Database Systems and Oracle: Experiences and Lessons Learned

Deborah Dunn
Assistant Professor
Computer Science Department
Stephen F. Austin State University
Nacogdoches, TX 75961
(936) 468-2508
ddunn@sfasu.edu

Abstract

In a tight job market, IT professionals with database experience are likely to be in great demand. Companies need database personnel who can help improve access to and security of data. The events of September 11 have increased business' awareness of the need for database security, backup, and recovery procedures. It is our responsibility to prepare our students to meet the challenges of database development and system administration. This knowledge of creating, manipulating and analyzing data for decision making purposes is fundamental for those managers making business decisions. Our decision to change to Oracle is a perfect example of the need to make students more aware that a company's database platform may change and they must be capable of meeting the challenges of the change. This paper will focus on the pedagogical changes necessary to teach fundamental database concepts and the current technology needed to implement a system using Oracle.

Introduction

In the final report of the Computing Curricula 2001 project, a set of recommendations for an undergraduate program in Computer Science was identified [1]. The task force identified the *body of knowledge* as the fundamental topics which are essential for an undergraduate degree in computer science. Each of the areas was further subdivided into units that were categorized as either being part of the minimal *core* or an *elective*. During the development of the report, several principles were articulated to guide the task force, one of which was the acknowledgment that we "must recognize the importance of remaining abreast of progress in both technology and pedagogy" [1]. In addition, it was desirable to keep the body of knowledge small enough that it could be supplemented at the individual institution level to include other topics. Due to the constantly changing nature of technology, many topics have emerged (and will continue to do so) that some might be tempted to include in the body of knowledge. The quandary lies in the decision as to what should be removed to accommodate a new topic. Therefore, it is commonly recognized that the body of knowledge does not dictate an entire curriculum. It merely represents those topics for which there is a broad consensus that the information and concepts should be part of an undergraduate curriculum.

As computer science educators, our goal is to (1) produce students with a strong background in the discipline and (2) equip them with the tools necessary to be productive and remain current in the field. We are able to achieve the first goal by providing them with topics that are part of the fundamental body of knowledge. The second goal is achieved by supplementing our courses with those topics that emerge as technology changes. The specific area in the body of knowledge that

will be addressed in this paper is that of Information Management. The methods employed to achieve the first goal will be discussed briefly. The emphasis will be on the methods utilized to accomplish the second goal.

Course Structure

In recent years there have been a number of discussions targeting the content of database systems and the database course [2, 5, 7]. The topics identified by the task force as core topics are information models and systems (3 hours), database systems (3 hours), and data modeling (four hours) [1]. The general consensus seems to be that these topics are fundamental to any database systems course and it is reasonable to expect them to be covered adequately. The elective topics and pedagogy used to convey the information are typically what seem to differentiate a database systems course at one institution from that of other institutions.

Like many other institutions [6], we have elected to cover selected topics more extensively, including relational databases, database query languages, and relational database design. Included in this coverage are topics such as relational algebra, SQL, PL/SQL, functional dependencies, and normalization up to Boyce-Codd normal form. It is standard practice to adopt some type of database management system with which to implement many of the theoretical concepts covered in the course [5, 7]. Recent surveys [7] and panel discussions [2, 5] seem to indicate that Access and Oracle are the database management systems used most extensively in the academic community.

Environment

Prior to our move to Oracle, we utilized Access in the database systems course. In the spring of 2001 the decision was made to introduce Oracle into the course and use it as the primary database management system for the undergraduate course as well as the graduate course. Initially we hosted Oracle on a Sun machine running Solaris, but our system administrator was dissatisfied with the performance after the first year. The following year we were able to acquire new hardware and since the spring of 2003 we have been hosting Oracle 9i Release 2 on a Gateway 975 server running Redhat Enterprise Linux 3. We experienced a variety of problems in the initial setup, most of which had to do with Java and the Java libraries. The Oracle Technology Network provides a substantial amount of documentation for installing and administering Oracle 9i on Unix systems, as well as information regarding many of our subsequent issues. In addition, the system administrator located a web site that acted as an excellent resource for our initial setup [4]. The web site was our primary guide during the installation process. The fall of 2004 was our first opportunity to design and implement student projects using our new configuration.

Team Projects

The purpose of the team project is to provide students the opportunity to put theory into practice and gain proficiency using current database management software. It is used as a mechanism for helping the students learn and actually implement different components of database languages (in the form of SQL and PL/SQL) and different features of a database management system. More recently, the students are given the opportunity to build web applications if they so desire.

As stated earlier, the database management system of choice for this course is Oracle 9i. The students are provided with the requirements specification, which was adapted and modified from one of the textbooks used in the course [8]. It states that each project will typically go through the following phases:

1. Design the database using entity-relationship diagrams.
2. Create the tables, including constraints such as primary keys, foreign keys, check constraints, and any other constraints.
3. Create triggers and active elements to maintain the integrity of the database and to perform appropriate actions on database updates.
4. Populate the database using SQL **insert** statements or by writing programs.
5. Design and create any necessary queries.
6. Design and create any necessary forms and reports.
7. Write applications programs in Visual Basic, Java, or PL/SQL.
8. Document the project.

The project specification includes the requirement that the students construct the application programs to implement some form of user interface. The students are provided with descriptions of several projects that are similar in functionality and complexity. Each project must differ in the combination of project description/database application interface utilized. For example, two teams may select the same project description, but the application interface has to be different. Given three different project descriptions and three different choices for application interfaces, this provides the opportunity for nine distinct types of projects to be implemented, which is usually more than adequate for a typical class size.

Experiences

The objective of the undergraduate database course was to blend the theoretical framework of database systems with the practical application of a real-world database implementation experience. With this in mind, two textbooks were selected for the course. The Kifer [3] text and the first several chapters of the Sunderraman [8] text were the basis for much of the lecture material. However, one of the primary reasons for selecting the Sunderraman text was to provide a reliable, tangible reference for the team projects, regardless of the application interface selected. The students were encouraged to utilize those chapters that were applicable to their particular project. While Sunderraman's text does not include a reference for connecting to Oracle using Visual Basic, many of the students enter the course with some Visual Basic experience, including database access and manipulation.

Prior to the course, several small experimental projects were implemented and a set of notes was developed for connecting to the Oracle database. One lecture was spent demonstrating these projects and connecting to the database using each of the interfaces. What follows is an example of the type of information and the notes with which the students were provided in order to connect appropriately. Comments have been inserted where our server or database name was specified.

Establishing a connection to the Database using Java and JDBC.

1) Add the path to the Java Virtual Machine to your PATH environment variable
    a. /u01/app/oracle/jre/1.4.2/bin
    b. Make sure this is earlier in the list than any other JVM paths
2) Add the path for connecting to the database using Java to your PATH environment variable
    a. /usr/java/j2sdk1.4.2_03/bin:/u01/app/oracle/product/9.2.0/jdk/bin:$PATH:$HOME/bin
3) Add Oracle Library paths to the LD_LIBRARY_PATH environment variable
    a. /u01/app/oracle/product/9.2.0/lib:/lib:/lib/i686
    b. Oracle provides three different JDBC clients – some require these libraries, some do not.
4) Add required classes to the CLASSPATH environment variable
    a. $ORACLE_HOME/jdbc/lib/ojdbc14.jar:$ORACLE_HOME/jdbc/lib/classes12.zip:/u01/app/oracle/product/9.2.0/lib:$ORACLE_BASE/jre/1.3.1/lib/i386:/u01/app/oracle/product/9.2.0/jdbc/lib:/u01/app/oracle/product/9.2.0/jdk/lib:/u01/app/oracle/product/9.2.0/JRE/lib:$CLASSPATH
    b. For Swing, AWT, etc, you may need additional entries in the CLASSPATH
5) Be sure to export all of your environment variables, source .bash_profile, and check for errors in typing if you have any problems getting connected to the database.
6) Run a simple program to make sure you can connect:

Establishing a connection to the Database using VB and OLE DB.

1) Go to Start->Programs->Oracle-OraHome 92 -> Configuration and Migration Tools ->net configuration assistant
2) Select Local net service name configuration
3) Select Add
4) Oracle 8i or later (We run 9i.)
5) The service name for your database is {our database service name}.
6) Connect using TCP to {our server} on the standard port.
7) Perform a test (You'll probably need to change the username/password)
8) Set a net service name (this is what you will use to connect with ADO.NET)

In VB, to get a new connection:

- Go to the first tab and select OLE DB provider for Oracle. The server name is {our server}.

Now you have your connection, and you're off. Enjoy!

Establishing a Database Access Descriptor for use with PL/SQL server pages.

1) Go to {we provided the link to our server}
2) Select Gateway Database Access Descriptor Settings
3) Select add new Database Access Descriptor (default)
4) Set the name to be an appropriate subdirectory for your site
5) Set the Oracle User Name and Password to be your account on {our database}
6) Use {our connection string} for the connect string

7) Leave all other values blank.
8) Access your page via {link to our server} (whatever name you gave to your DAD)/(PL/SQL module you created)

Lessons Learned

Most database instructors would agree it is difficult to cover the traditional database concepts in a three-credit hour course [9]. Some believe it is unrealistic to include the addition of a full-blown real-world database project in a course whose content is steadily increasing [5]. Others may even argue that using a database management system such as Oracle is more difficult to learn [7].

With this in mind, our students were instructed to experiment and implement as many different features, or "bells and whistles", without worrying about creating an entire working system. In other words, implementing one trigger, one stored procedure, and one update query is much more beneficial than implementing one update for a customer, one update for an inventory item, and one update for an employee.

Lesson #1: Students *want* to create an entire working system. They are uncomfortable submitting a project that is, in their opinion, incomplete.

Solution: It must be emphasized that they are being given a broad learning experience rather than focusing on project completion. Their database skills increase as the number of features they implement and/or are exposed to increases.

As in many other database courses [7], the number and complexity of projects was a concern. As stated earlier, it was decided that the teams would implement one of a variety of projects using one of a variety of interfaces.

Lesson #2: Due to the disparity in the background and experience of many of the students, some of the database applications required much more work than others.

Solution: In order to level the playing field as well as improve the learning experience, all of the students should be exposed to the same set of database application interfaces. Again, this only serves to augment the database skills of the students.

Summary and Future Plans

As a result of this experience, the next database systems course will be a bit different. The topics that are currently being taught in the course will remain. We believe it is important that the students receive a solid background in concepts as well as practical development of database skills. We will continue to use Oracle as the primary database management system. In addition, a brief overview of Access will be incorporated into the course.

It is important that the team project approach continue to be utilized. However, it will be modified in a number of different ways. First, it is essential that the students have the experience

gained by implementing a variety of system mechanisms. This might be better accomplished by assigning several small project components rather than assigning a project that is perceived by the students as a complete system. Second, it may be more feasible to assign small projects and provide each group the opportunity to implement portions of a project using different application interfaces. Regardless of the implementation, the earlier stated goals, (1) producing students with a strong background in the discipline, and (2) equipping them with the tools necessary to be productive and remain current in the field, can be met by utilizing a project-based pedagogy.

Acknowledgements

I am grateful to Mr. Dennis Lingerfelt, my graduate assistant, for the assistance he provided in testing the class environment, preparing the notes for database connections, and working with the students. Special thanks to the students in the class who worked hard throughout the semester and provided constructive feedback on the course project.

References

1. ACM/IEEE-CS, *Final Report of the Joint ACM/IEEE-CS Task Force on Computing Curricula 2001 for Computer Science*, http://www.computer.org/education/cc2001/final/index.htm

2. Adams, E., Goelman, D., Granger, M., and Ricardo, C. Managing the Introductory Database Course: What Goes In and What Comes Out?, *SIGCSE Bulletin*, Volume 36, Number 1 (March 2004), 497-498.

3. Kifer, Bernstein, and Lewis, Database Systems: An Application-Oriented Approach, 2nd Edition, Addison Wesley Publishing, 2005.

4. Puschitz, W. Oracle-Linux Web Page, http://www.puschitz.com/InstallingOracle9i.shtml

5. Robbert, M. The Database Course: What Must be Taught, *SIGCSE Bulletin*, Volume 32, Number 1 (March 2000), 403-405.

6. Robbert, M. and Ricardo, C. Trends in the Evolution of the Database Curriculum, *SIGCSE Bulletin*, Volume 35, Number 3 (September 2003), 139-143.

7. Springsteel, F., Robbert, M., and Ricardo, C. The Next Decade of the Database Course: Three Decades Speak to the Next, *SIGCSE Bulletin*, Volume 32, Number 1 (March 2000), 41-45.

8. Sunderraman, *Oracle 9i Programming: A Primer*, Addison Wesley Publishing, 2004.

9. Urban, S. and Dietrich, S. Advanced Database Concepts for Undergraduates: Experience with Teaching a Second Course, *SIGCSE Bulletin*, Volume 33, Number 1 (March 2001), 357-361.