversion takes place during storage or retrieval [36].

Furthermore, the length of varchar is defined based on the domain specifics. For example, one can look at Nomenclature Enzyme data on the NC-IUBMB web site, and note the maximum character length of an enzyme name is less than 75. Thus, the length of the enzyme name is defined as 75, varchar(75). For other varchar type columns, length is given according domains of the fungal genomics project. Namely, enzymes from 14

fungi are being examined. If 4,000 enzymes are produced by each fungus on average, then there might be about 60,000 enzymes totally to be examined. Thus, enzymeID is defined as varchar(10), in which two letter ('EN') indicate enzyme and rest is 8 digits or less. Similarly, experimentID is defined as INT(7).

The data types are also modified when compared to Shu's design. Generally speaking, most id types are changed from varchar(20) to integer(7) except for chemicalID, enzymeID, reactionID and protocolID. These ids refer to the same referedID column in the reference table. It may be a problem if these ids are integer types. Also, data types of attributes, in which their domain constraints restrict the values of attributes to be drawn from the set of approximate numeric values, are changed from varchar to float. Furthermore, data types of some attributes are changed from varchar (longer than varchar(200)) to text. The text type, which stores strings up to 2^{16} characters, is more suitable for an attribute like comment or description.

4.2 Table Normalization

The theoretical rules that the design of a relation meets are known as normal forms [18]. Each normal form represents an increasingly stringent set of rules. There are six nested normal forms, which are first normal form, second normal form, third normal form, Boyce-Codd normal form, fourth normal form, and fifth normal form. The higher the normal form, the better the design of the relation. In most cases, if our relations are placed in third normal form (3NF), then they will avoid most of the problems common to bad relational designs. Boyce-Codd (BCNF) and fourth normal form (4NF) handle special situations that arise occasionally. Fifth normal form is a complex set of criteria

that is extremely difficult to work with. It is very difficult to verify that a relation is in 5NF. Most practitioners do not bother with 5NF, knowing that if their relations are in 3NF, then their designs are generally problem free.

In order to create a functional and efficient database, thoroughly examine and eliminate redundancy and inconsistent dependency in table design by the rules of normalization.

4.2.1 First Normal Form

A table is in first normal form (1NF) if it meets the following criteria: the data are stored in a two-dimension table with no repeating groups. A repeating group is directly analogous to a multi-valued attribute in an E-R diagram. From previous design, there are a couple of tables containing a multi-value attribute. For example, in enzyme table, synonym and referenceID are two multi-value attributes. As we know, one enzyme may have several synonyms and be referenced by several articles. See Table 4.40.

Table 4.40 List Fields in the Enzyme Table (Shu's Design)

Field Name	Data type	Description
enzymeId	VARCHAR(20)	primary key.
enzymeName	VARCHAR (100)	the name used by our lab. It may come from references.
ECNumber	VARCHAR (20)	EC number of the enzyme
recommendedName	VARCHAR (100)	recommended name of the enzyme
synonym	VARCHAR (100)	other names used for the enzyme.
casNumber	VARCHAR (20)	CAS registry number
class	VARCHAR (50)	class of the enzyme
comment	VARCHAR (200)	description of storage temperature, specific activity, and physical form
unitDefinition	VARCHAR (200)	how the activity unit is defined.
source	VARCHAR (150)	the origin of the enzyme
referenceId	VARCHAR (20)	foreign key from reference table

Consequently, this will render searching the table difficult. A solution to this problem is to get rid of the repeating group altogether. We create another table called enzymeSynonym to handle multiple instances of the repeating groups. Please see Table 4.41, 4.42, 4.43 and 4.44.

Table 4.41 Fields in the Enzyme Table

Field Name	Data Type	Description
enzymeID	VARCHAR(10)	primary key, not null, the character number is no more than 10.
enzymeName	VARCHAR(75)	name of enzyme used
ECNumber	VARCHAR(15)	Enzyme commission number of IUBMB
CASNumber	VARCHAR(15)	CAS registry number
source	VARCHAR(20)	the origin of the enzyme
categoryNo	VARCHAR(10)	number of category
StorageTempID	INT(3)	foreign key from the storageTemp table
unitDefinition	TEXT	how activity unit is defined
comment	TEXT	description of specific activity or physical form

Table 4.42 A Record in the Enzyme Table

Field Name	Data
enzymeID	EN1
enzymeName	a-Amylase from Aspergillus oryzae
ECNumber	3.2.1.1
CASNumber	9000-85-5
source	Sigma
categoryNo	A6211
StorageTempID	1
unitDefinition	One unit will liberate 1.0 mg of maltose from starch in 3 min at pH 6.9 at 20
comment	150-250 units/mg protein (Biuret)

Table 4.43 List Fields in EnzymeSynonym Table

Field Name	Data Type	Description
enzymeID	Varchar(10)	foreign key from the enzyme table
synonym	Varchar(50)	different synonyms with same enzyme

Table 4.44 A Record in the EnzymeSynonym Table

Field Name	Data
enzymeID	EN1
synonym	1,4-a-D-Glucan-glucanohydrolase porcine pancreas

Table 4.45 List of Fields in the ChemicalSynonym Table

Field Name	Data Type	Description
chemicalID	VARCHAR(10)	foreign key from the chemical table
synonym	VARCHAR(30)	Synonym name

Table 4.46 A Record in the ChemicalSynonym Table

Field Name	Data
chemicalID	CH3
synonym	5-Cholesten-3b-ol 3-oleate

Similarly, we create a chemicalSynonym table to store all synonyms of chemicals. ChemicalID is foreign key from the chemical table. See Table 4.45 and 4.46.

However, relationship of enzyme and reference is one to many [21]. Therefore, we change reference direction, namely, include the foreign key, enzymeID, in the reference table so that it handles one enzyme is referenced by many articles.

Storage temperature is another multi-value attribute in the enzyme table since storage temperature can have a range of values, such as 0^0C to 5^0C. Thus, we break down into two tables: enzyme and storageTemp tables. We changed storageTemp to storageTempID in enzyme table and created storageTempID, minimumTemp and MaximumTemp three attributes in the storageTemp table. See Table 4.47 and 4.48.

Table 4.47 Fields in the StorageTemp Table

Field Name	Data Type	Description
storageTempID	INT(3)	primary key, unsigned int, not null, not zero
minimumTemp	INT(3)	the lowest temperature for storage, Celsius unit
maximumTemp	INT(3)	the highest temperature for storage, Celsius unit

Table 4.48 A Record in the StorageTemp Table

Field Name	Data
storageTempID	2
minimumTemp	2
maximumTemp	8

4.2.2 Second Normal Form

Second normal form is defined as follows: the relation is in first normal form and all nonkey attributes are functionally dependent on the entire primary key. Each relation was checked according to rules of second normal form. All relations are in 2NF.

4.2.3 Third Normal Form

Third normal definition is that the relation is in second normal form and there are no transitive dependencies. After respecting all the tables, we find transitive dependencies in

the enzyme and chemical tables. Namely, quantity is functionally dependent on enzymeID because professor's lab (location) is functionally dependent on enzymeId and quantity is functionally dependent on professor's lab. A table called location is created with primary keys of labName and objectID. Quantity and unit are the other attributes. See Table 4.49 and 4.50. Therefore, this removes the transitive dependency and its associated anomalies. All relations are in third normal form.

Table 4.49 Fields in the Location Table

Field Name	Data Type	Description
labName	VARCHAR(25)	primary key, name of lab
objectID	VARCHAR(10)	primary key from object table
quantity	FLOAT	amount of enzyme or chemical
unit	VARCHAR(10)	unit of object

Table 4.50 A Record in the Location Table

Field Name	Data
labName	Justin Powlowski lab
objectID	EN1
quantity	200000
unit	units

4.2.4 Boyce-Codd Normal Form and Fourth Normal Form

To be in BCNF, a relation must meet the following rule: the relation is in third normal form and all determinants are candidate keys. Similarly, the rule for fourth normal form is that the relation is in Boyce-Codd normal form and there are no multivalued dependencies. BCNF and 4NF are designed to handle relations that exhibit a special characteristic that does not arise too often. We have examined our database schemas

carefully. Fortunately, all relational schemas in the EAMDB are fourth normal form relations. Therefore, it is now possible to insert and delete any data without a modification anomaly.

Through the above validation steps, the database schemas have been refined as Figure 4.1 shows. Each rectangle stands for a relation in the EAMDB. An arrow connecting two rectangles stands for a relationship between two relations. The relation which the arrowhead points to is referenced by the relation on the other side of the arrow. However, for clarity, a few arrows are not drawn in the schemas, such as one connection beween procedure and enzymePurification. From top to bottom in the rectangle, a table name is written in the first line. The primary key of the table is written in the second field (PK indicates primary key). All nonkey attributes are written in third field.

Normalizing relations in the database usually separates entities into their own relations and makes it possible for users to enter, modify and delete data without disturbing entities other than the one directly being modified. However, when relations are split, relationships are represented by matching primary and foreign keys. We force the DBMS to perform matching operations between relations whenever a query requires data from more than one table. The fourth section of this chapter will further validate our EAMDB by using a set of queries.

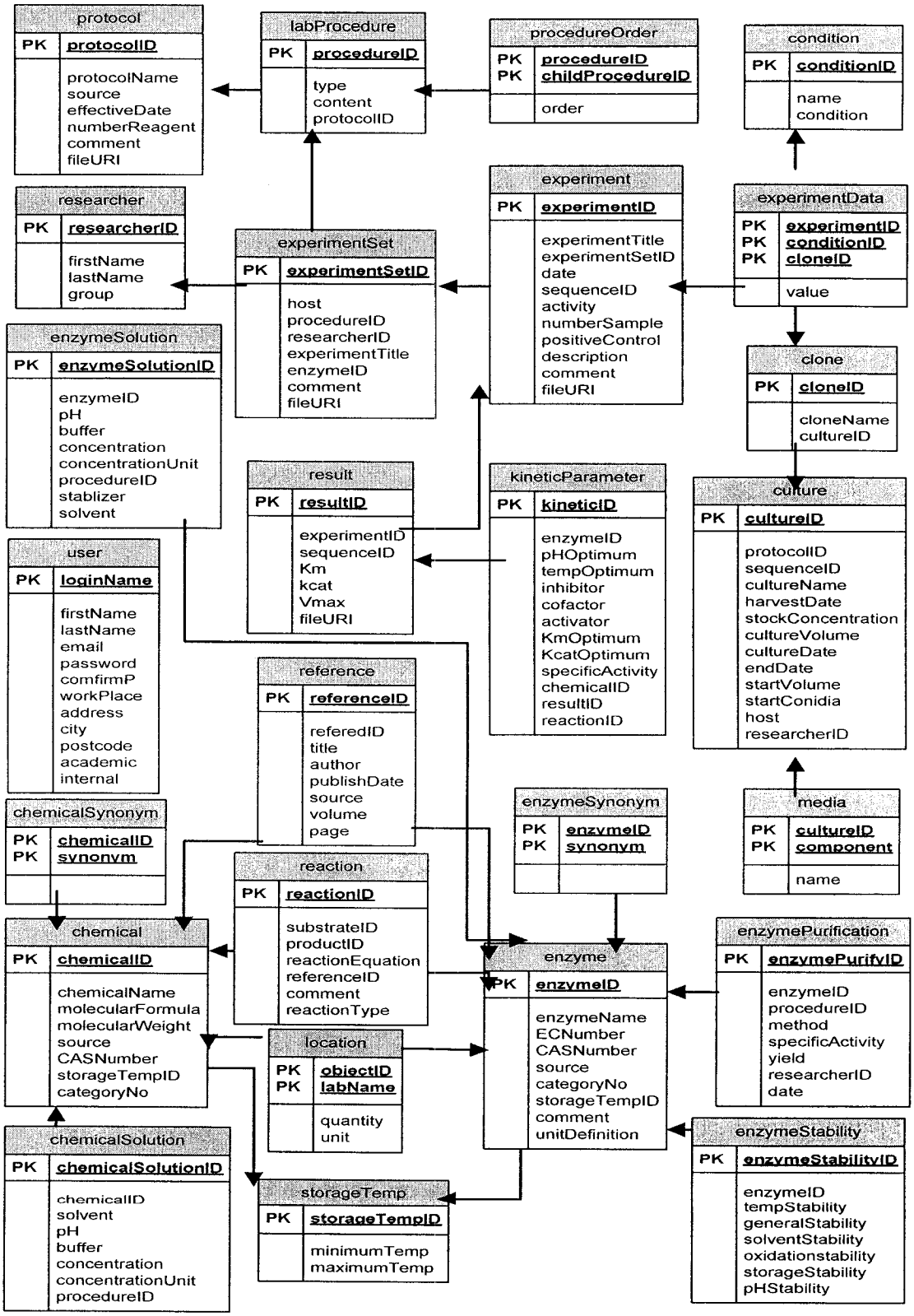


Figure 4.1 The EAMDB Schemas after Validation

4. 3 Query Ability

After having normalized relations, the EAMDB can be queried. First, database schemas are converted to SQL schemas, which can be found in appendix B. Then, each relation is populated with certain records. All queries listed in Chapter 3 (Page37) and some SQL function queries have been implemented to test whether EAMDB can answer these queries. [36] However, a few queries don't have any outputs because some tables do not have data. The reasons are explained in the next chapter.

4.3.1 Implement Queries Listed in Chapter 3

For all of the given queries, queries are numbered first corresponding to the numbers in Chapter 3 and explanations about queries are given. Then, the SQL sentence and outputs from the database are presented.

Query 1. What is used to link to protocol files? This query selects all data of the fileURI column from the protocol table in EAMDB. The protocol ID is a primary key indexed. Moreover, the table does not contain many records because experiments do not ever require hundreds of protocols. Thus, the query is answered quickly.

```
mysql> select fileURI from protocol;
+-------------------------------------------------------------------------------+
| fileURI                                                                       |
+-------------------------------------------------------------------------------+
| http://www.cs.concordia.ca/~grad/l_fu/protocol/protocolprotease.htm           |
| http://www.cs.concordia.ca/~grad/l_fu/protocol/protocolcellulase.htm          |
| http://www.cs.concordia.ca/~grad/l_fu/protocol/protocolLignin.htm             |
| http:/www.cs.concordia.ca/~grad/l_fu/protocol/protocolPectin.htm              |
+-------------------------------------------------------------------------------+
4 rows in set (0.00 sec)
```

Query 2. What are the experimental procedures used in a specific protease experiment? This query retrieves procedures for a protease experiment from the lab procedure and procedure order tables. The equijoin between tables is based on indexed and integer attributes, procedureID in labProcedure table and childProcedureID in procedureOrder table, respectively. The size of the dataset in the table also is small. Thus, this query is done efficiently.

```
mysql> select P.orderOf, L.content from labProcedure L, procedureOrder P  where (
P.procedureID = '1' and L.type = 'step' and L.procedureID = P.childProcedureID);
+---------+-------------------------------------------------------------------------------------------------+
| orderOf | content
+---------+------------------------------------------------------------------------------------------- --+
|       1 | Prepare 1X digestion buffer (PBS, pH7.8): dilute 2.5 ml of the 20X digestion
          buffer with dH2O to a final volume of 50 ml.                                                    |
|       2 | Prepare a 1.0 mg/ml stock solution of the BODIPY FL casein by adding 0.2 ml
          of dH2O.                                                                                        |
|       3 | Prepare a 10 micro-g/ml working solution of the BODIPY casein: Add 0.2 ml of
          the stock solution prepared in step 2 to one of the vials containing the lyophilized
          substrate.                                                                                      |
|       4 | To detect enzyme activity in supernatant, add 100 micro-l of the supernatant to
          100 micro-l of the BODIPY casein working solution prepared in step 3 for
          microplate assay.                                                                               |
|       5 | Incubate the samples for one hour, protected from light.                                          |
|       6 | Read the fluorescence in a fluorescence microplate reader.                                        |
+---------+-------------------------------------------------------------------------------------------------+
6 rows in set (0.00 sec)
```

Query 3. What are the EC numbers and names for enzymes stored in Luc Varin lab? This query retrieves all enzyme names and EC numbers from the enzyme table. These enzymes are located at the Luc Varin Lab. The equijoin between enzyme and location tables is based on indexed attributes, enzymeID and objectID. This query is done effectively.

```
mysql> select E.enzymeName, E.ECNumber  from enzyme E, location L where
E.enzymeID = L.objectID and labName = 'Luc Varin(room530-3)';

+-----------------------------------------------------------+-----------+
| enzymeName                                                | ECNumber  |
+-----------------------------------------------------------+-----------+
| Cellulase Aspergillus niger                               | 3.2.1.4   |
| Cellobiohydrolase II from Trichoderma longibrachiatum     | 3.2.1.91  |
| exo-1,3-beta-Glucanase from Trichoderma sp.               | 3.2.1.58  |
| Endoproteinase Glu-C Staphylococcus aureus V8             | 3.4.21.19 |
| Thermolysin Bacillus thermoproteolyticus rokko            | 3.4.24.27 |
+-----------------------------------------------------------+-----------+
5 rows in set (0.00 sec)
```

Query 4. What are the catalytic constants of a given enzyme produced by a fungus? This query retrieves constants for a specific enzyme from the kinetic parameter table. The join between the enzyme and kinetic parameter tables are indexed attributes, enzymeID in both tables. Unfortunately, this table does not have any records yet.

```
mysql> select E.enzymeName, K.pHOptimum, K.tempOptimum, K.KmOptimum,
K.KcatOptimum from enzyme E, kineticParameter K where E.enzymeID = K.enzymeID
and enzymeName ='manganese peroxidase';

Empty set (0.00 sec)
```

Query 5. What are fluorescence counts for a two-day culture and the sequence ID of CcinSEQ3957 in the protease activity experiment? This query retrieves the information from four tables: experiment, experimentData, condition, and clone. They all joined by the indexed and integer attributes (cloneID, conditioned, and experimented). Despite the four tables are joined, it is done efficiently.

```
mysql> select C.condition, L.cloneName, E.value from condition C, clone L,
experimentData E, experiment X where (L.cloneID = E.cloneID and C.conditionID =
E.conditionID and C.name = 'average counts of day' and C.condition = '2' and
X.experimentID = E.experimentID and X.experimentTitle like '%activity of
CcinSEQ3957');
```

```
+-----------+-------------------------+---------+
| condition | cloneName               | value   |
+-----------+-------------------------+---------+
| 2         | CcinSEQ3957a            | 498354  |
| 2         | boiled CcinSEQ3957a     | 121895  |
| 2         | CcinSEQ3957b            | 537629  |
| 2         | boiled CcinSEQ3957b     |  71250  |
| 2         | ANEP2b                  |  80466  |
| 2         | boiled ANEP2b           |  14575  |
+-----------+-------------------------+---------+
6 rows in set (0.00 sec)
```

Query 6. What is the culture information for a specific gene sequence? This query selects information from the culture table for the gene sequence ID of PchrSEQ0108. It queries the single table, and the size of dataset of the table is small. It is done effectively.

```
mysql> select cultureID, sequenceID, host, startVolume, startConidia, cultureDate,
endDate from culture where sequenceID = 'PchrSEQ0108';
+-----------+----------------+---------+-------------+-------------+-------------+-------------+
| cultureID | sequenceID     | host    | startVolume | startConidia | cultureDate | endDate     |
+-----------+----------------+---------+-------------+-------------+-------------+-------------+
|         6 | PchrSEQ0108    | A.niger | 800ul       |       1e+08 | 2003-09-04  | 2003-09-09  |
|         7 | PchrSEQ0108    | A.niger | 1250ul      |       1e+08 | 2003-09-04  | 2003-09-09  |
+-----------+----------------+---------+-------------+-------------+-------------+-------------+
2 rows in set (0.00 sec)
```

Query 7. What are the temperature and solvent stability of a given enzyme? This query retrieves temperature and solvent stability information for lignin peroxydase. It queries from two tables: enzyme and enzymeStability. The join between the tables is based on enzymeID indexed. It should be done effectively. However, there are no data in the enzymeStability table. Thus, an empty set is returned.

```
mysql> select E.enzymeName, S.tempStability, S.solventStability from enzymeStability
S, enzyme E where E.enzymeID = S.enzymeID and  enzymeName = 'Lignin peroxydase';

Empty set (0.00 sec)
```

Query 8. What are all of the experiments that have been done for enzymes coming from *Coprinus cinereus* in the laboratory? This query retrieves what kinds of experiments have been done through gene sequences of *Coprinus cinereus*, which has string 'Ccin' in the middle of sequenceID. It queries the experiment table indexed on experiment title. It is done efficiently.

```
mysql> select experimentTitle from experiment where experimentTitle like '%Ccin%';
+----------------------------------------+
| experimentTitle                        |
+----------------------------------------+
| protease activity of CcinSEQ3957       |
| protease activity of CcinSEQ4030       |
| pH determination of CcinSEQ3957        |
| pH determination of CcinSEQ4030        |
| temperature of CcinSEQ3957             |
| Inhibition of CcinSEQ3957              |
+----------------------------------------+
6 rows in set (0.00 sec)
```

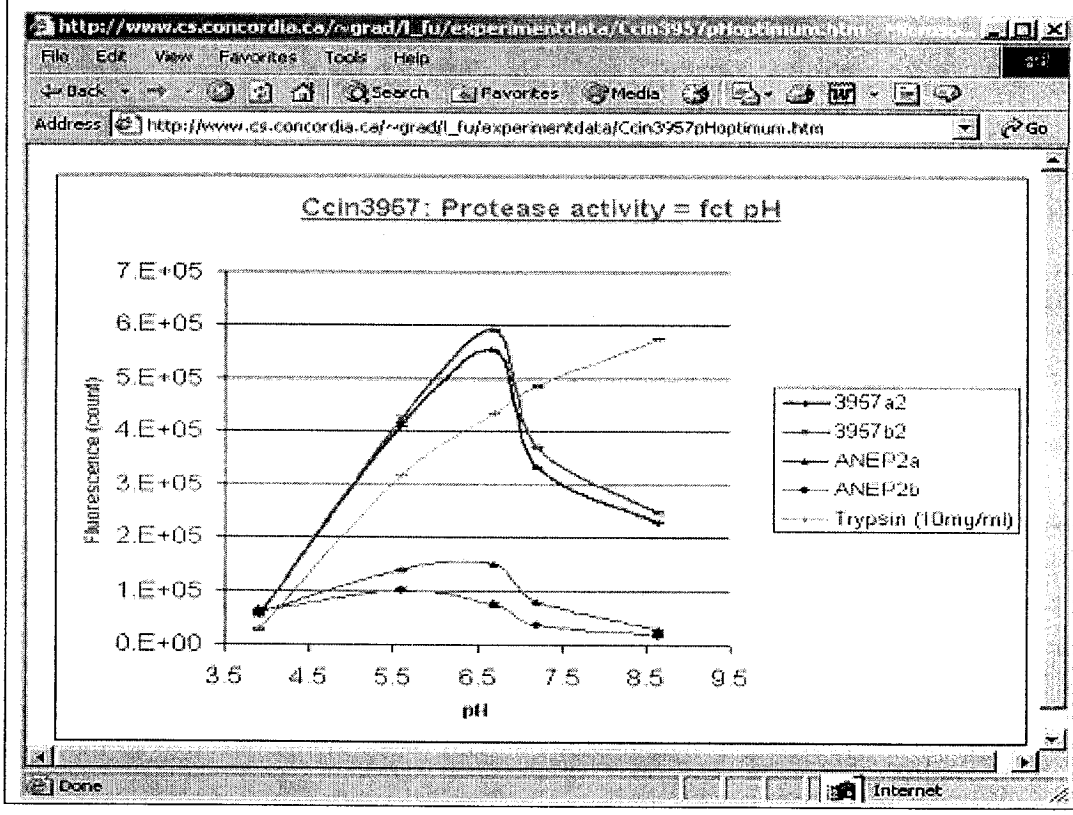
Query 9. What are all the enzymes produced by a specific fungus? This query retrieves all enzymes of a given fungus. However, in the enzyme table, all enzymes are commercial enzymes. Target enzymes of our experiments have not been inserted yet. So, the result of query is empty set. This query only accesses the enzyme table and enzyme name is indexed.

```
mysql> select enzymeName, source from enzyme where source = 'Gloeophyllum
trabeum';
Empty set (0.00 sec)
```

Query 10. How the graph look in the pH determination experiment for a specific enzyme? This query retrieves the graph of the pH determination experiment for the gene sequence CcinSEQ3957. It first queries the experiment table to get the file URI of the graph, then shows the graph through the Web by that URI. The query accesses the

experiment table, and the search attribute experimentTitle is indexed. So, the query is executed very fast.

```
mysql> select fileURI from experiment where sequenceID = 'CcinSEQ3957'and
experimentTitle like 'pH determination%';
+-----------------------------------------------------------------------------------------+
| fileURI                                                                                 |
+-----------------------------------------------------------------------------------------+
| http://www.cs.concordia.ca/~grad/l_fu/experimentdata/Ccin3957pHoptimum.htm |
+-----------------------------------------------------------------------------------------+
1 row in set (0.00sec)
```



Query 11. Who has conducted the optimum pH determination experiment? Here, the query retrieves the name of the researcher who has conducted the optimum pH determination experiment. The equijoin attribute, researcherID, is integer and indexed in the researcher table. Moreover, there are not many records in the tables. So, the query is answered quickly.

```
mysql> select R.name, R.groupName from researcher R, experimentSet E where
E.researcherID = R.researcherID and E.experimentSetTitle = 'Optimum pH
determination';

+------------------------+-----------------+
| name                   | groupName       |
+------------------------+-----------------+
| Regis-Olivier Benech   | enzyme group    |
+------------------------+-----------------+
1 row in set (0.00 sec)
```

Query 12. What are chemical synonyms for N-Acetyl-D-glucosamine? The query retrieves the chemical name and chemical synonyms from chemical and chemicalSynonym tables, respectively. It uses the attribute chemicalID to join two tables, and chemicalID is indexed. So, it is done efficiently.

```
mysql> select C.chemicalName, S.synonym from chemical C, chemicalSynonym S
where C.chemicalName = 'N-Acetyl-D-glucosamine' and C.chemicalID = S.chemicalID;

+---------------------------+-------------------------------------+
| chemicalName              | synonym                             |
+---------------------------+-------------------------------------+
| N-Acetyl-D-glucosamine    | 2-Acetamido-2-deoxy-D-glucose       |
| N-Acetyl-D-glucosamine    | D-GlcNAc                            |
+---------------------------+-------------------------------------+
2 rows in set (0.00 sec)
```

Query 13. Which article references trypsin from bovine pancreas in the reference relation? This query retrieves the title of the reference article for the enzyme called trypsin from bovine pancreas. The join keys are referedID and enzymeID indexed in their tables. Search key, enzymeName in the enzyme table, is indexed. Thus, the query is done effectively.

```
mysql> select R.title from reference R, enzyme E where E.enzymeID = R.referedID and
E.enzymeName = 'Trypsin from bovine pancreas';

+-----------------------------------------------------------------------------+
| title                                                                       |
+-----------------------------------------------------------------------------+
| Molecular kinetic properties of crystalline diisopropyl phosphoryl trypsin. |
+-----------------------------------------------------------------------------+
1 row in set (0.00 sec)
```

Query 14. What are the maximum and minimum temperatures for the Cellulase from *Aspergillus niger*? The query retrieves the maximum and the minimum temperatures of storage for the enzyme called cellulase from *Aspergillus niger*. The equijoin key, storageTempID, is a primary key in the storageTemp table, and there are not many records in the table. So, the query is done effectively.

```
mysql> select enzymeName, minimumTemp, maximumTemp from enzyme E,
storageTemp S where S.storageTempID = E.storageTempID and enzymeName =
'Cellulase Aspergillus niger';

+-----------------------------------+-----------------+-----------------+
| enzymeName                        | minimumTemp | maximumTemp |
+-----------------------------------+-----------------+-----------------+
| Cellulase from Aspergillus Niger |               2 |               8 |
+-----------------------------------+-----------------+-----------------+
1 row in set (0.00 sec)
```

Query 15. How many reagents are used in the lignin peroxidase protocol? A number of reagents are used to define a specific enzyme activity in a protocol. This query retrieves the number of reagents from the protocol table. The protocol name is indexed so that the query is answered effectively.

```
mysql> select protocolName, numberReagent from protocol where protocolName =
'lignin peroxidase';

+-------------------+-------------------+
| protocolName      | numberReagent |
+-------------------+-------------------+
| Lignin Peroxidase |                 4 |
+-------------------+-------------------+
1 row in set (0.00 sec)
```

Query 16. What is the positive control in the inhibition experiment of CcinSEQ3957? The positive control is used in inhibition experiments. This query retrieves the description of the positive control for enzyme of CcinSEQ3957 in the experiment table. SequenceID is indexed so that it is done efficiently.

```
mysql> select positiveControl from experiment where sequenceID = 'CcinSEQ3957' and
experimentTitle like 'inhibition%';

+-----------------------------------------------------------------------------------------------------+
| positiveControl
+-----------------------------------------------------------------------------------------------------+
| Supernatants of the clones Ccin3957a and b from culture 1: Ccin3957a : 472 260 +/-
 1523 counts (Fluorescence) Ccin3957b : 529 181 +/- 6630 counts (Fluorescence).          |
+-----------------------------------------------------------------------------------------------------+
1 row in set (0.00 sec)
```

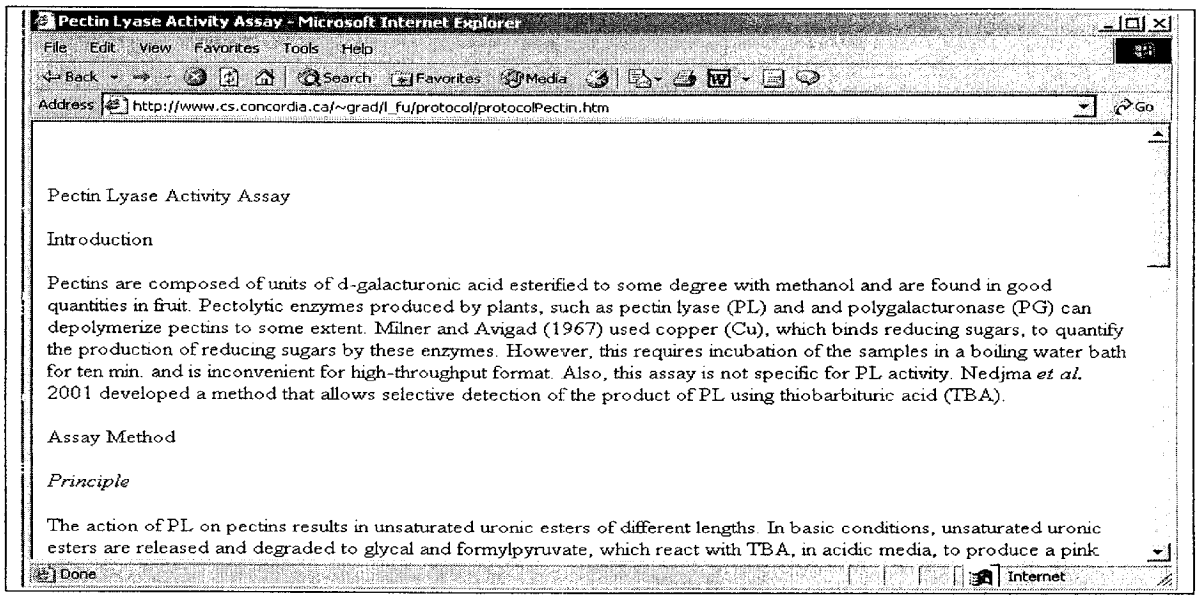
Query 17. What are the chemical names and category numbers for chemicals coming from company Sigma? These chemicals are brought from different companies. This query retrieves all chemicals from the chemical table where source is equal to Sigma. Chemical name is indexed. So, the query is done fast.

```
mysql> select chemicalName, categoryNo from chemical where source = 'Sigma';
+-----------------------------------------------------------------+------------+
| chemicalName                                                    | categoryNo |
+-----------------------------------------------------------------+------------+
| Cholesteryl oleate                                              | C9253      |
| Chitin crab shells                                              | C7170      |
| Chitin crab shells                                              | C9752      |
| Digalacturonic acid                                             | D4288      |
| 3-Methyl-2-benzothiazolinone hydrazone hydrochloride hydrate    | M8006      |
| N-Acetyl-D-glucosamine                                          | A8625      |
| Polygalacturonic acid sodium salt                               | P3850      |
| Starch, soluble                                                 | S9765      |
| 4-Nitrophenyl b-D-glucopyranoside                               | N7006      |
| 4-Nitrophenyl b-D-cellobioside                                  | N5759      |
| 2-Nitrophenyl ?-D-galactopyranoside                             | N-1127     |
| Laminarin from Laminaria digitata                               | L9634      |
| Xylan from birch wood                                           | X0502      |
| 4-Nitrophenyl b-D-cellobioside                                  | N5759      |
| Xylan from beechwood                                            | N4252      |
| Xylan from birch wood                                           | X0502      |
| Xylan oat spelts                                                | X0627      |
+-----------------------------------------------------------------+------------+
17 rows in set (0.00 sec)
```

Query 18. Which protocol is used in pectin lyase activity assay experiments? This query retrieves file path of pectin lyase protocol and then users can get real protocol file from the identical file path. ProtocolName is indexed so that it is answered quickly.

```
mysql> select fileURI from protocol where protocolName like 'pectin%';

+-----------------------------------------------------------------+
| fileURI                                                         |
+-----------------------------------------------------------------+
| http:/www.cs.concordia.ca/~grad/l_fu/protocol/protocolPectin.htm |
+-----------------------------------------------------------------+
1 row in set (0.00 sec)
```



Query 19. What is the concentration of enzyme solution for Chitinase from *Streptomyces griseus*? This query retrieves the concentration of chitinase from the enzyme solution table. However, there is no record in this table now. The answer is empty set.

```
mysql> select E.concentration from enzymeSolution E, enzyme N  where E.enzymeID =
N.enzymeID and N.enzymeName like 'Chitinase%';

Empty set (0.01 sec)
```

Query 20. What is the inhibitor for the enzyme manganese peroxidase? This query retrieves the inhibitor from the kineticParameter table, which equijoin key is enzymeID indexed in enzyme and kineticParameter tables. However, there is no record in the kineticParameter table, so the answer is empty set.

```
mysql> select E.enzymeName, K.inhibitor from enzyme E, kineticParameter K where
E.enzymeName = 'manganese peroxidase' and E.enzymeID = K.enzymeID;

Empty set (0.00 sec)
```

Query 21. Which media are used to culture enzyme for the CcinSEQ484 gene? This query retrieves the media names from the media table. However, it needs to join two tables of culture and media because one culture may be dependent on several media. The equijoin key cultureID indexed and there are not many records in the media table so that the query is answered fast.

```
mysql> select DISTINCT C.sequenceID, M.name from culture C, media M where
C.cultureID = M.cultureID and C.sequenceID = 'CcinSEQ484';

+--------------+-----------------------------------------+
| sequenceID   | name                                    |
+--------------+-----------------------------------------+
| CcinSEQ484 | Glycerol-complex Medium (GCM)   |
| CcinSEQ484 | Methanol-complex Medium (MCM)  |
+--------------+-----------------------------------------+
2 rows in set (0.00 sec)
```

Query 22. What sequence ids have been transformed into hosts and cultured in the EAMDB? The user would like to see all cultured sequenceIDs decide how to culture the transformant of a new gene sequence. This query selects all of sequenceID in the culture table. CultureID is indexed, so this query should be done efficiently.

```
mysql> select sequenceID from culture;
+----------------+
| sequenceID     |
+----------------+
| CcinSEQ3957 |
| CcinSEQ3957 |
| null           |
| CcinSEQ4030 |
| CcinSEQ3957 |
| PchrSEQ0108 |
| PchrSEQ0108 |
| CcinSEQ484   |
+----------------+
8 rows in set (0.00 sec)
```

Query 23. What are the enzyme names and category numbers for enzymes coming from Molecular Probes? After getting category number of Molecular Probes, the user uses it to search related information easily. This query is simply a selection from the enzyme table.

```
mysql> select enzymeName, categoryNo from enzyme where source = 'Molecular
Probes';

+-------------------------------+--------------+
| enzymeName                    | categoryNo |
+-------------------------------+--------------+
| Protease-Enzcheck assay kit | E6638        |
+-------------------------------+--------------+
1 row in set (0.00 sec)
```

Query 24. What is the comment written in the experiment set of molecular characterization? In each experiment set the researcher writes down the special comments. This query retrieves the comment from the experimentSet table, however, it takes a little more time.

```
mysql> select comment from experimentSet where experimentSetTitle = 'molecular
characterization';

+-----------------------------------------------------------------------------------------------+
| comment
+-----------------------------------------------------------------------------------------------+
| Next step will be concentration of the supernatants of the clone Ccin3957b
expressed in A. niger, using amicon then gel electrophoresis (SDS-PAGE). Perform
the protease assays with the use of the supernatants of the six clones of Ccin3957
expressed by Pichia pastoris, provided by Tsang Lab (Linda): P. Pastoris is known to
do not produce any excreted proteases                                          |
+-----------------------------------------------------------------------------------------------+
1 row in set (0.04 sec)
```

Query 25. What is the pH value in enzyme solution for α-amylase? This query retrieves a pH value from the enzymeSolution table. It joins the two tables of enzyme and

enzymeSolution with key enzymeID indexed. Nonetheless, there is no record in the table, so the answer is empty set.

```
mysql> select E.enzymeName, S.pH from enzyme E, enzymeSolution S where
E.enzymeID = S.enzymeID and E.enzymeName = 'a-amylase';
Empty set (0.00 sec)
```

Query 26. How much b-Glucosidase from almond is left in all laboratories? The user would check how much enzyme is left in the labs and then decide to buy some if needed. The query selects the quantity and unit from the location table. The join keys are enzymeID in the enzyme table and objectID indexed in the location table, so the query is done effectively.

```
mysql> select E.enzymeName, L.quantity, L.unit from enzyme E, location L where
E.enzymeID = L.objectID and E.enzymeName = 'b-Glucosidase from almonds';

+-------------------------------+----------+-------+
| enzymeName                    | quantity | unit  |
+-------------------------------+----------+-------+
| b-Glucosidase from almonds    |      100 | units |
+-------------------------------+----------+-------+
1 row in set (0.00 sec)
```

Query 27. What method is used for purifying endoglucanase? This query selects the method used in endoglucanase purification. Nevertheless, there is no record in the enzyme purification table. So, result of query is empty set.

```
mysql> select E.enzymeName, P.method from enzyme E, enzymePurification P where
P.enzymeID = E.enzymeID and E.enzymeName = 'endoglucanase';
Empty set (0.00 sec)
```

Query 28. What is the result for enzyme of CcinSEQ3957 in the temperature experiments? For each experiment, the researcher draws some conclusions. This query selects the conclusion of the experiment for enzyme of sequence CcinSEQ3957. The

query is performed effectively because experimentTitle is indexed and there are not many records in the table.

```
mysql> select result from experiment where sequenceID = 'CcinSEQ3957' and
experimentTitle like 'temperature%';
+-----------------------------------------+
| result                                  |
+-----------------------------------------+
| 45 degree Celsius with weak activity    |
+-----------------------------------------+
1 row in set (0.00 sec)
```

Query 29. What is the reaction equation for sodium carboxymethyl cellulose? The query retrieves the reaction equation from the reaction table. However, there is not any record in the table, so the answer is empty set.

```
mysql> select R.reactionEquation from reaction R, chemical C  where
C.chemicalName = 'Sodium carboxymethyl cellulose' and R.substrateID =
C.chemicalID;
Empty set (0.01 sec)
```

Query 30. What are sequences tested in experiments? The user likes to see all gene sequenceIDs in experiments. This query just selects all sequenceIDs in the experiment table, and experimentID is indexed so that the query is answered efficiently.

```
mysql> select sequenceID from experiment;
+----------------+
| sequenceID     |
+----------------+
| CcinSEQ3957 |
| CcinSEQ4030 |
| CcinSEQ3957 |
| CcinSEQ4030 |
| CcinSEQ3957 |
| CcinSEQ3957 |
+----------------+
6 rows in set (0.00 sec)
```

4.3.2 Query Information by Functions

In order to validate our schemas further, we use a few SQL function queries below to query EAMDB.

4.3.2.1 Query Information Using SQL Function COUNT()

Count the start volume for each record in the culture table. This query retrieves start volume of each culture, for which its end culture date is prior to 2003-09-08. Meanwhile, it counts the numbers of start volumes when they are equal and displays results in increasing chronological order. The date in MySQL is compared as numeric values, and the dataset is not large. So, query is answered fast.

```
mysql> select startVolume, count(*) from culture where endDate < "2003-9-8" group by
startVolume;

+---------------+-----------+
| startVolume | count(*)  |
+---------------+-----------+
| 1904ul        |     1     |
| 2000ul        |     2     |
| 2740ul        |     1     |
| 2858ul        |     1     |
+---------------+-----------+
4 rows in set (0.00 sec)
```

4.3.2.2 Query Information Using Functions SUM() and AVG()

What is an average value of the maximum temperatures in the storage temperature table? This query retrieves sum and average of the maximum temperatures from the storage temperature table. It adds all values of the maximum temperatures, counts records, and gets an average. All records in this table are numbers, so it is done quickly.

```
mysql> select SUM(maximumTemp), count(*), AVG(maximumTemp) as average from
storageTemp;
```

```
+-------------------------+-----------+------------+
| SUM(maximumTemp) | count(*)   | average    |
+-------------------------+-----------+------------+
|                      -6 |         7 | -0.8571    |
+-------------------------+-----------+------------+
1 row in set (0.00 sec)
```

4.3.3 Query Information by Pattern Matching

Who are the researchers with string 'yun' in their names? This query retrieves name and group name of a researcher from the researcher table. This query is done quickly because the researcher table does not contain many records.

```
mysql> select name, groupName from researcher where name like 'yun%';

+------------+--------------------------+
| name       | groupName                |
+------------+--------------------------+
| yun zheng | gene expression group |
+------------+--------------------------+
1 row in set (0.00 sec)
```

During querying the EAMDB, we have observed that the MySQL system generated several error messages. Some of them are typing errors, such as attribute minimum has been typed as "minmum". Some of them are insufficient data type, namely varchar(15) is not long enough for experiment set titles. We have corrected all errors found so far, populated the database again, and continued to query the system. Finally, we are satisfied with the answers from queries mentioned above. Thus, the system is ready to be implemented.

Chapter 5 Implementation

Several issues are discussed in this chapter. First, system overview describes relational database MySQL and a scripting language PHP. The following section describes metadata, a method to handle biochemical data. The subsequent four sections give examples of using database through web interfaces. Some screen shots of web pages are presented here. Lastly, two sections present the total number of tables in the EAMDB and availability of data currently.

5.1 System Overview

PHP and MySQL are the world's best combination for creating data-driven sites [24]. Both of them are Open Source/Free Software. PHP (PHP Hypertext Preprocessor) is a widely used general-purpose scripting language that facilitates the creation of dynamic Web pages by embedding PHP-coded logic in HTML documents. It combines many of the finest features of Perl, C, and Java. It also adds its own elements to the concoction to give web programmers great flexibility and power in designing and implementing dynamic, content-directed web pages. The adopted solution to the problem is to implement a three-tier client server approach where web server as a middle layer would act as a tier between the MySQL and web application. See Figure 5.1. Currently, we use MySQL Version 3.23.49 and PHP Version 4.2.3 installed in Apache/1.3.24 at the Computer Science Department, Concordia University.

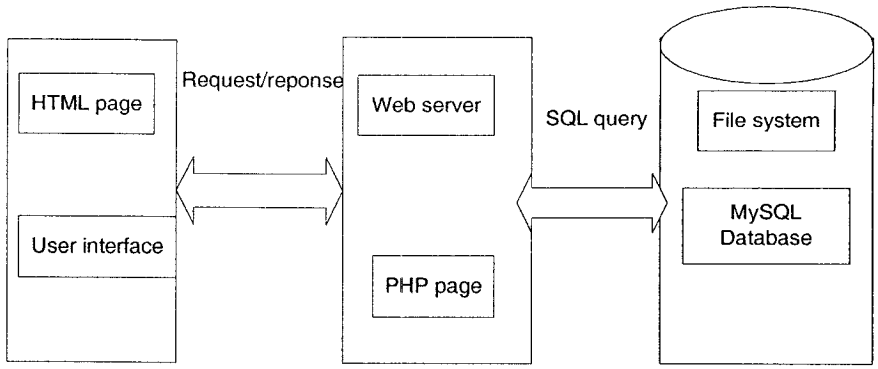


Figure 5.1 The Control Model Diagram of System Architecture

5.2 Metadata

The biochemical data usually include different levels of interpretation, i.e., raw data (also called sensor data), calibrated data, validated data, derived data, and interpreted data [5]. Moreover, relational databases have limitations, for example, data representation is "a set of (flat) tuples". Thus, bioinformaticians use metadata, information about the data, to identify the real data in some situations. The information required to identify data is based on content, validity, sources, preprocessing, or other selected properties. In our database, the protocol table stores the metadata about different protocols. The real protocols are converted to HTML files and stored in the file system. There is a link called fileURI, uniform resource identifier, in the protocol table. One can easily use fileURI to retrieve a real protocol file. Moreover, the SOP documents, graphs and raw data also are converted into HTML files and stored in the directory of U:\grad\www\l_fu temporarily. They will be moved eventually to the file directory of the fungal genomics project at Loyola campus of Concordia University. Figure 5.2 shows that the URI of cellulose protocol file is http://www.cs.concordia.ca/~l_fu/protocol/protocolCellulase.htm.

Additionally, we have set up links between enzymes, chemicals in protocols or SOP documents and their information stored in MySQL. For instance, a user clicks on cellulase in the cellulase protocol file. Next a web page pops out and shows specific cellulase information from the enzyme table in EAMDB. Please see Figure 5.3.

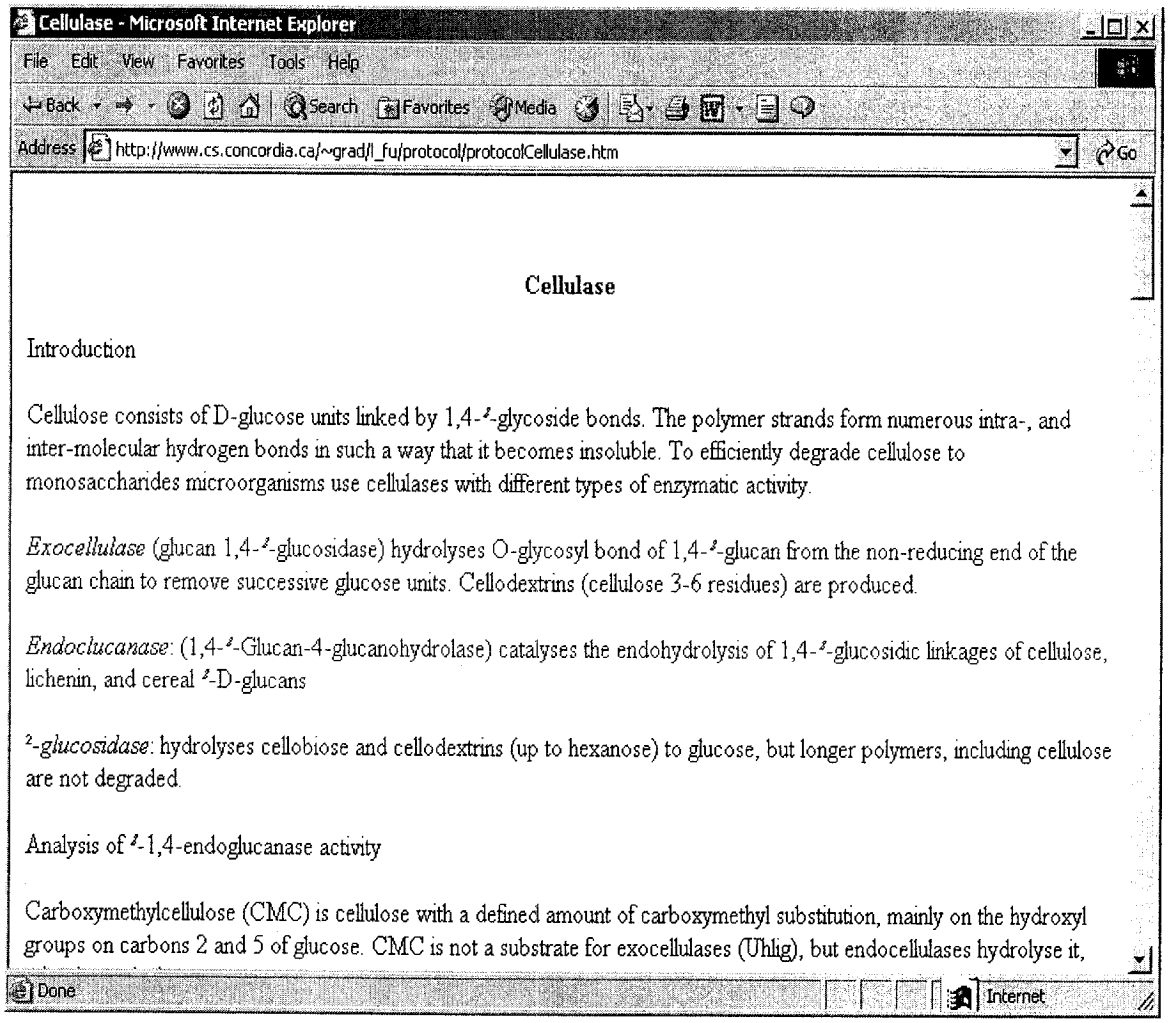


Figure 5.2 The HTML File Converted from the Cellulase Protocol

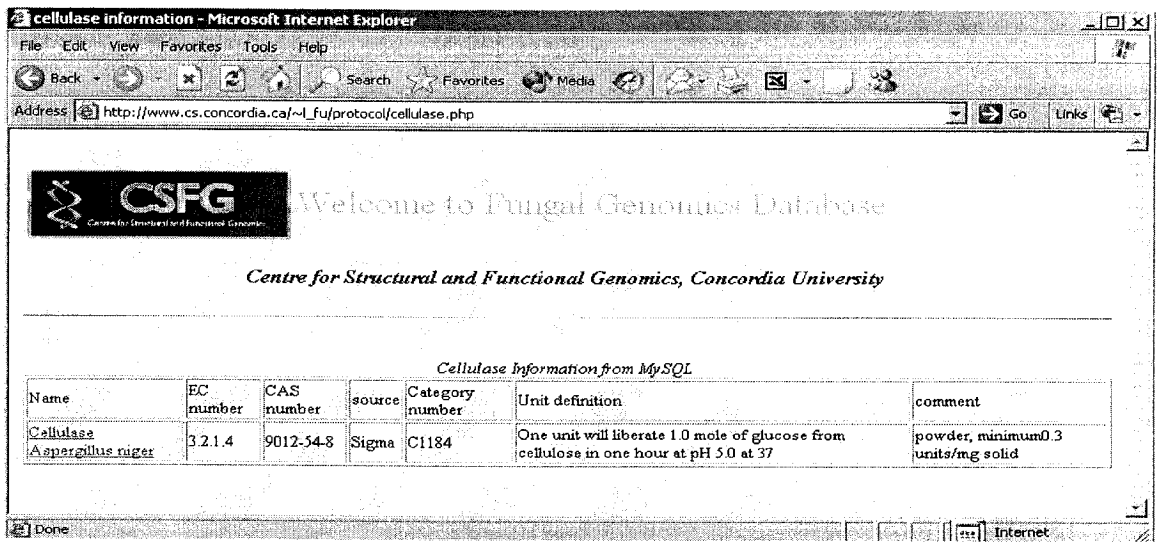


Figure 5.3 A Screen Shot of Cellulase Information from MySQL

The following are PHP and HTML code, which explain how to connect the EAMDB from web pages. When users click on the word cellulose in the protocol file, the following file called cellulase.php is executed. The <?php and ?> tags are used to denote blocks of PHP code. In the PHP code, we use the PHP function mysql_connect ("jeeves", "l_fu", "******") to create a connection with the EAMDB. Three parameters are passed: 1) the computer name (host name) of MySQL server, 2) the MySQL user name and 3) password. As soon as a connection is established, function mysql_select_db is used to select the default database. Then, the function mysql_query sends a query to the currently active database on the server. Moreover, the function mysql_fetch_row() gets a result row as an enumerated array and fetches one row of data from the result associated with the specified result identifier. The row is returned as an array. Each result column is stored in an array offset, starting at offset 0. Finally, the function printf() displays all the rows to the screen, and function mysql_close closes the connection to MySQL.

```
<html><head> <title>cellulase information</title></head>
<body BGCOLOR = "FFFFFF">
  <?php
   $db_link = mysql_connect("jeeves","l_fu", "zhaox9") or die("could not connect to the
      database\n");
   mysql_select_db("l_fu") or die("could not  select database\n");
   $query = "select enzymeName, ECNumber, CASNumber,source, categoryNo,
     unitDefinition, comments  from enzyme where enzymeID = 'EN11'";
   $result_set = mysql_query($query) or die("query failed\n");
   echo"<center> Cellulase Information from MySQL</center>\n";
   echo "<table border=1>\n";
   echo "<tr><td>Name</td><td>EC number</td><td>CAS
       number</td><td>source</td><td>Category number</td><td>Unit
       definition</td><td>comment</td></tr>\n";
   while ($myrow = mysql_fetch_row($result_set)) {
       printf("<tr><td>%s</td><td>%s</td><td>%s</td><td>%s</td><td>%s</td>
            <td>%s</td><td>%s</td></tr>\n", $myrow[0], $myrow[1],  $myrow[2],
            $myrow[3], $myrow[4], $myrow[5],$myrow[6], $myrow[7],$myrow[8]);
   }
  echo "</table>\n";
  mysql_close($db_link);
  ?></body></html>
```

5.3 Login

The data produced by enzyme activity experiments need to be stored in the database. A number of web pages need to be created to handle the data entrances because internal scientists are not familiar enough with SQL to input the data directly. These interfaces are created through using HTML and PHP depending on the different data, such as chemical data and enzyme data. First, a login page is created to let users input their user name and password [22]. The system checks the user name and password for each user and sets up access permissions for various category users. See Figure 5.4. After the user successfully logs in, the user sees a main page, which contains news and links for viewing the information from and inserting data into the database. If one has two roles, one should have two login names and passwords for each role in order to access information fully.

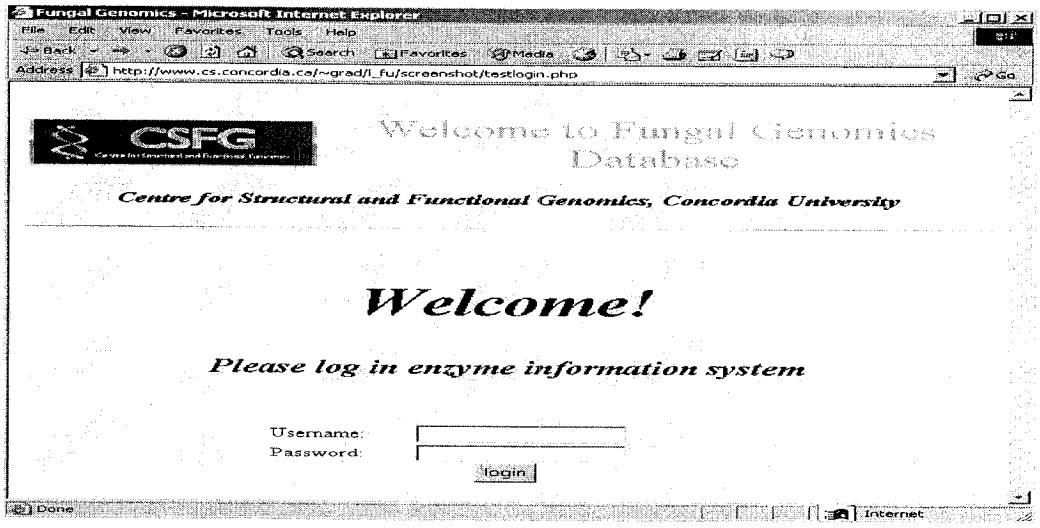


Figure 5.4 The Login Page of the Enzyme Activity Mapping System

5.4 Insert Data

Internal scientists collect experimental data on a daily basis. They will input data into the database based on their working schedule. As an example, we give a web page for inserting a chemical record into EAMDB. See Figure 5.5. Internal scientists type each value into each input field beside labeled names. Then, they need to click on insert button after checking that all data are correct. The system shows a phrase of "Insert successfully" in next page. See Figure 5.6. If internal scientists wish to insert more data into the database, they can click on back to the insertion page and repeat the previous processes again until all data is inserted. If any wrong data is inserted into the database by mistakes, the errors might be detected by internal scientists later. It also should be corrected by internal scientists.

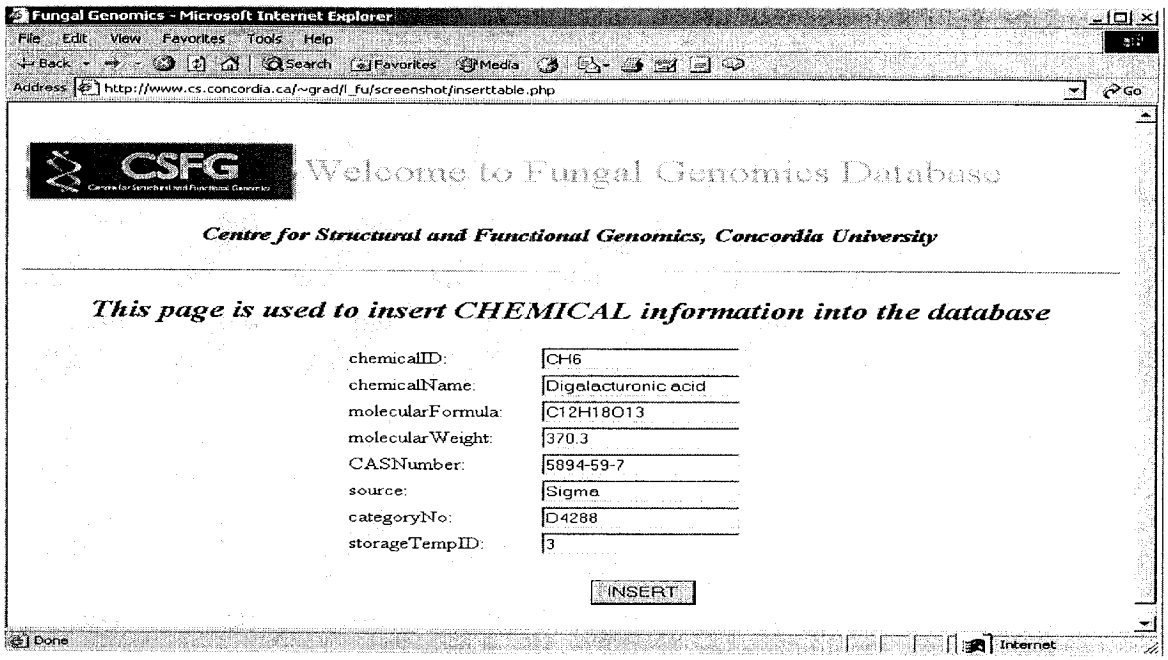


Figure 5.5 The Web Page of Inserting the Chemical Information into EAMDB

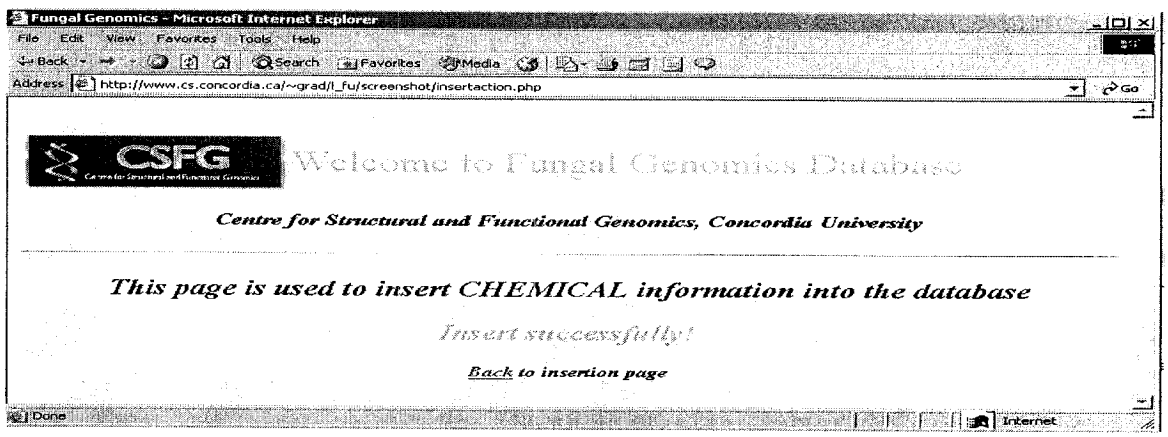


Figure 5.6 The Web Page of Inserting the Data Successfully

These functions of the system are implemented by the following HTML and PHP codes. The first file name is insertion.php. This file is a normal HTML file, which displays an input form. When the user clicks the button INSERT, the input data are sent

to the WWW server, and insertaction.php specified to the attribute action in the form is executed.

```
<html><head><title>Fungal Genomics</title>
<meta http-equiv="Content-Type" content="text/html; charset=iso-8859-1"></head>
<body bgcolor="#FFFFFF" text="#000000">
<body background="../picture/thesisback.gif">
<p><p><table><tr><td><IMG SRC="../picture/logocs.gif" HEIGHT=60 BORDER=0
ALT=""></td>    <td><center><font color= "orange"
size="+3">Welcome to Fungal Genomics Database</font></center></td></tr></table>
<p>
<center><h3><I>Centre for Structural and Functional Genomics, Concordia University
<I></h3><P><P></center>
<center><img vspace="5" src="../picture/bar.gif" width="100%" alt='a moving bar'>
</center><br><br>
<center> <h2><B>This page is used to insert CHEMICAL information into the
database</B></h2></center>

/*create form*/
<form action="insertaction.php" method="post">
<table cellspacing=2 cellpadding=0 width=320 align="center">
<tr><td>chemicalID:</td><td><input name="chemicalID" type="text" width=10>
</td></tr>
<tr><td>chemicalName:</td><td><input name="chemicalName" type="text" width=10>
</td></tr>
<tr><td>molecularFormula:</td><td><input name="molecularFormula" type="text"
width=10></td></tr>
<tr><td>molecularWeight:</td><td><input name="molecularWeight" type="float"
width=10></td></tr>
<tr><td>CASNumber:</td><td><input name="CASNumber" type="text" width=10>
</td></tr>
<tr><td>source:</td><td><input name="source" type="text" width=10></td></tr>
<tr><td>categoryNo:</td><td><input name="categoryNo" type="text" width=10>
</td></tr>
<tr><td>storageTempID:</td><td><input name="storageTempID" type="integer"
width=10></td></tr></table><br><br>
<table cellspacing=2 cellpadding=0 width = 380 align="right"><tr><td><input
type="submit" value = "INSERT"></td></tr></table>
</form></body></html>
```

Below is the content of insertaction.php. In the block of PHP code, the data input is assigned to each variable. Then, the functions, mysql_connect, mysql_select,

mysql_query are performed in order to connect to MySQL and insert the data into the chemical table. At the end of the PHP block, an output message indicates whether insertion is successful or not, and the function mysql_close closes the connection to MySQL.

```
<html><head><title>Fungal Genomics</title>
<meta http-equiv="Content-Type" content="text/html; charset=iso-8859-1"></head>
<body bgcolor="#FFFFFF" text="#000000">
<body background="../picture/thesisback.gif"> <br>
<table><tr><td><IMG SRC="../picture/logocs.gif" HEIGHT=60 BORDER=0
ALT=""></td>    <td><center><font color= "orange"
size="+3"> Welcome to Fungal Genomics Database</font></center></td></tr></table>
<p>
<center><h3><I>Centre for Structural and Functional Genomics, Concordia University
<I></h3><P><P></center>
<center><img vspace="5" src="../picture/bar.gif" width="100%" alt='a moving bar'>
</center><br><br>
<center> <h2><B>This page is used to insert CHEMICAL information into the database
</B></h2></center><br><br>
<?php
$chemicalID = $_POST["chemicalID"];
$chemicalName = $_POST["chemicalName"];
$molecularFormula = $_POST["molecularFormula"];
$molecularWeight = $_POST["molecularWeight"];
$CASNumber = $_POST["CASNumber"];
$source = $_POST["source"];
$categoryNo = $_POST["categoryNo"];
$storageTempID = $_POST["storageTempID"];
   $db_link = mysql_connect("jeeves", "l_fu", "zhaox9") or
      die("could not connect");
   mysql_select_db("l_fu");
   mysql_query("INSERT INTO chemical values ('$chemicalID',
      '$chemicalName','$molecularFormula', '$molecularWeight',
      '$CASNumber','$source',  '$categoryNo', '$storageTempID')");
   mysql_close($db_link);
?>
<center><font color = "#FF9988"><h2>Insert successfully!</h2></font></center>
<br><br><center><H4><a href = "inserttable.php">Back</a> to insertion page</H4>
</body></html>
```

5.5 Query Records

The data information stored in EAMDB should be retrieved through interfaces. Here, we give a simple example of retrieving records inserted by the previous example. Users can visit querying data pages through the main page links. In order to retrieve chemical information, users can either type the chemical ID or chemical name in this page (Figure 5.7). Then, users click on the query button and see related information from the database on the next page. The retrieved information appearing in table form is just an example and may appear in other forms (Figure 5.8).

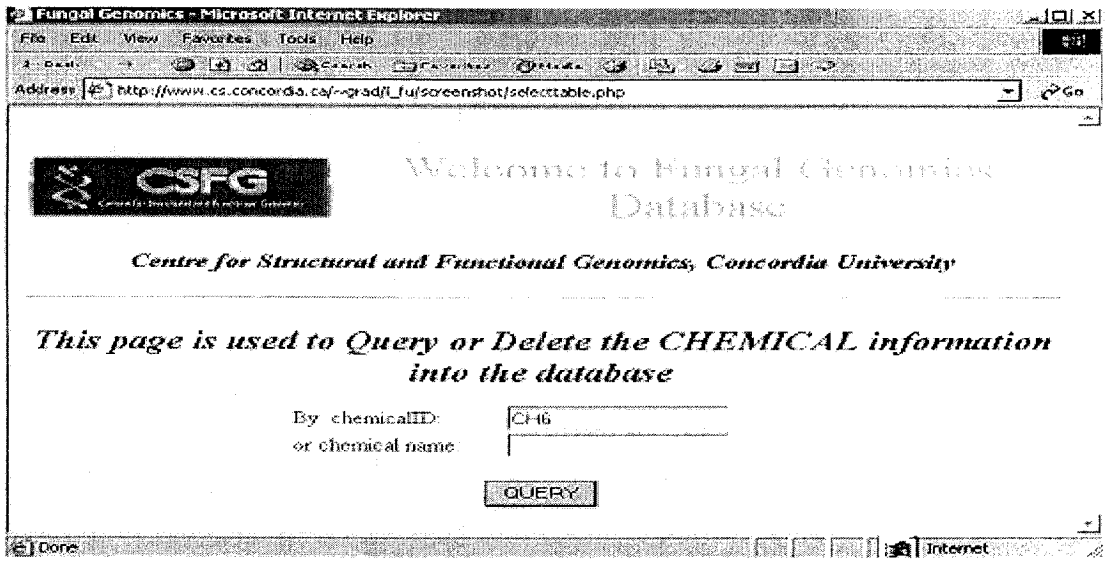


Figure5.7 The Web Page of Querying the Chemical Record in EAMDB

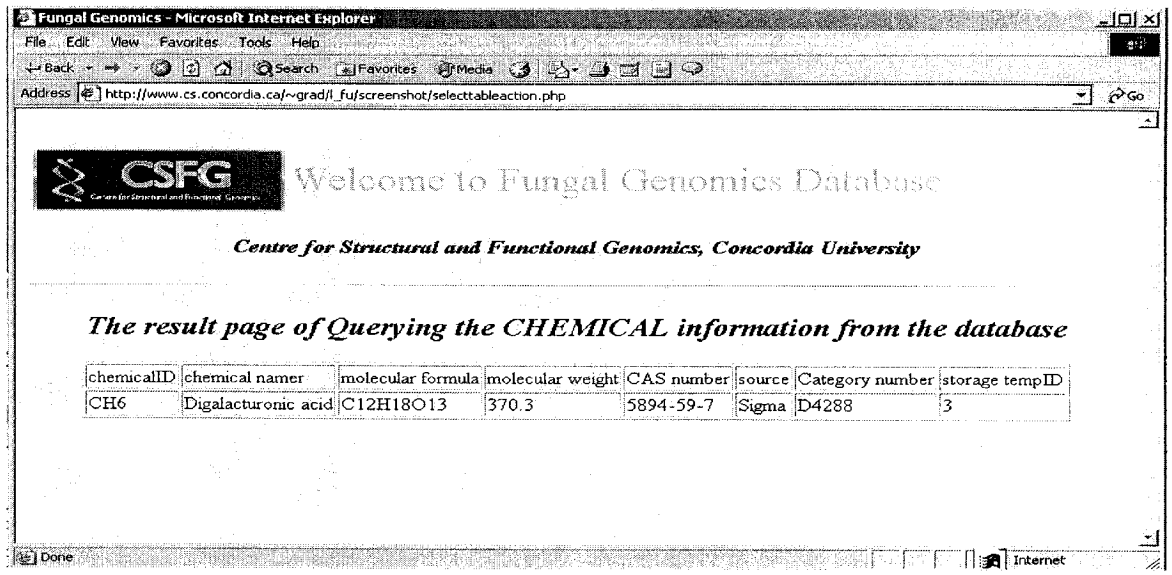


Figure 5.8 The Web Page of Selecting a Chemical Record from the EAMDB

One possibility is that queried records do not exist in the database. In this case, users may see a page like the one below. (Figure 5.9)

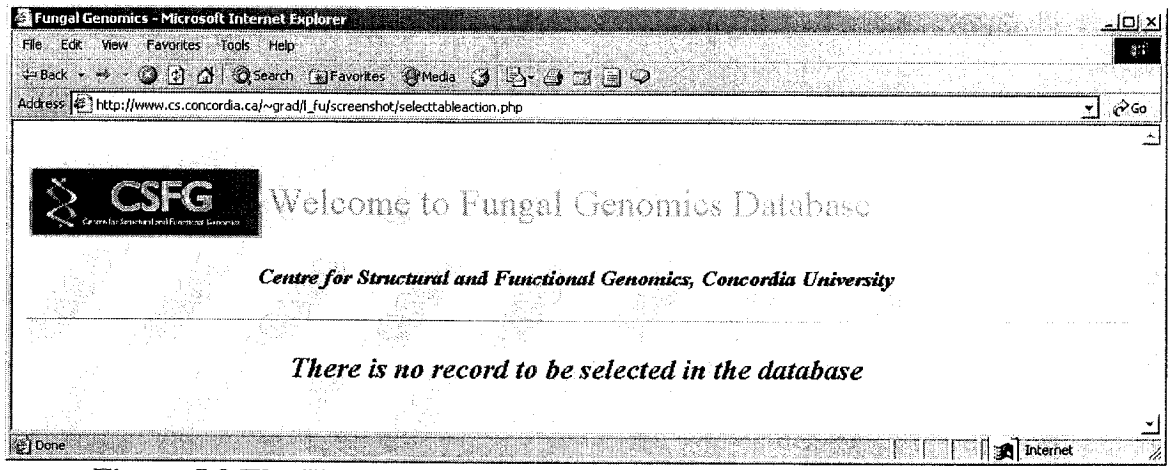


Figure 5.9 The Web Page Showing no Record to be Selected in EAMDB

5.6 Delete Records

The interfaces also include sections for deleting records when internal scientists believe that they should be omitted. Users can link to these pages through the main page when they have the required permissions (Figure 5.10). Public users definitely do not have these permissions. When unauthorized users try to perform these functions, they get warning messages. However, after authorized users input chemical ID or chemical name and click on delete button, they will receive a message indicating that the record has been deleted successfully. (Figure 5.11)

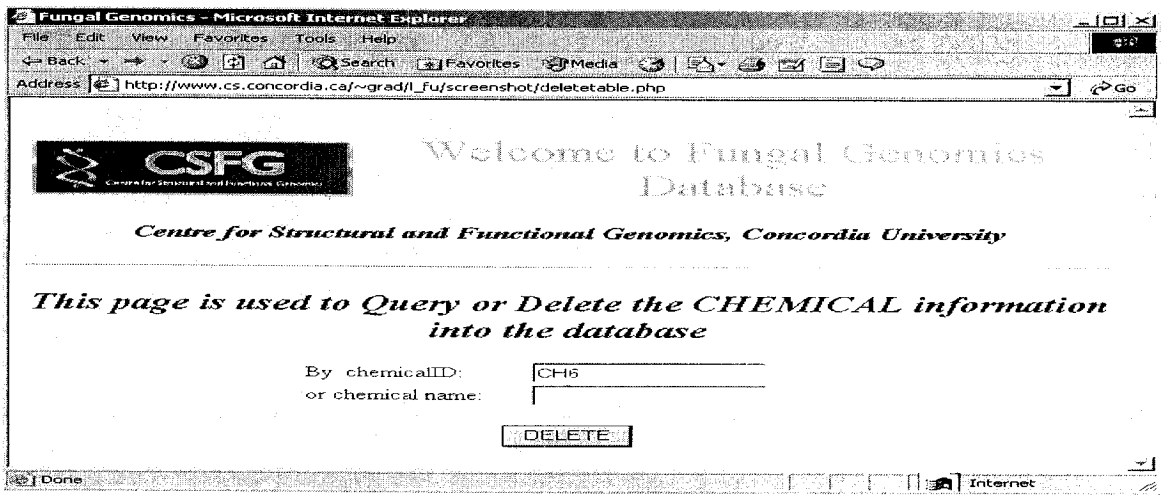


Figure 5.10 The Web Page for Deleting Chemical Records

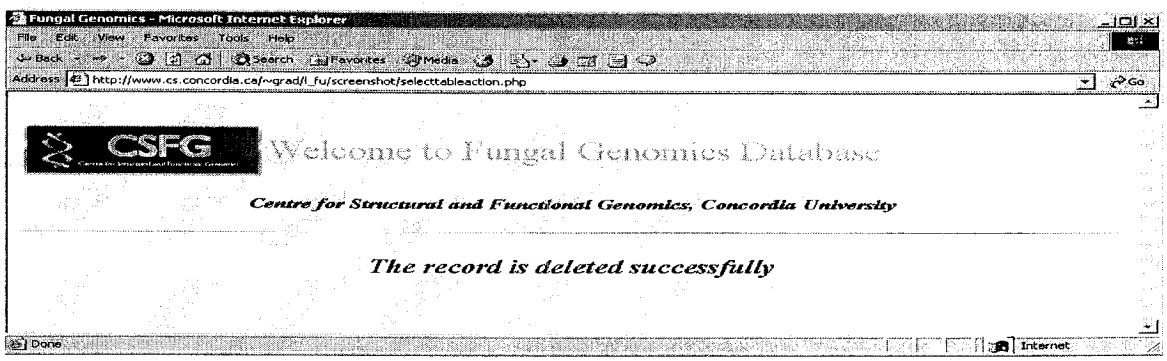


Figure 5.11 The Web Page for Deleting a Chemical Record Successfully

5.7 Possible Standard Reports

When the fungal genomic project is completed, public users can access certain information through user interfaces. Several possible standard reports are described in this section. The first standard report page is the introduction page, which presents general information about this project. For example, it may include a project overview, partners, funding, examined fungi, and significant findings. The user gets basic information about the project from this report. Next, a standard report is the enzyme information page reporting characteristic information of all enzymes examined in the experiments of this project. It may list these enzymes by EC Number or according to the fungus from which an enzyme comes. Each enzyme name links to its own individual page. Through links, the user could look at detailed information, such as enzyme synonyms, reaction equation, catalytic constants and enzyme stability. Another standard report page could be the publications page. The internal scientists in this project will publish their research papers eventually. The user can view all publications related to the fungal genomic project. Moreover, a search page will be provided to public users. It may include a few search methods such as sequence search, taxonomy tree search and quick search. The user inputs a gene sequence, an enzyme name or EC Number to retrieve related information from the database. Furthermore, a list of useful Internet resources related to fungal genomics should be presented because the user may like to view corresponding information on other web sites. Resources might be grouped into different categories, for instance, enzymes and their genes, chemical compounds, microorganisms and scientific literatures. The next report page might be a frequently asked questions page. It lists a number of questions asked by public users and possible answers to these

questions. An example might be which method is used in characterizing cellulases. The last report page contains the contact information. The user may want to know more information about this project, and they may contact us by e-mail or phone. Thus, these types of information should be provided in this page.

5.8 Some Statistics about Tables

There are 17 tables in Shu's design. Through validation processes, we have added 12 tables, modified attributes in unchanged tables and deleted 3 tables based on impact of detailed requirements and normalization techniques. Therefore, the total number of tables in EAMDB is 26. We have implemented partial data into 18 tables currently because in this three-year project, some experiments have not begun yet. Thus, we do not have any data for insertion. Table 5.1 shows the overview of refinement.

5.9 Available Data

To validate the database, we have inserted partial data of 4 tuples to 60 tuples into each table. Moreover, HTML files converted from four protocols, five graphs, and one data file are stored in the file system. These files are linked to the EAMDB by uniform resource identifiers (fileURI). We will store more data in the database when they become available.

At the time of writing, researchers working in the experiments of the enzyme activity assay have made much progress. They have examined 51 different enzymes. They have also written 13 standard operation procedures (SOP) and four protocols, which are still under revision and have not been released yet.

Table 5.1 Overview of Changes in EAMDB

Main Modules	Tables in Shu's Design	Tables after refinement	Tables populated with partial data
Enzyme activity mapping module	reaction kineticParameter	reaction kineticParameter	none
Enzyme module	Enzyme enzymeStability enzymeSolution	enzyme enzymeSynonym enzymeStability enzymeSolution	enzyme enzymeSynonym
Chemical compound module	chemical chemicalSolution	chemical chemicalSynonym chemicalSolution	chemical chemicalSynonym
Experiment module	assayExperiment protocol enzymePurification inhibitionExperiment researcher	experimentSet experiment culture enzymePurification	experimentSet experiment culture media
Data module	assayData assayResult inhibitionData inhibitionResult	ExperimentData condition clone result	experimentData condition clone
Reference module	reference	reference researcher user location storageTemp	reference researcher location storageTemp
Protocol module	none	protocol labProcedure procedureOrder	protocol labProcedure procedureOrder
Total tables	17	26	18

Chapter 6 Conclusion and Future Work

6.1 Conclusion

Enzymes are biological catalysts with special properties that are not found in other catalysts. Enzymes bind temporarily to one or more of the reactants of the reaction they catalyze. As a result, they lower the activation energy needed and thus speed up the reaction. In order to find fungal enzymes that can be used in pulp and paper processing, researchers at Concordia University are conducting high throughput enzyme assay experiments. The data produced by these experiments need to be stored in the EAMDB for scientific uses. Validation and implementation of the Enzyme Activity Mapping Database (EAMDB) presented in this thesis is part of the effort to achieve the goal.

Several techniques used to capture detailed requirements are discussed. They are informal discussion with researchers, analyzing some data files, using use cases approach and utilizing query approach. First, informal discussion allow us to gain greater insights into the enzyme activity assay experiments and query any points unclear to ourselves, vice versa. Next, analyzing data files directly gives us an idea about what should be stored in the database.

Moreover, in use cases approach, four potential users are internal scientists, internal bioinformaticians, a database administrator, and public users. The nine use cases in the system are load SOP (standard operation procedures) data, load experiment data, load graph data, generate reports, view reports, modify experiment data, modify graph data, retrieve experiment data, and retrieve graph data. Each use case is described respectively.

Additionally, in the query approach, internal scientists are both the data source and users. This means that they have full access rights to the database. Their tasks with the database are to load all of the necessary data and to make use of information stored. Usually, they retrieve the information from the database to guide their research activities. Internal bioinformaticians only query the data in the database. They may retrieve the information to update other databases or use it to develop various software tools. They also give some feedback to a database administrator about maintaining the database. Furthermore, a database administrator (DBA) has complete control of the database. The DBA's responsibility is to ensure that the database is updated accurately and regularly in order to meet new user needs. Other responsibilities include maintaining data integrity, security and backup. In addition, public users consist of external scientists, external bioinformaticians, and general users. They all can only view the standard reports generated by internal scientists through interfaces of web pages because experimental details may be protected as intellectual property. The thirty possible queries are described to extract detailed requirements.

There are significant impacts of the detailed requirements during development of the EAMDB. At this stage, six tables have been added into EAMDB. They are experimentSet table, condition table, clone table, labProcedure table, procedureOrder table, and culture table. Besides, some attributes in existing tables are modified in order to store the experimental data from data files adequately. Primary key and foreign keys are used to obtain data integrity during the validation process. However, the unnecessary tables like inhibitionExperiment, inhibitionData, and inhibitionResult were deleted.

A main step taken in the validation process is normalizing all the relations. All relations are arranged into logical groups. Thus, we normalize each group respectively. Through first normal form, four tables, enzymeSynonym table, chemicalSynonym table, storageTemp table and media table were added into EAMDB because they are related to multivalue attributes. Through third normal form, a location table is added to the database because a transitive dependency exists in chemical and enzyme tables. After checking Boyce-Codd normal form and fourth normal form, we conclude that all relations are in 4NF.

The next step is querying the database. First, populate 18 tables with data of 4 tuples to 60 tuples according to available data. Then, the thirty queries listed in capturing requirement stage have been implemented to query the EAMDB. MySQL query sentences and outputs are presented as well. Queries with SQL functions, such as COUNT(), SUM(), AVG() and pattern matching, are also implemented. During querying process, some error messages are generated because of mistypes of attributes and insufficient data types in tables. These errors were corrected immediately.

Additionally, in order to handle experimental protocols, metadata is stored in the protocol table and a fileURI (uniform resource identifier) is used to retrieve a corresponding protocol file. So do the graph, SOP, and data files.

Interfaces between users and the EAMDB have been established by using PHP. The certified users can insert data into and query data from the database through user interfaces. Some screen shots and source codes are presented to illustrate the uses of interfaces.

We use all possible techniques to validate and implement the EAMDB. Currently, most relations are ready to be queried despite limited amounts of data available. However, the fungal genomic project is a three-year project, so some experiments have not begun yet. Seven tables in the database still do not contain any data. As it is generated, data will be loaded into tables.

6.2 Contribution

This thesis presents a validation and implementation of enzyme activity mapping database into MySQL. The contribution of the thesis to the EAMDB falls into the following categories:

- Capture the detailed requirements by several techniques: a) discussing informally with researchers; b) analyzing some data files which exactly show what should be stored in the database; c) using use cases approach; and d) utilizing query approach.
- Refine the EAMDB design by splitting, adding, and deleting tables and their relationships according the detailed requirements.
- Normalize the relational schemas. Add and break down tables so that all of relations are in 4NF.
- Implement partial data into the EAMDB from Excel and Word files, which are provided by researchers in the enzyme assay group.
- Query EAMDB using a set of queries, which users would like to query the EAMDB, to validate the database design. Correct some errors generated by MySQL due to mistypes or insufficient data types.

- Create HTML files for all protocols, graphs, SOP and raw data. Set up an attribute, uniform resource identifiers (URI), in each related table. They can be used to link corresponding files.
- Set up links in HTML files so that users can click on specific items in the file and see related information stored in the EAMDB.
- Build up user interfaces for the EAMDB by using PHP.

6.3 Suggested Future Work

The following work is suggested in order to create a highly accurate, efficient and convenient system.

A. Further performance test is needed. Currently, only a limited amount of data is populated in the EAMDB. A full test requires a large amount of data stored in the database. Moreover, the process of software development is iterated to ensure the final product is correct.

B. Refine the user interfaces on the web. A good database design needs the support of wonderful user interfaces to enforce its requirements. This requires that the interfaces have a distinctive fashion, including the background, icons and text fields. One may code this web application other than using PHP, such as Perl and Java Servlets. However, one should be aware that the web application is coded to use the MySQL database. There are a number of places where the code had to accommodate MySQL's lack of functionality.

C. Create the user interfaces, which can be used to load experimental data automatically from external files. It will reduce the heavy work of inserting data into EAMDB.

D. Create the user interfaces to handle random natural language queries. It helps scientists use the EAMDB efficiently and effectively.

E. Integrate the EAMDB into an integrated database for the fungal genomics project. The information of enzyme catalytic activities is part of the functional information about the enzymes. It should be connected with data in the gene expression system, microarray system and cDNA library. To complete the general database for the whole project, EAMDB should be regarded as an essential part.

Bibliography

[1] Susan M Andrews. Design & Implementation of a Relational Database and Graphical User Interface to Store Microarray Data. Master Thesis, Department of Information Technology, University Glasgow. September 2001.

[2] Tim Berners-Lee, Roy T. Fielding, Larry Masinter. Uniform Resource Identifiers (URI): Generic Syntax. RFC2396, Aug. 1998. http://www.ietf.org/rfc/rfc2396.txt

[3] Gordon Bickerstaff. Enzyme catalyzed reactions. Paisley University. URL *http://www-biol.paisley.ac.uk/courses/stfunmac/glossary/enzymes.html*

[4] Greg Butler. Bioinformatics Algorithms. Course slides, Concordia University, 2003. URL http://bio-it.cs.concordia.ca/internal-students/comp691R-summer2003/

[5] Greg Butler. Scientific databases and major conceptual models for bioinformatics, lecture slides, Concordia University. 2003. URL http://bio-it.cs.concordia.ca/internal-students/comp691S-summer2003/lectures/week2-lectures.pdf

[6] Greg Butler. What is Bioinformatics? n.p., June 2000. URL http://www.cs.concordia.ca/~faculty/gregb/home/PDF/bioinfo-overview.pdf

[7] John Carlis, Joseph Maguire. Mastering Data Modeling: a user-driven approach. Addison-Wesley. 2001. ISBN 0-201-70045-X.

[8] Corpus Christi. Enzymes. Enzyme course slides, Island University. Oct. 2001. URL *http://www.tamucc.edu/~plarkin/4301folder/HTMLCH8%20%20/sld022.htm*

[9] Robert A. Copeland, *Enzymes: A Practical Introduction to Structure, Mechanism, and Data Analysis*, VCH Publishers. New York. 1996.

[10] Damian Counsel. Bioinformatics Frequently Asked Questions. Bioinformatics.org. URL *http://bioinformatics.org/faq/#definitionOfCheminformatics*

[11] C. J. Date. An Introduction to Database Systems. Seventh edition, Addison-Wesley. 2000. ISBN 0-201-38590-2.

[12] Lynda B.M. Ellis, C. Douglas Hershberger and Lawrence P. Wackett. The University of Minnesota Biocatalysis/Biodegradation Database: microorganisms, genomics and prediction. *Nucleic Acids Research.* 28. 2000: 377-379.

[13] Principles of Enzyme Catalysis. Edvotek: The Biotechnology Education Company. URL *http://www.edvotek.com/pdfs/282.pdf*

[14] ExPASy—Biochemical Pathways (2000), URL http://www.expasy.org/cgi-bin/search-biochem-index

[15] Gerald Frenkel. Michaelis-Menten Model of Enzyme Kinetics. Course slides, Rutgers University, NJ. URL http://tecn.rutgers.edu/bio301s/michaelismenten_kinetics.htm

[16] Hector Garcia-Molina, Jeff Ullman, Jennifer Widom. Database Systems: the Complete Book, 1st edition. Prentice Hall, 2001. ISBN: 0130319953.

[17] Susumu Goto, Yasushi Okuno, Masahiro Hattori, Takaaki Nishioka and Minoru Kanehisa. LIGAND: database of chemical compounds and reactions in biological pathways. *Nucleic Acids Research*, 30 (2002), 402-404.

[18] Jan L. Harrington. *Relational Database Design Clearly Explained.* Morgan Kaufmann, CA. ISBN: 0-12-326425-1.

[19] Ivar Jacobson, Magnus Christerson, Patrik Jonsson, and Gunnar Overgaard. *Object-Oriented Software Engineering: A Use Case Driven Approach.* Addison-Wesley, USA. 1992.

[20] John W. Kimball. Enzyme Kinetics. On-line. June 2003. URL http://users.rcn.com/jkimball.ma.ultranet/BiologyPages/E/EnzymeKinetics.html

[21] Vijay Kumar. Database Implementation and Validation. Course slides, Kansas City. 2002. URL http://www.sce.umkc.edu/~kumarv/

[22] Yanhong Li. Enhancement of the CINDI system. Master Thesis, Department of Computer Science, Concordia University. Aug. 2003.

[23] Neil Matthew and Richard Stones. *Beginning Database with MySQL.* Chicago: Wrox Press. 2002.

[24] Graeme Merrall. PHP/MySQL Tutorial. Carnegie Mellon University. 2003. URL http://hotwired.lycos.com/webmonkey/programming/php/tutorials/tutorial4.html

[25] Molecular Probes. EnzChek Protease Assay Kits. Product information. Jan. 2003. URL. http://www.interchim.fr/bio/molprobes/cd/docs/media/pis/mp06638.pdf

[26] Harvey Motulsky, Arthur Christopoulos. Fitting Models to Biological Data using Linear and Nonlinear Regression: a practical guide to curve fitting. GraphPad Software. 2003. URL http://www.graphpad.com/curvefit/find_vmax___km.htm

[27] Michael Nilges and Jens P. Linge. Bioinformatics-a definition. Institut Pasteur. URL http://www.pasteur.fr/recherche/unites/Binfs/definition/bioinformatics_definition.pdf

[28] Frank Norman. URI. SearchWebServices.Com definitions. Dec.2002. URL http://searchwebservices.techtarget.com/sDefinition/0,290660,sid26_gci214160,00.html

[29] Jean-Louis Reymond. New High-Throughput Screening Assays for Biocatalysis. *Chimia* 55, (2001), 1049-1052.

[30] Geri Schneider, Jason P. Winters. Applying Use Cases: a practical guide. Addison-Wesley. 1998.

[31] Ida Schomburg, Antje Chang and Dietmar Schomburg. BRENDA: enzyme data and metabolic information. *Nucleic Acids Research*. 30 (2002), 47-49.

[32] Ronghua Shu. Design of an Enzyme Activity Mapping Database. Master Thesis, Department of Computer Science, Concordia University. Spring 2003.

[33] Ian Sommerville. Requirement Engineering Process. *Software engineering*, 6th edition. Addison-Wesley, 2001.

[34] Michael A. Sypes. Enzyme Kinetics. Course slides, Pennsylvania University. 2003. URL http://www.bmb.psu.edu/courses/bmb401/GandGPowerPoints/chapter14

[35] Adrian Tsang et al. The Overview of the Fungal Genomics Project. Concordia University. 2002. URL https://fungalgenomics.concordia.ca/public/index.html

[36] Michael Widenius, David Axmark. MySQL Reference Manual. On-line 2002. URL http://www.mysql.com/doc/en/index.html

[37] Ulrike Wittig and Ann De Beuckelaer. Analysis and comparison of metabolic pathway databases. *Briefings in Bioinformatics*. 2 (2001), 126-142.

Appendix

The following is a conversion from Relational schemas of EAMDB to MySQL syntax. They are generated from MySQL database. First, an explanation about a table structure is presented. Then, a table structure is followed and all table types are MyISAM. However, we just give references to tables, which have been described in the thesis.

1. Table structure for table 'chemical' is a chemical table definition in MySQL. It shows 8 attributes describing the chemical. Each attribute has its own data type, such as, from varchar(10) to varchar(60). Molecular weight has data type float. The primary key in this table is chemicalID and cannot be null. Others attributes values can be null by default.

```
CREATE TABLE chemical (
  chemicalID varchar(10) NOT NULL default '',
  chemicalName varchar(60) default NULL,
  molecularFormula varchar(20) default NULL,
  molecularWeight float default NULL,
  CASNumber varchar(15) default NULL,
  source varchar(20) default NULL,
  categoryNo varchar(10) default NULL,
  storageTempID int(3) default NULL,
  PRIMARY KEY  (chemicalID)
  KEY idxchename (chemicalName)
) TYPE=MyISAM;
```

2. Table structure for table 'chemicalSolution' is a chemical solution table definition in MySQL. It shows 6 attributes describing the chemical solution. Each attribute has its own type. Primary key is chemicalSolutionID with integer type

```
CREATE TABLE chemicalSolution (
  chemicalSolutionID int(10) NOT NULL default '0',
  chemicalID varchar(10) default NULL,
  procedureID int(5) default NULL,
  solvent varchar(20) default NULL,
  pH float default NULL,
  buffer varchar(20) default NULL,
  concentration float default NULL,
  PRIMARY KEY  (chemicalSolutionID)
) TYPE=MyISAM;
```

3. Table structure for table 'chemicalSynonym' is a chemical sysnonym table definition in MySQL. It includes two attributes and has many to one relationship with chemical

```
CREATE TABLE chemicalSynonym (
 chemicalID varchar(10) default NULL,
 synonym varchar(30) default NULL
) TYPE=MyISAM;
```

4. Table structure for table 'clone' is a clone table definition in MySQL. The detailed Explanation is presented inside the table 4.16 in the thesis.

```
CREATE TABLE clone (
 cloneID int(7) NOT NULL default '0',
 cloneName varchar(20) default NULL,
 cultureID int(7) default NULL,
 PRIMARY KEY  (cloneID)
) TYPE=MyISAM;
```

5. Table structure for table 'condition' is a condition table definition in MySQL. The detailed Explanation is presented inside the table 4.18 in the thesis.

```
CREATE TABLE condition (
 conditionID int(4) NOT NULL default '0',
 name varchar(25) default NULL,
 condition varchar(10) default NULL,
 PRIMARY KEY  (conditionID)
) TYPE=MyISAM;
```

6. Table structure for table 'culture' is a culture table definition in MySQL. The detailed Explanation is presented inside the table 4.22 in the thesis.

```
CREATE TABLE culture (
 cultureID int(7) NOT NULL default '0',
 name varchar(35) default NULL,
 procedureID int(3) default NULL,
 sequenceID varchar(15) default NULL,
 researcherID int(3) default NULL,
 host varchar(20) default NULL,
 stackConcentration float default NULL,
 cultureVolume varchar(10) default NULL,
 startVolume varchar(10) default NULL,
 startConidia float default NULL,
 harvestDate date default NULL,
 cultureDat date default NULL,
 endDate date default NULL,
 PRIMARY KEY  (cultureID)
) TYPE=MyISAM;
```

7. Table structure for table 'enzyme' is an enzyme table definition in MySQL. The detailed Explanation is presented inside the table 4.41 in the thesis.

```
CREATE TABLE enzyme (
 enzymeID varchar(10) NOT NULL default '',
 enzymeName varchar(75) default NULL,
 ECNumber varchar(15) default NULL,
 CASNumber varchar(15) default NULL,
 source varchar(20) default NULL,
 categoryNo varchar(10) default NULL,
 storageTempID int(3) default NULL,
 comments text,
 unitDefinition text,
 PRIMARY KEY  (enzymeID)
 KEY idxenzname (enzymeName)
) TYPE=MyISAM;
```

8. Table structure for table 'enzymePurification' is an enzyme purification table definition in MySQL. It shows 6 attributes with different data types. EnzymePurificationID is primary key with data type int(7), which means values of primary key are not more than 7 digits. EnzymeID and procedureID are foreign keys.

```
CREATE TABLE enzymePurification (
 enzymePurificationID int(7) NOT NULL default '0',
 enzymeID varchar(10) default NULL,
 method text,
 procedureID int(5) default NULL,
 comment text default NULL,
 yield varchar(10) default NULL,
 PRIMARY KEY  (enzymePurificationID)
) TYPE=MyISAM;
```

9. Table structure for table 'enzymeSolution' is an enzyme solution table definition in MySQL. It has 8 attributes with different data types. Enzyme solution data will be stored in this table, and enzymeID is a foreign key from the enzyme table.

```
CREATE TABLE enzymeSolution (
 enzymeSolutionID int(7) NOT NULL default '0',
 enzymeID varchar(10) default NULL,
 solvent varchar(20) default NULL,
 pH float default NULL,
 buffer varchar(20) default NULL,
 stabilizer varchar(20) default NULL,
 concentration float default NULL,
 concentrationUnit varchar(5) default NULL,
```

```
 PRIMARY KEY  (enzymeSolutionID)
) TYPE=MyISAM;
```

10. Table structure for table 'enzymeStability' is an enzyme stability table definition in MySLQL. It has 8 attributes with different data types. Primary key is enzymeStabilityId and enzymeID is a foreign key from the enzyme table.

```
CREATE TABLE enzymeStability (
 enzymeStabilityID int(7) NOT NULL default '0',
 enzymeID varchar(10) default NULL,
 pHStability varchar(10) default NULL,
 tempStability varchar(10) default NULL,
 generalStability text,
 solventStability text,
 oxidationStability varchar(100) default NULL,
 storageStability varchar(100) default NULL,
 PRIMARY KEY  (enzymeStabilityID)
) TYPE=MyISAM;
```

11. Table structure for table 'enzymeSynonym' is a enzyme synonym table definition in MySQL. The detail explanation is given inside the table 4.43 in the thesis.

```
CREATE TABLE enzymeSynonym (
 enzymeID varchar(10) default NULL,
 synonym varchar(50) default NULL
) TYPE=MyISAM;
```

12. Table structure for table 'experiment' is an experiment table definition in MySQL. The detailed explanation is given inside the table 4.4 in the thesis.

```
CREATE TABLE experiment (
 experimentID int(7) NOT NULL default '0',
 experimentTitle varchar(30) default NULL,
 experimentSetID int(7) default NULL,
 sequenceID varchar(15) default NULL,
 numberSample int(1) default NULL,
 result varchar(20) default NULL,
 date date default NULL,
 positiveControl text,
 description text,
 comment text,
 fileURI varchar(70) default NULL,
 PRIMARY KEY  (experimentID)
 KEY idxexptitle (experimentTitle),
 KEY idxexpid (experimentID)
```

```
) TYPE=MyISAM;
```

13. Table structure for table 'experimentData' is an experiment data table definition in MySQL. The detailed explanation is given inside the table 4.14 in the thesis.

```
CREATE TABLE experimentData (
  experimentID int(7) NOT NULL default '0',
  conditioned int(4) NOT NULL default '0',
  cloneID int(7) NOT NULL default '0',
  value float default NULL
) TYPE=MyISAM;
```

14. Table structure for table 'experimentSet' is an experiment set table definition in MySQL. The detailed explanation is given inside the table 4.2 in the thesis.

```
CREATE TABLE experimentSet (
  experimentSetID int(7) NOT NULL default '0',
  experimentSetTitle varchar(45) default NULL,
  host varchar(20) default NULL,
  procedureID int(7) default NULL,
  resarcherID int(3) default NULL,
  controlEnzymeID int(7) default NULL,
  comment text,
  fileID varchar(70) default NULL,
  PRIMARY KEY  (experimentSetID)
  KEY idxsettitle (experimentSetTitle)
) TYPE=MyISAM;
```

15. Table structure for table 'kineticParameter' is a kinetic parameter table definition in MySQL. It has 17 attributes with different data types and would store all information about kinetic constants. Primary key is kineticiD integer type with maximum 7 digits length. ReactionID, enzymeID, resultID are foreign keys form related tables.

```
CREATE TABLE kineticParameter (
  kineticID int(7) NOT NULL default '0',
  reactionID varchar(10) default NULL,
  enzymeID varchar(10) default NULL,
  chemicalID varchar(10) default NULL,
  pHHigh float default NULL,
  pHLow float default NULL,
  pHOptimum float default NULL,
  tempHigh float default NULL,
  tempLow float default NULL,
  tempOptimum float default NULL,
  cofactior varchar(20) default NULL,
```

```
 activator varchar(20) default NULL,
 KmOptimum float default NULL,
 KmRoom float default NULL,
 KcatOptimum float default NULL,
 KcatRoom float default NULL,
 specificActivity varchar(20) default NULL,
 resultID int(7) default NULL,
 PRIMARY KEY  (kineticID)
) TYPE=MyISAM;
```

16. Table structure for table 'labProcedure' is a lab procedure table definition in MySQL. The detailed explanation is given inside the table 4.9 in the thesis.

```
CREATE TABLE labProcedure (
 procedureID int(5) NOT NULL default '0',
 type varchar(10) default NULL,
 content text,
 protocolID int(3) default NULL,
 PRIMARY KEY  (procedureID)
) TYPE=MyISAM;
```

17. Table structure for table 'location' is a location table definition in MySQL. The detailed explanation is given inside the table 4.49 in the thesis.

```
CREATE TABLE location (
 objectID varchar(10) NOT NULL default '',
 labName varchar(25) default NULL,
 quantity float default NULL,
 unit varchar(10) default NULL
) TYPE=MyISAM;
```

18. Table structure for table 'media' is a media table definition in MySQL. It has many to one relation to the culture table. CultureID is foreign key from the culture table. It stores components of each media culturing the host.

```
CREATE TABLE media (
 cultureID int(7) default NULL,
 component varchar(10) default NULL
) TYPE=MyISAM;
```

19. Table structure for table 'procedureOrder' is the procedure order table definition in MySQL. The detailed explanation is given inside the table 4.12 in the thesis.

```
CREATE TABLE procedureOrder (
 procedureID int(5) NOT NULL default '0',
 childProcedureID int(5) default NULL,
```

```
  orderOf int(2) default NULL
) TYPE=MyISAM;
```

20. Table structure for table 'protocol' is the protocol table definition in MySQL. The detailed explanation is given inside the table 4.7 in the thesis.

```
CREATE TABLE protocol (
  protocolID int(3) NOT NULL default '0',
  protocolName varchar(50) default NULL,
  source varchar(15) default NULL,
  effectiveDate date default NULL,
  numberReagent int(2) default NULL,
  comment text,
  fileURI varchar(70) default NULL,
  PRIMARY KEY  (protocolID)
) TYPE=MyISAM;
```

21. Table structure for table 'reaction' is the reaction table definition in MySQL. It has 6 attributes with different data types. Varchar(30) means the data input in this column can not over the 30 characters otherwise data will be truncated. Three foreign keys are from substrate table, product table and reference table.

```
CREATE TABLE reaction (
  reactionID varchar(10) NOT NULL default '0',
  reactionType varchar(30) default NULL,
  substrateID varchar(20) default NULL,
  productID varchar(20) default NULL,
  reactionEquation varchar(200) default NULL,
  note text,
  PRIMARY KEY  (reactionID)
) TYPE=MyISAM;
```

22. Table structure for table 'reference' is the reference table definition in MySQL. It stores information about all references of enzymes, chemicals, and reaction. ReferedID can be one of the enzymeID, chemicalID, and reactionID. The data type text indicates that sorting and comparison is performed in case-sensitive fashion and storage required L+2 bytes, where L < 2^16.

```
CREATE TABLE reference (
  referenceID int(6) NOT NULL default '0',
  referedID varchar(10) default NULL,
  title text,
  author varchar(25) default NULL,
  publisherDate varchar(10) default NULL,
  source varchar(30) default NULL,
  volume varchar(10) default NULL,
```

```
 page varchar(10) default NULL,
 PRIMARY KEY  (referenceID)
) TYPE=MyISAM;
```

23. Table structure for table 'researcher' is the researcher table definition in MySQL. It stores all researchers who participates the enzyme activity assays. It generally has attributes of name and group plus Id, which is enough to distinguish researchers. Other information like researcher address is not included.

```
CREATE TABLE researcher (
 researcherID int(3) NOT NULL default '0',
 name varchar(25) default NULL,
 groupName varchar(25) default NULL,
 PRIMARY KEY  (researcherID)
) TYPE=MyISAM;
```

24. Table structure for table 'result' is the result table definition in MySQL. It stores the enzyme characteristic results from experiments. It has 7 attributes with three data types: integer, varchar, and float. KineticParameter table has its primary key resultID as foreign key. ExperimentID in this table is foreign key form the experiment table.

```
CREATE TABLE result (
 resultID int(7) NOT NULL default '0',
 experimentID int(7) default NULL,
 sequenceID varchar(15) default NULL,
 fileURI varchar(70) default NULL,
 Km float default NULL,
 Kcat float default NULL,
 Vmax float default NULL,
 PRIMARY KEY  (resultID)
) TYPE=MyISAM;
```

25. Table structure for table 'storageTemp' is the storage temperature table definition in MySQL. The detailed explanation is given inside table 4.47 in the thesis.

```
CREATE TABLE storageTemp (
 storageTempID int(3) NOT NULL default '0',
 minimumTemp int(3) default NULL,
 maximumTemp int(3) default NULL,
 PRIMARY KEY  (storageTempID)
) TYPE=MyISAM;
```

26. Table structure for table 'user' is the user table definition in MySQL. The detail explanation is given inside table 4.24.

```
CREATE TABLE user (
  loginName varchar(8) default NULL,
  firstName varchar(20) default NULL,
  lastName varchar(20) default NULL,
  email varchar(40) default NULL,
  password varchar(8) default NULL,
  comfirmPassword varchar(8) default NULL,
  workPlace varchar(25) default NULL,
  address varchar(50) default NULL,
  city varchar(20) default NULL,
  postcode varchar(10) default NULL,
  academic tinyint(1) default NULL,
  internal tinyint(1) default NULL
) TYPE=MyISAM;
```

Glossary

apoenzyme A protein that forms an active enzyme system by combination with a coenzyme and determines the specificity of this system for a substrate.

CAS number A number assigned by the Chemical Abstracts Service that uniquely identifies a chemical substance.

clone A collection of genetically identical copies of a gene, cell, or organism.

conidia Generic term refering to a spore produced in vegetative reproduction of a fungus and located on sporulating blotches.

culture Cultivation of living material in prepared nutrient media; *also* : a product of such cultivation

EC number A number assigned to a type of enzyme according to a scheme standardized enzyme nomenclature developed by the Enzyme Commission of the Nomenclature Committee of the International Union of Biochemistry and Molecular Biology (IUBMB).

fungus Any of a major group (Fungi) of saprophytic and parasitic spore-producing organisms usually classified as plants that lack chlorophyll and include molds, rusts, mildews, smuts, mushrooms, and yeasts

gene A DNA segment containing biological information and hence coding for an RNA and/or polypeptide molecule

genome All the genetic material in the chromosomes of a particular organism. Its size is generally given as its total number of base pairs.

holoenzyme A catalytically active enzyme consisting of an apoenzyme combined with its cofactor.

host An organism on which a parasite lives and by which it is nourished (also applied, loosely, to a plant supporting an epiphyte).

fluorescence The emission of electromagnetic radiation, especially of visible light, stimulated in a substance by the absorption of incident radiation and persisting only as long as the stimulating radiation is continued.

medium Any liquid or solid materials which is prepared for the growth, maintenance, or storage of microorganisms. There are different types of media according to the purposes: basal medium, defined medium, complex medium, differential medium, selective medium, enriched medium, enrichment medium, etc.

pH Negative logarithm of the concentration (mol/L) of the $H_3O^+[H^+]$ ion; scale is commonly used over a range 0 to 14.

plasmid Autonomously replicating, extrachromosomal circular DNA molecules, distinct from the normal bacterial genome and nonessential for cell survival under nonselective conditions. Some plasmids are capable of integrating into the host genome. A number of artificially constructed plasmids are used as cloning vectors.

protease Any enzyme involved in proteolysis or An enzyme that cuts (digests) proteins.

proteolysis The hydrolysis of proteins into simpler compounds by the action of enzymes: occurs esp. during digestion.

proteomics It evokes not only all the proteins in any given cell, but also the set of all protein isoforms and modifications, the interactions between them, the structural description of proteins and their higher-order complexes, and for that matter almost everything 'post-genomic'.

substrate The base on which an organism lives.

supernatant The usually clear liquid overlying material deposited by settling, precipitation, or centrifugation

vector A DNA molecule that replicates on its own in a host cell and can be used as a vehicle in the laboratory for replicating other types of DNA.