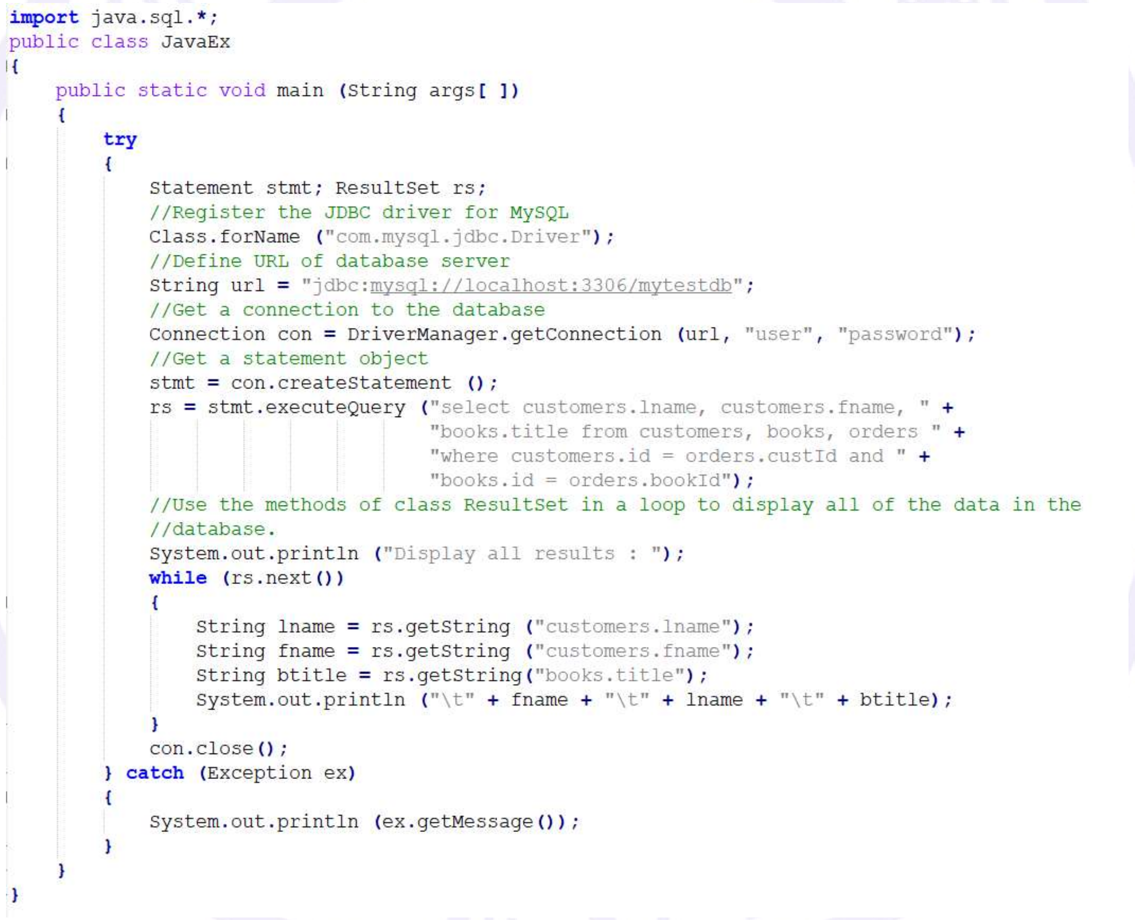
Database Language Bindings

Slides by Rogers, Modifications by Brown

Language “Popularity”

- Why include this section
  - ABET
  - Language Comprehension
- TIOBE Index
  - Java and C++
  - History Lesson
    - Java / TCL
    - Open Database Connections
- Language Bindings

Java (MySQL)

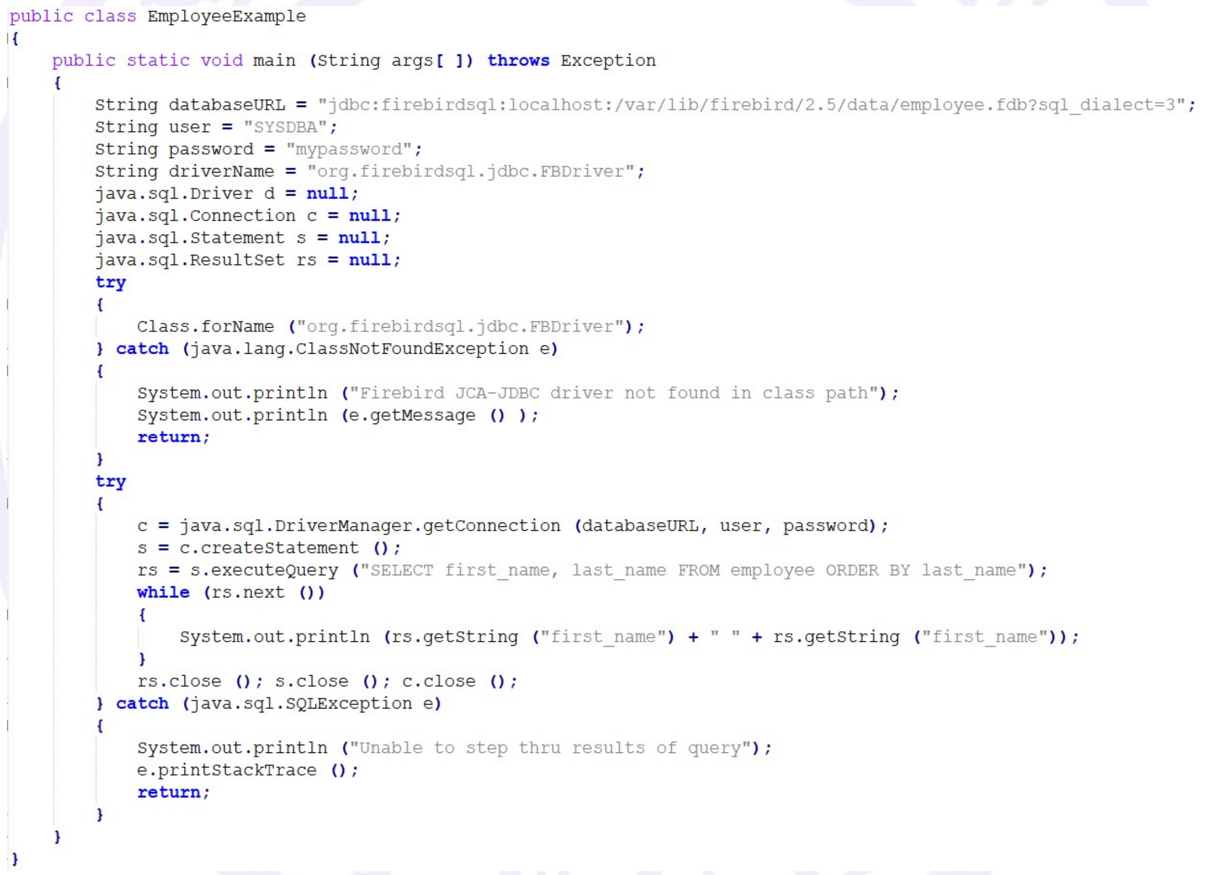


```
import java.sql.*;
public class JavaEx
{
    public static void main (String args[ ])
    {
        try
        {
            Statement stmt; ResultSet rs;
            //Register the JDBC driver for MySQL
            Class.forName ("com.mysql.jdbc.Driver");
            //Define URL of database server
            String url = "jdbc:mysql://localhost:3306/mytestdb";
            //Get a connection to the database
            Connection con = DriverManager.getConnection (url, "user", "password");
            //Get a statement object
            stmt = con.createStatement ();
            rs = stmt.executeQuery ("select customers.lname, customers.fname, " +
                                    "books.title from customers, books, orders " +
                                    "where customers.id = orders.custId and " +
                                    "books.id = orders.bookId");
            //Use the methods of class ResultSet in a loop to display all of the data in the
            //database.
            System.out.println ("Display all results : ");
            while (rs.next())
            {
                String lname = rs.getString ("customers.lname");
                String fname = rs.getString ("customers.fname");
                String btitle = rs.getString("books.title");
                System.out.println ("\t" + fname + "\t" + lname + "\t" + btitle);
            }
            con.close();
        } catch (Exception ex)
        {
            System.out.println (ex.getMessage());
        }
    }
}
```

Java Connector / J

- Compile with:
  - javav –cp $path_to_connector/mysql-connector-java.jar:./ JavaEx.java
  - On your VM:
    - javav –cp ./:/usr/share/java/mysql.jar JavaEx.java
- Run with:
  - java –cp $path_to_connector/mysql-connector-java.jar:./ JavaEx
  - On your VM:
    - java –cp ./:/usr/share/java/mysql.jar JavaEx
- Documentation: http://dev.mysql.com/doc/refman/5.0/en/connector-j.html

Java (Firebird)



```java
public class EmployeeExample
{
    public static void main (String args[ ]) throws Exception
    {
        String databaseURL = "jdbc:firebirdsql:localhost:/var/lib/firebird/2.5/data/employee.fdb?sql_dialect=3";
        String user = "SYSDBA";
        String password = "mypassword";
        String driverName = "org.firebirdsql.jdbc.FBDriver";
        java.sql.Driver d = null;
        java.sql.Connection c = null;
        java.sql.Statement s = null;
        java.sql.ResultSet rs = null;
        try
        {
            Class.forName ("org.firebirdsql.jdbc.FBDriver");
        } catch (java.lang.ClassNotFoundException e)
        {
            System.out.println ("Firebird JCA-JDBC driver not found in class path");
            System.out.println (e.getMessage () );
            return;
        }
        try
        {
            c = java.sql.DriverManager.getConnection (databaseURL, user, password);
            s = c.createStatement ();
            rs = s.executeQuery ("SELECT first_name, last_name FROM employee ORDER BY last_name");
            while (rs.next ())
            {
                System.out.println (rs.getString ("first_name") + " " + rs.getString ("first_name"));
            }
            rs.close (); s.close (); c.close ();
        } catch (java.sql.SQLException e)
        {
            System.out.println ("Unable to step thru results of query");
            e.printStackTrace ();
            return;
        }
    }
}
```

Java Jaybird Documentation

- Compile with:
  - javac –cp ./:$path_to_jaybird/jaybird-full-2.1.6.jar EmployeeExample.java
- Run with:
  - java –cp ./$path_to_jaybird/jaybird-full-2.1.6.jar EmployeeExample
- Documentation:
  http://www.firebirdsql.org/index.php?op=devel&sub=jdbc

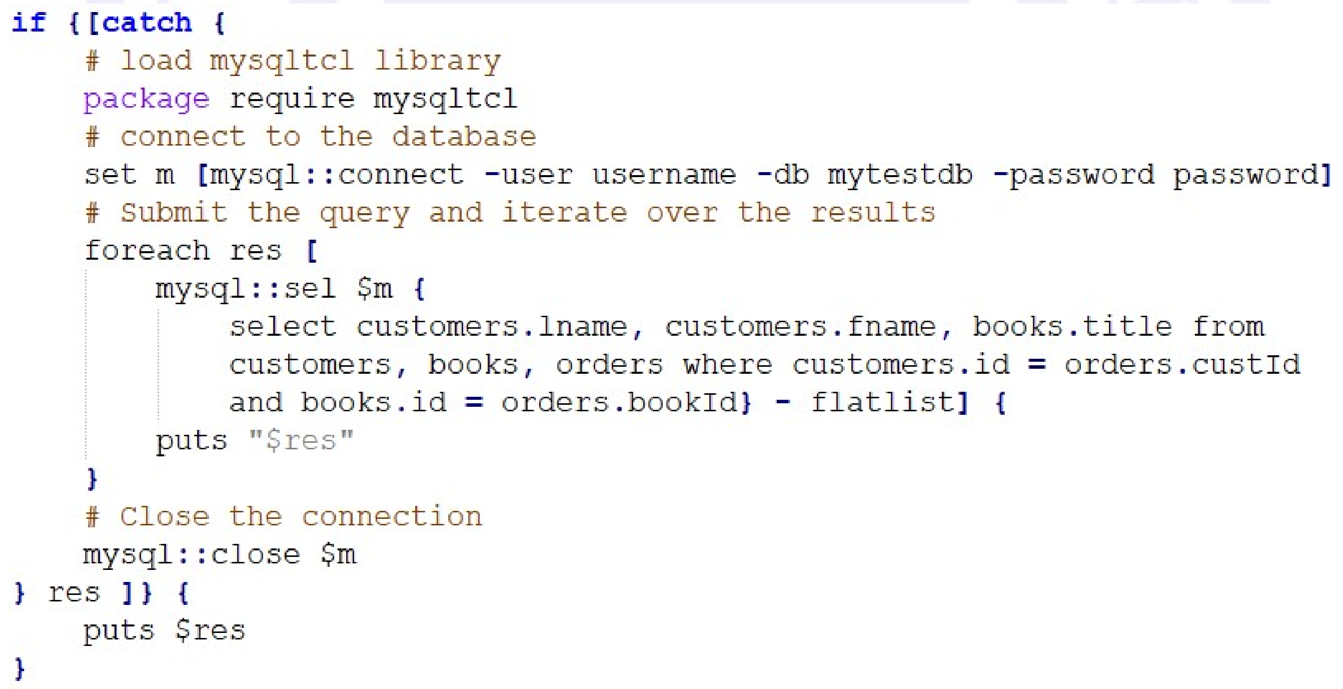
Java and Updating SQL

- INSERT, DELETE, UPDATE or any DDL Command

  execute (**String** sql)

  - True if the first result is a ResultSet object
  - False if it is an update count or there are no results
  - Can be used for SELECT queries also

TCL (MySQL) – “Tickle”



```
if {[catch {
    # load mysqltcl library
    package require mysqltcl
    # connect to the database
    set m [mysql::connect -user username -db mytestdb -password password]
    # Submit the query and iterate over the results
    foreach res [
        mysql::sel $m {
            select customers.lname, customers.fname, books.title from
            customers, books, orders where customers.id = orders.custId
            and books.id = orders.bookId} - flatlist] {
        puts "$res"
    }
    # Close the connection
    mysql::close $m
} res ]} {
    puts $res
}
```

TCL (MySQL)

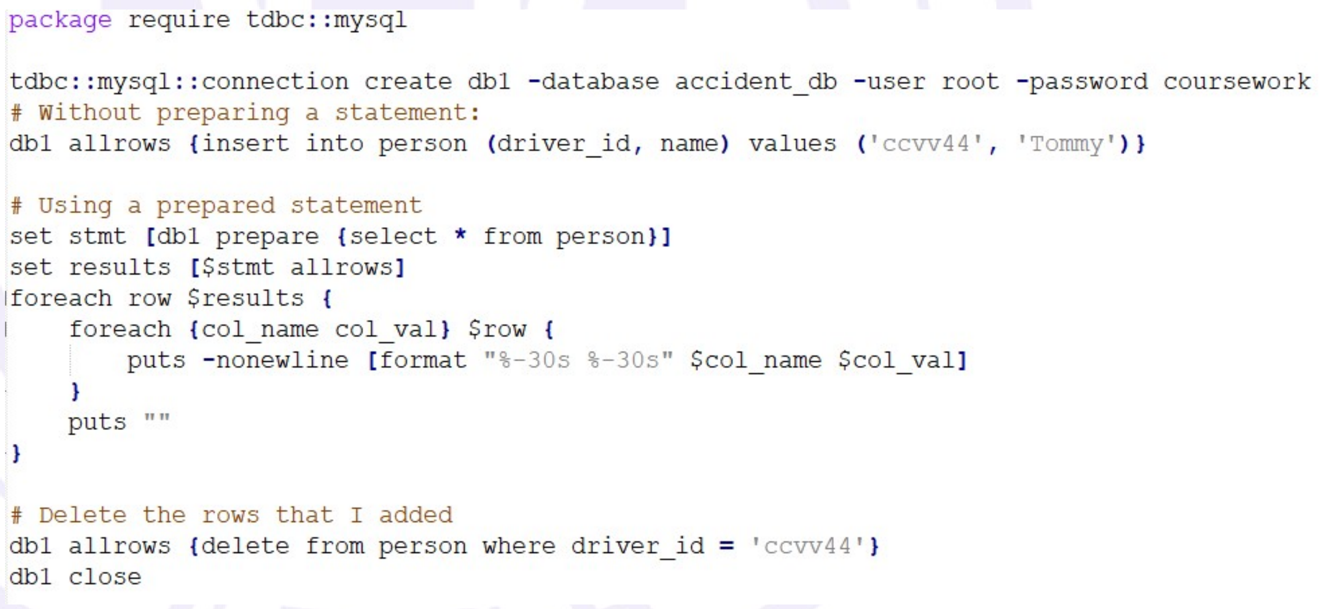
- INSERT, DELETE, UPDATE or any DDL

  mysql::exec handle sql-statement

  - Returns the number of affected rows for DELETE and UPDATE
  - In case of multiple statements, mysql::exec returns a list of number or affected rows
- Run with:
  - tclsh tclEx.tcl
- Documentation
  - http://www.xdobry.de/mysqltcl/mysqltcl.html

TCL and TDBC

- TDBC is an abstraction layer like JDBC
  - Database independent
  - MySQL Example



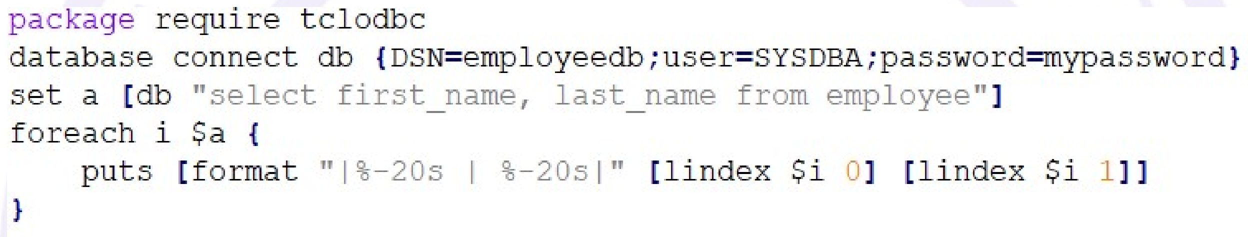
```
package require tdbc::mysql

tdbc::mysql::connection create db1 -database accident_db -user root -password coursework
# Without preparing a statement:
db1 allrows {insert into person (driver_id, name) values ('ccvv44', 'Tommy')}

# Using a prepared statement
set stmt [db1 prepare {select * from person}]
set results [$stmt allrows]
foreach row $results {
    foreach {col_name col_val} $row {
        puts -nonewline [format "%-30s %-30s" $col_name $col_val]
    }
    puts ""
}

# Delete the rows that I added
db1 allrows {delete from person where driver_id = 'ccvv44'}
db1 close
```

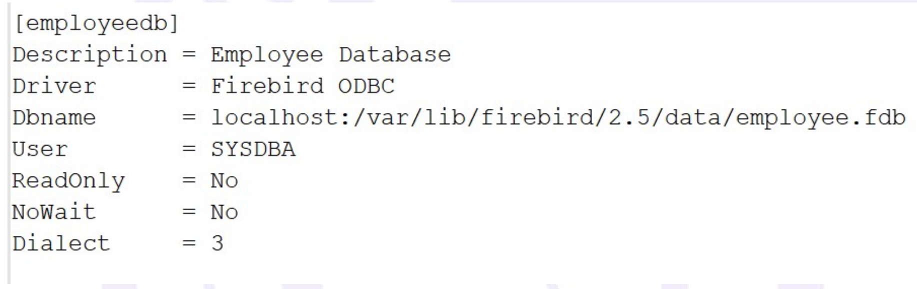
TCL (Firebird via ODBC)



```
package require tclodbc
database connect db {DSN=employeedb;user=SYSDBA;password=mypassword}
set a [db "select first_name, last_name from employee"]
foreach i $a {
    puts [format "|%-20s | %-20s|" [lindex $i 0] [lindex $i 1]]
}
```

TCL:ODBC

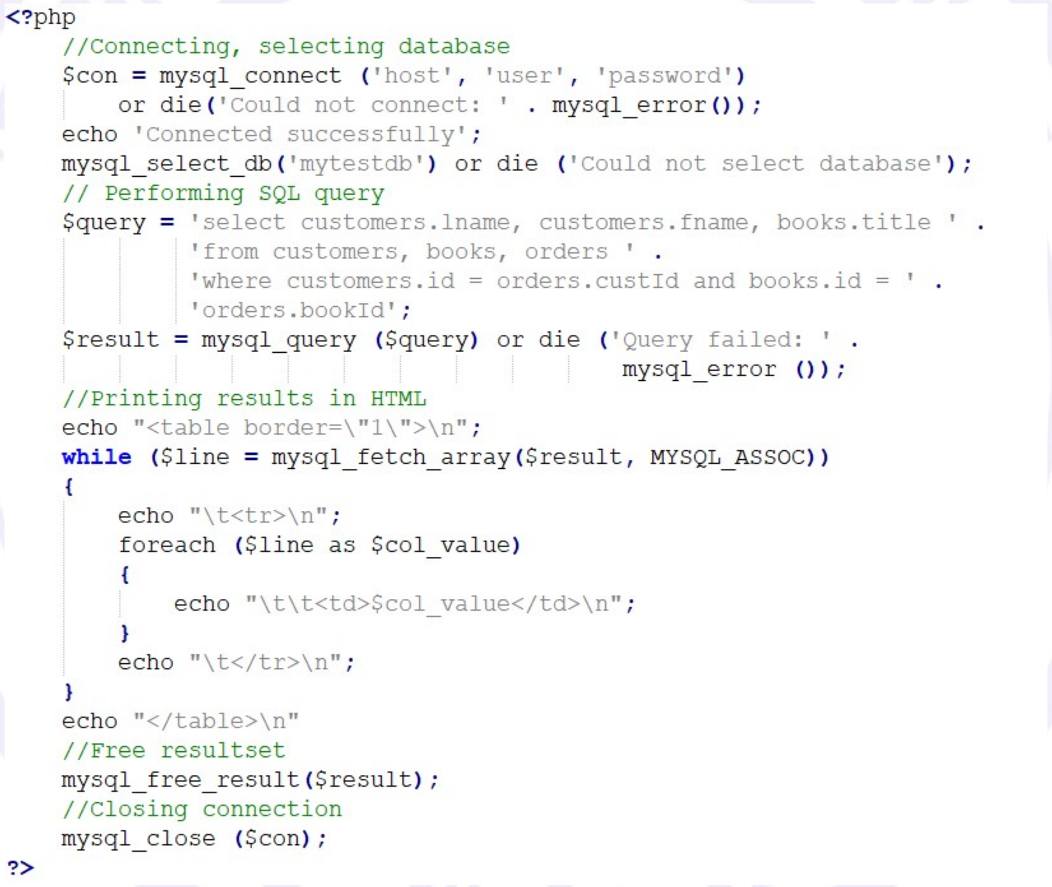
- Need the following .odb.ini file:



```
[employeedb]
Description = Employee Database
Driver      = Firebird ODBC
Dbname      = localhost:/var/lib/firebird/2.5/data/employee.fdb
User        = SYSDBA
ReadOnly    = No
NoWait      = No
Dialect     = 3
```

- Run with:
  - tclsh tclfb.tcl
- Documentation (on Debian/Ubuntu machines)
  - /usr/share/doc/tclodbc

PHP (MySQL)

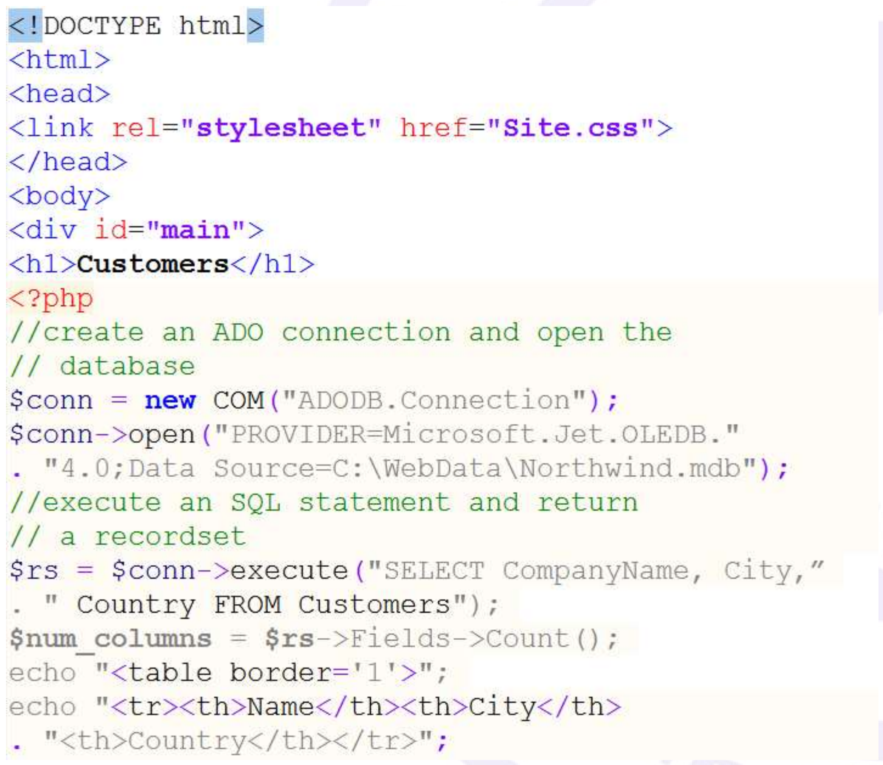


```php
<?php
    //Connecting, selecting database
    $con = mysql_connect ('host', 'user', 'password')
        or die('Could not connect: ' . mysql_error());
    echo 'Connected successfully';
    mysql_select_db('mytestdb') or die ('Could not select database');
    // Performing SQL query
    $query = 'select customers.lname, customers.fname, books.title ' .
             'from customers, books, orders ' .
             'where customers.id = orders.custId and books.id = ' .
             'orders.bookId';
    $result = mysql_query ($query) or die ('Query failed: ' .
                                        mysql_error ());
    //Printing results in HTML
    echo "<table border=\"1\">\n";
    while ($line = mysql_fetch_array($result, MYSQL_ASSOC))
    {
        echo "\t<tr>\n";
        foreach ($line as $col_value)
        {
            echo "\t\t<td>$col_value</td>\n";
        }
        echo "\t</tr>\n";
    }
    echo "</table>\n"
    //Free resultset
    mysql_free_result($result);
    //Closing connection
    mysql_close ($con);
?>
```

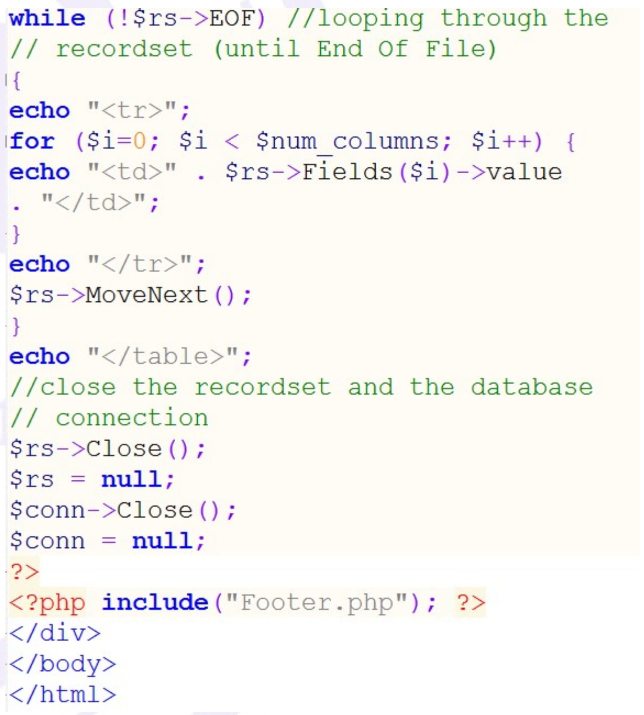
PHP MySQL Documentation

- Run with:
  - Copy phpEx.php to WWW directory
  - Point browser to:
    - http://csc.tntech.edu/~user/phpEx.php
- Documentation: http://php.net/mysql
- PHP has firebird drivers, but I do not include examples
  - Documentation found here: http://php.net/manual/en/book.ibase.php

PHP ADO Example



```
<!DOCTYPE html>
<html>
<head>
<link rel="stylesheet" href="Site.css">
</head>
<body>
<div id="main">
<h1>Customers</h1>
<?php
//create an ADO connection and open the
// database
$conn = new COM("ADODB.Connection");
$conn->open("PROVIDER=Microsoft.Jet.OLEDB."
. "4.0;Data Source=C:\WebData\Northwind.mdb");
//execute an SQL statement and return
// a recordset
$rs = $conn->execute("SELECT CompanyName, City,”
. " Country FROM Customers");
$num_columns = $rs->Fields->Count();
echo "<table border='1'>";
echo "<tr><th>Name</th><th>City</th>
. "<th>Country</th></tr>";
```



```
while (!$rs->EOF) //looping through the
// recordset (until End Of File)
{
echo "<tr>";
for ($i=0; $i < $num_columns; $i++) {
echo "<td>" . $rs->Fields($i)->value
. "</td>";
}
echo "</tr>";
$rs->MoveNext();
}
echo "</table>";
//close the recordset and the database
// connection
$rs->Close();
$rs = null;
$conn->Close();
$conn = null;
?>
<?php include("Footer.php"); ?>
</div>
</body>
</html>
```

Examples from http://www.w3schools.com

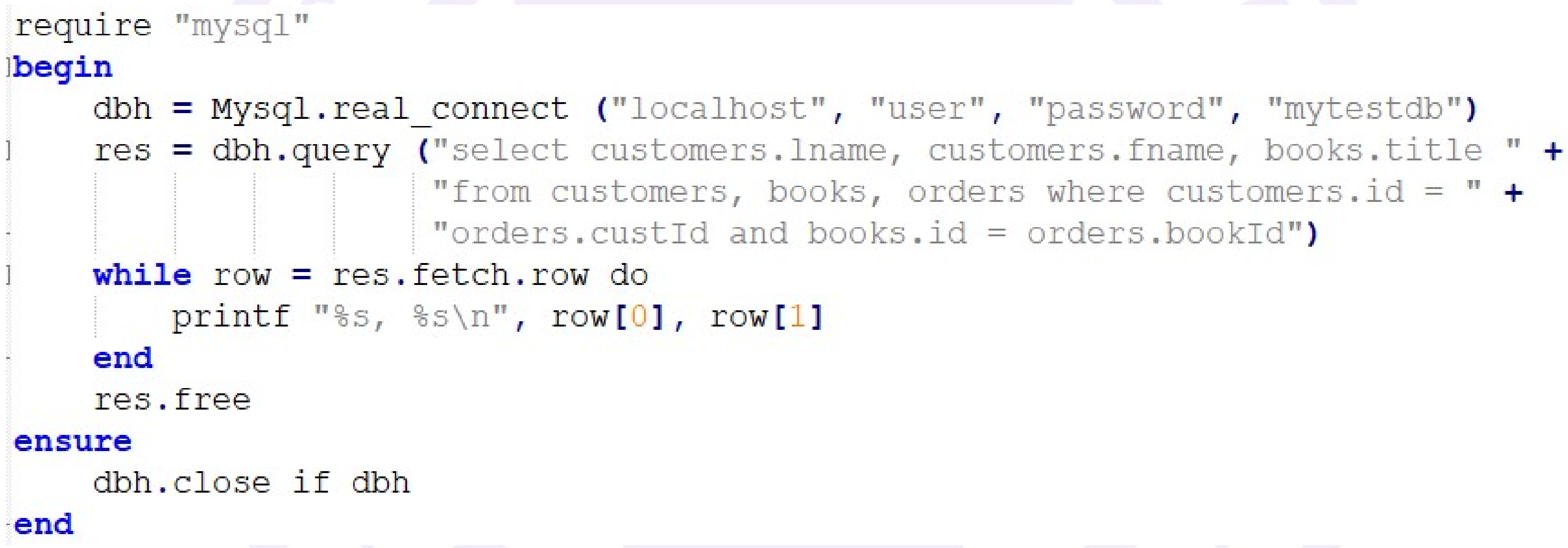
Python (MySQL)

```python
import sys
import MySQLdb
try:
    conn = MySQLdb.connect (host = "localhost", user = "username",
                            passwd = "password", db = "mytestdb")
except MySQLdb.Error, e:
    print "Error %d: %s" % (e.args[0], e.args[1])
    sys.exit (1)
cursor = conn.cursor ()
cursor.execute ("select customers.lname, customers.fname, " +
                "books.title from customers, books, orders where customers.id = " +
                "orders.custId and books.id = orders.bookId")
while (1):
    row = cursor.fetchone()
    if row == None:
        break
    print "%s, %s, %s" % (row[0], row[1], row[2])
```

Python MySQL Documentation

- Run with:
  - python pythonEx.py
- Documentation: http://mysql-python.sourceforge.net

Ruby (MySQL)

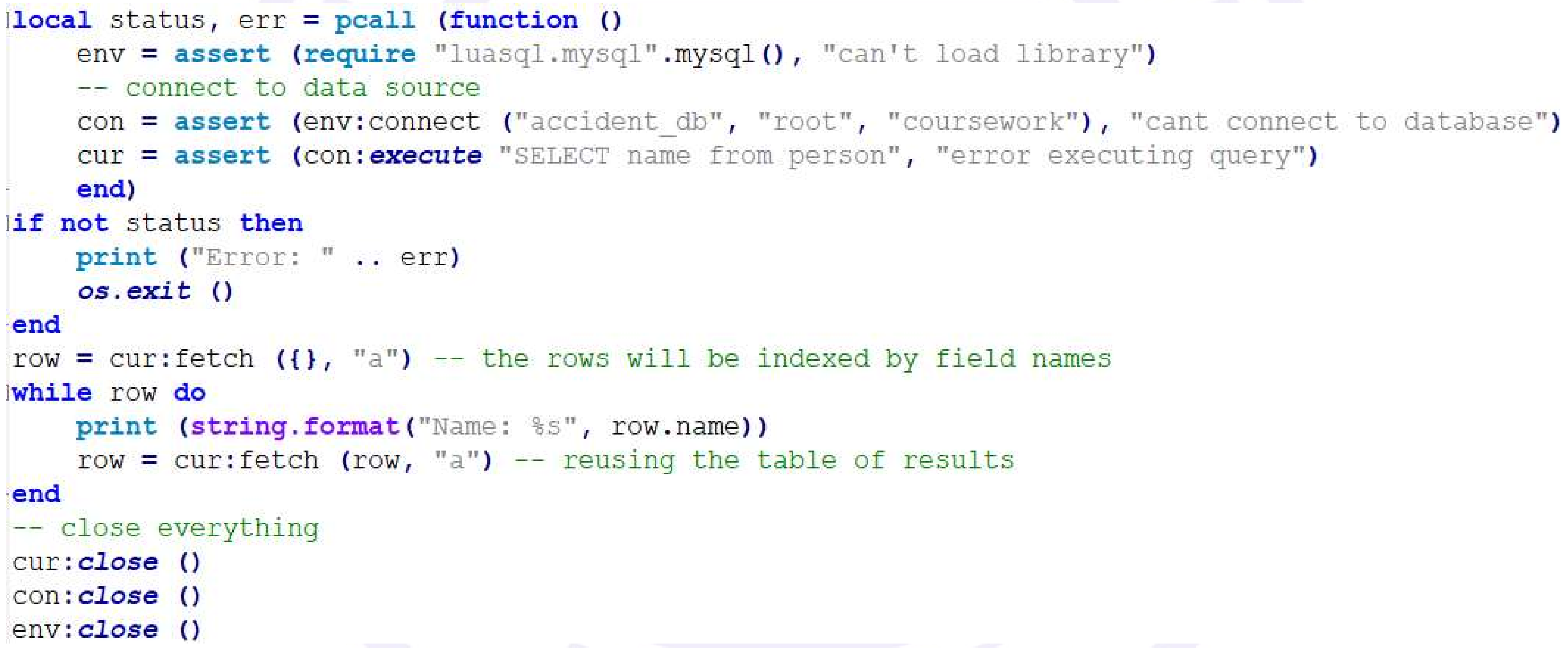


```
require "mysql"
begin
    dbh = Mysql.real_connect ("localhost", "user", "password", "mytestdb")
    res = dbh.query ("select customers.lname, customers.fname, books.title " +
                     "from customers, books, orders where customers.id = " +
                     "orders.custId and books.id = orders.bookId")
    while row = res.fetch.row do
        printf "%s, %s\n", row[0], row[1]
    end
    res.free
ensure
    dbh.close if dbh
end
```

Ruby MySQL Documentation

- Run with:
  - ruby rubyEx.rb
- Documentation: http://www.tmtm.org/en/mysql/ruby/

Lua (MySQL)

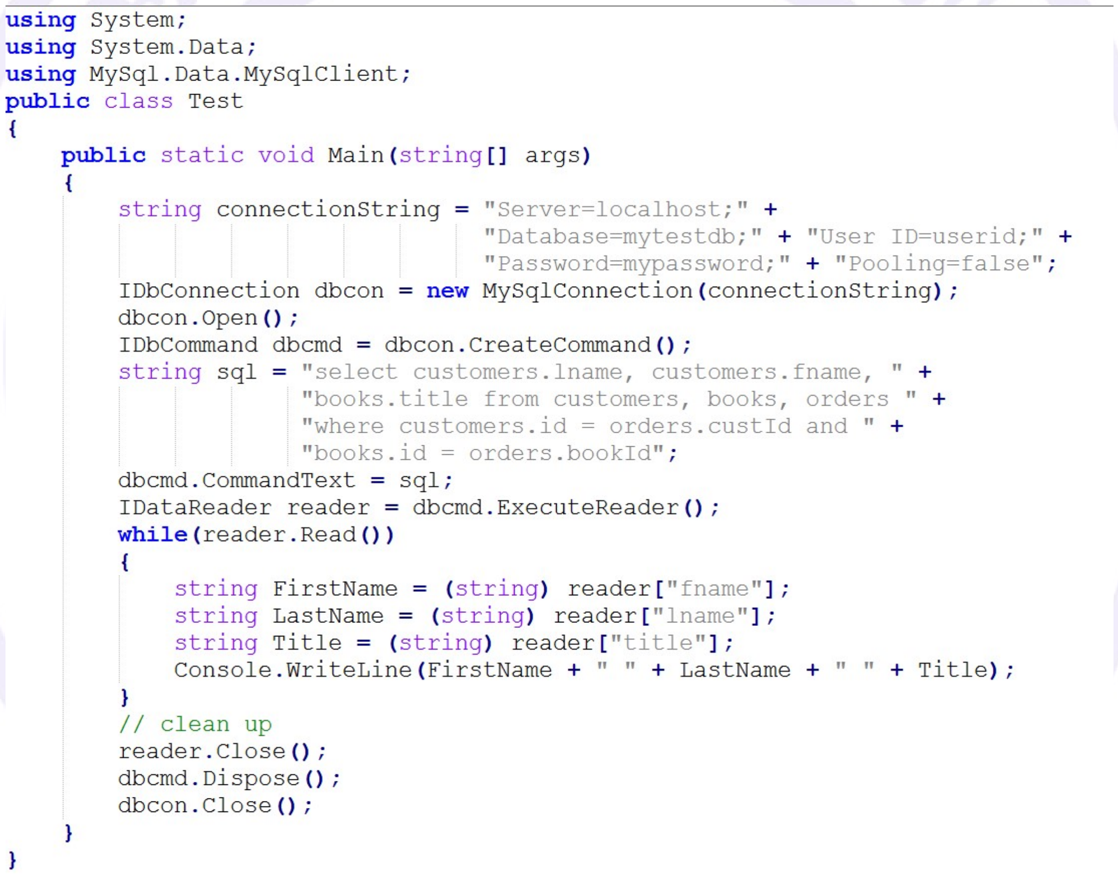


```lua
local status, err = pcall (function ()
    env = assert (require "luasql.mysql".mysql(), "can't load library")
    -- connect to data source
    con = assert (env:connect ("accident_db", "root", "coursework"), "cant connect to database")
    cur = assert (con:execute "SELECT name from person", "error executing query")
    end)
if not status then
    print ("Error: " .. err)
    os.exit ()
end
row = cur:fetch ({}, "a") -- the rows will be indexed by field names
while row do
    print (string.format("Name: %s", row.name))
    row = cur:fetch (row, "a") -- reusing the table of results
end
-- close everything
cur:close ()
con:close ()
env:close ()
```

Lua Documentation

- Run with
  - lua ex.lua
- Documentation for LuaSQL found here:
  - http://keplerproject.github.io/luasql/doc/us/manual.html

C# Mono (MySQL)

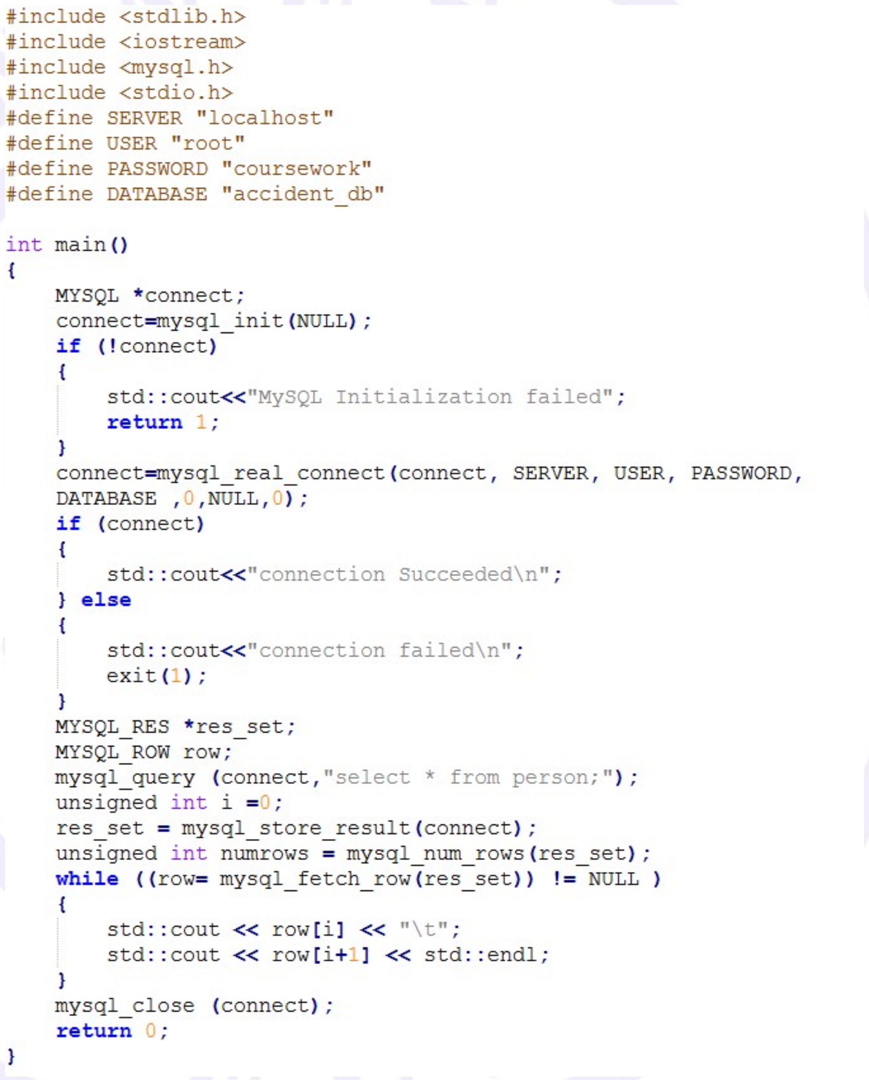


```
using System;
using System.Data;
using MySql.Data.MySqlClient;
public class Test
{
    public static void Main(string[] args)
    {
        string connectionString = "Server=localhost;" +
                                  "Database=mytestdb;" + "User ID=userid;" +
                                  "Password=mypassword;" + "Pooling=false";
        IDbConnection dbcon = new MySqlConnection(connectionString);
        dbcon.Open();
        IDbCommand dbcmd = dbcon.CreateCommand();
        string sql = "select customers.lname, customers.fname, " +
                     "books.title from customers, books, orders " +
                     "where customers.id = orders.custId and " +
                     "books.id = orders.bookId";
        dbcmd.CommandText = sql;
        IDataReader reader = dbcmd.ExecuteReader();
        while(reader.Read())
        {
            string FirstName = (string) reader["fname"];
            string LastName = (string) reader["lname"];
            string Title = (string) reader["title"];
            Console.WriteLine(FirstName + " " + LastName + " " + Title);
        }
        // clean up
        reader.Close();
        dbcmd.Dispose();
        dbcon.Close();
    }
}
```

MySQL Connector / .NET Documentation

- Compile with:
  - dmcs csharpex.cs –r:System.Data.dll –r:/usr/lib/cli/MySql.Data-6.4/MySql.Data.dll
- Run with:
  - mono csharpEx.exe
- Documentation:
  http://dev.mysql.com/doc/refman/5.0/en/connector-net.html
- Firebird example left as an example for the class ☺

C++ (MySQL)



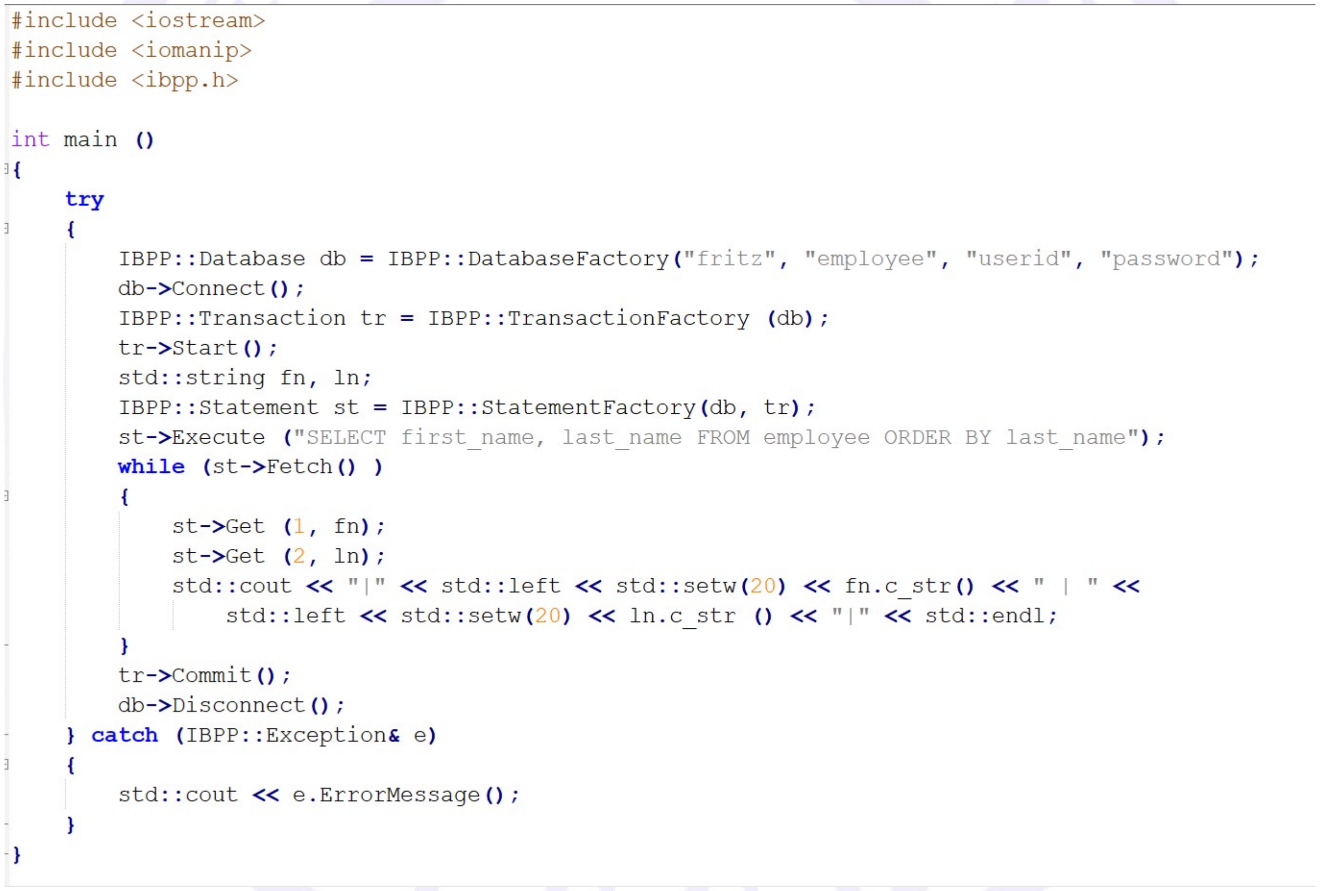
```
#include <stdlib.h>
#include <iostream>
#include <mysql.h>
#include <stdio.h>
#define SERVER "localhost"
#define USER "root"
#define PASSWORD "coursework"
#define DATABASE "accident_db"

int main()
{
    MYSQL *connect;
    connect=mysql_init(NULL);
    if (!connect)
    {
        std::cout<<"MySQL Initialization failed";
        return 1;
    }
    connect=mysql_real_connect(connect, SERVER, USER, PASSWORD,
    DATABASE ,0,NULL,0);
    if (connect)
    {
        std::cout<<"connection Succeeded\n";
    } else
    {
        std::cout<<"connection failed\n";
        exit(1);
    }
    MYSQL_RES *res_set;
    MYSQL_ROW row;
    mysql_query (connect,"select * from person;");
    unsigned int i =0;
    res_set = mysql_store_result(connect);
    unsigned int numrows = mysql_num_rows(res_set);
    while ((row= mysql_fetch_row(res_set)) != NULL )
    {
        std::cout << row[i] << "\t";
        std::cout << row[i+1] << std::endl;
    }
    mysql_close (connect);
    return 0;
}
```

C++ Documentation

- Compile with:
  - g++ -lmysqlclient –o tst –I/usr/include/mysql tst.cpp
- Run with:
  - ./tst
- Documentation: http://dev.mysql.com/doc/refman/5.6/en/c-api.html
- Good tutorial: http://zetcode.com/db/mysqlc/

C++ (Firebird)



```cpp
#include <iostream>
#include <iomanip>
#include <ibpp.h>

int main ()
{
    try
    {
        IBPP::Database db = IBPP::DatabaseFactory("fritz", "employee", "userid", "password");
        db->Connect();
        IBPP::Transaction tr = IBPP::TransactionFactory (db);
        tr->Start();
        std::string fn, ln;
        IBPP::Statement st = IBPP::StatementFactory(db, tr);
        st->Execute ("SELECT first_name, last_name FROM employee ORDER BY last_name");
        while (st->Fetch() )
        {
            st->Get (1, fn);
            st->Get (2, ln);
            std::cout << "|" << std::left << std::setw(20) << fn.c_str() << " | " <<
                std::left << std::setw(20) << ln.c_str () << "|" << std::endl;
        }
        tr->Commit();
        db->Disconnect();
    } catch (IBPP::Exception& e)
    {
        std::cout << e.ErrorMessage();
    }
}
```

IBPP Documentation

- Compile with:
  - g++ -D$os_flag –libpp –o ex2 ex2.cpp
    - Where $os_flag is IBPP_WINDOWS, IBPP_LINUX, or IBPP_DARWIN
    - On your VM, it looks like:

      g++ -DIBPP_LINUX –o ex2 –lfbclient –libpp ex2.cpp
- Run with:
  - ./ex2
- Documentation: http://www.ibpp.org/reference
- See the following for C++ MySQL examples: http://dev.mysql.com/tech-resources/articles/mysql-connector-cpp.html

End of Language Bindings