What is a Database?

A database is more than just a collection of data. It organizes the way we access it, so that we have the ability to:

- Store information in a reliable and accessible way
- Access data via network
- Easily select a specific “view” of the data
- Have multiple users access it simultaneously
A database has a Server and a Client

Client software runs on users’ computers

Server software runs on computer where data is actually stored

(client software can also run on server computer)
Databases organize data in *Tables*

Like a spreadsheet, databases organize the data into tables with *rows* and *columns*.

Unlike a spreadsheet, each entry in a database is a complete row with a value for every column.
Databases organize data in Tables

- Tables are 2-dimensional. Modern spreadsheets add a 3\textsuperscript{rd} dimension with sheets.
- Similarly, databases add additional tables to gain a 3\textsuperscript{rd} dimension.
- Databases that can \textit{relate} columns from one table to another are called \textit{relational} databases

![Spreadsheet Image]

David Lawrence  JLab  6/10/08
Relational tables avoid redundancy

Avoid designing “super tables” that contain redundant information

Here, the Participants table links rows in the Experiments table with rows in the Researchers table.
Why MySQL?

- MySQL is a popular, commercial-quality database
- MySQL is well documented
- MySQL is free
- MySQL comes (optionally) installed on most common flavors of Linux
SQL is an ANSI standard

- SQL stands for *Structured Query Language*
- The ANSI SQL specification is independent of any specific database (i.e. MySQL, Postgres, Oracle, …)
- All commercial-grade databases extend their implementation of the language beyond the ANSI specification
- However for most small projects, the SQL can be written in a ANSI complaint way making the bulk of the code independent of the database itself
Introduction to SQL

- SQL queries tend to read like an English sentence:
  - `SELECT first_name FROM Friends`
  - `DELETE FROM Friends WHERE first_name="Bob"`

- A query starts with a command (verb) followed by a subject and then possibly additional clauses that qualify the command

  `SELECT first_name FROM Friends WHERE status="like"`
CREATE-ing a table

Create a table with the CREATE TABLE command

CREATE TABLE IF NOT EXISTS Experiments(
    name char(255),
    number char(32),
    expid int PRIMARY KEY AUTO_INCREMENT,
    created datetime,
    modified timestamp
);

(The “IF NOT EXISTS” clause is not in ANSI standard)
MySQL Data types

- BOOL
- BIT
- TINYINT (1 byte)
- SMALLINT (2 byte)
- INT (4 byte)
- BIGINT (8 byte)
- FLOAT (4 byte)
- DOUBLE (8 byte)
- CHAR or VARCHAR
- TEXT
- BLOB (<65kB)
- LONGBLOB (<4GB)
- ENUM
- SET
- DATETIME
- TIMESTAMP

*partial list*
INSERTing data into a table

- The INSERT command is used to create new rows in a table.

```sql
25  INSERT INTO Experiments VALUES("HAPPEX", "E-99-115", 1, NOW(), NOW());
27  INSERT INTO Experiments VALUES("PrimEx", "E-02-103", 2, NOW(), NOW());
28  INSERT INTO Experiments VALUES("QWeak", "E-08-016", 3, NOW(), NOW());
29  INSERT INTO Experiments VALUES("GlueX", "E12-06-102", 4, NOW(), NOW());
```

- This example specifies values for all columns of the Experiments table. However, one may specify values for only certain columns and default values will be used for the unspecified ones.
SELECT-ing data from a table

- To select more than one column, give a comma-separated list.
- To select all columns, use the wildcard "*".

```
mysql> SELECT name FROM Experiments;
+----------+
| name     |
+----------+
| HAPPEX   |
| PrimEx   |
| QWeak    |
| GlueX    |
+----------+
4 rows in set (0.00 sec)
```

David Lawrence   JLab   6/10/08
Multiple tables in a SELECT

- We want a list of all experiments that the UGR institution is participating in.

```sql
mysql> SELECT Experiments.name FROM Experiments,Researchers,Participants
   -> WHERE Experiments.expid=Participants.expid
   -> AND Researchers.userid=Participants.userid
   -> AND institution="UGR";

+------------------+
| name             |
| HAPPEX           |
| HAPPEX           |
| GlueX            |
| GlueX            |
+------------------+
4 rows in set (0.04 sec)
```
We want to know how many experiments each researcher is participating in.

```sql
mysql> SELECT name, count(Participants.userid) AS numexperiments
    -> FROM Participants, Researchers
    -> WHERE Participants.userid=Researchers.userid
    -> GROUP BY Participants.userid;
```

<table>
<thead>
<tr>
<th>name</th>
<th>numexperiments</th>
</tr>
</thead>
<tbody>
<tr>
<td>John Doe</td>
<td>2</td>
</tr>
<tr>
<td>Jane Smith</td>
<td>2</td>
</tr>
<tr>
<td>Kelley Cook</td>
<td>2</td>
</tr>
<tr>
<td>Pat Jones</td>
<td>2</td>
</tr>
<tr>
<td>Carl Stanley</td>
<td>1</td>
</tr>
<tr>
<td>David Lawrence</td>
<td>2</td>
</tr>
</tbody>
</table>

6 rows in set (0.34 sec)
ORDER BY and LIMIT

- With ORDER BY and LIMIT, we can have the server re-order the rows by the contents of a column and only return us a subset of rows.
- Useful for displaying a web page with N items per page.
**SELECT** can combine columns

The **SELECT** command can combine columns mathematically to dynamically form a new column (temporarily) for the query.
UPDATE-ing data in a table

mysql> SELECT * FROM Researchers;
+----------+----------------+-------------------+-------+-------------------+-------------+
<table>
<thead>
<tr>
<th>userid</th>
<th>institution</th>
<th>email</th>
<th>name</th>
<th>created</th>
<th>modified</th>
</tr>
</thead>
<tbody>
<tr>
<td>37</td>
<td>UGR</td>
<td><a href="mailto:jdoe@ugr.edu">jdoe@ugr.edu</a></td>
<td>John Doe</td>
<td>2008-06-08 13:11:58</td>
<td>2008-06-08 13:11:58</td>
</tr>
<tr>
<td>63</td>
<td>URT</td>
<td><a href="mailto:k127@hotmail.net">k127@hotmail.net</a></td>
<td>Kelley Cook</td>
<td>2008-06-08 13:11:58</td>
<td>2008-06-08 13:11:58</td>
</tr>
<tr>
<td>108</td>
<td>UGR</td>
<td><a href="mailto:pjohns@ugr.edu">pjohns@ugr.edu</a></td>
<td>Pat Jones</td>
<td>2008-06-08 13:11:58</td>
<td>2008-06-08 13:11:58</td>
</tr>
<tr>
<td>122</td>
<td>IOP</td>
<td><a href="mailto:carl@iop.ru">carl@iop.ru</a></td>
<td>Carl Stanley</td>
<td>2008-06-08 13:11:58</td>
<td>2008-06-08 13:11:58</td>
</tr>
<tr>
<td>137</td>
<td>JLab</td>
<td><a href="mailto:davidl@jlab.org">davidl@jlab.org</a></td>
<td>David Lawrence</td>
<td>2008-06-08 13:11:58</td>
<td>2008-06-08 13:11:58</td>
</tr>
</tbody>
</table>
+----------+----------------+-------------------+-------+-------------------+-------------+
6 rows in set (0.00 sec)

mysql> UPDATE Researchers SET email="patj@ugr.edu" WHERE userid=108;
Query OK, 1 row affected (0.04 sec)
Rows matched: 1  Changed: 1  Warnings: 0

mysql> SELECT * FROM Researchers;
+----------+----------------+-------------------+-------+-------------------+-------------+
<table>
<thead>
<tr>
<th>userid</th>
<th>institution</th>
<th>email</th>
<th>name</th>
<th>created</th>
<th>modified</th>
</tr>
</thead>
<tbody>
<tr>
<td>37</td>
<td>UGR</td>
<td><a href="mailto:jdoe@ugr.edu">jdoe@ugr.edu</a></td>
<td>John Doe</td>
<td>2008-06-08 13:11:58</td>
<td>2008-06-08 13:11:58</td>
</tr>
<tr>
<td>63</td>
<td>URT</td>
<td><a href="mailto:k127@hotmail.net">k127@hotmail.net</a></td>
<td>Kelley Cook</td>
<td>2008-06-08 13:11:58</td>
<td>2008-06-08 13:11:58</td>
</tr>
<tr>
<td>108</td>
<td>UGR</td>
<td><a href="mailto:patj@ugr.edu">patj@ugr.edu</a></td>
<td>Pat Jones</td>
<td>2008-06-08 13:11:58</td>
<td>2008-06-08 13:11:58</td>
</tr>
<tr>
<td>122</td>
<td>IOP</td>
<td><a href="mailto:carl@iop.ru">carl@iop.ru</a></td>
<td>Carl Stanley</td>
<td>2008-06-08 13:11:58</td>
<td>2008-06-08 13:11:58</td>
</tr>
<tr>
<td>137</td>
<td>JLab</td>
<td><a href="mailto:davidl@jlab.org">davidl@jlab.org</a></td>
<td>David Lawrence</td>
<td>2008-06-08 13:11:58</td>
<td>2008-06-08 13:11:58</td>
</tr>
</tbody>
</table>
+----------+----------------+-------------------+-------+-------------------+-------------+
6 rows in set (0.00 sec)
DELETE-ing data from a table

```
mysql> INSERT INTO Experiments (name,number,created) VALUES("dummy","E09-123",NOW());
Query OK. 1 row affected (0.00 sec)

mysql> SELECT * FROM Experiments;
+---------------------+-------+-----+-------+-------+
<table>
<thead>
<tr>
<th>name</th>
<th>number</th>
<th>expid</th>
<th>created</th>
<th>modified</th>
</tr>
</thead>
</table>
+---------------------+-------+-----+-------+-------+
5 rows in set (0.00 sec)

mysql> DELETE FROM Experiments WHERE expid=5;
Query OK. 1 row affected (0.00 sec)
```
The mysql tools

- `mysql` – interactive command line tool
- `mysqldump` – dump contents (including table definitions) of a database
- `mysqlshow` – show info about tables, databases, etc.
- `mysql_config` – print C/C++ compiler options for current platform
import java.sql.*;

public class java_api_test {

    static public void main (String[] args) {

        try{
            // load driver and connect to database
            Class.forName("com.mysql.jdbc.Driver");
            java.sql.Connection con =
                DriverManager.getConnection("jdbc:mysql://localhost/test","davidl",null);

            // send query to database
            Statement s = con.createStatement();
            ResultSet res = s.executeQuery("SELECT * FROM Researchers");

            // loop over results
            while(res.next()){
                System.out.println(res.getString("name") + " +" + res.getString("email");
            }
        } catch(Exception e) {System.out.println(e.toString());}

    } // main
}
#include <stdio.h>
#include <mysql.h>

int main(int argc, char *argv[]) {
    // Initialize MYSQL handle and connect to database
    MYSQL *mysql = mysql_init(NULL);
    mysql_real_connect(mysql, "localhost", "davidl", NULL, "test", 0, NULL, 0);

    // Send query to database
    mysql_query(mysql, "SELECT * FROM Researchers");

    // Loop over rows in result
    MYSQL_RES *res = mysql_store_result(mysql);
    while(MYSQL_ROW row = mysql_fetch_row(res)){
        unsigned long *lengths = mysql_fetch_lengths(res);

        // Loop over fields in row, printing each to screen
        for(int i=0; i<mysql_num_fields(res); i++){
            printf("%s", lengths[i], row[i] ? row[i] : "NULL");
        }
        printf("\n");
    }

    // Close connection to database
    mysql_close(mysql);
}

David Lawrence   JLab   6/10/08
Accessing the database with PHP

```php
<?php

try{
    // Connect to database
    $user = "davidl";
    $dbh = new PDO('mysql:host=localhost;dbname=test', $user);
    // Send query to database and loop over results
    foreach($dbh->query('SELECT * FROM Researchers') as $row){
        print($row[name]." ".$row[email]."\n");
    }
    } catch (PDOException $e) {
        print "Error connecting to database : ".$e->getMessage();
        die();
}
?>
```
PHP embedded in HTML

```php
<?php
try{
   // Connect to database
   $user = "davidl";
   $dbh = new PDO('mysql:host=localhost;dbname=test', $user);
   // Send query to database and loop over results
   foreach($dbh->query("SELECT * FROM Researchers") as $row){
      print("<TR><TD>".$row[name]."</TD><TD>".$row[email]."</TD></TR>");
   }
} catch(PDOException $e) {
      print "Error connecting to database : ".$e->getMessage();
      die();
}?
</table>
</BODY>
</HTML>
```
PHP embedded in HTML
Other features of MySQL

- Stored Procedures / Functions
- Transactions
- Triggers
- Partitions
- Views
- Indexes
- Replication
- Scheduled Events
Summary

- Databases organize data in a reliable, accessible way that allow remote users to access the data from any number of “views”
- MySQL is a commercial-grade, freely available database that provides ANSI SQL compliance
- SQL is a well-documented and a relatively easy syntax to learn
- MySQL databases can be accessed from most any programming language