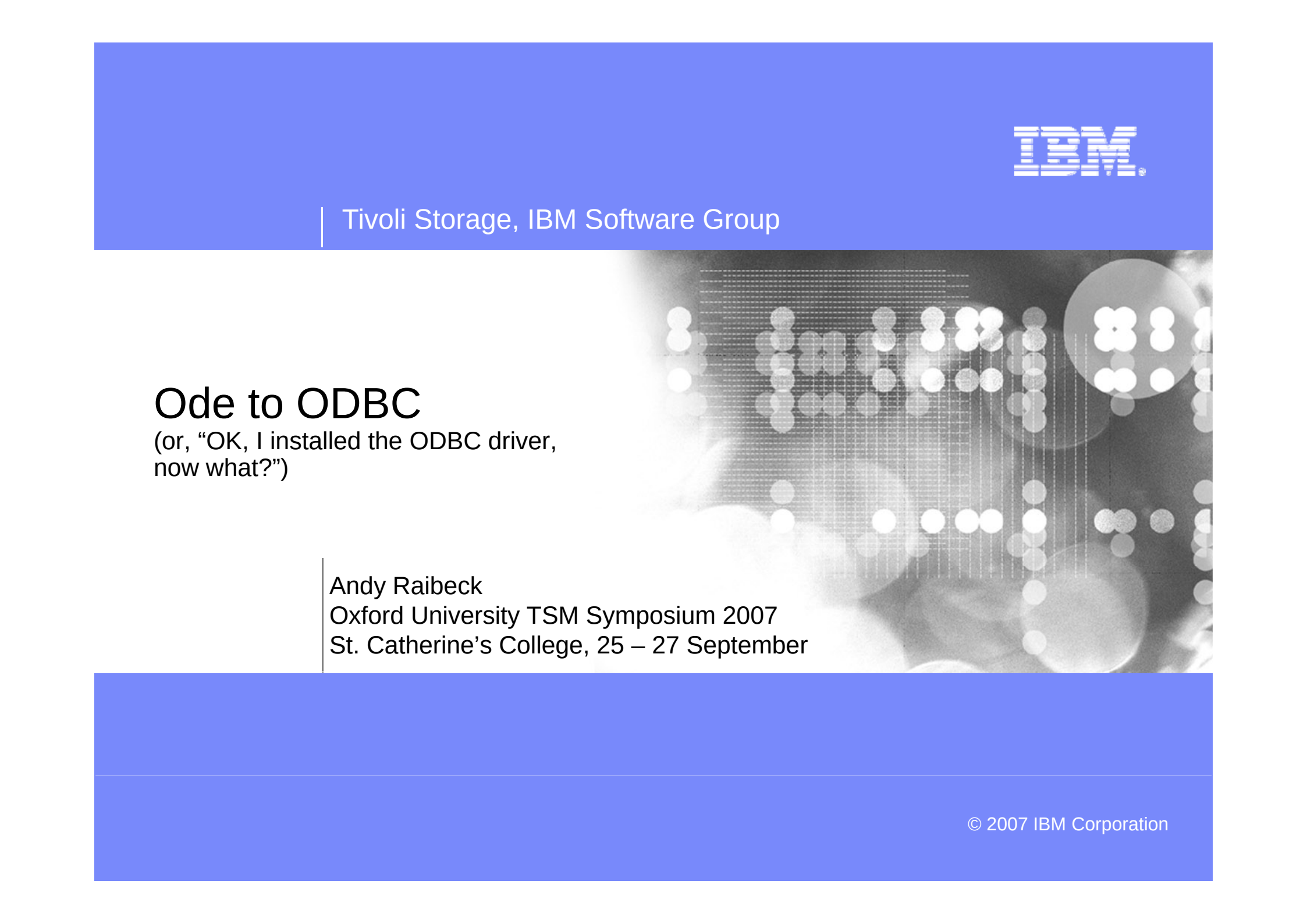


Ode to ODBC

(or, “OK, I installed the ODBC driver, now what?”)

Andy Raibeck

Oxford University TSM Symposium 2007
St. Catherine’s College, 25 – 27 September

Trademarks

- The following terms are trademarks of International Business Machines Corporation in the United States, other countries, or both:
  - AIX
  - IBM
  - Tivoli
- Other company, product, and service names may be trademarks or service marks of others.
- Microsoft, Windows, Windows NT, and the Windows logo are trademarks of Microsoft Corporation in the United States, other countries, or both.
- UNIX is a registered trademark of The Open Group in the United States and other countries.

Abstract

One of the most frequently asked questions about Tivoli Storage Manager is how to query the TSM server database to generate custom reports. The answers often entail some form of the SQL SELECT statement, usually issued from the TSM Administrative command line interface. Despite having an ODBC driver since Version 2, it seems that the ODBC driver is among the least understood of TSM's many standard components. This session will focus on the TSM ODBC driver.

Agenda

- What ODBC is
- How ODBC works
- How to configure the TSM ODBC driver
- ODBC programming fundamentals
- Examples

Agenda

- **What ODBC is**
- How ODBC works
- How to configure the TSM ODBC driver
- ODBC programming fundamentals
- Examples

What ODBC is

- ODBC: Open Database Connectivity
- ODBC is a specification for a generalized database application programming interface (API).
- Designed for maximum operability: the ability for a single application to access different database management systems (DBMS's) using the same source code.
- Uses Structured Query Language (SQL) as the database access language.

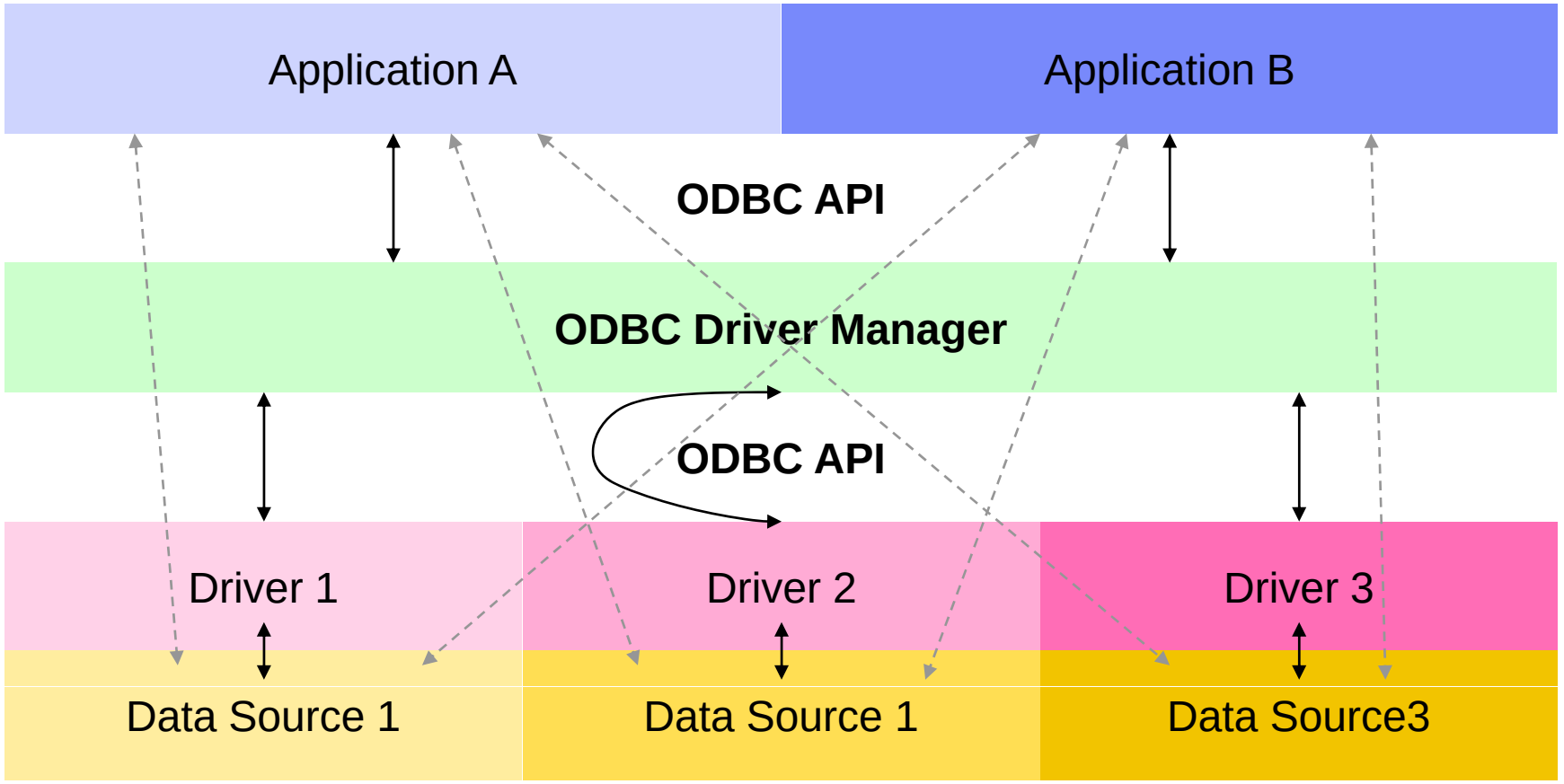
What ODBC is (*continued*)

- About ODBC drivers:
  - An ODBC driver is an implementation of the ODBC API .
  - Presents the same external interface (the ODBC API) to application developers.
  - The internal implementation is specific to the given DBMS.
  - Usually implemented by the DBMS developers.

What ODBC is (*continued*)

- About the TSM ODBC driver:
  - Based on Microsoft's ODBC 3.51 specification
  - Supports multithreading (version 5.4.1.2)
  - Supports ODBC 2.x and 3.x applications
  - Can be used by a variety of applications and programming languages custom reporting
  - A specialized type of administrative client
  - A programmatic interface to the TSM server SELECT command
  - Is read-only: cannot be used to modify the TSM server database

What ODBC is (*continued*)



Agenda

- What ODBC is
- **How ODBC works**
- How to configure the TSM ODBC driver
- ODBC programming fundamentals
- Examples

Driver-to-server communication

- TSM ODBC support uses a cursor mechanism is used to process SELECT statements and return the results.
- A cursor is a pointer to the current row in a result set.
- The TSM ODBC implementation uses several undocumented admin commands to create, manipulate, and delete cursors.
-  Described here for informational purposes only. If you are not the TSM ODBC driver, you should not use them.

Cursor commands

- `DEFINE CURSOR cursorname SQL="select_statement"`
  - Creates a cursor which will point to the result set of the specified SELECT statement.
  - Multiple cursors can be defined within a given admin session.
  - A given cursor can be used only by the admin session in which it is created.
- `DELETE CURSOR cursorname`
  - Closes the named cursor, then deletes it.
  - Any pending results are lost.
  - The cursor cannot be opened again.

Cursor commands (*continued*)

- `OPEN CURSOR` *`cursorname`*
  - Creates the result set to which the specified cursor points.
  - Only one cursor can be open at a time in a given admin session.
- `FETCH NEXT` *`n`*
  - Obtains the next *n* rows from the result set pointed to by the currently open cursor.
- `CLOSE CURSOR`
  - Closes the currently open cursor.
  - Any pending results are lost.
  - The cursor can be opened again, but will point to the first record in the result set.

Cursor commands (*continued*)

- `QUERY CURSOR [`*`pattern`*`]`
  - *pattern* can be a fully-qualified name or a pattern using **?** or *****.
  - Default pattern is ***** (all cursors).
  - For each cursor, displays:
    - Cursor name
    - "Open" indicator
    - Number of rows fetched
    - "End-of-table" indicator
    - SELECT statement

Cursor commands (*continued*)

- `DISPLAY CURSOR` *`cursorname`*
  - For each column in the result set, displays:
    - The column name
    - The column data type, represented as a numeric value (undocumented)
    - The maximum length of the column
      - 0 if data type dictates the length
    - The scale (number of positions to the right of the decimal point) of the column
      - 0 if inappropriate to the data type

Cursor usage example

```
tsm: SS2_ADSM_GROUP_SERVER>define cursor amr_cursor sql="select node_name, files
pace_name, date(backup_end) as ""BACKUP END DATA"" from filespaces where backup_
end is null or backup_end < current_timestamp - 10 days"

tsm: SS2_ADSM_GROUP_SERVER>query cursor

Cursor      Open?     Num.  End Of  SQL Expression
Name                  Rows  Table?
                   Fetched
----------  -----  --------  ------  ------------------------------------------
AMR_CURSOR  No           0    No    select node_name, filespace_name,
                                     date(backup_end) as "BACKUP END DATA" from
                                     filespaces where backup_end is null or
                                     backup_end < current_timestamp - 10 days

tsm: SS2_ADSM_GROUP_SERVER>display cursor amr_cursor

NODE_NAME      FILESPACE_NAME      BACKUP END DATA
---------      --------------      ---------------
         7                   7                    8
        64               1,024                    0
         0                   0                    0
```

Cursor usage example (*continued*)

```
tsm: SS2_ADSM_GROUP_SERVER>open cursor amr_cursor

tsm: SS2_ADSM_GROUP_SERVER>query cursor

Cursor      Open?     Num.  End Of SQL Expression
Name                  Rows  Table?
                   Fetched
----------- ----- -------- ------ -------------------------------------------
AMR_CURSOR  Yes          0   No    select node_name, filespace_name,
                                    date(backup_end) as "BACKUP END DATA" from
                                    filespaces where backup_end is null or
                                    backup_end < current_timestamp - 10 days

tsm: SS2_ADSM_GROUP_SERVER>fetch next 1

NODE_NAME              FILESPACE_NAME         BACKUP END DATA
------------------     ------------------     ---------------
3A6929                 \\ibm-130af90178f-
                        \c$
```

Cursor usage example (*continued*)

```
tsm: SS2_ADSM_GROUP_SERVER>fetch next 4

NODE_NAME               FILESPACE_NAME         BACKUP END DATA
------------------      ------------------     ---------------
3A6929                  \\n3a6929b\c$
ADSMAPAR                \\apardude\c$               2005-03-14
ADSMAPAR                \\branch\c$
ADSMAPAR                \\broyhill\c$

tsm: SS2_ADSM_GROUP_SERVER>query cursor

Cursor     Open?     Num.  End Of SQL Expression
Name                 Rows  Table?
                  Fetched
---------- ----- -------- ------ -------------------------------------------
AMR_CURSOR  Yes         5   No   select node_name, filespace_name,
                                  date(backup_end) as "BACKUP END DATA" from
                                  filespaces where backup_end is null or
                                  backup_end < current_timestamp - 10 days
```

Cursor usage example (*continued*)

```
tsm: SS2_ADSM_GROUP_SERVER>fetch next 2

NODE_NAME               FILESPACE_NAME         BACKUP END DATA
------------------      ------------------     ---------------
ADSMAPAR                \\jee\c$
ADSMPERF                /                           2005-06-30

tsm: SS2_ADSM_GROUP_SERVER>query cursor

Cursor     Open?     Num.  End Of SQL Expression
Name                 Rows  Table?
                  Fetched
---------- ----- -------- ------ -------------------------------------------
AMR_CURSOR  Yes         7   No   select node_name, filespace_name,
                                  date(backup_end) as "BACKUP END DATA" from
                                  filespaces where backup_end is null or
                                  backup_end < current_timestamp - 10 days

tsm: SS2_ADSM_GROUP_SERVER>close cursor
```

Cursor usage example (*continued*)

```
tsm: SS2_ADSM_GROUP_SERVER>query cursor

Cursor      Open?     Num.  End Of  SQL Expression
Name                  Rows  Table?
                   Fetched
----------  -----  --------  ------  ------------------------------------------
AMR_CURSOR  No           0    No    select node_name, filespace_name,
                               date(backup_end) as "BACKUP END DATA" from
                               filespaces where backup_end is null or
                               backup_end < current_timestamp - 10 days

tsm: SS2_ADSM_GROUP_SERVER>delete cursor amr_cursor

tsm: SS2_ADSM_GROUP_SERVER>query cursor
ANR2034E QUERY CURSOR: No match found using this criteria.
ANS8001I Return code 11.

tsm: SS2_ADSM_GROUP_SERVER>
```

Agenda

- What ODBC is
- How ODBC works
- **How configure the TSM ODBC driver**
- ODBC programming fundamentals
- Examples

Getting the TSM ODBC driver

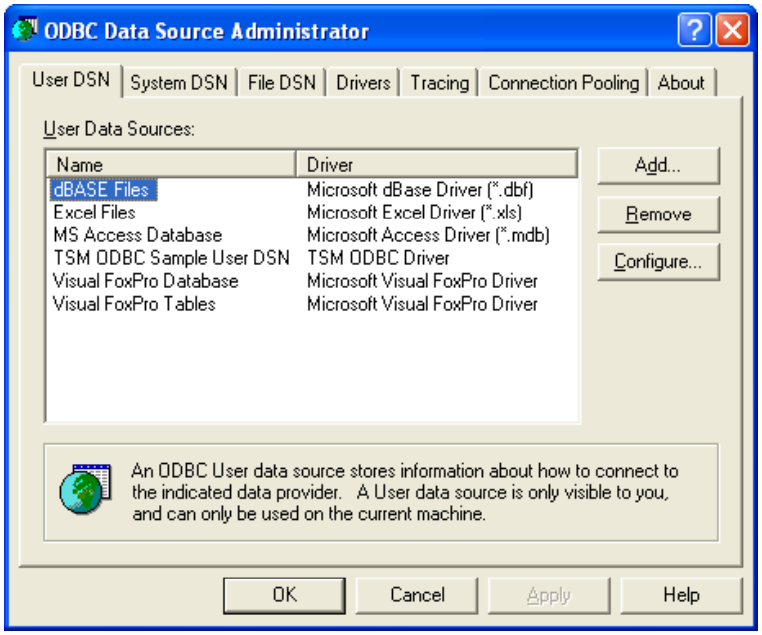
- The ODBC driver is considered a component of the Backup-Archive client for Windows, but is packaged separately from the client.
- Recommended minimum version: 5.3.5.0 or 5.4.1.0
- Start from:

  `http://www.ibm.com/software/sysmgmt/products/support/IBMTivoliStorageManager.html`

- "Fixes by version" link
- "Downloads for …" link for the desired client version
- "Windows x32 client" link
- Look for the files with "ODBC" in the name

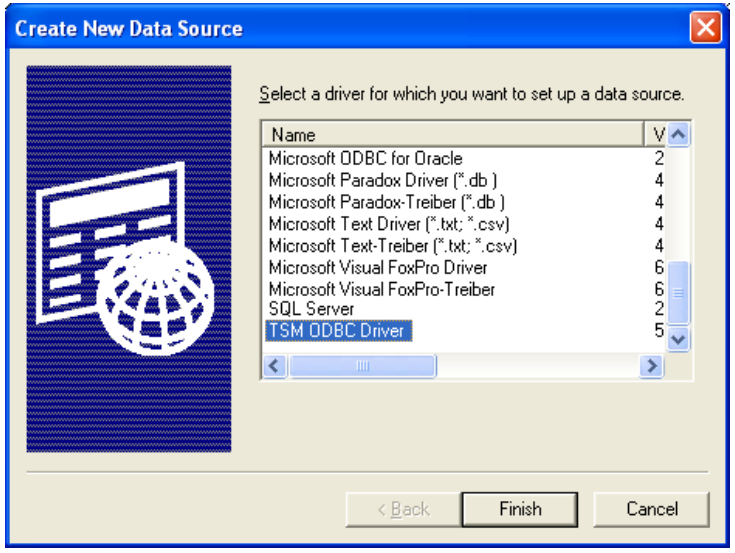
Configuring the ODBC driver

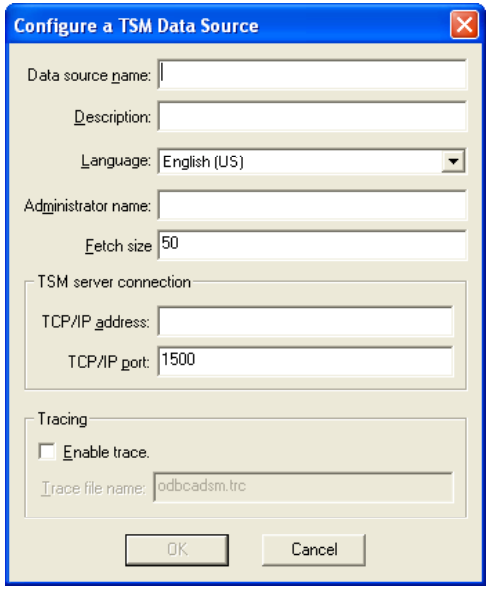
- Assumption: once you've downloaded the package, you can figure out how to install on your own. ☺
- After the driver is installed, start the ODBC Data Source Administrator (odbcad32.exe)



Configuring the ODBC driver (*continued*)

- Click "Add…" to bring up the "Create new data source" dialog.
- Select the TSM ODBC driver.
- Click "Finish" to bring up the "Configure a TSM Data Source dialog.



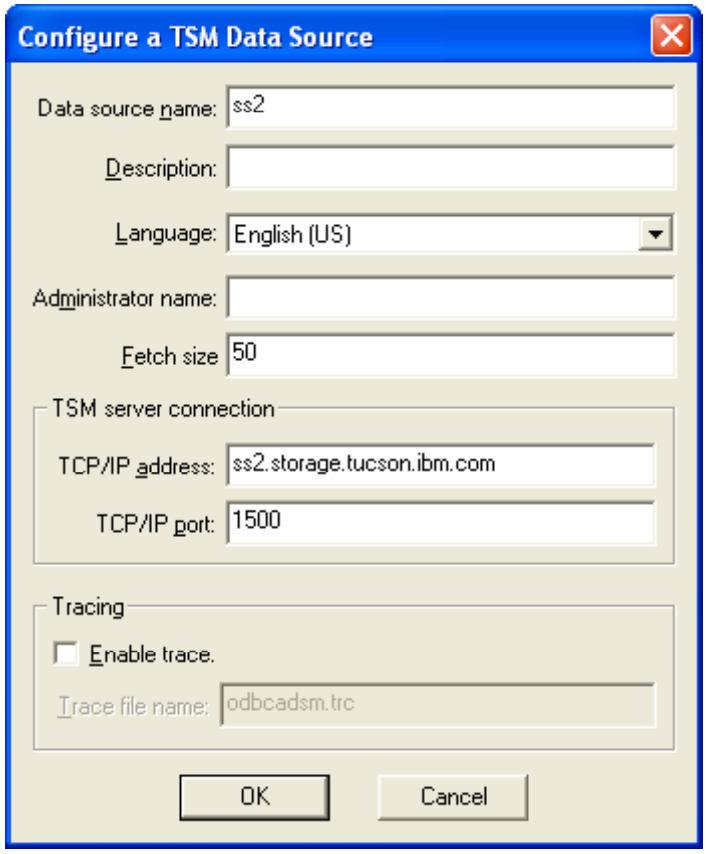


Configuring the ODBC driver (*continued*)

- Fill in the dialog fields:
  - "Data source name" must be unique, suggest using the TSM server name. (required)
  - "Description" is free-form text. (optional)
  - Available "Language" items depends on what was selected during the install. (required)
  - "Fetch" influences the FETCH NEXT command. Higher values can improve performance. (required)
  - "TCP/IP address" and "TCP/IP port" are for the TSM server. (required)
  - Click "OK" when done, then "OK" again to close the ODBC Data Source Administrator.

Configuring the ODBC driver (*continued*)

- Completed dialog:



Agenda

- What ODBC is
- How ODBC works
- How configure the TSM ODBC driver
- **ODBC programming fundamentals**
- Examples

Core program functionality

- Allocate and deallocate
  - Environment handle
  - Connection handle
  - Statement handle
- Connect
- Execute SELECT statements
- Retrieve results
- Cleanup

Core program functionality (*continued*)

- Environment handle
  - Stores global information.
  - Keeps track of connection handles.
- Connection handle
  - Stores information pertaining to a given connection to a TSM server.
  - Keeps track of statement handles.
  - Has one "parent" environment handle.
- Statement handle
  - Stores information pertaining to a given SELECT statement.
  - Has one "parent" connection handle.
- Each handle will establish a session with the TSM server.

Core program functionality (*continued*)

- Connect
  - After allocating the environment handle and statement handle, you can establish a connection to the TSM server.
- Execute SELECT statements
  - Once a connection handle has connected to the TSM server, a statement handle can be allocated.
  - The statement handle is used to execute the SELECT statement.
- Retrieve results
  - After the SELECT statement executes, the results are retrieved via the statement handle.
- Cleanup
  - When done, free up any statement, connection and environment handles.

Agenda

- What ODBC is
- How ODBC works
- How configure the TSM ODBC driver
- ODBC programming fundamentals
- **Examples**

C – The "native" ODBC interface

- Use the Microsoft Windows platform SDK and Microsoft C compiler (cl.exe)
- Platform SDK can be found here
  - http://www.microsoft.com/downloads
  - Search on

    "Windows Server 2003 R2 Platform SDK"

C – The "native" ODBC interface (*continued*)

- Code needs the following up front:

```
#include <windows.h>
#include <sql.h>
#include <sqlext.h>
```

- Allocate the handles:

```
SQLAllocHandle(SQL_HANDLE_ENV, SQL_NULL_HANDLE, &h_env);
SQLSetEnvAttr(h_env,
              SQL_ATTR_ODBC_VERSION,
              (SQLPOINTER)SQL_OV_ODBC3,
              SQL_IS_UINTEGER);
SQLAllocHandle(SQL_HANDLE_DBC, h_env, &h_dbc);
SQLAllocHandle(SQL_HANDLE_STMT, h_dbc, &h_stmt);
```

- The environment must be set to the ODBC version you intend to use (SQL_OV_ODBC2 or SQL_OV_ODBC3).

C – The "native" ODBC interface (*continued*)

- Execute the SQL SELECT statement:

```
SQLExecDirect(h_stmt, sql, SQL_NTS);
```

- Determine how many columns are in the result set.
  - If the SELECT statement is fixed, you should already know how many columns to expect.

```
SQLNumResultCols(h_stmt, &num_cols);
```

C – The "native" ODBC interface (*continued*)

- For each column in the result set, get the column metadata:
  - The columns are numbered starting with '1'.

```
for (i = 1; i <= num_cols; i++)
{
   rc = SQLDescribeCol(h_stmt,
                       i,
                       col_name,
                       sizeof(col_name),
                       &col_name_len,
                       &col_type,
                       &col_size[i],
                       &col_scale,
                       &col_nullable);
   ...
}
```

- Some or all of this info should be saved to an array, one element per column.
- This data can be used to size the data areas used to store the results.

C – The "native" ODBC interface (*continued*)

- Bind data areas to each column in the result set:

```
for (i = 1; i <= num_cols; i++)
{
   rc = SQLBindCol(h_stmt,
                   i,
                   SQL_C_CHAR,
                   &data[i],
                   col_size[i],
                   &size_ind[i]);
}
```

C – The "native" ODBC interface (*continued*)

- Now we are ready to retrieve the data:

```
rc = SQL_SUCCESS;

while (rc != SQL_NO_DATA)
{
   rc = SQLFetch(h_stmt);

   for (i = 1; i <= num_cols; i++)
   {
      printf("%s", data[i]);

      if (i < num_cols)
         printf(",");
      else
         printf("\n");
   }
}
```

C – The "native" ODBC interface (*continued*)

- Cleanup

```
/* Make sure to free any malloc'd memory */

SQLFreeHandle(SQL_HANDLE_STMT, h_stmt);

SQLDisconnect(h_dbc);

SQLFreeHandle(SQL_HANDLE_DBC, h_dbc);

SQLFreeHandle(SQL_HANDLE_ENV, h_env);
```

- Notice that the deallocation is done in the reverse order of the allocation.

Java

- Java has an analogous interface to ODBC called JDBC.
- Java provides a JDBC/ODBC "bridge" that allows Java programs to use ODBC.
- JDBC is very rich in functionality.
- The next few slides show a simple[r] implementation of programming a SELECT statement.

Java (*continued*)

- Code needs the following up front:

```
import java.sql.*;
```

- In main(), declare these objects:

```
Connection  dbc    = null;
Statement   stmt   = null;
ResultSet   rs     = null;
```

- Notice that no environment object is needed.
- Load the JDBC/ODBC bridge:

```
Class.forName("sun.jdbc.odbc.JdbcOdbcDriver");
```

Java (*continued*)

- Establish the ODBC connection:

```
dbc = DriverManager.getConnection("jdbc:odbc:" + dsn,
                                  uid, pwd);
```

- Allocate the statement handle and execute the query:

```
stmt = dbc.createStatement();
rs = stmt.executeQuery(query);
```

Java (*continued*)

- Get the results, row by row, and display them:

```
if (!rs.next())
  System.out.println("No records to display");
else
{
  do
  {
    System.out.println(rs.getString("NODE_NAME") + ","
    + rs.getString("FILESPACE_NAME") + ","
    + rs.getString("Unnamed[3]"));
  }
  while (rs.next());
}
```

- Finally, close up shop:

```
stmt.close();
dbc.close();
```

Perl

- Other "scripting" languages such as Perl, Python, Ruby, and VBScript, can hide a lot of the complexity of using ODBC.
- For Per, The TSM ODBC driver is known to work with ActiveState Perl 5.6.1 build 638 and the Roth Win32::ODBC package (Win32-ODBCBeta, 0,1991.12.21). This package can be installed with PPM using the following command:

```
ppm install http://www.roth.net/perl/packages/win32-odbcbeta.ppd
```

- The next two slides show a simple implementation of our SELECT statement.

Perl (*continued*)

- Pick up the Win32::ODBC package:

```
use WIN32::ODBC;
```

- Connect to the TSM server:

```
my $db = new Win32::ODBC("DSN=$dsn;UID=$uid;PWD=$pwd");
```

- Notice that we don't even bother with "handles".
- Execute the query:

```
$db->Run($query)
```

Perl (*continued*)

- Display the results:

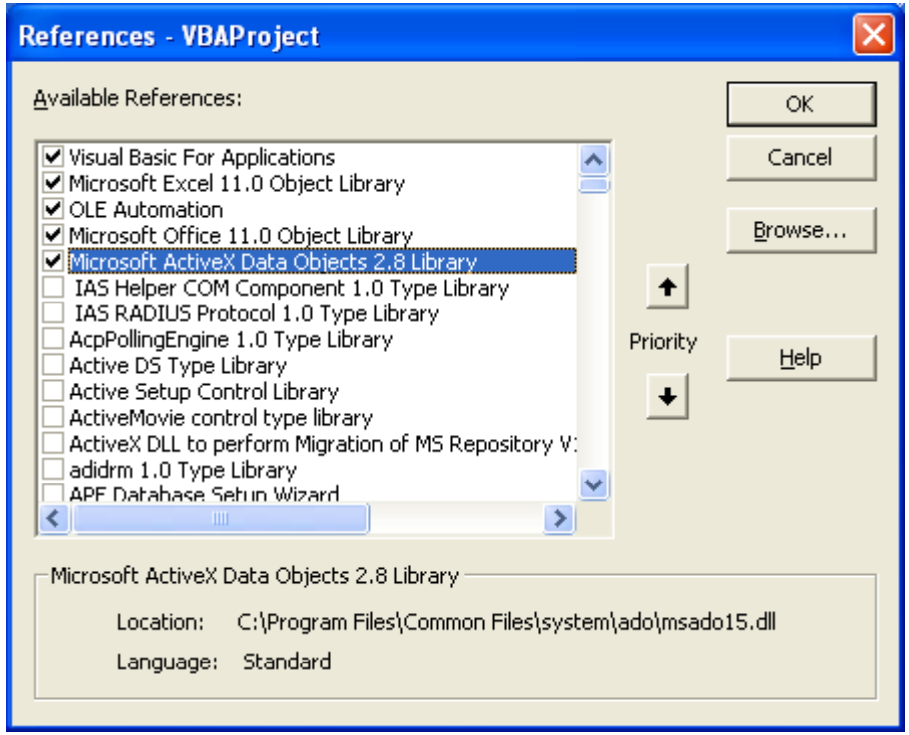
```
while ($db->FetchRow()) {
  print "NODE_NAME", ": ", $db->Data("NODE_NAME"), "\n";
  print "FILESPACE_NAME", ": ", $db->Data("FILESPACE_NAME"), "\n";
  print "Unnamed[3]", ": ", $db->Data("Unnamed[3]"), "\n";
  print "\n";
}
```

- Cleanup is easy!

```
$db->Close();
```

Excel

- You can code a macro in Excel that runs a SELECT statement, then populates the spreadsheet with the results:
  - Start Excel.
  - Select the "Tools/Macro/Visual Basic Editor" menu item.
  - Select the 'Tools/References..." menu item. Make sure that the "Microsoft ActiveX Data Objects 2.5 Library" item is checked. Note: The library version can be 2.5 or higher.



Excel (*continued*)

- In the Visual Basic editor, locate the "ThisWorkbook" object in the "Project - VBAProject" window.
- Double-click on "ThisWorkbook" to bring up the code editor for the workbook object.
- Enter your VBA code in the code editor. The sample macro provided here will require you to customize the dsn, admin ID, and password values.
- Select the "File/Close and Return to Microsoft Excel" menu item.

Excel (*continued*)

- Select the "Tools/Macro/Security" menu item. Normally security is set to high, in which case the macro will not run. You will need to set security to medium in order to be prompted to run the macro.
- From the Excel menu, select "File/Exit". When prompted to save your changes, click "Yes". Save the workbook with whatever name you want.
- Start Excel again, then open the workbook you saved in the prior step. You will be prompted to enable macros. If you enable macros, the VBA macro should run.
  - Depending on the time it takes to process the query, you will have to wait several seconds or more before the data loads.

Excel (*continued*)

- Like Perl, much of ODBC's complexity is wrapped by VBA.
- Open a connection to the TSM server:

```
conn.ConnectionString = "DSN=" & dsn & ";UID=" &
  uid & ";PWD=" & pwd
conn.Open
```

- Run the SELECT:

```
rs.ActiveConnection = conn
rs.Source = sql_query
rs.Open
```

Excel (*continued*)

- Populate the spreadsheet:

```
Do Until rs.EOF
   row = row + 1
   sheet.Cells(row, 1).Value = rs!node_name
   sheet.Cells(row, 2).Value = rs!filespace_name
   sheet.Cells(row, 3).Value = Format(rs!backup_end,
                               "yyyy/mm/dd")
   rs.MoveNext
Loop
```

- Close up shop:

```
rs.Close
conn.Close
```

VBScript

- VBScript similarly hides many of ODBC's complexities.
- Create connection and result set ("record set") objects:

```
set dbc = WScript.CreateObject("ADODB.Connection")
set recordSet = WScript.CreateObject("ADODB.Recordset")
```

- Open the connection and result set, and run the query:

```
dbc.Open dsn, uid, pwd
recordSet.Open sql, dbc
```

VBScript (*continued*)

- Display the results:

```
myArray = recordSet.GetRows()

for r = 0 to UBound(myArray, 2)
  fieldIndex = 0

  for each field in recordSet.Fields
    WScript.Echo field.Name & ": " &myArray(fieldIndex, r)
    fieldIndex = fieldIndex + 1
  next
next
```

- Close up shop:
  - `recordSet.Close()`
  - `dbc.Close()`

The End

Questions?