Part I

Database Concept and Design

The first part of the book is a stepping stone that will prepare you to embark on the journey of learning database programming. You will begin to know database systems, database language, relational data model, and will learn how to design efficient database systems using ER Modeling and normalization.

CHAPTER

1

Introduction to Database Systems

Objectives

- To know what a database system is.
- To understand database schemas: internal schemas, logical schemas, and external schemas, and their relationships.
- To understand logical data independence and physical data independence.
- To become familiar with the roles of database administrators, database application developers, and applications users.
- To become familiar with the functions of database management systems.
- To use simple SQL commands and execute them from MySQL command prompt, from Oracle SQL*PLUS and iSQL*Plus, and from Access.

1.1 Introduction

By now you may have heard a lot about database systems. Database systems are everywhere. Your social security information is stored in a database by the government. If you shop online, your purchase information is stored in a database by the company. If you attend a university, your academic information is stored in a database by the university. Database systems not only just simply store data, but also provide means of accessing, updating, manipulating, and analyzing data. Your social security information is updated periodically and you can register courses online. Database systems have played an important role for the society and for the commerce. This chapter

introduces how database systems work and sets the tone for the book.

1.2 What is a Database System?

A database system consists of a database, the software that stores and manages data in the database, and the application programs that present data and enable the user to interact with the database system, as shown in Figure 1.1.

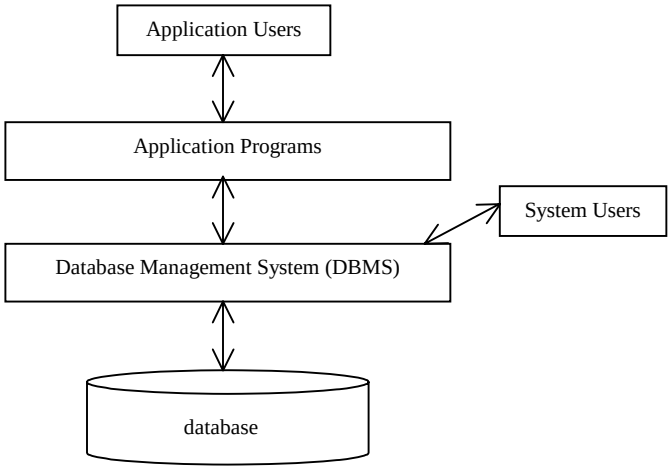

Figure 1.1

A database system consists of data, DBMS, and application programs.

Literally, a database is a repository of data that forms information. Strictly speaking, information and data are two different terms. Information is an interpretation of data, and data is stored in the database represented as numbers, characters, texts, images, etc. Nevertheless, the distinction between them can be ignored most of the time, so a database system is also known as an *information system*. The software that stores and manages data is known as *database management system*, or *DBMS*. DBMS is usually provided by database vendors. When you purchase a database system from a software vendor such as Oracle, IBM, Microsoft, and Sybase, you actually purchase the DBMS from the vendor. The database management systems are designed for use by the professional programmers and they are not suitable for the ordinary customers to use. The application programs are built on the DBMS for the customers to access and update the database. The application programs can be viewed as the interfaces between the database system and its users. The application programs may be standalone GUI applications or Web applications. The application programs may access several different database systems in the

network, as shown in Figure 1.2.

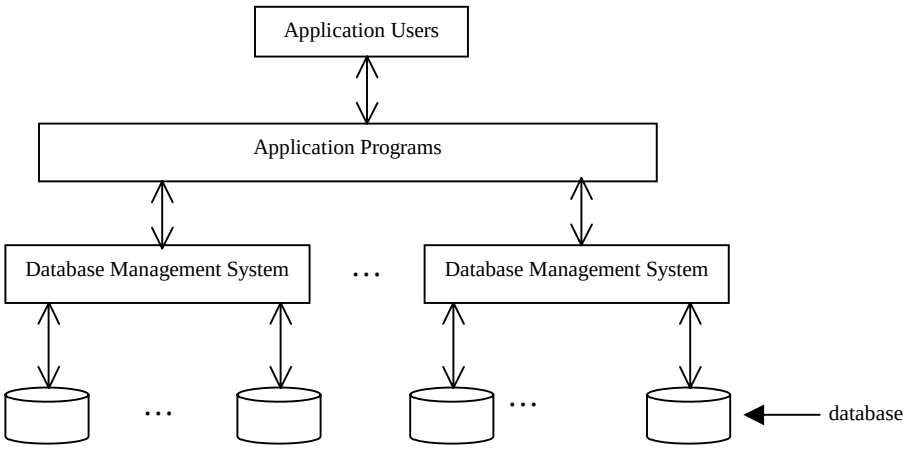

Figure 1.2

An application program may access multiple database systems.

Most of today's database systems are *relational database systems*, in which all of its data are stored in tables. A row of a table represents a record and a column of a table represents the value of a single attribute of the record. The relational data representation and manipulation is introduced in Chapter 2, "Relational Data Model." The ubiquitous language for accessing relational databases is the SQL (Structured Query Language), which will be introduced in Part II, "SQL." The focus of this book is to develop database application programs using SQL and Java.

1.3 Database Schemas

Database schemas describe the relationships, structures and constraints of the data in the database. There are three levels of schemas: *internal schema*, *logical schema*, and *external schema*. An internal schema defines how data is stored internally in the database. A logical schema presents a logical view of the data. An external schema presents part of the database that is interested to the users. This is also known as the *three-schema architecture*. The relationship among these schemas is shown in Figure 1.3.

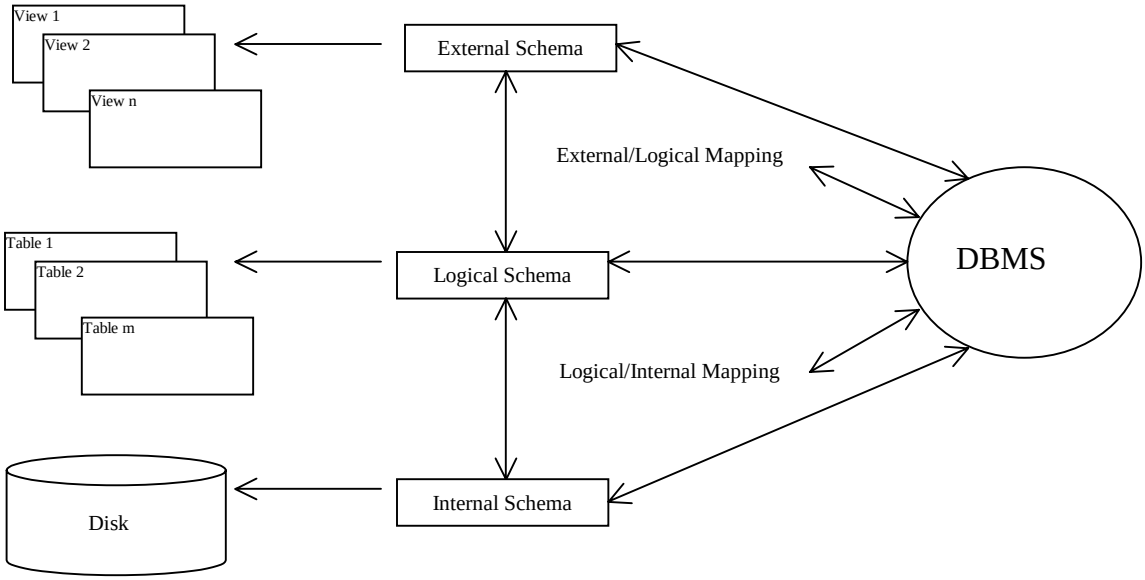

Figure 1.3

Database systems define data schemas at three levels.

1.3.1 Logical Schemas

Consider the Course table as shown in Figure 1.4. The logical schema of the Course table is shown in Figure 1.5. The schema defines the table name, the column names and types, and the constraints on the column values. The types define the range of the values. In the Course table, subjectId is a string of four characters, courseNumber and numOfCredits are integers, title is a variable-length string with maximum size 50. Constraints are additional restrictions on the columns. For example, the constraint in the Course schema is that numOfCredits must be greater than or equal to 1.

Course Table

courseId	subjectId	courseNumber	title	numOfCredits
11111	CSCI	1301	Introduction to Java I	4
11112	CSCI	1302	Introduction to Java II	3
11113	CSCI	3720	Database Systems	3
11114	CSCI	4750	Rapid Java Application	3
11115	MATH	2750	Calculus I	5
11116	MATH	3750	Calculus II	5
11117	EDUC	1111	Reading	3
11118	ITEC	1344	Database Administration	3

Figure 1.4

A course is described by course ID, subject ID, title, and credit hours.

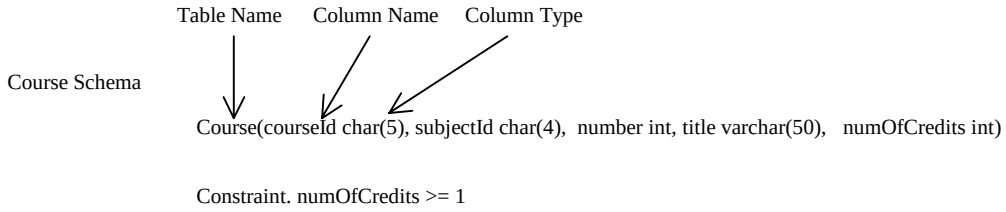

Figure 1.5

The schema of the Course table defines the table name, column names and types, and column constraints.

The schema is stored in the database. Who defines the schema and how the schema is defined? Usually the designer of the database defines the schema using the data definition language. The DBMS system processes the data definition language and stores schema into the database. For example, the Course table can be defined as follows:

```
create table Course(
  courseId char(5),
  subjectId char(4),
  courseNumber int,
  title varchar(50),
  numOfCredits int);
```

> NOTE: The table definition is descriptive and easy to read. However, you don't need to know the complete language syntax for defining database schemas at this time. The syntax for creating tables will be introduced in Chapter 4, "SQL."
>
> NOTE: This book adopts the following naming conventions. The tables are named in the same way as Java classes, and the attributes are named in the same was as Java variables. The SQL keywords are named in the same way as Java keywords.

1.3.2 Internal Schemas

The preceding statement creates a table. It defines table columns, types, and constraints, but it does not specify how a table is stored in the database. The internal schema can be used to specify a desired data structure for organizing the data in the table. If no internal schema is specified,

the system's default internal schema is used. If the table is used frequently for search on titles, you can create an index on the title attribute in the Course table to improve efficiency. The statement to create such a structure can be as follows:

```
create index CourseIndex on Course (title);
```

The internal schema defines how data is stored in the database, but not where data is stored. Where data is stored is specified by the database administrator and depends on the configuration of the database system.

1.3.3 External Schemas

The logic schemas define the entire logical view of the database system. A particular user needs to see only part of the database. An external schema can be used to define external views (usually abbreviated to just "view") that present the part of the database interested to the user. For example, you can create a view that consists of only the MATH courses as follows:

```
create view MathCourse
  as select * from Course where subjectId = 'MATH';
```

> NOTE: MySQL and Access currently don't support views.

1.3.1 Mappings of the Schemas

The external schema describes the part of the database to a particular user group, the logical schema describes the entire database to all users, and the internal schema describes the physical storage structures of the database. All the schemas are the descriptions of the database at three levels; the actual data is stored on disks. A user's request to access data through an external view involves actual tables and data structures for the table. The process of transforming requests and results between levels are called *mappings*. There are two types of mappings – *external/logical mapping* and *logical/internal mapping*. External/logical mapping handles the correspondents between the external schemas and logical schemas, and the logical/internal mapping handles the correspondents between the logical schema and internal schema. Mappings are handled by the DBMS and they are transparent to the users.

1.3.2 Data Independence

Obviously it takes time to handle mappings. So why do we need three schemas? The three-schema architecture achieves logical data independence and physical data independence and makes developing and maintaining database systems easier.

Logical data independence means that you can change the logical schema without changing the external schema. For example, you may have to modify the logical schema to expand the database by adding more tables. The existing external schemas are not changed. Therefore, the application programs that use the external schema remain intact.

Physical data independence means that you can change the internal schema without changing the logical schema. For example, you may have to modify the internal schema to add a new index in a table to improve the performance or drop an index if it is no longer needed. These changes do not affect the existing logical schema. So the applications that use the external schema or the logical schema are not affected.

1.4 Database Administrators, Developers, and End Users

A database system involves many people with different roles. Database administrators are the managers of the database systems. The developers are the people who build database applications for the users. Database end users are the people who use the applications. For example, a bank clerk who uses the database system to access the customer’s account is an end user for the bank database system. All the people who use the database are referred to as *database users*.

Database administrators (DBA) are responsible for creating user accounts, defining database schemas, allocating storage space for the database, maintaining database integrity, controlling access to the database system, and monitoring and optimizing database performance. DBAs coordinate with the developers and the users on database design and ensure the database is available and secured.

Database developers are the software professionals who develop database applications that extend the functionality of the database system. These applications can be standalone client/server or Web-based applications, report generating programs, and a variety of business applications. Database developers work with the users to determine the system requirements and design the database schemas and user interfaces. The developers work with the DBAs to create the schemas in the database.

1.5 Database Management Systems

A database management system (DBMS) is the software that handles all accesses and operations to the database. It provides an interface for database administrators to manage the database and for application programs to access the database. This section discusses its major functions.

1.5.1 Supporting Database Languages

Every DBMS supports the database language for accessing the database. The database language consists of three sublanguages: *data definition language (DDL), data manipulation language (DML), and data control language (DCL)*. Data definition language is for defining database schemas, data manipulation language is for querying and updating databases, and data control language is for managing transactions and maintaining securities. The statements may be issued interactively or embedded in a programming language such as Java. The DBMS compiles and executes the statements. The DBMS usually provides software that enables the user to enter commands interactively. For example, you can use MySQL's command line tool and Oracle's SQL*Plus or iSQL*Plus to enter SQL commands to access and manipulate MySQL and Oracle databases. In Chapter 7, "JDBC," you will learn to develop an interactive SQL interface using Java.

1.5.2 Implementing Database Schemas

Database administrators define the database schemas using the commands in the data definition language. The DBMS interprets the commands and implements the schemas in the database. Specifically, the DBMS supports the following functions:

- Mappings between the external schema and the logical schema and mappings between the logical schema and internal schema.

- Enforcing the constraints.

- Maintaining data dictionary. The schemas itself are stored as data in the database. This data is known as the database *meta data*.

1.5.3 Managing Transactions

Since a database is often accessed by multiple users simultaneously, it is important to control concurrent accesses to avoid database inconsistency. For example, a

bank database stores customer account information. Two users may access the same account at the same time to withdraw money from two different ATM machines. Without concurrency control, both customers may be able to withdraw all money from the account. To avoid problems like this, DBMS employs the mechanism to manage concurrent transactions.

It is possible that a transaction failure may occur before the transaction is completed. This would cause data corruption. To avoid it, it is the responsibility of the DBMS to recover the data and restore the database to the state prior to the execution of the transaction. DBMS usually employs the mechanism to log database updates and keep the copy of both old value and new value in a log file. In the event of a transaction failure, DBMS can restore the data using the log file. You will learn currency control and recovery in Chapter 6, “Transaction and Concurrency.”

1.5.4 Maintaining Security

A database is shared by many users, but not all the users can access all the information in the database and perform all types of operations in the database. For example, the students can query the Course table, but cannot update the Course table. The DBA can use the database language to grant or revoke privileges for accessing data and for performing certain operations. The DBMS checks the privileges to enforce the security. Typically, DBA creates accounts for users. The privileges and restrictions are associated with the accounts. You will learn maintaining database securities in Chapter 7, “Security.”

1.6 History of Database Systems

Prior to database systems, data were stored using file systems. File systems are inflexible. There are many drawbacks associated with the file systems. To develop applications using file-based systems, you have to write the code to deal with file structures at the operating systems level.

The first significant development in the database system dates back to early 1960s with the success of the Integrated Data Store (IDS) developed by a group led by Charles Bachman at General Electric. IDS was the major drive for the development of the first standard data model known as *networked DBMS* by the Conference on Data Systems Languages (CODASYL). In the middle 1960s, IBM developed a product called Information Management System (IMS) that uses a hierarchical data model to describe data and their relationship.

The relational data model was proposed by E. F. Codd in a 1970 research paper, "A Relational Model of Data for Large Shared Data Banks." Codd's research led to the development of the relational database systems. The experimental relational database *System R* developed at IBM's San Jose Research Lab in California demonstrated that a relational database system has the simplicity and flexibility for supporting database applications without sacrificing performance. The *Sequel* language used in System R is the basis for the SQL language (still pronounced "sequel"). In 1979, Relational Software, Inc. (now Oracle Corporation) introduced the first commercially implementation of SQL. The standardization of the SQL language by American National Standards Institute (ANSI) and International Standards Organization (ISO) in 1986 boosts the success of the relational database systems.

Today relational database systems dominate the database market. The trend will continue as companies migrate their legacy systems into Web-friendly relational database systems. There are hundreds of relational database management systems. They share the common SQL language, but not all the DBMSs support all the features in SQL. For example, MySQL does not support views. Some systems have their own extensions to SQL. For example, MySQL, Oracle and Access have their own proprietary functions. This book introduces standard SQL as well as SQL extensions in MySQL Oracle, and Access.

1.7 SQL

To access or write applications for database systems, you need to use the Structured Query Language (SQL). SQL is the universal language for accessing relational database systems. Application programs may allow users to access database without directly using SQL, but these applications themselves must use SQL to access the database.

You have used SQL to define database schemas. You can also use SQL to query and modify database. This section gives some examples of simple SQL statements.

To drop a table, use the drop table statement. For example, the following statement drops the Department table. Be careful, this command drops the table and its contents from the database.

```
drop table Department;
```

To see the definition of the table, use the describe

command. For example, the following statement shows the schema of the Department table.

```
describe Department; (Oracle and MySQL only)
```

To insert a record into a table, use the insert statement. For example, the following statement inserts a record ('SC', 'Science', '999001111') into the College table.

```
insert into College values ('SC', 'Science', '999001111');
```

To update a record, use the update statement. For example, the following statement changes the deanId to '888001111' for the Science college.

```
update College set deanId = '888001111' where name = 'Science';
```

To delete a record, use the delete statement. For example, the following statement deletes the Science college from the College table.

To query the database, use the select statement. For example, the following statement selects the departments in the Science college.

```
select name from Department where collegeCode = 'SC';
```

To find how many departments are in the Science college, you can use the following query:

```
select count(*) from Department where collegeCode = 'SC';
```

You will learn to write SQL statement in Chapters 4 and 5. SQL is a *non-procedural language*. As shown in the preceding statements, a SQL statement tells the database what to do, but not how to do. The DBMS translates SQL into procedures using the operators similar to relational algebra. Furthermore, the DBMS can optimize queries and choose the most efficient strategy to execute queries. All of these details are performed behind the scenes to make SQL simple and easy to use.

1.8 Database Applications Using Java

Before 1970s, database systems were mainly developed for use within organizations. The systems were primarily used for storing data and for performing tasks such as printing bank statements, telephone bills, and student transcripts. The users usually access the database through text-based terminals. These systems often don’t communicate with each other. With the advancement of computer network technologies, the networked database systems were developed

to enable companies and their partners to share data. The personal computers and workstations brought graphical user interfaces to the clients. The client/server database systems became a popular solution for database applications. Clients connect to a DBMS through the network. The database server is at the back end to process data and the client is at the front end to present the data and interact with the user. The clients were typically developed using Microsoft Visual Basic, Borland Delphi, and other development tools provided by database vendors, such as PowerBuilder by Sybase and Oracle Developer 2000 by Oracle. These are excellent rapid application development tools, but the programs developed suffer three problems. First, they can only run on certain platforms. Second, they have to be installed and updated on the clients. Third, they cannot run from a Web browser. Java came to the rescue. Java programs are platform independent so that they can run on any platform with a Java virtual machine.

Java is a full-featured, general-purpose, object-oriented programming language that is capable of developing robust mission-critical systems. Java can be used to develop standalone applications, applets, servlets and JSP. Java applications are like a traditional database client developed using MS Visual Basic. Java applets are special type of Java programs that can run from a Web browser. All these programs interact with the databases through JDBC, which is a Java API that provides a uniform interface for accessing and manipulating relational databases. You will learn how to develop database applications and applets using JDBC in Chapter 32, “Java Database Programming,” in the text.

Java servlets are the Java programs that run from the Web server to generate dynamic Web contents. Web browsers are universal client to interact with databases. Data is stored in databases. Servlets can generate Web pages that contain current information from the database. Servlets can also be used to update information in the database. Servlets can connect to any relational database via JDBC. For example, you can develop an HTML form to accept student registration information. Upon clicking the Submit button in the form, the servlet is invoked to store the registration information to the database. You will learn to develop dynamic Web pages using Java servlets in Chapter 10, “Servlets.” Using servlets, you have to embed HTML tags and text inside the Java source code. This makes the code difficult to maintain. Java ServerPages can be used to separate HTML tags and text from the Java code. JSP enables you to write regular static HTML code in the normal way and embed Java code to produce dynamic contents. JSP will be introduced in Chapter 11,

"Java ServerPages."

Using Java, you can not only develop programs that access relational databases, but also write functions, procedures and triggers in the database server such as in Oracle. Oracle has a built-in Java virtual machine that is capable of executing Java programs inside the database server. Java Stored Procedures are introduced in Appendix G, "Java Stored Procedures."

1.9 Getting Started with Database Systems

To develop database applications, you need to use a database server. There are more than one hundred DBMSs in the market. You can use any relational database like MySQL, Microsoft SQL server, Microsoft Access, IBM DB2, Sybase, Borland InterBase, and Oracle. This book chooses MySQL, Oracle, and Access for the reasons stated in the preface. It is important to emphasize that the database design and development concepts are universal and applicable to all database systems and the SQL examples in the book can run on all relational database systems provided that the DBMS supports SQL2.

1.9.1 Using MySQL

MySQL is a popular database with more than 4 million users. It is one of the fastest relational databases in the market. Many companies are using it to support their websites, data warehouses, and business applications. MySQL was developed by a Swedish company named MySQL AB. The product is distributed under GNU General Public License (GPL). You can download it free from www.mysql.com. MySQL runs on Windows, Linux and Solaris. It can support multiple users concurrently on the network. Students can access a MySQL database server standalone on their own computer or from the network. The installation is simple and straightforward. For help on installation, please see www.cs.armstrong.edu/liang/dbbook.html. This book demonstrates using MySQL from the Windows operating system.

Assume that you have installed MySQL with the default configuration; you can access MySQL from the DOS command prompt using the command mysql from the c:\mysql\bin directory, as shown in Figure 1.6.

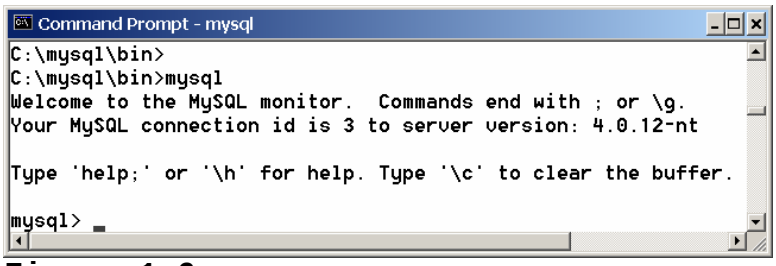

Figure 1.6

You can access a MySQL database server from the command window.

> NOTE: On Windows, your MySQL database server starts every time your computer starts. You can stop it by typing the command net stop mysql and restart it by typing the command net start mysql.

By default, the server contains two databases named mysql and test. You can see these two databases displayed in Figure 1.7 using the command show databases.

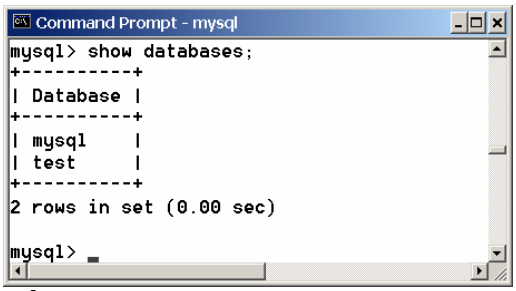

Figure 1.7

The show databases command displays all available databases in the MySQL database server.

The mysql database contains the tables that store the information about the server and its users. This database is intended for the server administrator to use. For example, the administrator can use it to create users and grant or revoke user privileges. Since you are the owner of the server installed on your system, you have full access to the mysql database. However, you should not create user tables in the mysql database. You can use the test database to store data or create new databases. You can also create a new database using the command create database *databasename* or drop an existing database using the command drop database *databasename*.

To select a database for use, type the use databasename command. Since the test database is created by default in

every MySQL database, let us use it to demonstrate SQL commands. As shown in Figure 1.8, the test database is selected.

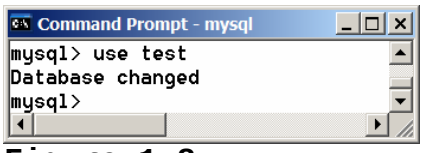

Figure 1.8

The use test command selects the test database.

Enter the following SQL statements from the MySQL command prompt, as shown in Figure 1.9. The create table statement creates a table named State. The insert statement inserts the values into the table. The select statement displays the contents from the table.

```
create table State(
  name varchar(15) not null,
  capital varchar(25),
  population integer);

insert into State values ('Georgia', 'Atlanta', 8383915);
insert into State values ('New York', 'Albany', 19011378);

select * from State;
```

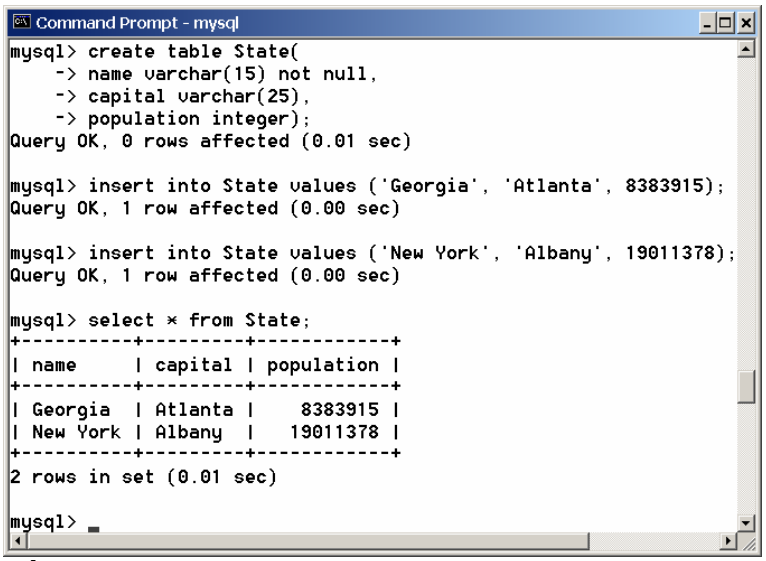

Figure 1.9

The execution result of the SQL statements is displayed in the MySQL command tool.

If you have typing errors, you have to retype the whole command. To avoid retyping the whole command, you can save the command in a file, and then run the command from the file. To do so, create a text file, e.g., named test.sql,

which contains the commands. You can create the text file using any text editor, e.g., NotePad, as shown in Figure 1.10. To comment a line, precede with two dashes. You can now run the script file by typing source test.sql from the SQL command prompt, as shown in Figure 1.11.

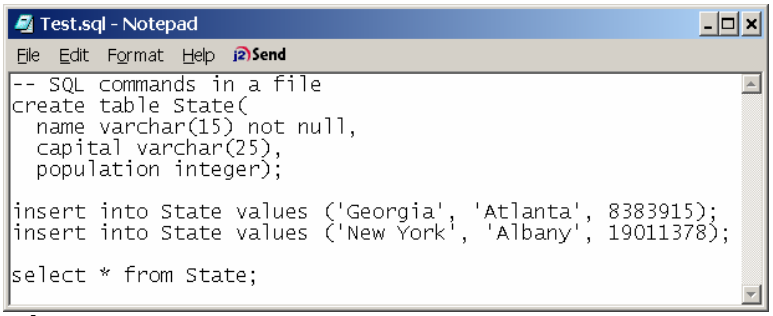

Figure 1.10

You can use Notepad to create a text file for SQL commands.

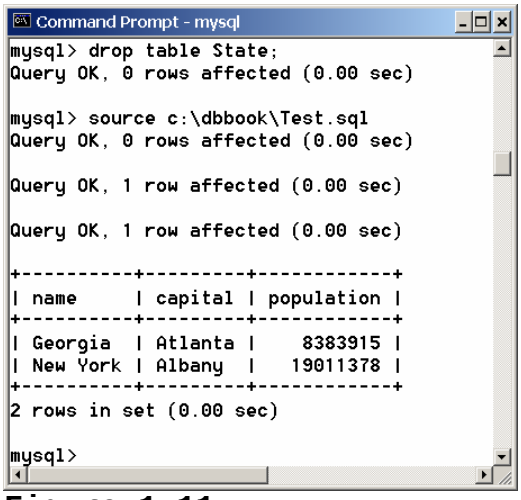

Figure 1.11

You can run the SQL commands in a script file from MySQL.

1.9.2 Using Oracle

Oracle 10g Enterprise runs on Windows 2000/NT/XP, Linux and Solaris. It can support multiple users concurrently on the network. Instructors can set up an Oracle 10g database on a server and let students access it from a Web browser so students are not required to install any Oracle software.

> NOTE: Students may choose to install Oracle 10g on their personal computer. Oracle 10g can be downloaded free from

http://www.oracle.com/technology/software/products/database/oracle10g/index.html.

There are many ways to access Oracle. The easiest is to use iSQL*Plus, which enables you to access Oracle from a Web browser. It requires no installation by the user. Suppose an Oracle 10g Enterprise database has been installed on the host liang.armstrong.edu with HTTP server enabled, the user can access it from a Web browser using the URL http://liang.armstrong.edu:5560/isqlplus, as shown in Figure 1.12.

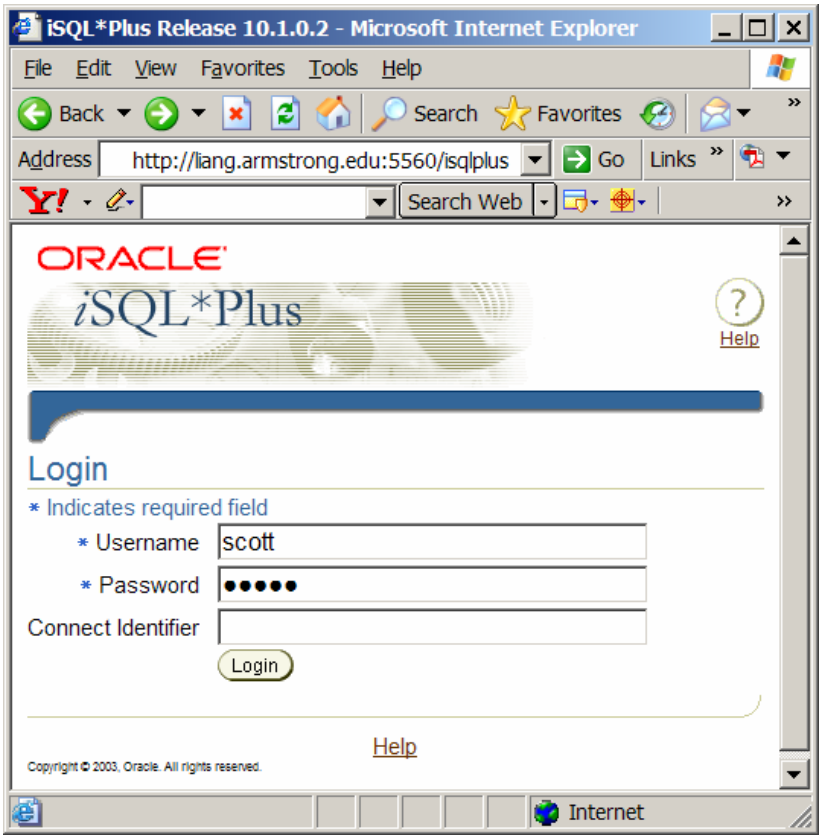

Figure 1.12

*You can start iSQL*Plus from a Web browser.*

Enter the username (e.g., scott) and password (e.g., tiger) and click Log In to log into the database. The *i*SQL*Plus user interface is shown in Figure 1.13.

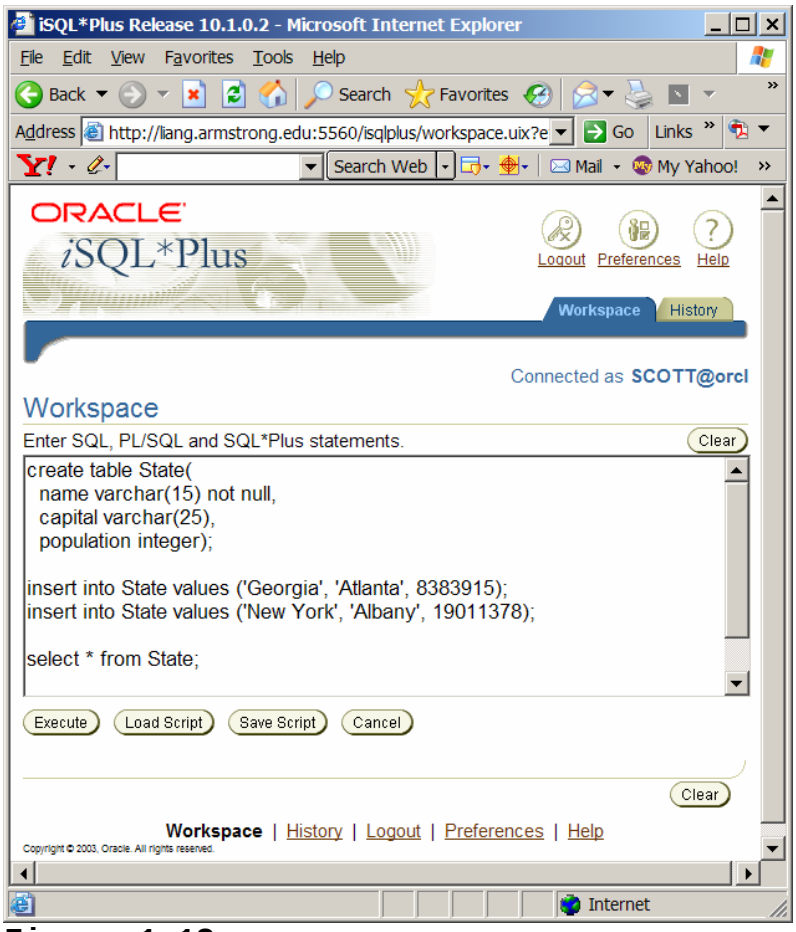

Figure 1.13

*You can enter SQL statements, save a SQL script, and load SQL script from iSQL*Plus.*

Enter the following SQL statements in the Enter Statement text box and click the Execute button. The execution result is displayed on the work screen, as shown in Figure 1.14.

```
create table State(
  name varchar(15) not null,
  capital varchar(25),
  population integer);

insert into State values ('Georgia', 'Atlanta', 8383915);
insert into State values ('New York', 'Albany', 19011378);

select * from State;

commit;
```

The commit statement ends the current transaction and makes permanent all changes performed in the transaction. Transactions will be covered in Chapter 6.

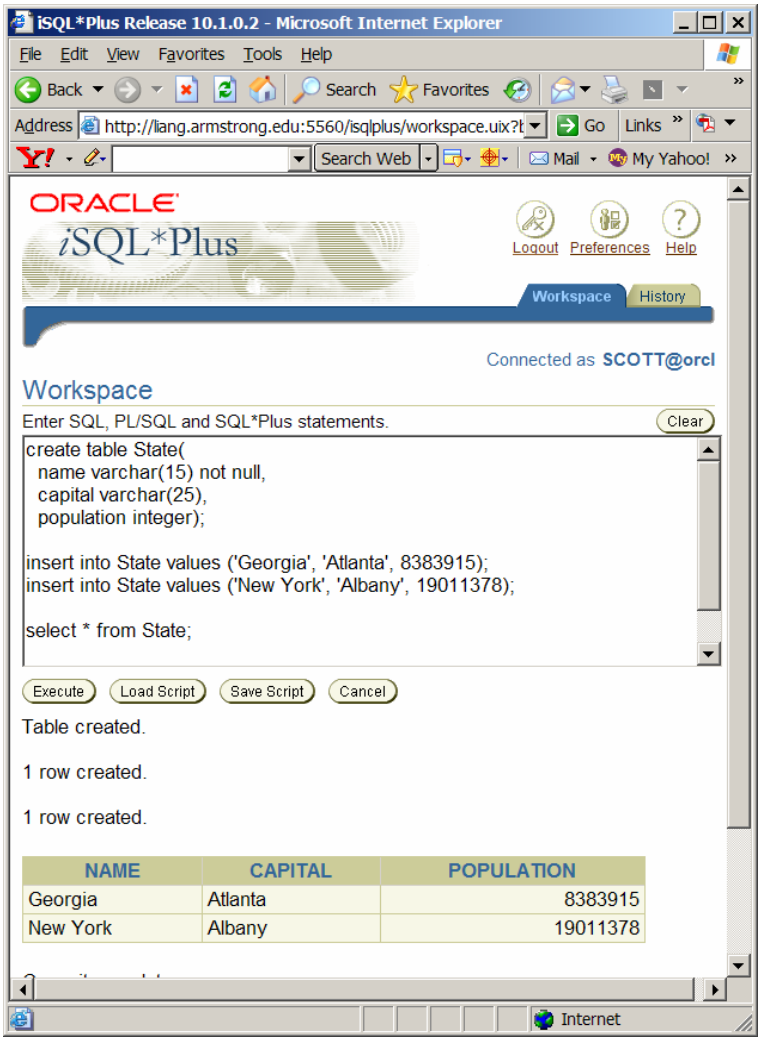

Figure 1.14

The execution result of the SQL statements is displayed in the work screen.

You can choose the Output type to display the result in the work screen, file, or a separate window. You can save the statements in a script file by clicking the Save Script button. You can load the script file from the local client by clicking the Browse button.

1.9.2.1 Accessing Oracle from Command Window (Optional)

Another way to access Oracle is through SQL*Plus. You have to first install SQL*Plus on your computer. If Oracle database is also installed on your machine, you can access it by typing **sqlplus scott/tiger** from the command window. If Oracle database is on a different machine, you need to create an Oracle network service name for the remote database. For the information on Oracle network service name, please see www.prenhall.edu/liang/db.html. Figure 1.15

shows an example of using SQL*Plus to access the oracle database on the host liang.armstrong.edu. The alias name for the database is liangorcl.

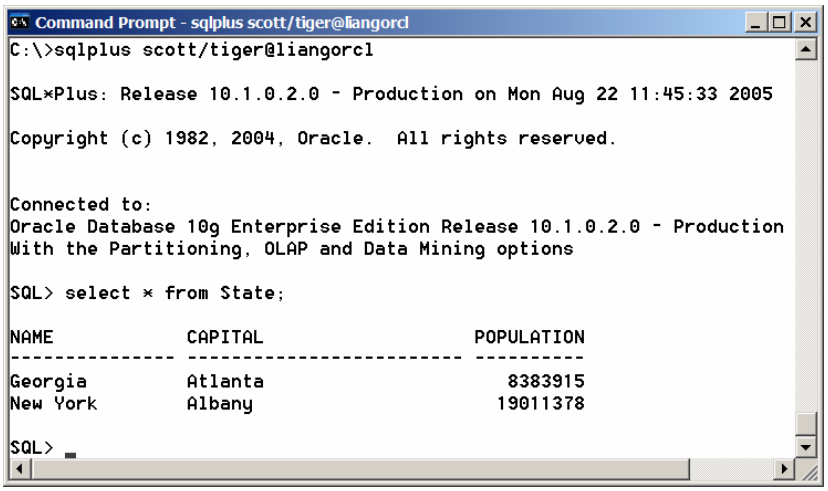

Figure 1.15

*You can start SQL*Plus from the DOS command prompt.*

If you have typing errors, you have to retype the whole command. To avoid retyping the whole command, you can save the command in a file, and then run the command from the file. To do so, type **edit *filename*.sql** from the SQL command prompt as shown in Figure 1.16. This command invokes Windows Notepad as shown in Figure 1.17. Type SQL commands in the Notepad and save and exit Notepad. To comment a line, precede with two dashes. You can now run the script file by typing *@filename*.sql, as shown in Figure 1.18.

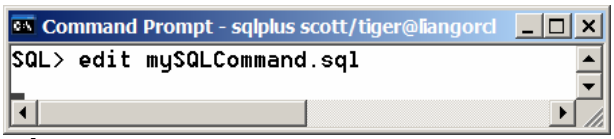

Figure 1.16

You can use the edit command to launch an editor.

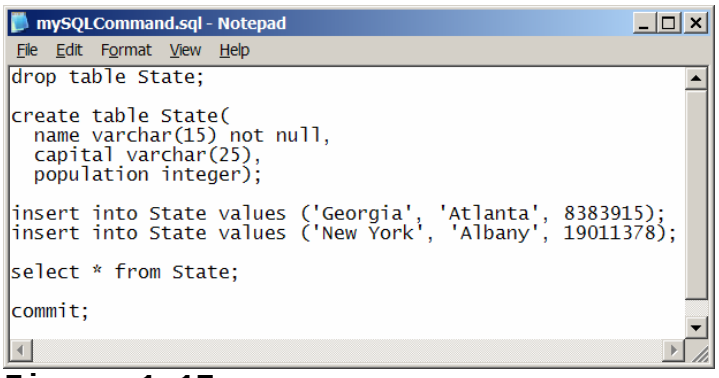

Figure 1.17

You can save the SQL commands using Notepad.

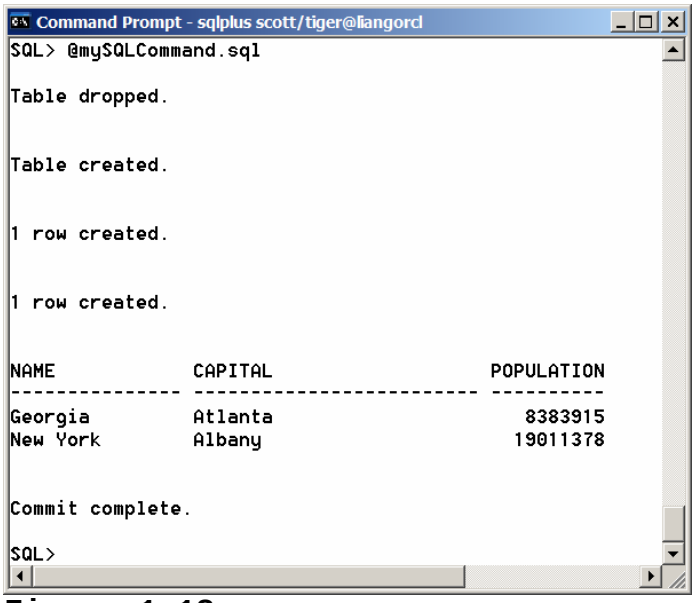

Figure 1.18

You can run the SQL commands in a script file from Oracle.

As shown in Figure 1.18, an error is generated when the create table statement is executed, because the table already exists. The insert statements cause errors because they violate the primary key constraint.

1.9.3 Using Access

Access is a ubiquitous database running on Microsoft Windows. It is usually used by a single user. Access provides an intuitive graphical user interface that enables you to create tables and insert, update, and delete data from the windows without using the SQL commands. However, to develop database applications using Java, you still have to learn and use SQL. This chapter demonstrates using SQL with Access.

Here are the steps to create a database and execute SQL statements from Microsoft Access:

1. Launch Microsoft Access database as shown in Figure 1.19. Check *Blank Access database* in the Create a new database section. Click *OK* to display the File New Database dialog box, as shown in Figure 1.20. Create and select the directory dbbook in the Save in field and type DBBook_MDB.mdb in the File name field. Click *Create* to create a new database. The DBBook_MDB database is created as shown in Figure 1.21.

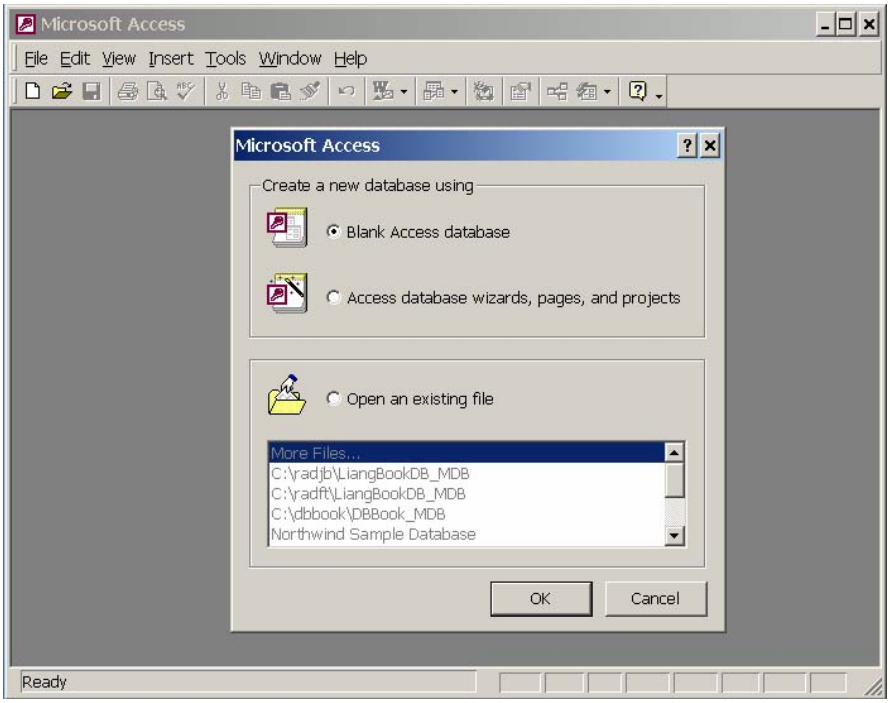

Figure 1.19

You can create a new database or use an existing database.

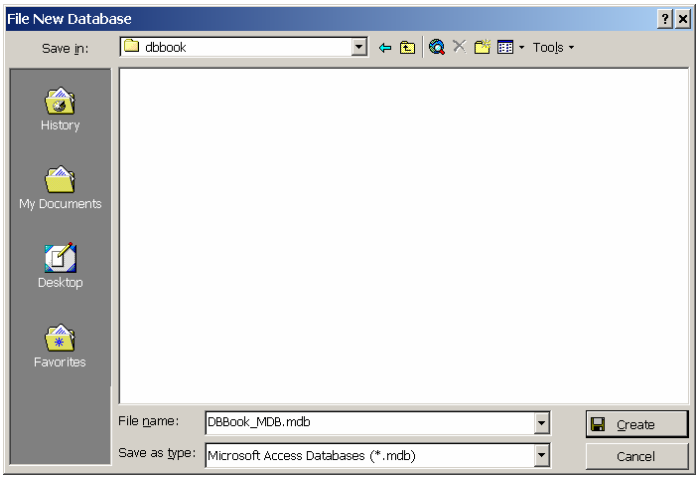

Figure 1.20

The File New Database dialog box enables you to specify a new database.

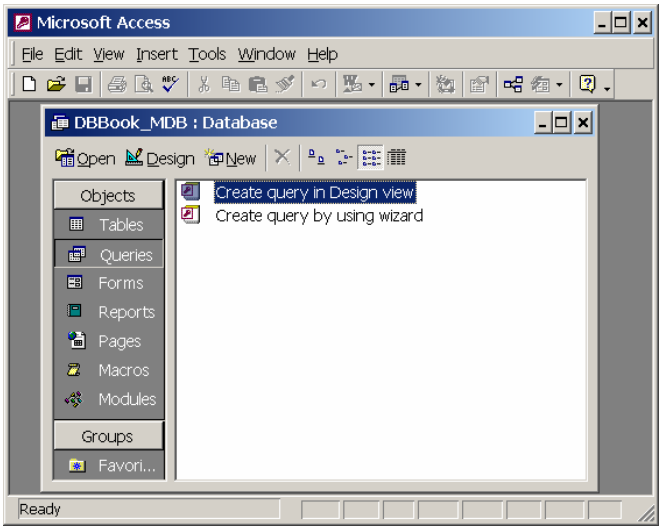

Figure 1.21

The DBBook_MDB database is created in c:\dbbook\DBBook_MDB.mdb.

> NOTE: An Access database is contained in a single file with a .mdb extension. If you have an existing database, you can open it without having to create a new database.

2. Choose *Queries* in the Objects column and click *Create query in Design View* (see Figure 1.21) to display the Query Design view (see Figure 1.22). Click *Close* to close the Show Table dialog box.

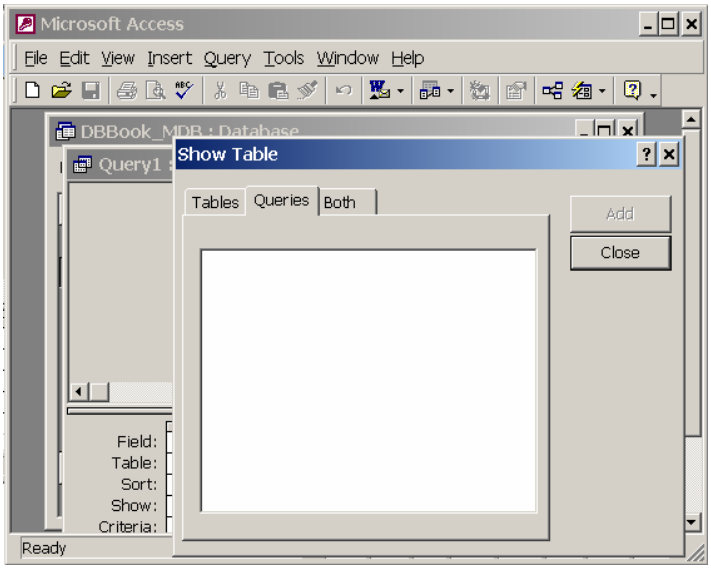

Figure 1.22

The Show Table dialog box must be closed to display the SQL view.

3. Select *SQL View* in the SQL combo box (see Figure 1.23) to display the SQL command window, as shown in Figure 1.24.

4. Type in the statement for creating the State table (see Figure 22.12) and click the *Run* toolbar button to execute the statement.

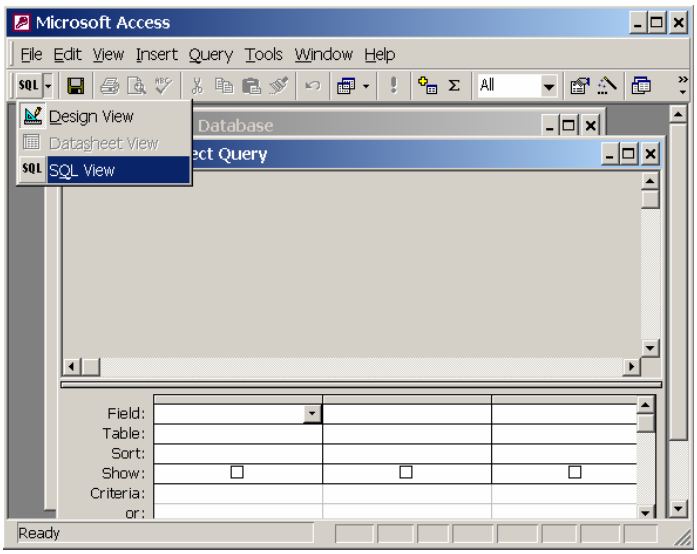

Figure 1.23

The SQL command window can be displayed by choosing the SQL View command.

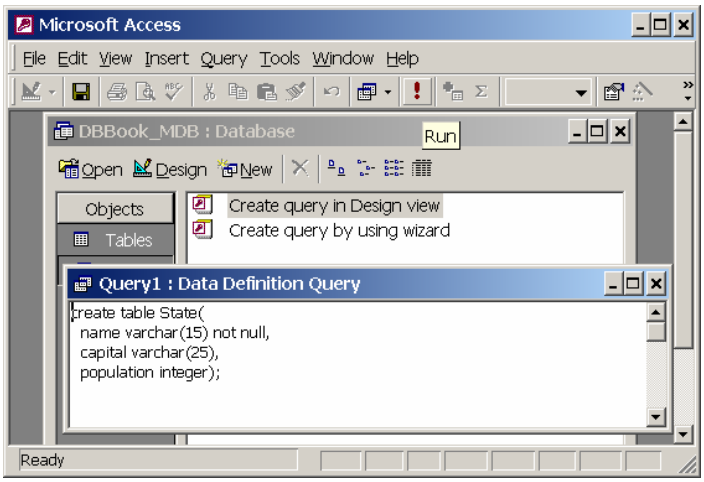

Figure 1.24

You can type the SQL command in the Query window and execute it.

5. Type the following SQL statements one at a time and execute each statement to insert data into the State table and select them from the table. The selection result is shown in Figure 1.25.

```
insert into State values ('Georgia', 'Atlanta', 8383915);
insert into State values ('New York', 'Albany', 19011378);
select * from State;
```

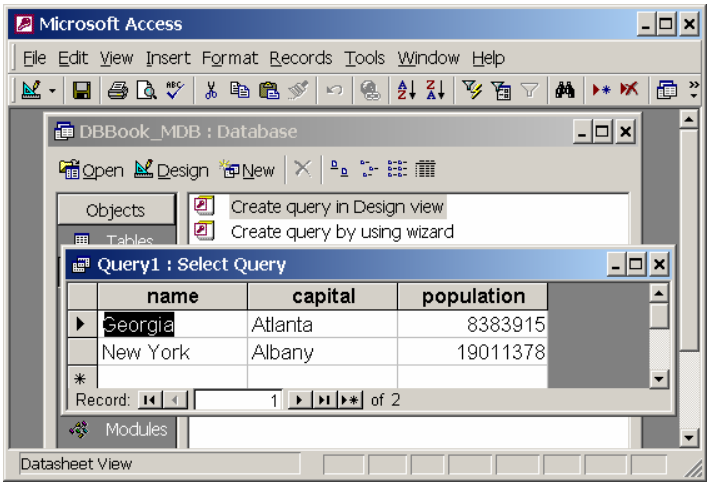

Figure 1.25

The selection result is displayed in a window on Access.

> NOTE: Access cannot execute SQL commands from a script file. You have to type and execute one SQL command at a time from the query window.

Chapter Summary

This chapter introduced the concept of database systems. You learned database management systems. You also learned how to access Oracle database from iSQL*Plus and SQL*Plus, and how to use simple SQL commands to create tables and manipulate data in the tables.

Review Questions

1.1 What is a database system?
1.2 What is a database schema? What are external schema, logical schema, and internal schema?
1.3 What is logical data independence? What is physical data independence?
1.4 What are the responsibilities of a DBA?
1.5 What are the functions of a DBMS?
1.6 What is the difference between a procedural language and non-procedural language?

Exercises

1.1 Create a table for Course, insert records into the table as shown in Figure 1.4.