INTRODUCTION TO MySQL

Powered by

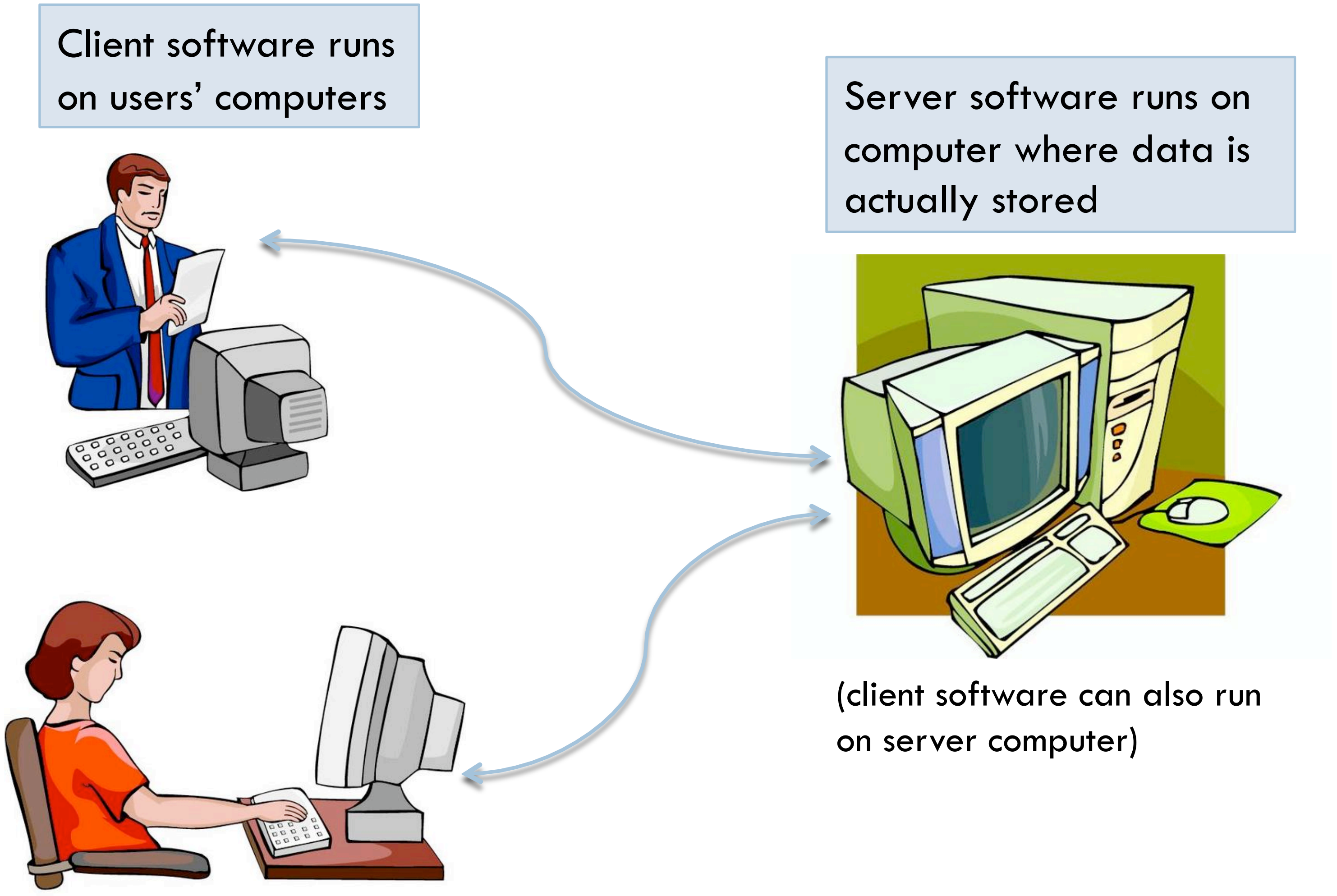
David Lawrence, Jlab June 10, 2008

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

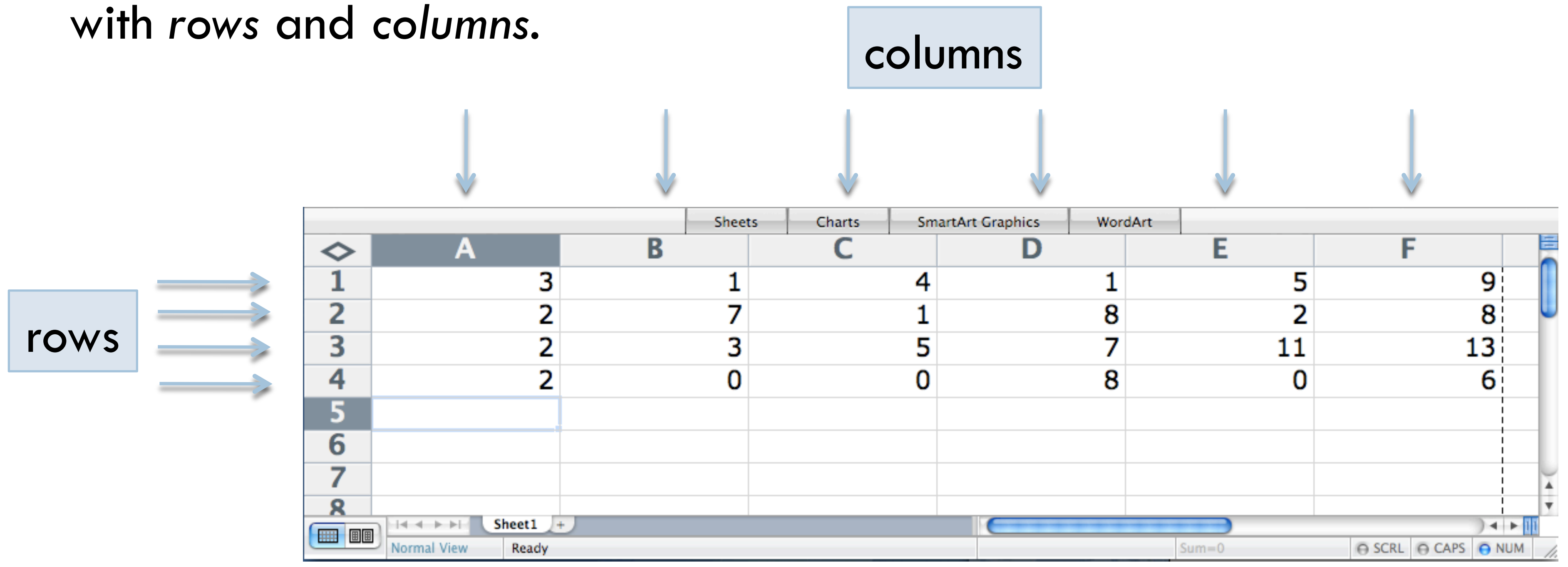
A database has a *Server* and a *Client*


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*.

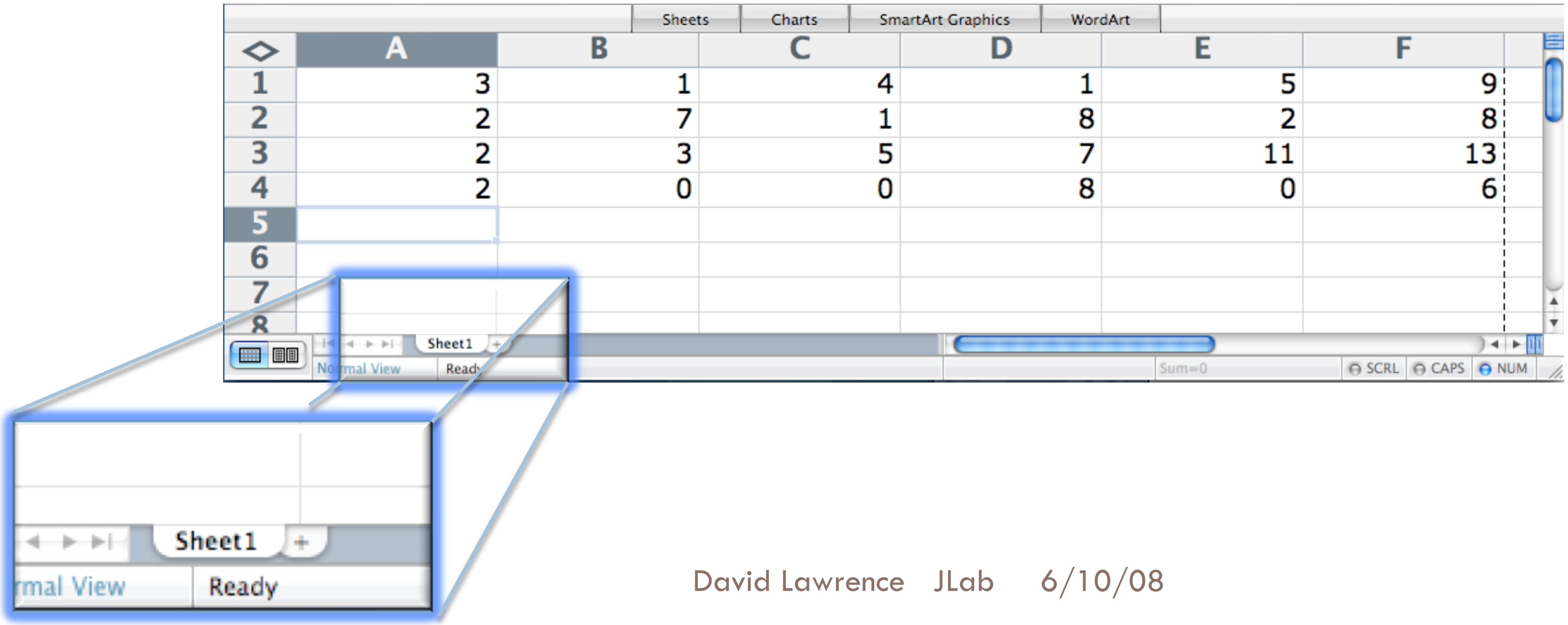


	A	B	C	D	E	F
1	3	1	4	1	5	9
2	2	7	1	8	2	8
3	2	3	5	7	11	13
4	2	0	0	8	0	6
5						
6						
7						
8						

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 3rd dimension with *sheets.*
- Similarly, databases add additional tables to gain a 3rd dimension.
- Databases that can *relate* columns from one table to another are called *relational databases*



	A	B	C	D	E	F
1	3	1	4	1	5	9
2	2	7	1	8	2	8
3	2	3	5	7	11	13
4	2	0	0	8	0	6
5						
6						
7						
8						

Relational tables avoid redundancy

Experiments

name	number	expid
HAPPEX	E-99-115	1
PrimEx	E-02-103	2
QWeak	E-08-016	3
GlueX	E12-06-102	4

Researchers

userid	institution	email	name
37	UGR	jdoe@ugr.edu	John Doe
45	IPET	janes@ipet.gov	Jane Smith
63	URT	k127@hotmail.net	Kelley Cook
108	UGR	pjones@ugr.edu	Pat Jones
122	IOP	carl@iop.ru	Carl Stanley
137	JLab	davidl@jlab.org	David Lawrence

Participants

expid	userid
1	37
1	108
1	137
2	122
3	45
3	63
4	37
4	45
4	63
4	108
4	137

- 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

Powered by MySQL®

- 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

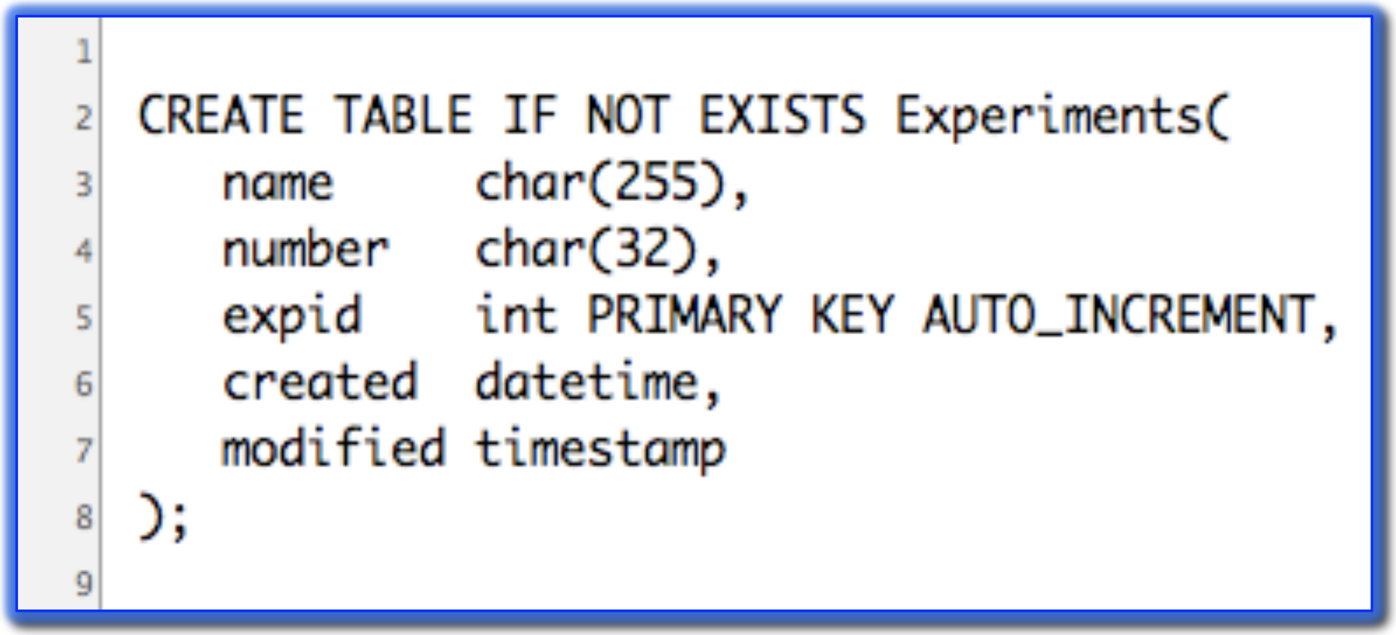
- SQL queries tend to read like an English sentence:
  - *SELECT first_name FROM Friends*
  - *DELETE FROM Friends WHERE first_name="Bob"*

- 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)

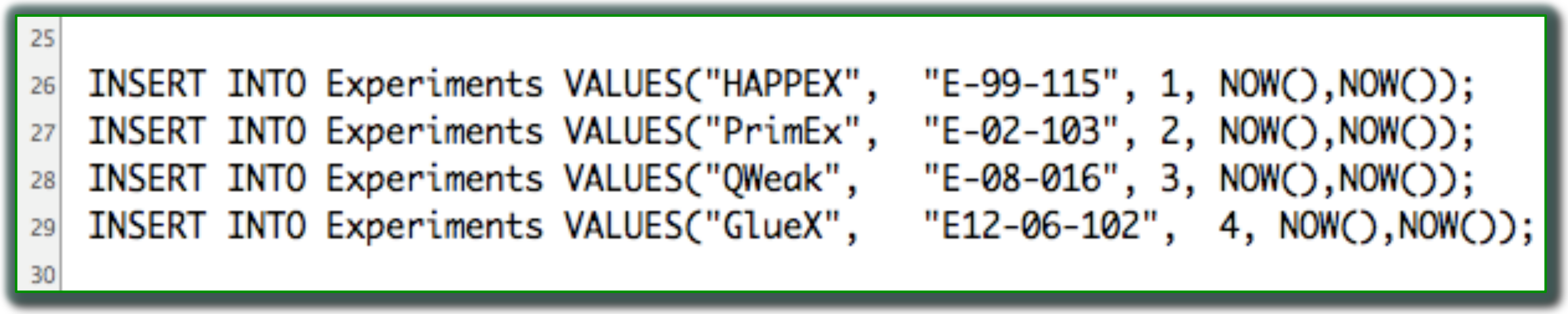
MySQL Data types

*partial list

- 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

*INSERT*ing data into a table

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



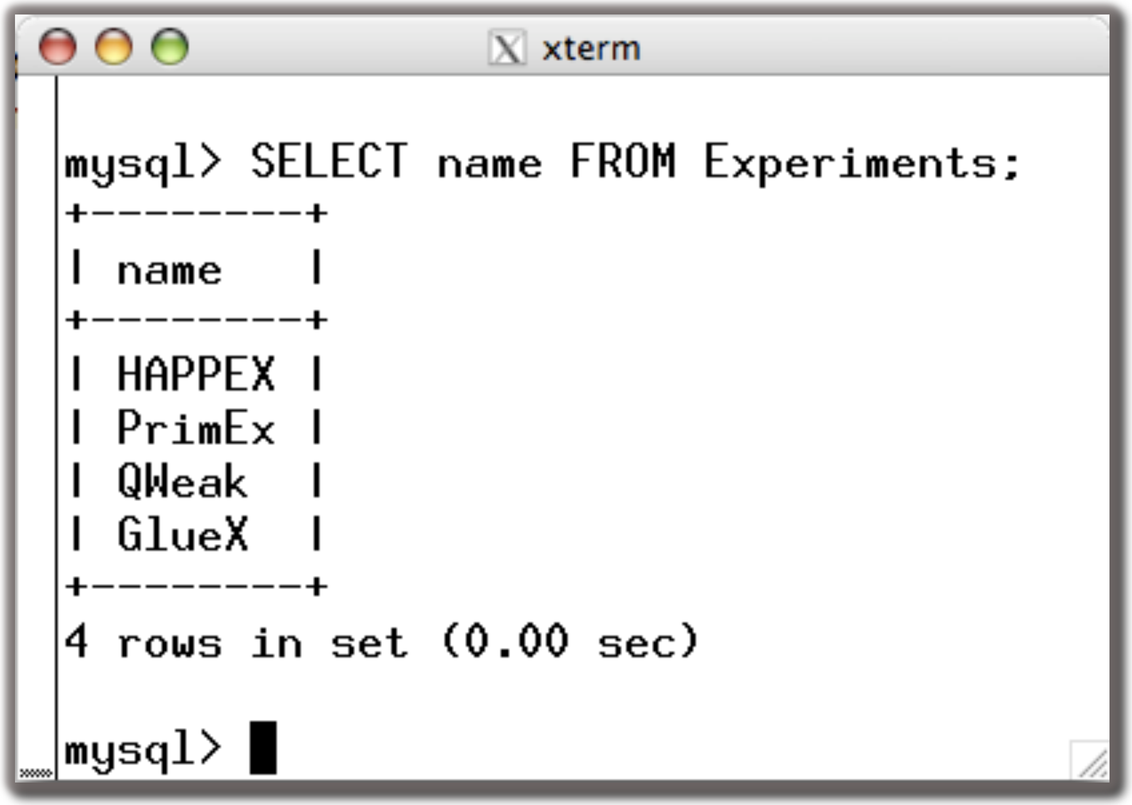
```
INSERT INTO Experiments VALUES("HAPPEX",  "E-99-115", 1, NOW(),NOW());
INSERT INTO Experiments VALUES("PrimEx",  "E-02-103", 2, NOW(),NOW());
INSERT INTO Experiments VALUES("QWeak",   "E-08-016", 3, NOW(),NOW());
INSERT INTO Experiments VALUES("GlueX",   "E12-06-102",  4, NOW(),NOW());
```

- 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

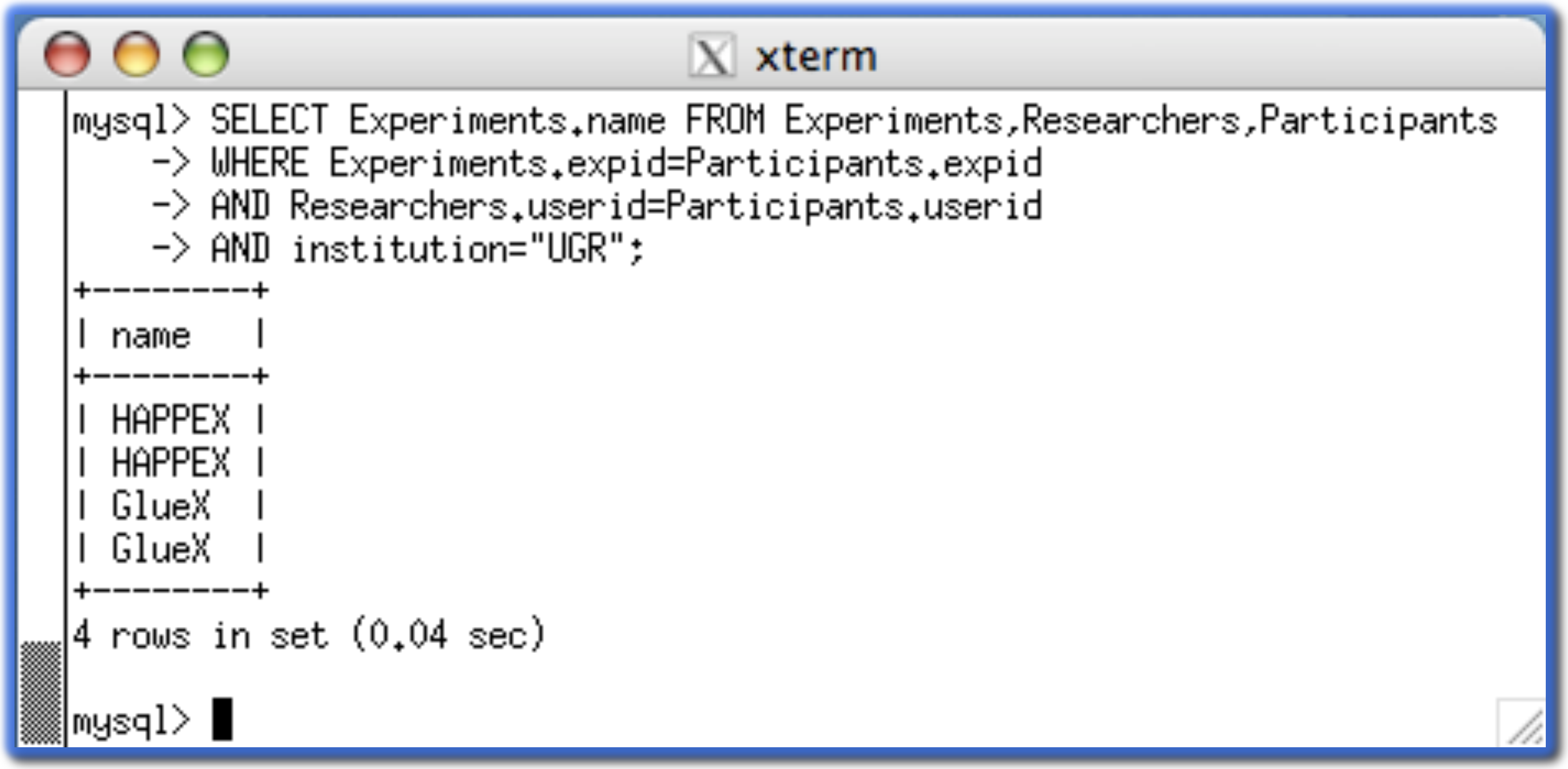
- To select all columns, use the wildcard "*"



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

mysql> 
```

Multiple tables in a *SELECT*

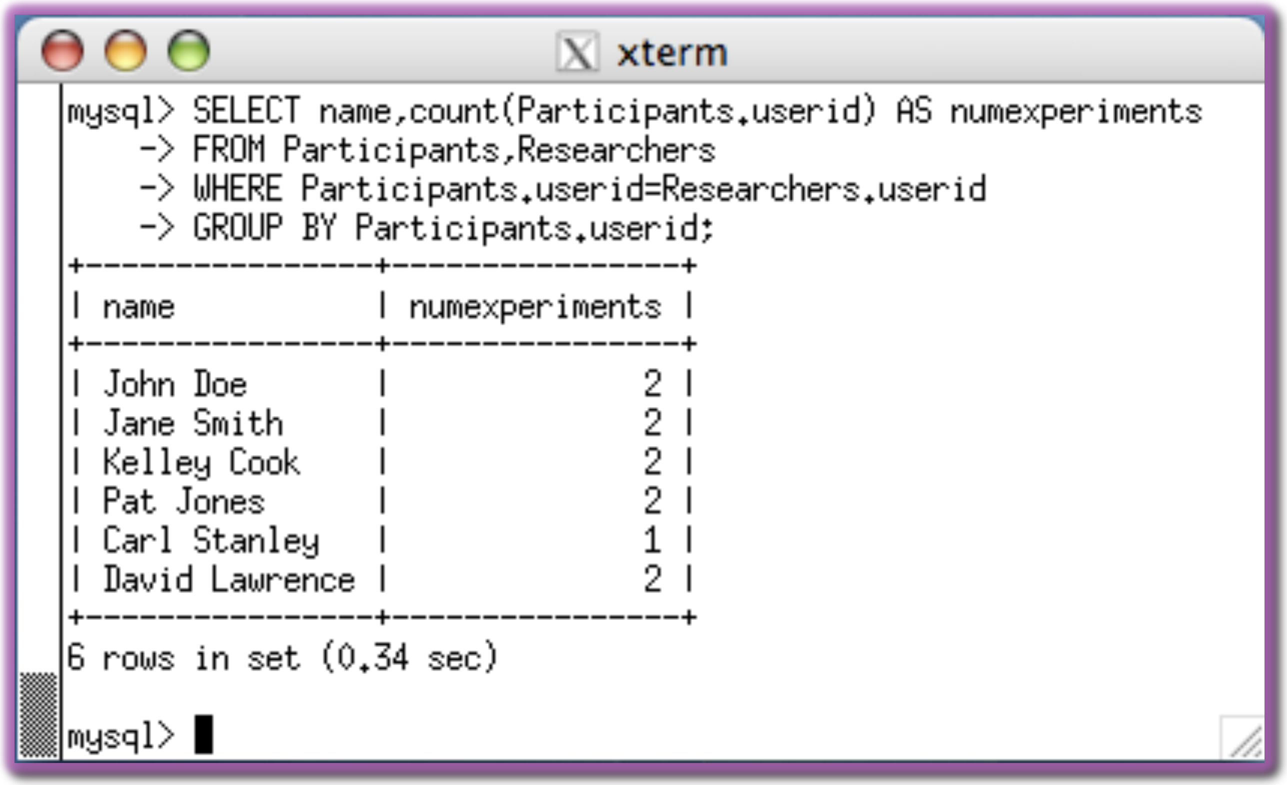
- We want a list of all experiments that the UGR institution is participating in.



```
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)

mysql> 
```

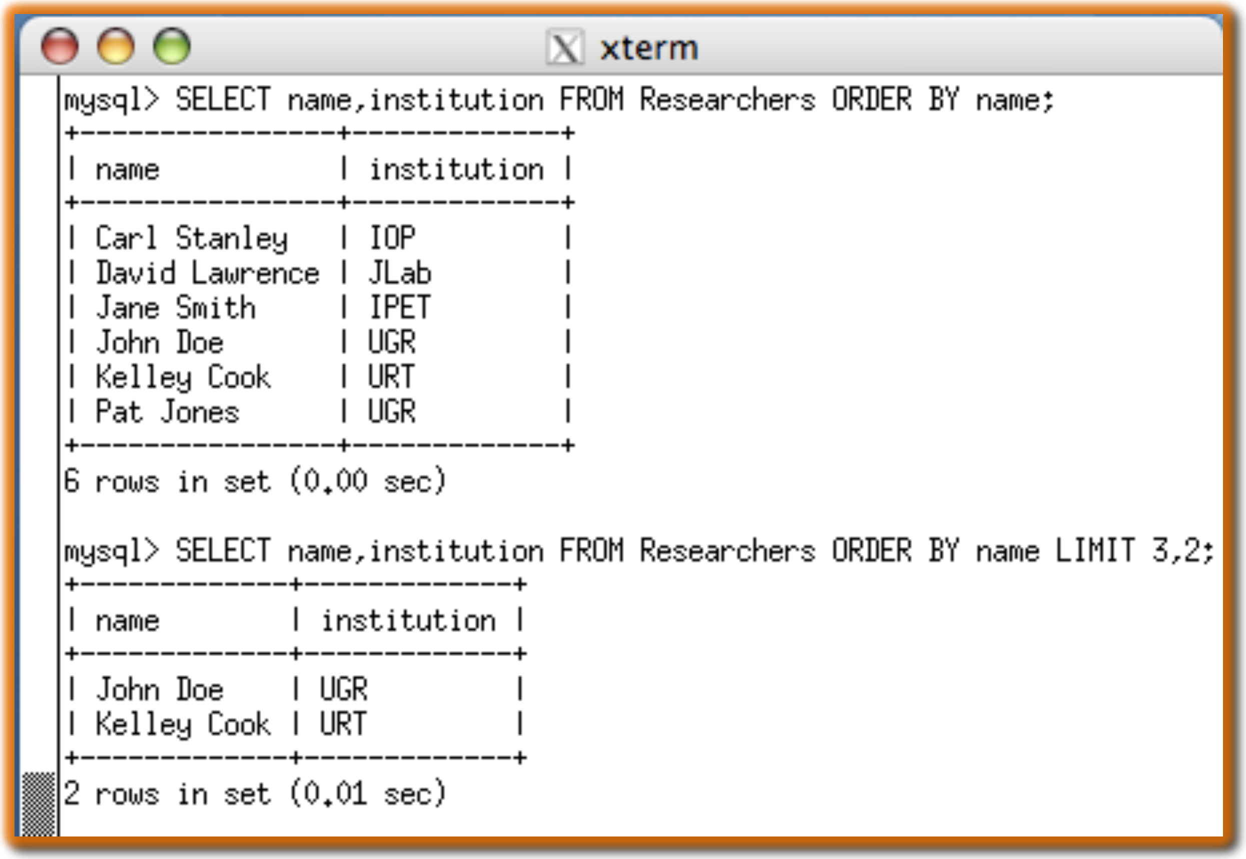
GROUP BY

- We want to know how many experiments each researcher is participating in



```
mysql> SELECT name,count(Participants.userid) AS numexperiments
    -> FROM Participants,Researchers
    -> WHERE Participants.userid=Researchers.userid
    -> GROUP BY Participants.userid;
+----------------+----------------+
| name           | numexperiments |
+----------------+----------------+
| John Doe       |              2 |
| Jane Smith     |              2 |
| Kelley Cook    |              2 |
| Pat Jones      |              2 |
| Carl Stanley   |              1 |
| David Lawrence |              2 |
+----------------+----------------+
6 rows in set (0.34 sec)

mysql> 
```

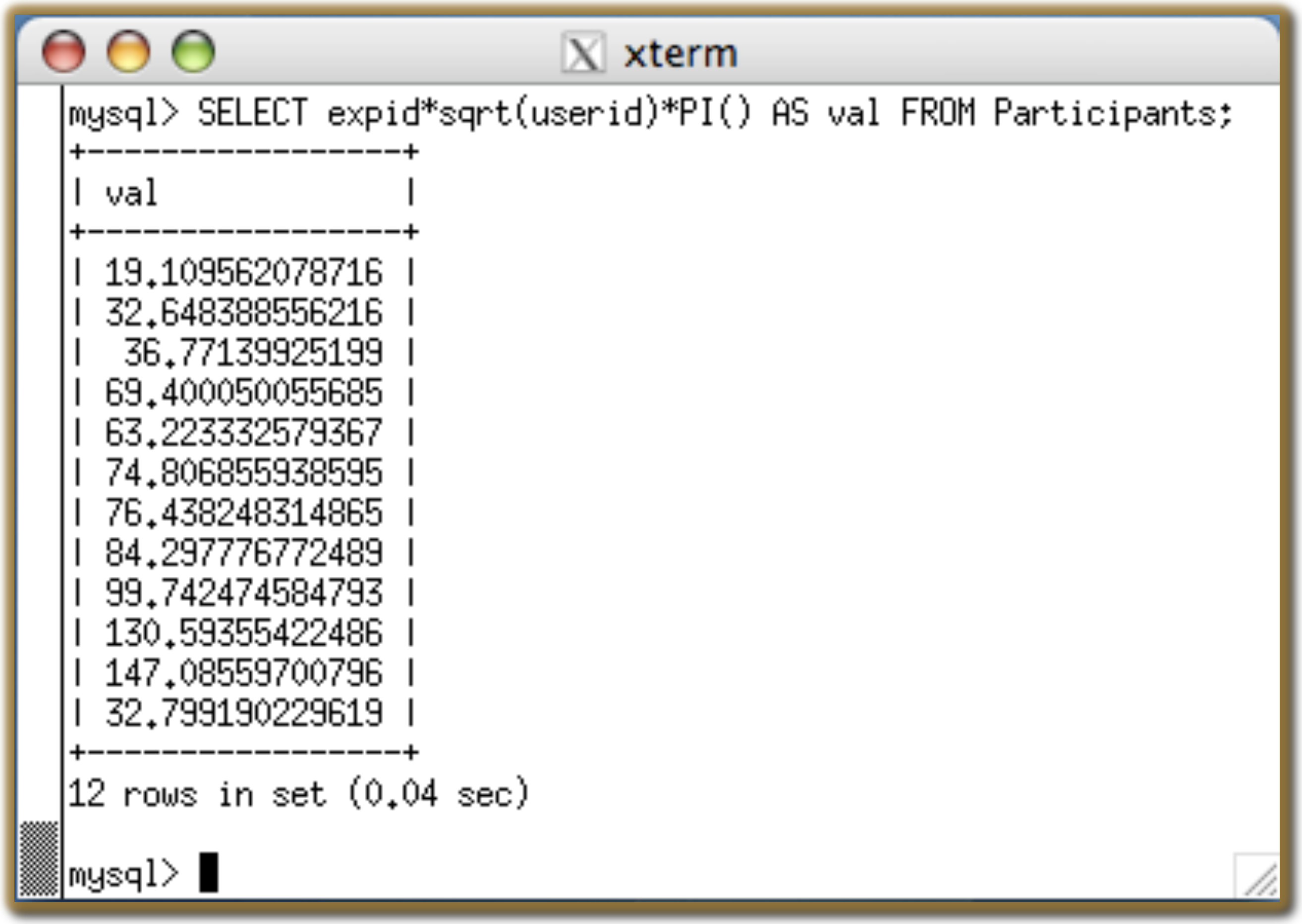
ORDER BY and *LIMIT*



```
mysql> SELECT name,institution FROM Researchers ORDER BY name;
+----------------+-------------+
| name           | institution |
+----------------+-------------+
| Carl Stanley   | IOP         |
| David Lawrence | JLab        |
| Jane Smith     | IPET        |
| John Doe       | UGR         |
| Kelley Cook    | URT         |
| Pat Jones      | UGR         |
+----------------+-------------+
6 rows in set (0.00 sec)

mysql> SELECT name,institution FROM Researchers ORDER BY name LIMIT 3,2;
+-------------+-------------+
| name        | institution |
+-------------+-------------+
| John Doe    | UGR         |
| Kelley Cook | URT         |
+-------------+-------------+
2 rows in set (0.01 sec)
```

- 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

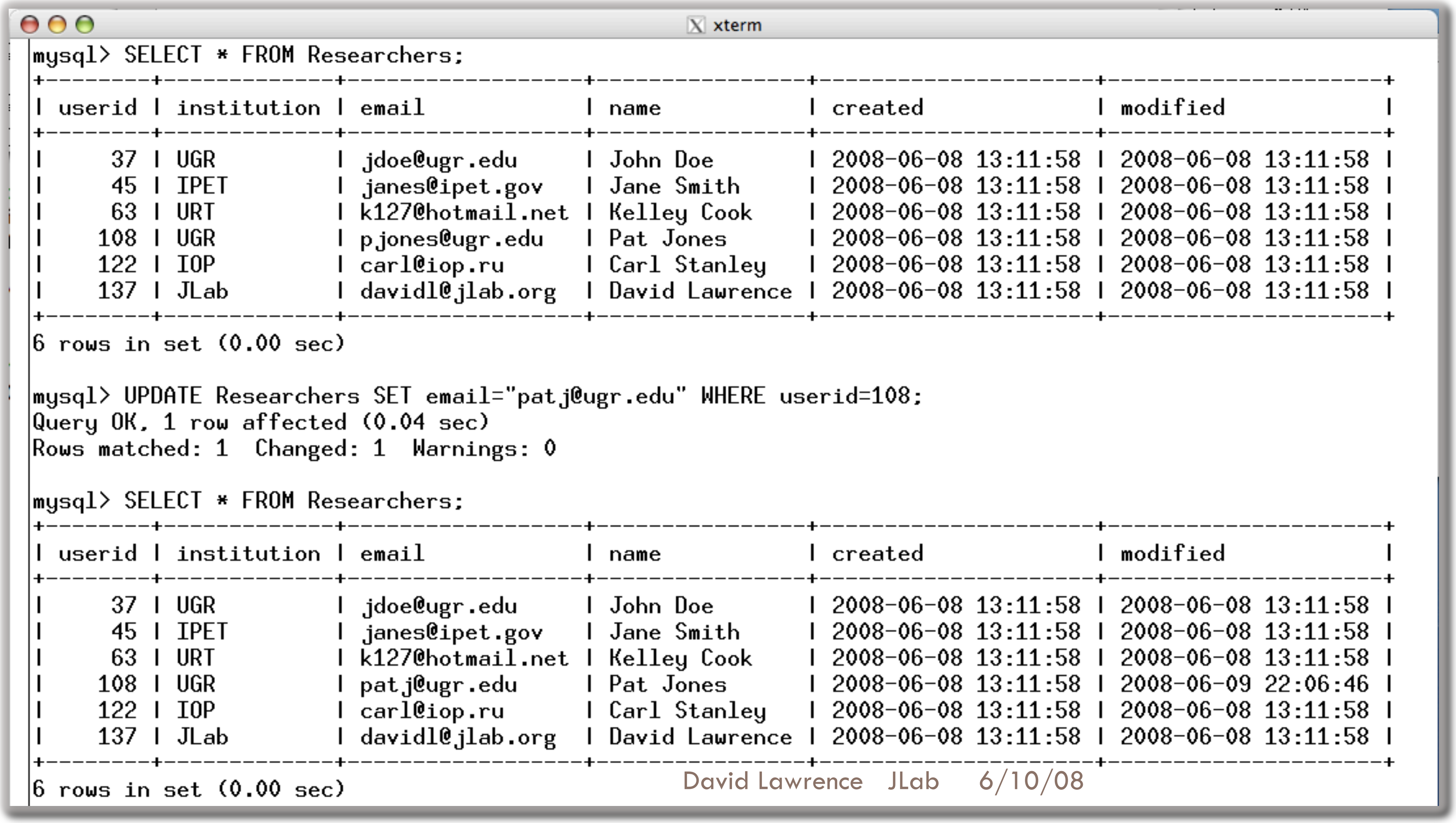
SELECT can combine columns



```
mysql> SELECT expid*sqrt(userid)*PI() AS val FROM Participants;
+-----------------+
| val             |
+-----------------+
| 19.109562078716 |
| 32.648388556216 |
|  36.77139925199 |
| 69.400050055685 |
| 63.223332579367 |
| 74.806855938595 |
| 76.438248314865 |
| 84.297776772489 |
| 99.742474584793 |
| 130.59355422486 |
| 147.08559700796 |
| 32.799190229619 |
+-----------------+
12 rows in set (0.04 sec)

mysql> 
```

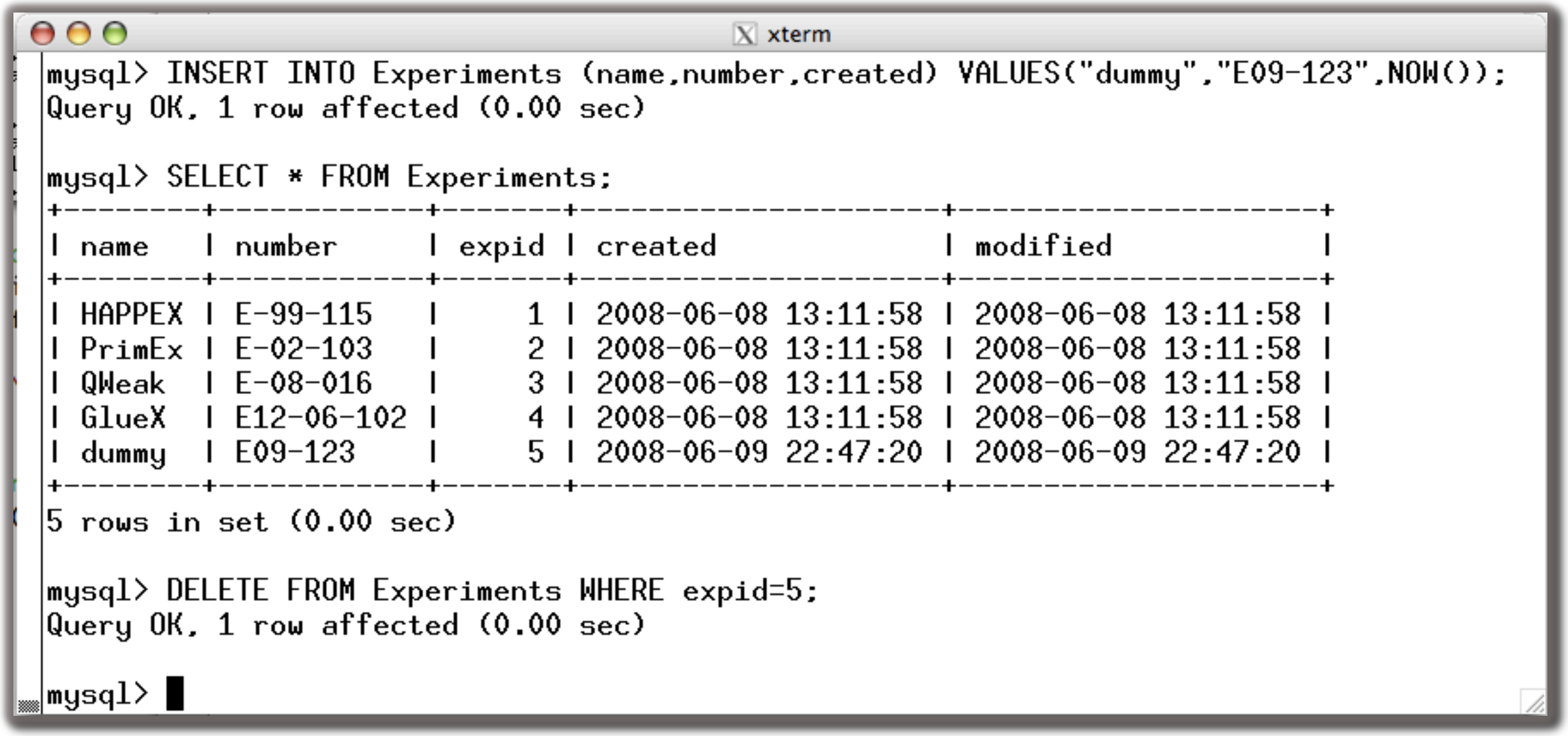
- The *SELECT* command can combine columns mathematically to dynamically form a new column (temporarily) for the query

UPDATE-ing data in a table



```
mysql> SELECT * FROM Researchers;
+--------+-------------+-------------------+-----------------+---------------------+---------------------+
| userid | institution | email             | name            | created             | modified            |
+--------+-------------+-------------------+-----------------+---------------------+---------------------+
|     37 | UGR         | jdoe@ugr.edu      | John Doe        | 2008-06-08 13:11:58 | 2008-06-08 13:11:58 |
|     45 | IPET        | janes@ipet.gov    | Jane Smith      | 2008-06-08 13:11:58 | 2008-06-08 13:11:58 |
|     63 | URT         | k127@hotmail.net  | Kelley Cook     | 2008-06-08 13:11:58 | 2008-06-08 13:11:58 |
|    108 | UGR         | pjones@ugr.edu    | Pat Jones       | 2008-06-08 13:11:58 | 2008-06-08 13:11:58 |
|    122 | IOP         | carl@iop.ru       | Carl Stanley    | 2008-06-08 13:11:58 | 2008-06-08 13:11:58 |
|    137 | JLab        | davidl@jlab.org   | David Lawrence  | 2008-06-08 13:11:58 | 2008-06-08 13:11:58 |
+--------+-------------+-------------------+-----------------+---------------------+---------------------+
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;
+--------+-------------+-------------------+-----------------+---------------------+---------------------+
| userid | institution | email             | name            | created             | modified            |
+--------+-------------+-------------------+-----------------+---------------------+---------------------+
|     37 | UGR         | jdoe@ugr.edu      | John Doe        | 2008-06-08 13:11:58 | 2008-06-08 13:11:58 |
|     45 | IPET        | janes@ipet.gov    | Jane Smith      | 2008-06-08 13:11:58 | 2008-06-08 13:11:58 |
|     63 | URT         | k127@hotmail.net  | Kelley Cook     | 2008-06-08 13:11:58 | 2008-06-08 13:11:58 |
|    108 | UGR         | patj@ugr.edu      | Pat Jones       | 2008-06-08 13:11:58 | 2008-06-09 22:06:46 |
|    122 | IOP         | carl@iop.ru       | Carl Stanley    | 2008-06-08 13:11:58 | 2008-06-08 13:11:58 |
|    137 | JLab        | davidl@jlab.org   | David Lawrence  | 2008-06-08 13:11:58 | 2008-06-08 13:11:58 |
+--------+-------------+-------------------+-----------------+---------------------+---------------------+
6 rows in set (0.00 sec)
```

DELETE-ing data from a table

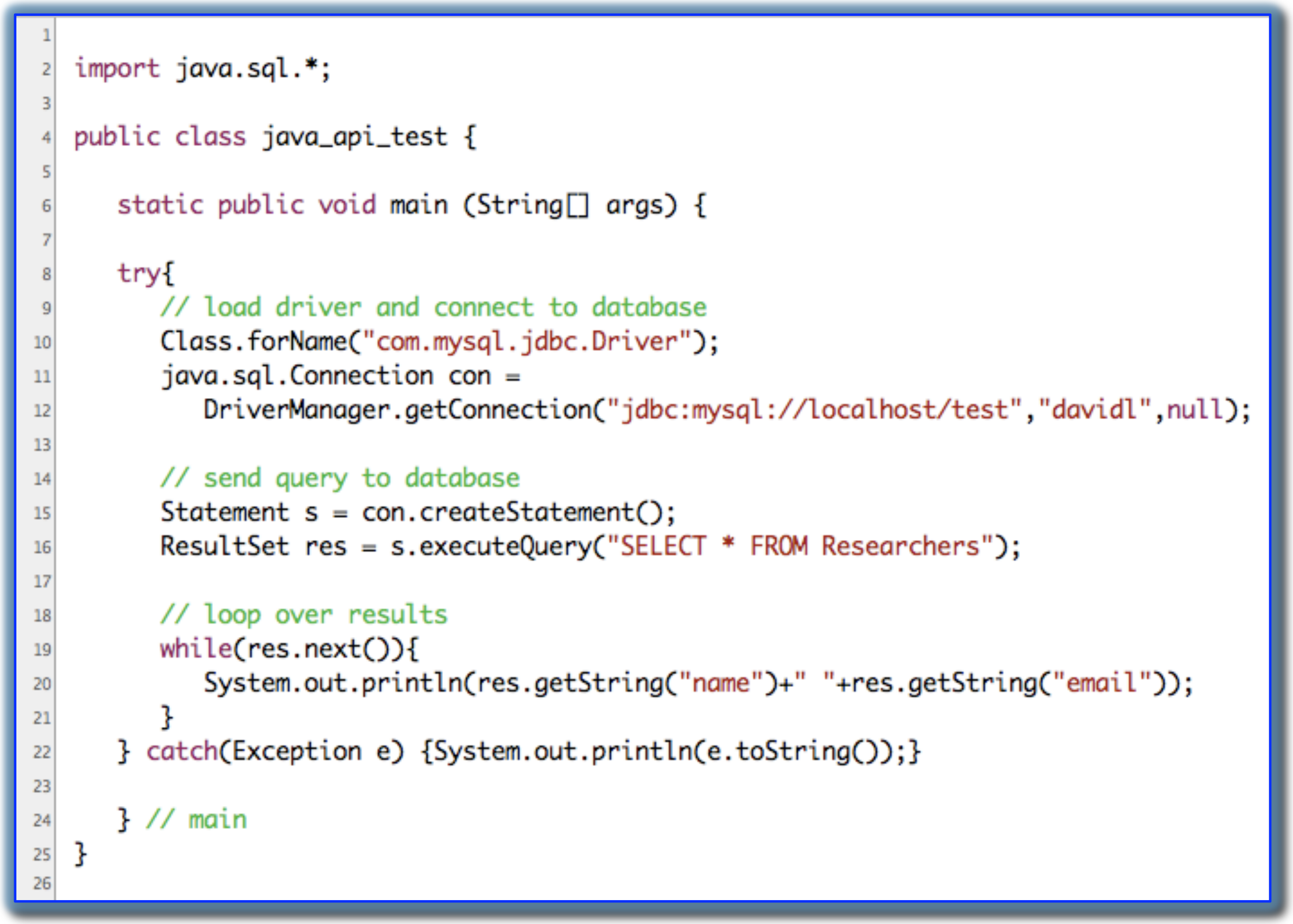


```
mysql> INSERT INTO Experiments (name,number,created) VALUES("dummy","E09-123",NOW());
Query OK, 1 row affected (0.00 sec)

mysql> SELECT * FROM Experiments;
+--------+------------+-------+---------------------+---------------------+
| name   | number     | expid | created             | modified            |
+--------+------------+-------+---------------------+---------------------+
| HAPPEX | E-99-115   |     1 | 2008-06-08 13:11:58 | 2008-06-08 13:11:58 |
| PrimEx | E-02-103   |     2 | 2008-06-08 13:11:58 | 2008-06-08 13:11:58 |
| QWeak  | E-08-016   |     3 | 2008-06-08 13:11:58 | 2008-06-08 13:11:58 |
| GlueX  | E12-06-102 |     4 | 2008-06-08 13:11:58 | 2008-06-08 13:11:58 |
| dummy  | E09-123    |     5 | 2008-06-09 22:47:20 | 2008-06-09 22:47:20 |
+--------+------------+-------+---------------------+---------------------+
5 rows in set (0.00 sec)

mysql> DELETE FROM Experiments WHERE expid=5;
Query OK, 1 row affected (0.00 sec)

mysql> 
```

The mysql tools

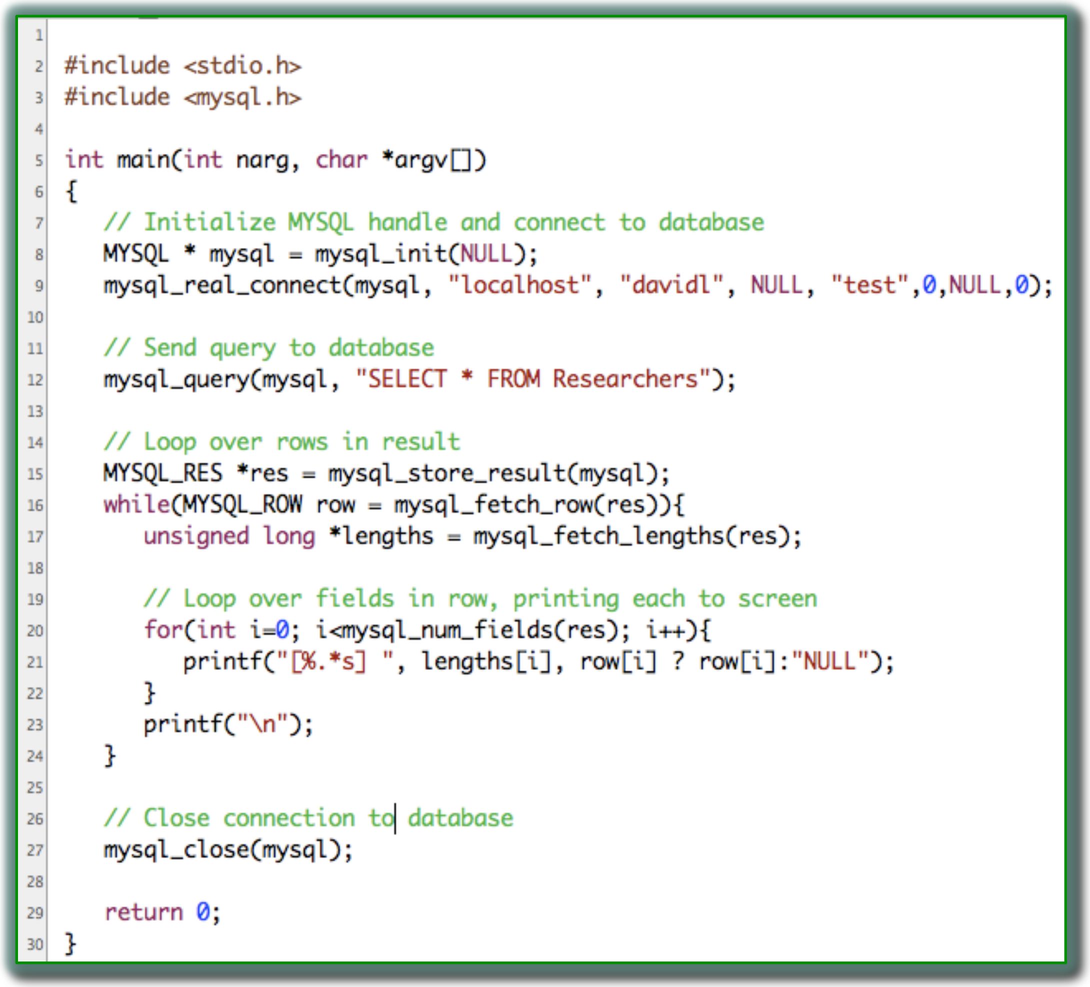
- *mysql* – interactive command line tool
- *mysqldump* – dump contents (including table definitions) of a database
- *mysqlshow* – show info about tables, databases, etc.
- *mysql_config* – print C/C++ compiler options for current platform

Accessing the database with JAVA



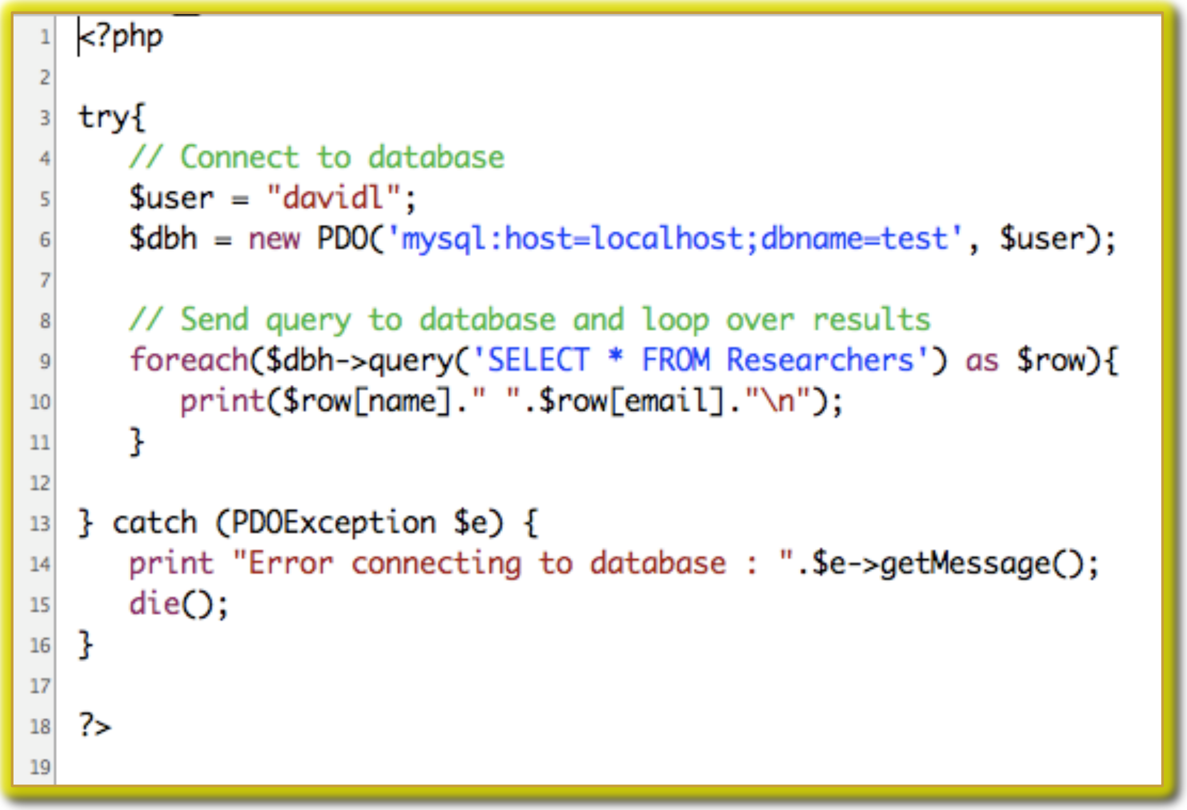
```
import java.sql.*;

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
}
```

Accessing the database with C/C++



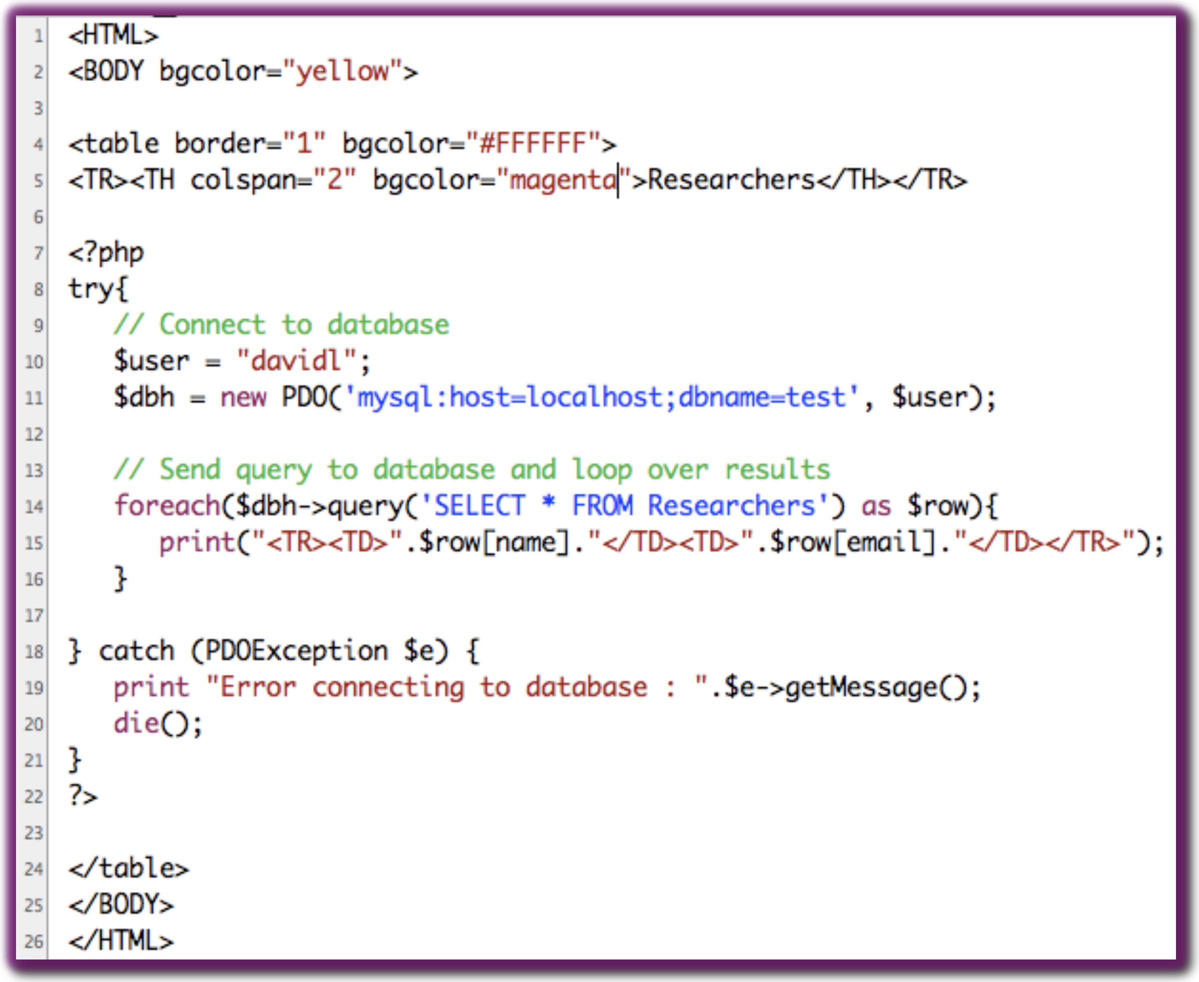
```
#include <stdio.h>
#include <mysql.h>

int main(int narg, 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);

   return 0;
}
```

Accessing the database with PHP



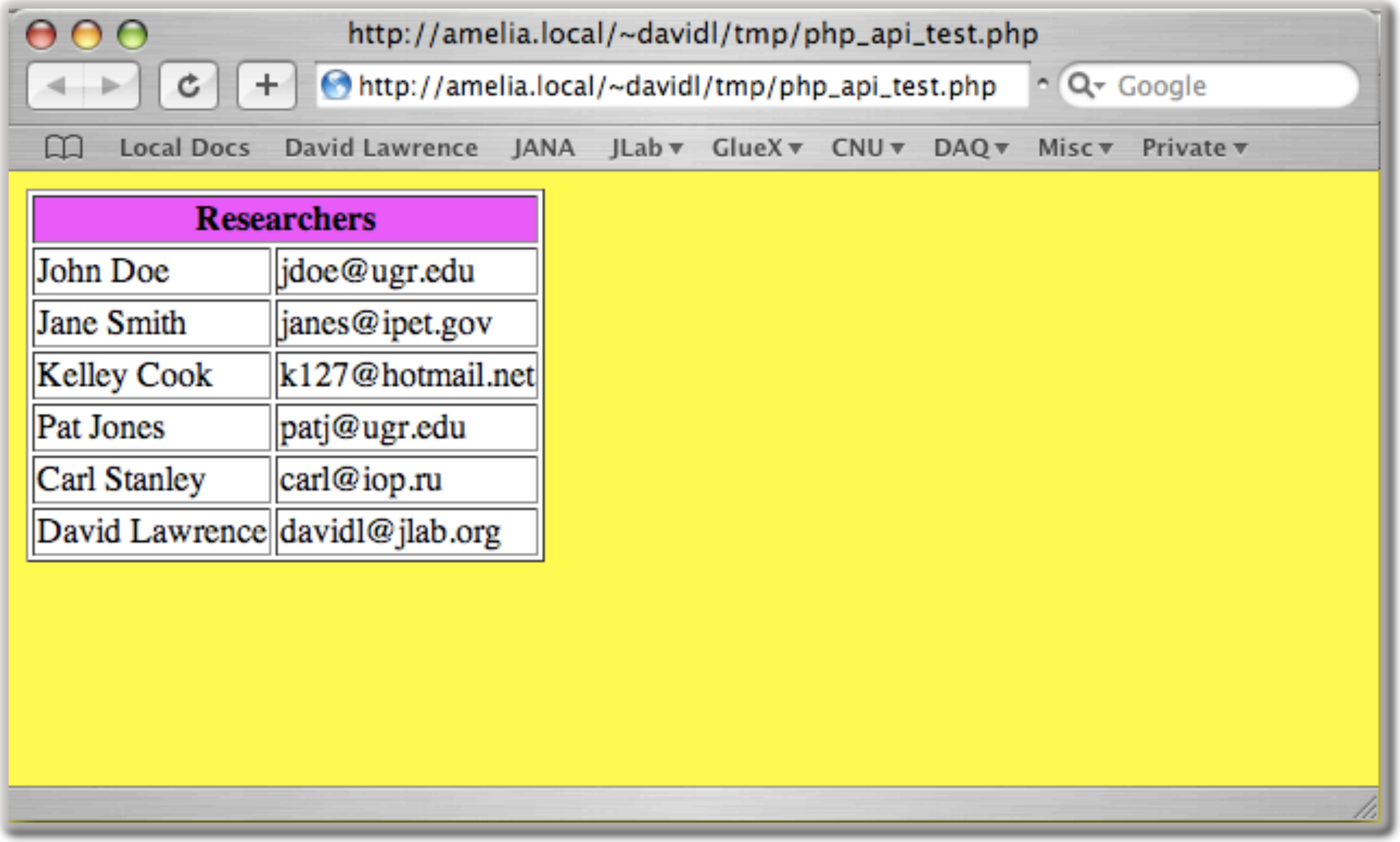
```
<?php

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



```
<HTML>
<BODY bgcolor="yellow">

<table border="1" bgcolor="#FFFFFF">
<TR><TH colspan="2" bgcolor="magenta">Researchers</TH></TR>

<?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>
```

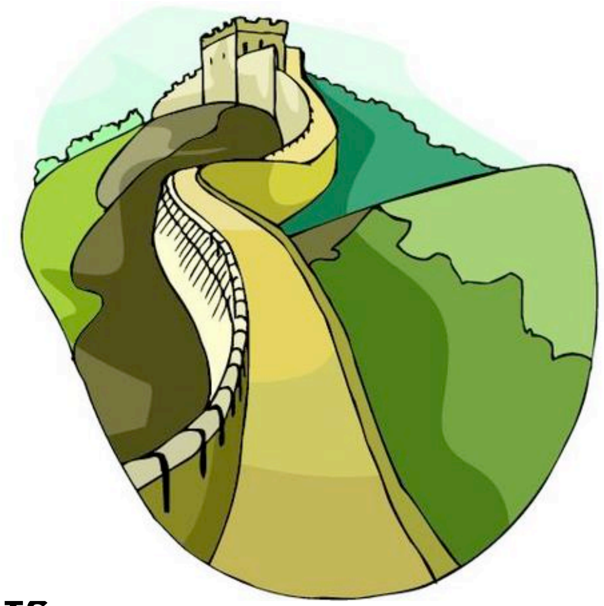
PHP embedded in HTML


http://amelia.local/~davidl/tmp/php_api_test.php
http://amelia.local/~davidl/tmp/php_api_test.php
Google
Local Docs
David Lawrence
JANA
JLab
GlueX
CNU
DAQ
Misc
Private

Researchers	
John Doe	jdoe@ugr.edu
Jane Smith	janes@ipet.gov
Kelley Cook	k127@hotmail.net
Pat Jones	patj@ugr.edu
Carl Stanley	carl@iop.ru
David Lawrence	davidl@jlab.org

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