Application Links

Chapter 7.1

V3.0

Dr Gordon Russell

NAPIER UNIVERSITY
EDINBURGH

Introduction

- Up till now we have controlled our databases using an interactive tutorial window…
- In reality we will be writing application code to communicate with the database.
- Some development environments hide the database.
- Some approaches to application writing make the database access explicit.

4GL

- The term 4GL has largely been overused and is falling out of fashion.
- Paradox and Access are examples of the 4GL approach.
- It provides:
  - A database engine
  - Table creation and data editing facilities
  - Application Builder
  - Report builder
- With this approach the database is almost invisible to the programmer.

Concerns

- While such environments can allow you to build database applications rapidly…
  - Fast development can cause prototypes to appear as (weak) products.
  - Proprietary. Migration and upgrade problems.
  - May tie development to a particular OS.
  - Fast vendor releases, leading to supporting you product on obsolete releases of the system.
  - Can be hard to find experts for future developments.

Databases in other languages

- Rather than proprietary languages (VBA, dBase) use standard (C, C++) languages with database extensions.
- There are a few approaches:
  - SQL embedding
  - A database API
  - Visual programming (e.g. Delphi).
- These approaches all make use of CURSORS.

Cursor

- A cursor is a concept which is a pointer to the result of some sort of database query.
- It points to the result 1 row at a time.
- Cursor commands allow you to step to the next row of the result.
- Some cursor implementations allow you to step back up through results too.
- The fields of the row can be reached via the API, or Program Variables, or via Text Boxes.

API calls

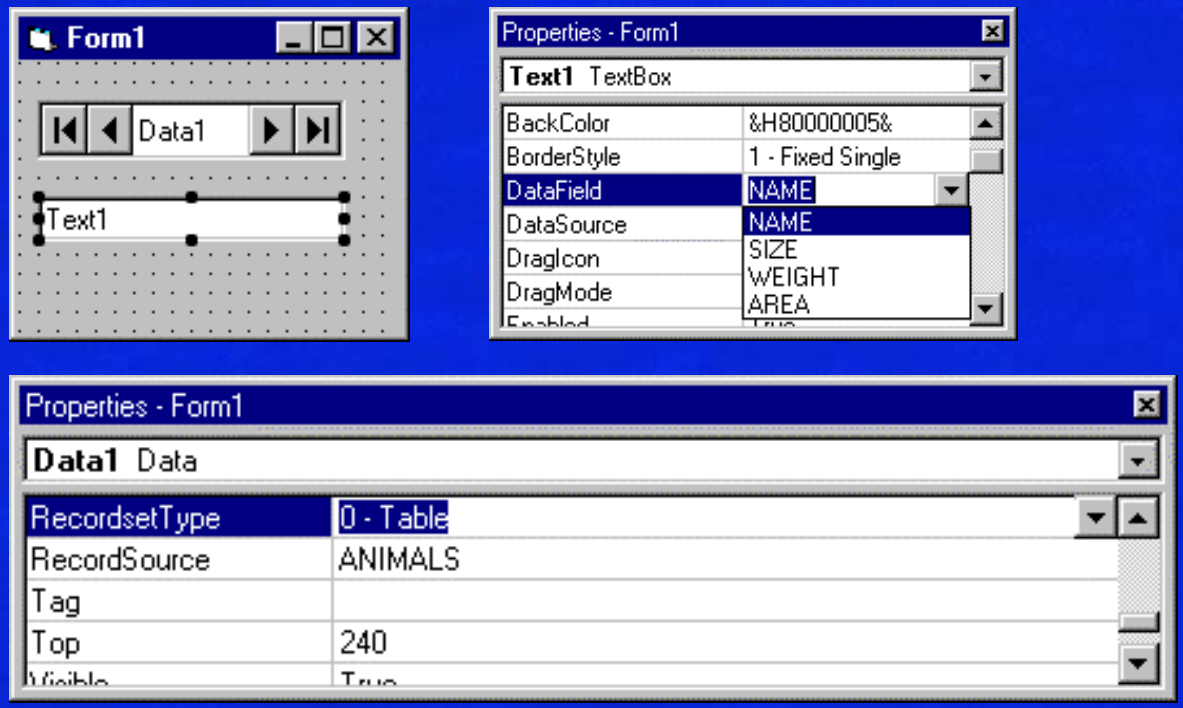
- An API is a set of calls to control an SQL database. Typical instructions include:
  - Connect
  - Execute
  - Fetch
  - Advance
  - Test
  - close

- Having a standard API is ideal.
- You can also have proprietary API interfaces.
- Standard approaches mean
  - You can switch between vendors easily.
- Of course there are companies out there with almost-standard APIs. Attractive to port to, yet hard to port from. Very naughty!

Delphi Example

```
Table1.First;
while not Table1.EOF do
begin
        Memo1.lines.Add(Table1.FieldByName('NAME').
                                AsString);
Table1.Next;
end;
```

Data Linked Visual Components: Visual Basic



Spreadsheets: Excel

- In excel equations, you can have cells like:
  =vlookup(B1, Sheet2!$A$1:$B$3, 2)
- For simple databases this can work quite well!
  - Store one record per row (1NF)
  - Rely on VLOOKUP indexing
  - Worry about key ordering yourself.
  - Slow and error prone
  - Does not scale

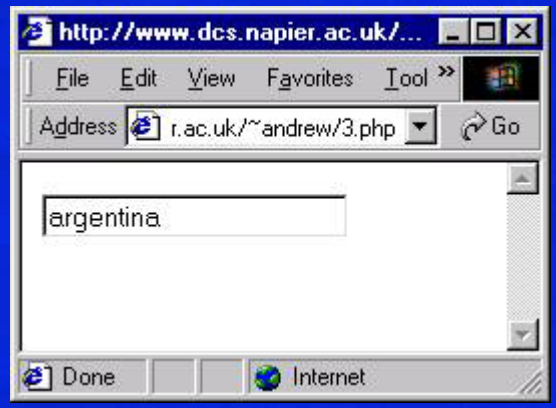
PHP and MySQL

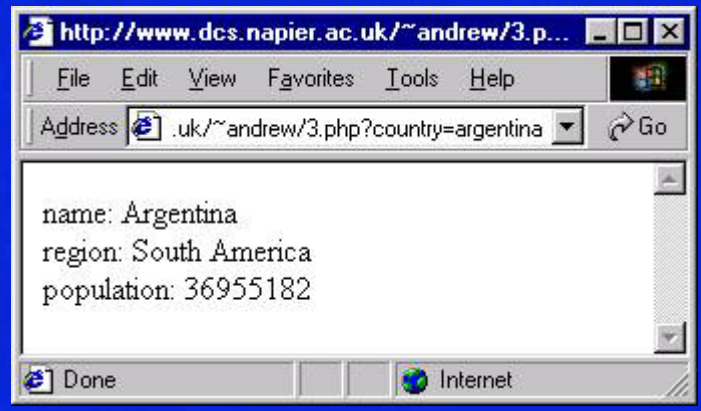
- Given…

```
selene% /usr/local/mysql/bin/mysql -hzeus -u andrew –p
mysql> use andrew
mysql> show tables;
    +------------------------+
    | Tables_in_andrew |
    +------------------------+
    | one                          |
    | cia                          |
    +------------------------+
    2 rows in set (0.05 sec)
```

```
mysql> select * from cia where population>200000000;
+---------------+-----------------+---------+----------------+--------------------+
| name          | region          | area    | population     | gdp                |
+---------------+-----------------+---------+----------------+--------------------+
| China         | Asia            | 9596960 | 1261832482     | 4800000000000      |
| India         | Asia            | 3287590 | 1014003817     | 1805000000000      |
| Indonesia     | Southeast Asia  | 1919440 |  224784210     |  610000000000      |
| United States | N. America      | 9629091 |  275562673     | 9255000000000      |
+---------------+-----------------+---------+----------------+--------------------+
4 rows in set (0.11 sec)
```

Looks like:


http://www.dcs.napier.ac.uk/...
File Edit View Favorites Tool
Address r.ac.uk/~andrew/3.php
Go
argentina
Done
Internet


http://www.dcs.napier.ac.uk/~andrew/3.p...
File Edit View Favorites Tools Help
Address .uk/~andrew/3.php?country=argentina
Go
name: Argentina
region: South America
population: 36955182
Done
Internet

With:

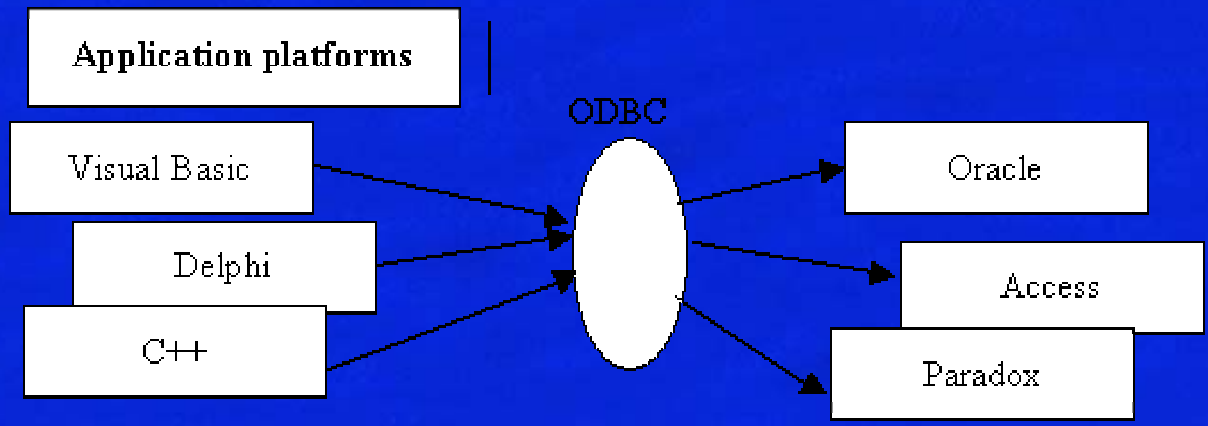
```
<?php
if ($country) {
  $link = mysql_connect("zeus", "andrew", "******")
        or die("Could not connect");
  mysql_select_db("andrew") or die("Could not select database");
  $query = "SELECT name, region, population
                        FROM cia WHERE name='$country'";
  $result = mysql_query($query) or die("Query failed");
  while ($row = mysql_fetch_array($result)) {
    extract($row);    print "name: $name<br>\n";
    print "region: $region<br>\n";
    print "population: $population<br>\n";  }
  print "</table>\n";  mysql_free_result($result);
  mysql_close($link);}
else {  print "<form><input name='country'></form>\n"; }
?>
```

SQL Embedding

- Here is a fragment of the embedding for C:

```
/* Execute the SQL query */
 EXEC SQL SELECT CustID, SalesPerson, Status
 FROM Orders
 WHERE OrderID = :OrderID
 INTO :CustID, :SalesPerson, :Status;
```

Advantages of standard API



Popular API

- Popular options include
  - ODBC
  - JDBC
  - DBI/DBD
  - ASP

ODBC

- Open Database Connectivity.
- Work through a client/server “brokerage”.
- Can be tricky to set up.
- Can be expensive, with relatively few working free options.

JDBC

- ODBC for Java
- Similar problems to ODBC.
- A good standard to base things on.
- Perhaps more choice than ODBC.

DBI/DBD

- Becoming more popular.
- Good free option.
- API quite consistent between platforms and vendors.

In PERL.

```
my $dbh = DBI->connect("dbname","username","password");
my $depno = 3;
my $cmd = $dbh->prepare("SELECT surname
    FROM employee where depno=?");
my $res = $cmd->execute($depno);
while (my ($name) = $res->fetchrow_array()) {
  print "The employee name is $name\n";
}
```

ASP

```
<%SQL="SELECT carName FROM Cars ORDER BY carName“
set conn = server.createobject("ADODB.Connection")conn.open
                                                    "parking“
set cars=conn.execute(SQL) %>
<% do while not cars.eof %>
  <%= cars(0) %> <br>
  <%cars.movenext     loop%>
<% cars.close %>
```

Efficiency

- Often the web server and the DB on different machines
- Each request->app->db->result->web cycle can be slow.
- DB connection creation is often the slowest part.
- Queries used should be considered carefully to minimise
  - The number of queries
  - The size of the result returned.
- Some optimisation options, for instance
  - connection pooling
  - data cashing