PIC 40A

Lecture 20: Introduction to SQL

What is a database?

A database is a way of storing data.

Databases are specifically designed to be very efficient in storing and retrieving data.

The most common way of structuring data in a database is relational database system.

Relational databases

•A relational database is a structured collection of tables.

•Each table consists of rows called records.

•Columns of the table have keys called fields. Each cell contains data.

•Most tables have a special column that identifies the rows of the table. The values in this column are called primary keys.

Relational database table example

Fields

	IP_address	num_visits	timestamp
Record 1	123.25.67	7	12911193
Record 2	98.104.22	3	13056782

A more complex relational database

Vette_id	Body_style	Miles	Year	State	Equip
1	coupe	18.0	1997	Arkansas	Automatic,leather, CD
2	hatchback	58.0	1996	Connecticut	Automatic,leather, CD
3	convertible	13.5	2001	Alabama	Automatic,leather
4	hatchback	19.5	1995	Alaska	4-speed,leather
5	hatchback	25.0	1991	California	Automatic,leather
6	hardtop	15.0	2000	Alaska	4-speed
7	coupe	55.0	1979	Georgia	6-speed,leather
8	convertible	17.0	1999	California	6-speed,leather,CD
9	hardtop	17.0	2000	California	6-speed,leather,CD
10	hatchback	50.0	1995	Connecticut	Automatic,CD

Corvettes table

Example continued

Simplification: Move some data to a new table. To accomplish this we create a separate equipment table.

Equip_id	Equip
1	Automatic
2	4-speed
3	5-speed
4	6-speed
5	CD
6	Leather

Equipment table

Example continued.

Next we need a way to indicate which cars have what equipment. To accomplish this we create a cross reference table. To emphasize that this table ties the Corvettes table to equipment table we call it Corvettes_Equipment.

Vette_id	Equip
1	1
1	5
1	6
2	1
2	5
2	6
3	1
3	6
4	2
4	6

Example continued

A further simplification is to code the state names with a number.

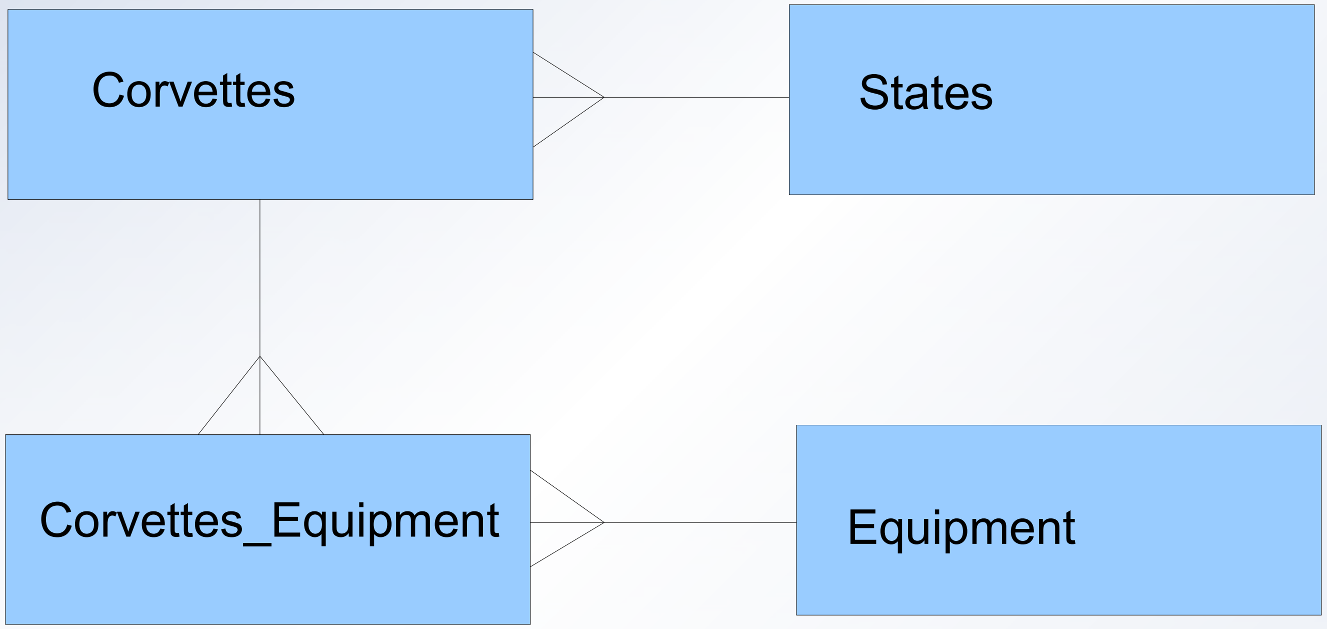
State_ID	State
1	Alabama
2	Alaska
3	Arizona
4	Arkansas
5	California
6	Colorado
7	Connecticut
8	Delaware
9	Florida
10	Georgia

Example continued

What we have accomplished:

Vette_id	Body_style	Miles	Year	State
1	coupe	18.0	1997	4
2	hatchback	58.0	1996	7
3	convertible	13.5	2001	1
4	hatchback	19.5	1995	2
5	hatchback	25.0	1991	5
6	hardtop	15.0	2000	2
7	coupe	55.0	1979	10
8	convertible	17.0	1999	5
9	hardtop	17.0	2000	5
10	hatchback	50.0	1995	7

Example continued



Had we not used a cross reference table the relationship between Corvettes and Equipment would have been many to many.

What is SQL?

SQL stands for **S**tructured **Q**uery **L**anguage.

It is a standard language developed for accessing and modifying relational databases.

SQL in turn is used by a database management system. Some common database management systems are:

- MySQL
- SQLite
- PostgreSQL
- Oracle
- Microsoft SQL Server

SQLite

- SQLite is free. Open source.
- Very widely used and implements most of the SQL standard
- We can use SQLite without an additional administrator assistance.
- We will use Sqlite 3

SQL from the Unix command line

First login to your PIC Unix account

From your Unix account type:

```
laguna> sqlite3 my_first_database

SQLite version 3.7.17 2013-05-20 00:56:22
Enter ".help" for instructions
Enter SQL statements terminated with a ";"
```

You have now created a database called `my_first_database`. Subsequent commands will be applied to this database.

Basic MySQL commands

SQLite is case insensitive although field names are case sensitive.

SQLite commands end in a ; so if you do not end your command with a ; SQLite assumes you want to keep typing.

Useful dot commands

.help

.databases

.tables

.quit

Do not use ; with . commands

CREATE TABLE

To create a new table we use the `CREATE TABLE` command.

Syntax for creating a new table is:

```
CREATE TABLE IF NOT EXISTS tablename (
fieldname1 type options default 'defaultvalue',
fieldname2 type options default 'defaultvalue',
...
);
```

There are number of possibilities for type including: `varchar` (or `char`) `int`, `float`(significant digits, digits after decimal).

After specifying type, you may specify the max length of the variable.

eg. `varchar(10)`

Example

```
CREATE TABLE IF NOT EXISTS students (
name varchar(100),
sid int(9),
gpa decimal(3,1),
phone varchar(12)
);
```

SELECT

Is used to ask for data from a table.

It tells the database to retrieve info from any number of tables and return it as a result set.

Syntax:

```
SELECT [fields here] FROM [tables here];

SELECT [fields here] FROM [tables here]
WHERE [condition here];
```

Example

Show all records from table "students" where the field called "name" has the value "Joe".

```
SELECT * FROM students WHERE name = "Joe";
```

Show all records from table "people" where the field called "name" has the value "Joe" AND the phone number "555-6789".

```
SELECT * FROM people WHERE name = "Joe" AND
phone_number = "555-6789";
```

Example

Show all records from table "students" where the field called "name" has the value "Joe" OR "Bob".

```
SELECT * FROM students WHERE name = "Joe" OR name = "Bob";
```

Show "student_id" and "name" fields from table "students" with GPA greater than or equal to 3.5.

```
SELECT sid, name FROM students WHERE GPA >= 3.5;
```

INSERT

Add a record to a table in a database.

Syntax:

```
INSERT INTO tablename (field1, field2, ...)
VALUES ('val1', 'val2', ...);
```

Example:

```
INSERT INTO students (name,sid,gpa,phone)
VALUES ('Zoidberg',77843211, -4.0,"N/A");
```

UPDATE

Modify an existing record in the database

Syntax:
```
UPDATE tablename SET fieldname = value WHERE
condition;
```

Example:
```
UPDATE students SET GPA ='3.8' WHERE person =
'Joe';
```

DELETE

Removes an existing record in a database

Syntax:

```
DELETE FROM tablename WHERE condition;
```

Example:

```
DELETE FROM students WHERE name="Fry";
```

Using PHP to run SQLite3

```
try
{

     $db = new SQLite3('my_first_database.db');
}
catch (Exception $exception)
{
    echo '<p>There was an error connecting to the
database!</p>';

    if ($db)
    {
        echo $exception->getMessage();
    }

}
```

INSERT

```
$table = "students";
$field1 = "name";
$field2 = "sid";
$field3 = "gpa";
$field4 = "phone";

$value1 = "Zoidberg";
$value2 = 123456789;
$value3 = -4.0;
$value4 = "N/A";

$sql= "INSERT INTO $table ($field1, $field2,
$field3)
VALUES('$value1','$value2','$value3')";
```

UPDATE

Define a SQL query for updating records in a database table.

```
$newgrade= 3.1;
$newid= "222444888";

$sql= "UPDATE $table SET $field2
='$newid',
$field3 = '$newgrade' WHERE
$field1='$value1'";

$result = $db->query($sql);
```

DELETE

Define a SQL query for deleting records from a table in a database.

```
$sql= "DELETE FROM $table WHERE $field1 =
'Fry'";

$result = $db->query($sql);
```

SELECT

```
$sql= "SELECT $field1, $field2, $field3
FROM $table WHERE $field3 = '4.0'";

$result = $db->query($sql);
```

Now the result is potentially many records and we have to have a way to parse data from this result.

SELECT

We use `fetchArray()` to extract one record from the result. It works a lot like the `next` function we used for arrays.

```
while($record = $result->fetchArray())
{
print " <tr>\n";
print " <td>" . $record[$field1]. "</td>\n";
print " <td>" . $record[$field2]. "</td>\n";
print " <td>" . $record[$field3]. "</td>\n";
print " </tr>\n";
}
```

Note: `$record` is an array where keys are the fields. Corresponding values are the data.

Counting number of records

To figure out how many (if any) records meeting our criteria there are we can do:

```
  $sql = "SELECT count(*) FROM table WHERE 
condition";
  $result = $db->query($sql);
  $record=$result->fetchArray();
  $number = $record['count(*)'];
```