13.1 Relational Databases

- A database is a collection of data organized to allow relatively easy access for retrievals, additions, and deletions.
- A relational database is a collection of tables of data, each of which has one special column that stores the primary keys of the table.
- Rows are sometimes called entities.
- Designing a relational database for used Corvettes that are for sale.
- Could just put all data in a single table, whose key would be a simple sequence number.
- The table could have information about various equipment the cars could have.
- Better to put the equipment in a different table and use a cross-reference table to relate cars to equipment.
- Use a separate table for state names, with only references in the main table.

13.2 Intro to SQL

- A standard language to create, query, and modify databases.
- Supported by all major database vendors.
- More like structured English than a programming language.
- We cover only six basic commands: CREATE TABLE, SELECT, INSERT, UPDATE, DELETE, and DROP.
- SQL reserved words are case insensitive.

- There are several constraints possible.
- E.g., NOT NULL, PRIMARY KEY.

- The SELECT command.
- Used to specify queries.
- Three clauses: SELECT, FROM, and WHERE.
- General form:

  SELECT column names
  FROM table names
  WHERE condition

- There are many different data types:

  INTEGER, FLOAT, CHAR(length),...

- The CREATE TABLE command:

  CREATE TABLE table_name (
    column_name1 data_type, constraints,
    column_name2 data_type, constraints,
    ...
  )
13.2 Intro to SQL (continued)

- **The INSERT Command**

  ```sql
  INSERT INTO table_name (col_name1, ..., col_namen)
  VALUES (value1, ..., value_n);
  ```

  - The correspondence between column names and values is positional

  ```sql
  INSERT INTO Corvettes(Vette_id, Body_style, Miles, Year, State)
  VALUES (37, 'convertible', 25.5, 1986, 17);
  ```

- **The UPDATE Command**

  - To change one or more values of a row in a table

  ```sql
  UPDATE table_name
  SET col_name1 = value1, ...
  WHERE col_name = value;
  ```

  - The `WHERE` clause is the primary key of the row to be updated

  ```sql
  UPDATE Corvettes
  SET Year = 1996
  WHERE Vette_id = 17;
  ```

- **The DELETE Command**

  ```sql
  DELETE FROM table_name
  WHERE col_name = value;
  ```

  - The `WHERE` clause could specify more than one row of the table

  ```sql
  DELETE FROM Corvettes
  WHERE Vette_id = 27;
  ```

- **The DROP Command**

  ```sql
  DROP (TABLE | DATABASE) [IF EXISTS] name
  ```

  ```sql
  DROP TABLE IF EXISTS States;
  ```

**Joins**

- If you want all cars that have CD players, you need information from two tables, `Corvettes` and `Equipment`.

- A `SELECT` that does a join operation specifies two tables in its `FROM` clause and also has a compound `WHERE` clause.

```sql
FROM Corvettes, Equipment
WHERE Corvettes.Vette_id = Corvettes_Equipment.Vette_id
AND Corvettes_Equipment.Equip = Equipment.Equip_id
AND Equipment.Equip = 'CD';
```

This query produces:

<table>
<thead>
<tr>
<th>VETTE_ID</th>
<th>BODY_STYLE</th>
<th>MILES</th>
<th>YEAR</th>
<th>STATE</th>
<th>EQUIP</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>coupe</td>
<td>18.0</td>
<td>1997</td>
<td>4</td>
<td>CD</td>
</tr>
<tr>
<td>2</td>
<td>hatchback</td>
<td>50.6</td>
<td>1996</td>
<td>7</td>
<td>CD</td>
</tr>
<tr>
<td>8</td>
<td>convertible</td>
<td>17.0</td>
<td>1999</td>
<td>5</td>
<td>CD</td>
</tr>
<tr>
<td>9</td>
<td>hardtop</td>
<td>17.0</td>
<td>2000</td>
<td>5</td>
<td>CD</td>
</tr>
<tr>
<td>10</td>
<td>hatchback</td>
<td>50.0</td>
<td>1995</td>
<td>7</td>
<td>CD</td>
</tr>
</tbody>
</table>

13.3 Architectures for Database Access

- **Client-Server Architectures**

  - **Client tasks:**
    - Provide a way for users to submit queries
    - Run applications that use the results of queries
    - Display results of queries

  - **Server tasks:**
    - Implement a data manipulation language, which can directly access and update the database
    - A two-tier system has clients that are connected directly to the server
    - Problems with a two-tier system:
      - Because the relative power of clients has grown considerably, we could shift processing to the client, but then maintaining data integrity is difficult

  - A solution to the problems of two-tier systems is to add a component in the middle - create a three-tier system

  - For Web-based database access, the middle tier can run applications (client just gets results)

  ![Client-Server Architecture Diagram]

**Database Access with Embedded SQL**

- SQL commands are embedded in programs written in a host programming language, whose compiler is extended to accept some form of SQL commands

- Advantage:
  - One package has computational support of the programming language, as well as database access with SQL.
13.3 Architectures for Database Access

Disadvantage (of embedded SQL):
- Portability among database systems

Microsoft Access Architecture
- A tool to access any common database structure
- Use either the Jet database engine, or go through the Other Database Connectivity (ODBC) standard
- ODBC is an API for a set of objects and methods that are an interface to different databases
- Database vendors provide ODBC drivers for their products – the drivers implement the ODBC objects and methods
- An application can include SQL statements that work for any database for which a driver is available

13.3 Architectures for Database Access

The Perl DBI/DBD Architecture
- Database Interface (DBI) provides methods & attributes for generic SQL commands
- Database Driver (DBD) is an interface to a specific database system (MySQL, Oracle, etc.)
- Convenient for Web access to databases, because the Perl program can be run as CGI on the Web server system

PHP & Database Access
- An API for each specific database system
- Also convenient for Web access to databases, because PHP is run on the Web server

13.4 The MySQL Database System

- A free, efficient, widely used SQL implementation
- Available from http://www.mysql.org
- Logging on to MySQL (starting it):
  mysql [-h host] [-u username] [database name] [p]
  - Host is the name of the MySQL server
  - Default is the user’s machine
  - Username is that of the database
  - Default is the name used to log into the system
  - The given database name becomes the “focus” of MySQL
  - If it is an existing database, but it was not named in the mysql command, you must choose one on which to focus
  use cars;
  - Response is: Database changed

13.5 Database Access with Perl/MySQL

Needed:
1. DBI – a standard object-oriented module
2. A DBD for the specific database system

DBI Module
- Get complete documentation from perldoc DBI
- Interface is similar to Perl’s interface to external files – through a filehandle
- To provide access to DBI and create a DBI object:
  use DBI;
- Access to the object is through the reference variable, $db;
- To connect to the database:
  $dbh = DBI->connect("DBI:mysql:database_name", [username] [password]);
13.5 Database Access with Perl/MySQL (continued)

- Example:
  
  ```perl
  $dbh = DBI->connect("DBI:mysql:cars");
  ```

  - Creates the db handle
  - Assumes the user name of the person logged in
  - Assumes the db does not need a password

  - The `connect` method is usually used with `die`

  - A Perl program can have connections to any number of databases

  - To create a query, we usually compile the SQL command first, then use it against the database

  - To create a compiled query, use `prepare`, as in:

    ```perl
    $sth = $dbh->prepare("SELECT Vette_id, Body_style, Year, States.State FROM Corvettes, States WHERE Corvettes.State = States.State_id AND States.State = 'California'");
    ```

  - To execute a compiled query, use `execute`, as in:

    ```perl
    $sth->execute() or die "Error - query: $dbh->errstr"
    ```

  The `$sth` object now has the result of the query

  - To display the results, we would like column names, which are stored in a hash

    ```perl
    $col_names = $sth->{NAME};
    ```

  - Rows of the result are available with the `fetchrow_array` method, which returns a reference to an array that has the next row of the result. (returns false if there are no more rows)

  - Note: Putting query results in an HTML document can cause trouble (>, <, " , and &)

    - Avoid the problem by using the CGI function, `escapeHTML`

    ```html
    SHOW access_cars.pl
    ```

    Query Results
    ```html
    Year, State, Body_style, Vette_id
    2003, California, Hardtop 030 1865
    2006, California, Convertible 222 12993
    2007, California, Convertible 272 2000
    ```

Chapter 13 © 2003 by Addison Wesley Longman, Inc.