Embedded SQL in C - using libpq

1. PostgreSQL provides a library for C - libpq. It has functions - PQconnectdb, PQexec, PQfinish etc.  
   http://www.postgresql.org/docs/7.4/static/libpq.html

2. Connect to VPN. It is needed because our database server does not accept any connections from outside the network. Check the links below for how to connect to VPN.  
   http://cat.pdx.edu/windows/connecting-to-the-vpn-on-windows.html  
   http://cat.pdx.edu/linux/connecting-to-the-vpn-on-linux-2.html  
   http://cat.pdx.edu/mac/connecting-to-the-vpn-from-mac-2.html

3. Other files required to compile the program:  
   - libpq-fe.h  
   - libpq.lib  
   - libpq.dll  
   How to get these files?  
   I got them when I installed PostgreSQL on my machine. I can provide those for Windows machines.

For Mac : (Student post on piazza)  
If you've installed the Xcode tools (free on Apple's app store), it should have installed the libpq libraries as well. You can create a new project in Xcode, add libpq into the list of linked libraries, and include the header <libpq-fe.h> in main.c.

4. How to compile?  
   (Make sure that all the above files and your program are in the same folder.)  
   gcc   -L. -lpq -o movies movies.c

5. Here’s the sample program:

```c
#include <stdio.h>
#include "libpq-fe.h"

void main() {  
   const char *conninfo;  
   PGconn    *conn;  
   PGresult  *res;  
   int         i, j, nFields, tuples;

   conninfo = "host=131.252.208.122 port=5432 dbname=w13db2 user=w13db2 password=root";

   /* Make a connection to the database */  
   conn = PQconnectdb(conninfo);
```
if (PQstatus(conn) != CONNECTION_OK) {
    printf("Connection to database failed: %s",
            PQerrorMessage(conn));
} else {
    printf("Connection Successful!
");
}

/* Execute a query */
res = PQexec(conn, "SELECT * FROM movies");
if (PQresultStatus(res) != PGRES_TUPLES_OK) {
    printf("SELECT failed: %s", PQerrorMessage(conn));
    //PQclear(res);
} else {
    printf("SELECT Successful!
");
}

tuples = PQntuples(res);
printf ("Number of tuples: %d
\n", tuples);

/* first, print out the attribute names */
nFields = PQnfields(res);
for (i = 0; i < 2; i++)
    printf("%-50s", PQfname(res, i));
printf("\n");
for (i = 0; i < 60; i++)
    printf("%-c", '-');
printf("\n");

/* next, print out the rows */
for (i = 0; i < PQntuples(res); i++)
{
    for (j = 0; j < 2; j++)
        printf("%-15s", PQgetvalue(res, i, j));
    printf("\n");
}

PQclear(res);

/* close the connection to the database and cleanup */
PQfinish(conn);

cgetchar();
}