

WOW Builders Guide p.2

WOW Builders Guide p.2

Tabs

[Creating Tab Operations](#)

[Using Tabs](#)

[Tab Configuration](#)

[Default Tabs](#)

[Tab Fields](#)

[Tabs Per Line](#)

[Automatic Tab View](#)

[Allowing Tab Display](#)

[Empty Tab Results](#)

[Changing Tab Field Order](#)

[Hiding Search Parameters](#)

[Further Tab Customization](#)

Stored Procedures

[Calling Basic Stored Procedures](#)

[Passing Parameters to Stored Procedures](#)

[Inserting, Updating, and Deleting](#)

Running Your Applications

[Running SQL Queries and Operations](#)

[Running WOW Applications by URL](#)

[Running Applications in Application Libraries](#)

[Directly Executing Operations](#)

[Passing Parameters](#)

WOW Security Protocols

[Securing Applications](#)

[Local Users Only](#)

[Local Users Only or Operating System Profile](#)

[Operating System Profile](#)

[Personal Connection Pool](#)

[Operating System Profile Plus Operation](#)

[Steps for configuring Operating System Plus Operation Authentication](#)

[HTTP Referrer](#)

[SQL Operation](#)

[Unsecured](#)

[User List Authentication Operation](#)

[LDAP \[Minimum Version: WOW 7.0\]](#)

[Steps for configuring LDAP Authentication](#)

[1\) Create a Referrer Operation:](#)

[2\) Configure the Application:](#)

[Testing Without SSL:](#)

[Enabling SSL:](#)

[Restricting Group Access to an Application](#)

[Configure Group Search Properties](#)

[Add Group Properties](#)

[LDAP {} Property Group:](#)

[LDAP Plus Operation\[Minimum Version: WOW 7.0\]](#)

[Steps for configuring LDAP Plus Operation Authentication](#)

[User Groups \[Minimum Version: WOW 7.0\]](#)

[Securing Operations](#)

[Optional Sign On](#)

[Table Authorization](#)

[Securing Fields and Operations with User Authorization Operations](#)

[Overview](#)

[User Authorization List](#)

[User Authorization Operation](#)

[User Group Authorization List](#)

[User Group Authorization Operation](#)

[Field Level Authorization](#)

[Assigning an Authorization Operation to a Field](#)

[Assigning Authorization Operation to an SQL Operation](#)

[Deploying Applications](#)

WOW Utilities

[Users](#)

[Themes](#)

[Dynamic Themes with URL Parameter](#)

[Keyed Values](#)

Interfacing WOW with Excel

[Connecting WOW to an Excel File](#)

[Creating system DSN \(Windows Only\)](#)

[Pointing to desired Excel worksheet](#)

[Connecting WOW to created DSN \(screenshot below\)](#)

[Syntax of SQL select, update statement](#)

[SQL select statement syntax](#)

[Basic SQL Queries Using the SELECT Statement](#)

[Other Queries Using the SELECT Statement](#)

[Using a WHERE clause with the SELECT statement](#)

[SQL update statement syntax](#)

[Basic SQL Queries Using the UPDATE Command](#)

[Using a WHERE clause with the UPDATE statement](#)

[Creating Reports and Graphs with WOW and Excel](#)

[WOW Setup for Excel Web Query](#)

[Creating and Updating Excel Tables from WOW Web Data](#)

[Steps to create a web query](#)

[Setting up a New Web Query in Excel 2002](#)

[Setting up a New Web Query in Excel 2000 or earlier](#)

[Integrating WOW with Existing Excel Files](#)

[Setting WOW Operations to use Existing Excel Templates](#)

[Creating Reports from Data Imported from WOW into Excel](#)

[Restrictions](#)

Utilizing Existing RPG Applications

[Calling an RPG Program That Returns a Result Set](#)

[Add Code to Return a Result Set](#)

[Defining the Stored Procedure](#)

[Defining the WOW Operation](#)

[More Than One User Running the Operation at the Same Time](#)

[Calling an RPG Program That Returns a MODS \(Array\) in RPG Free](#)

[Add Code to Return an Array](#)

[Defining the Stored Procedure](#)

[Defining the WOW Operation](#)
[Calling an RPG Program That Returns a MODS \(Array\) in RPG IV](#)
[Add Code to Return an Array](#)
[Defining the Stored Procedure](#)
[Defining the WOW Operation](#)
[Calling an RPG Program That Returns Parameters](#)

[Advanced Development Techniques](#)

[Multi Value Reference Fields: \[PRO\]](#)
[Using the ReferenceField:](#)
[Considerations](#)

[WOW Performance](#)

[WOW's Built In High Performance Cache](#)
[Connection Properties](#)
[Controlling the Number of Records Returned](#)
[Controlling the Number of Fields Read](#)
[Optimizing SQL Performance for AS400 \(iSeries\)](#)
[Compare SQL Performance Against Non-WOW Methods](#)
[Using STRDBG](#)
[Using iSeries Navigator \(STRDBMON\)](#)
[Controlling How the Data is Accessed](#)
[Tomcat Server Performance](#)
[SQL Fragments \[PRO\]](#)

[Actions/Events \[PRO\]](#)

[Action](#)

[Creating an Action](#)
[Entry Information:](#)
[Operation Inheritance:](#)
[Action/Event Execution Information:](#)
[Action Location \(for Row action\):](#)
[Action Location \(for Row Collection action\):](#)
[Action Properties:](#)
[Action Messages:](#)
[Authorization:](#)
[Implementing the Drop Down Location:](#)

[Event](#)

[Creating an Event](#)
[Entry Information:](#)
[Event Scope:](#)
[Table That Triggers Event:](#)
[Action/Event Execution Information:](#)

[Example Uses:](#)

[Adding Action to save a copy of a single Row or Table Results:](#)
[Adding Event to automatically save changes to WOW Operations:](#)
[Adding Action to Run 2nd Operation That Remembers Rows Selected](#)
[Create 1st Operation:](#)
[Set Key Fields to Global:](#)
[Create 2nd Operation:](#)
[Create Action on 1st operation to Run 2nd Operation:](#)

[Troubleshooting and Debugging WOW](#)

[My iSeries DB2 files are locked by WOW which is affecting my saves and other programs!](#)
[WOW is unable to connect to the IBM iSeries Metadata Server because of restricted ports](#)

[Change User Name and Password of WOW for new metadata system](#)

[Configuring Logging \(Log4j\)](#)

[Log4J Configurations](#)

[WOW Log File \(output.log\)](#)

[Running a SQL Statement with Period in the name of a Database Table](#)

[When Running WOW off of a Linux or Unix machine and with MySQL, some operations don't work](#)

[When creating a row, the Current Date -CURRENT returns the wrong date](#)

[Changing Database Tables and Views:](#)

[WOW Administration and Support](#)

[Backing Up WOW Metadata](#)

[Backing Up WOW Metadata from AS/400](#)

[Backing Up WOW Application Code](#)

[Backing Up WOW Metadata Using DB2 on Windows](#)

[Disaster Recovery](#)

[Version Control](#)

[Web Resources \(Java, JSPs, HTML, etc\)](#)

[Metadata](#)

[Web Application Server Information](#)

[Getting Support for WOW](#)

[WOW Copyright Information](#)

Tabs

One of the options provided by WOW is to display the results of an operation in a tabbed layout. This layout is typically used where you have one primary operation, and multiple secondary operations related to the primary operation. For example, your primary operation might be to look up a customer, and your secondary operations might allow you to view that customer's history, current charges, and account options. While you are running the secondary operations, you still want to be able to view the results of your primary operation (e.g. the current customer resulting from your search).

Name LIKE

Search

Name Justin

Balance 4900

History Settings Charges

Description	Date	Amount
Breezeway	09/07/2004	140
Pavers	09/09/2004	400
Contract Work	09/12/2004	1400
Surveying	09/12/2004	300
10 Gallon Pickle Tub	09/15/2004	150
Ankle Protection	09/16/2004	90
Fire Retardent	09/18/2004	239

Creating Tab Operations

This section will describe how to configure your operations to use a tabbed interface, as shown above. The first step is to determine what your primary operation (also known as the "tab parent operation") will be. In our example, this is the query which looks up a customer based on a name search. The operation type of your tab parent operation should be set to "Tabbed Operation":

Basic

Label

Customers

Title

Operation Type*

Tabbed

Description

Operation Code

SELECT *FROM JETEMP.CUSTOMER WHERE NAME like ?

Once the tab parent has been defined, the next step is to create your secondary operations (also known as the "tab data operations"). Creating a tab data operation should be done exactly like creating an associated operation and associating it with the results of the tab parent operation. That process is summarized below:

- Each tab data operation should be an association operation type (such as 1-to-Many Association or 1-to-1 Association).

- The tab data operations can use the double question mark notation to refer to fields in the result of the tab parent operation.

Here is the tab data operation:

The screenshot shows a window titled "Basic" with several fields for configuring a tab data operation:

- Label:** Customers - History
- Operation Type:** Association 1-MANY (indicated by a dropdown arrow)
- Title:** (empty field)
- Description:** (empty field)
- Operation Code:** A large text area containing the SQL query: `SELECT *FROM JETEMP.CHIST where cid = ??ID`

- Each tab data operation should be listed as the associated operation on a field descriptor of a result field of the tab parent operation.

Basic Settings	
Field Name *	C_HISTORY
External Name:	History
Required:	<input type="checkbox"/>
Required On Search:	<input type="checkbox"/>
Default Value:	<input checked="" type="radio"/> -- None -- <input type="radio"/>
Auto Update Value:	

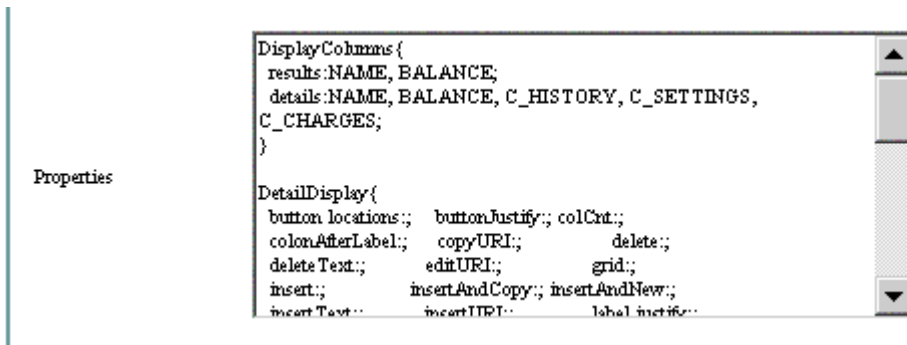
Display Settings	
Field Set:	<input checked="" type="radio"/> <input type="radio"/>
Display Order:	20
Display Rule *	Always
Display Component *	System
Help Text:	
Style Class:	
Display Width:	
Display Height:	

Possible Value Settings	
Possible Values Key:	-- None --
Possible Values Operation:	-- None --
Possible Value Class:	<input checked="" type="radio"/> Enter Class Name: <input type="radio"/> Select Existing Class: "DISTINCT*"

Advanced Settings	
Field Class:	<input checked="" type="radio"/> Enter Class Name: <input type="radio"/> Select Existing: Address 1
Field Descriptor Type *	Derived
Formatter Class:	<input checked="" type="radio"/> -- None -- <input type="radio"/>
Concurrency *	Concurrent Updates and Deletes Allowed
Getter Method:	
Remarks:	
Setter Method:	
Association Operation:	Customers - History
XML Tag:	
Notify Status Change:	No

There must be one such field (with an associated tab data operation) for every tab. The external name of the field will be the name of the tab as it is displayed to the user. It does not matter whether the field is a derived field or a physical field; what's important is that the field be selected as part of the tab parent operation, and that the field is listed in the

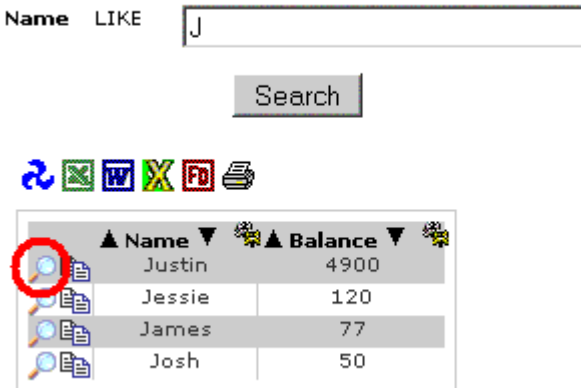
detail display columns (if the detail display columns are left blank then all fields are displayed by default, so not specifying detail display columns will work as well).



So in summary, the operation type of the tab parent operation must be "Tabbed Operation," and the tab data operations are each associated operations which are associated with a particular field of the tab parent.

Using Tabs

When using tabs in WOW, before the tab layout is displayed there must be a single tab parent row. What this means is that, if the tab parent operation is run and results in multiple rows, the tab layout is not displayed until the user has narrowed the results down to a single row. This can be done either by altering the search criteria so that a single row is returned, or by clicking on the detail view of one of the result rows.



Tab Configuration

The Tabs property group allows you to configure tabs in several ways:

Default Tabs

Once a single tab parent row has been selected, WOW will display all the tabs associated with that tab parent; however it will not automatically select a tab for the user. If you want WOW to automatically select one of the tabs by default, you can specify this in the Tabs properties group:

```
Tabs {  
  defaultTab: C_HISTORY;  
}
```

The above property group instructs WOW to initially display the C_HISTORY tab. In our example, this is the tab labeled "History." C_HISTORY is the internal name of the derived field associated with the operation for retrieving the customer's history.

The *defaultTab* property is specified in the tabbed operation (not any of the tab data operations).

Tab Fields

By default, any fields in the details of the tabbed operation which have associated operations are rendered as tabs. However, in some cases you may have fields with associated operations which are not tabs. You can use the *tabFieldsExclude* and *tabFields* properties to control which fields are displayed as tabs:

```
Tabs {
```

```

defaultTab: C_HISTORY;
tabFields: FIELD1, FIELD2;
}

```

In the above property group, fields FIELD1 and FIELD2 will be the only fields rendered as tabs, no matter how many other fields have association operations.

```

Tabs {
  defaultTab: C_HISTORY;
  tabFieldsExclude: FIELD_A, FIELD_B;
}

```

In the above property group, fields FIELD_A and FIELD_B will not be displayed as tabs, even if they have associated operations. The *tabFieldsExclude* and *tabFields* properties are always specified in the tabbed operation (not the tab data operations).

Tabs Per Line

If an operation has a large number of tabs, WOW may need to use multiple lines on the screen to display all of the tabs:

Call Center	Account	Credit Info	Load Info	History	Comments	Invoice
Security Lights	Location	Meter	Auxiliary Meter	Transformer	Service Orders	



▲ Load Month ▼	▲ Metered KWH ▼	▲ Monthly Charges ▼	▲ Energy Charges ▼	▲ KW Billed ▼
01/01/2004	3250	\$2,386.00	\$2,385.07	
11/01/2003	2980	\$604.00	\$603.18	
10/01/2003	31000	\$2,058.00	\$2,057.20	
06/01/2002	136654	\$17,467.38	\$17,467.38	
05/01/2002	623	\$155.90	\$60.90	
04/01/2002	615	\$114.00	\$60.10	
03/01/2002	335	\$114.00	\$47.25	
06/01/2001	944	\$25.15	\$25.15	
02/01/2001	1403	\$133.28	\$133.28	
12/01/2000	976	\$100.85	\$100.85	

By default, WOW will display a maximum of 10 tabs on the same line. You can change the maximum number of tabs on a single line with the *maxTabsPerLine* property:

```

Tabs {
  maxTabsPerLine: 7;
}

```

The *maxTabsPerLine* property is specified in the tabbed operation, not the tab data operations.

Automatic Tab View

As discussed in the previous section, when you run a tabbed operation you will only see the tabbed layout when viewing a single row from the results. Normally, when your query returns a single row WOW will take you directly to the tabbed view for that row instead of

displaying a list view containing a single row. However, this behavior can be overridden setting the *automaticTabView* property to false:

```
Tabs {  
  defaultTab: C_HISTORY;  
  maxTabsPerLine: 7;  
  automaticTabView: false;  
}
```

In this case, after running a tabbed operation, the results will always be displayed in a list view, even if there is only a single row to display. You can still click on the View Details link (the magnifying glass) for any row to view that row in a tabbed view.

The *automaticTabView* property is specified in the tabbed operation, not the tab data operations.

Allowing Tab Display

Sometimes, you may want the results of an operation to be displayed inside of a tab, but the details of that operation to be displayed outside of a tab. Using the *allowInTab* property you can specify exactly when an operation is allowed to be displayed in a tab:

```
Tabs {  
  allowInTab: results;  
}
```


There are 4 settings for this property:

- **always** – Displays both the results and details in a tab.
- **details** – Only the details (not the results) should be displayed in a tab.
- **never** – Neither the details nor results should be displayed in a tab.
- **results** – Only the results (not the details) should be displayed in a tab.

Unlike all other properties in the Tabs property group, the *allowInTab* property must be specified in the tab data operation, and not the tabbed operation.

Empty Tab Results



When a tabbed operation is run and there are no results, WOW can either display an empty table of the results, or can leave the results section of the screen blank.

 Messages

There is no information available for the month/year you have selected

Processing Month =

Year =

AccountJournalTB SummaryTB Detail Company Processing Month Processing Year
(no records exist)

Here is a tabbed search example that hides empty results:



The *hideWhenEmpty* property controls which method WOW uses to display empty tab results. When *hideWhenEmpty* is true, WOW will not display any results; if it is false then an empty Row Collection will be displayed. If the *hideWhenEmpty* property is not specified, it defaults to the same value as the *automaticTabView* property.

You can also define a message to display to the user when there are no results using the *emptyMessage* property.

```
Tabs {  
  hideWhenEmpty: true;  
  emptyMessage: There is no information available for the month/year that you have  
  selected;  
}
```

Changing Tab Field Order

On each field descriptor there is a display order. Fields with the lowest display order will get listed first.

The exact behavior is:

1. Find the field with the lowest display order.
2. Put that field and any other fields within the same field set together then get the next lowest field set and display those.

NOTE: In the FD Manager, there is an operation on the left called "display properties" that will allow you to set the display order and field set from an updateable table so you do not have to go into each FD one at a time.

Hiding Search Parameters

There are times when you may want the search parameters for the Tab Parent operation to only be displayed long enough to perform the search successfully. Using the *alwaysShowSearch* property, you can specify whether the search parameters should continue to be displayed (default behavior), or if they should be removed to provide more space for displaying the results from the Tab Parent operation:

```
Tabs {  
  alwaysShowSearch: false;  
}
```

There are 2 settings for this property:

- **true** – Always show the search parameters in the Tab Parent operation. This is the default value if this property is omitted.
- **false** – Once the Tab Parent operation successfully shows a record (row), hide the search parameters.

The *alwaysShowSearch* property is specified in the tabbed operation, not the tab data operation.

Further Tab Customization

[PRO] This section describes advanced techniques for customizing the layout and appearance of tabs, and is recommended for JSP programmers only. WOW uses 5 JSP's to display a typical tabbed screen:

Name LIKE

Search







Name Justin

Balance 4900

History

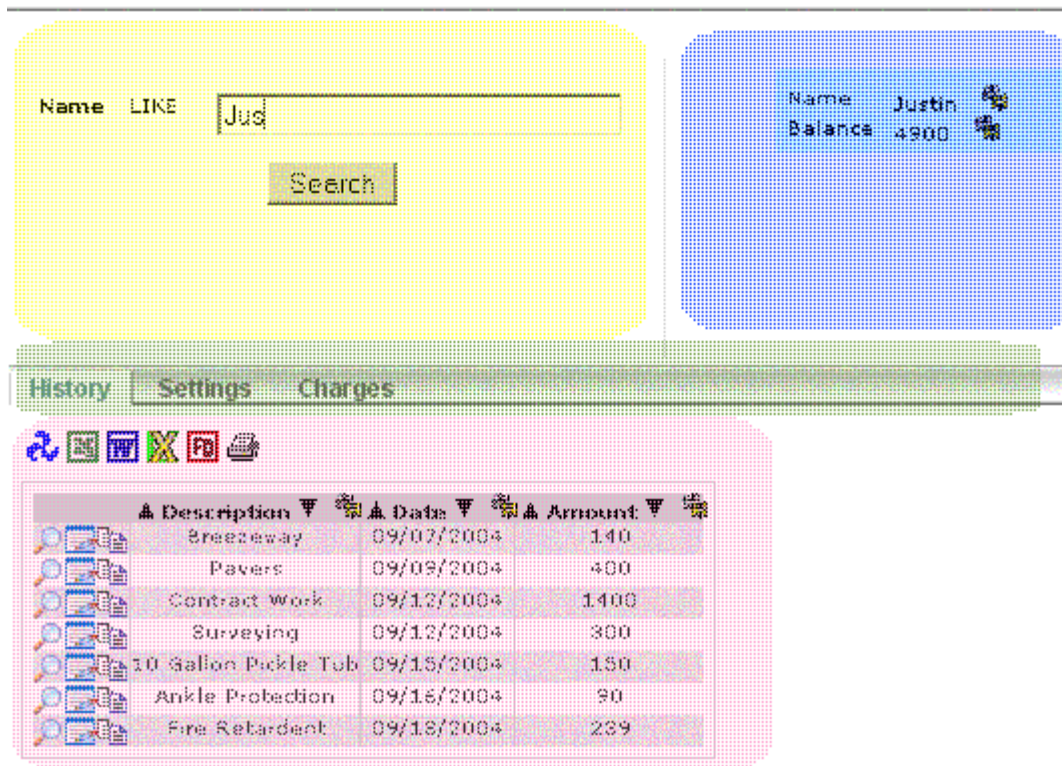
Settings

Charges



▲ Description ▼	▲ Date ▼	▲ Amount ▼
Breezeway	09/07/2004	140
Pavers	09/09/2004	400
Contract Work	09/12/2004	1400
Surveying	09/12/2004	300
10 Gallon Pickle Tub	09/15/2004	150
Ankle Protection	09/16/2004	90
Fire Retardent	09/18/2004	239

Here is the same tab screen, overlaid with different colors to distinguish the different JSP's:



By replacing one or more of the WOW default JSP's with your own custom JSP's you can completely control the manner in which tabs are displayed to the user. Each of the 5 different JSP's are discussed below:

- **Parameters JSP (yellow overlay)** – This JSP displays the parameters of the tabbed operation to the user. If you want to use a JSP other than the WOW default, specify your JSP in the "Parameters JSP" field of the tabbed operation.
- **Results JSP (pink overlay)** – This JSP displays the results of the tab data operation. If you want to use a JSP other than the WOW default, specify your JSP in the "JSP File" field of the tab data operation.
- **Tab Parent JSP (blue overlay)** – This JSP displays the contents of the tab parent row. If you want to use a JSP other than the WOW default, specify your JSP in the Tabs property group of the tabbed operation.
- **Tab Headings JSP (green overlay)** – This JSP displays the tab bar and labels. If you want to use a JSP other than the WOW default, specify your JSP in the Tabs property group of the tabbed operation: Tabs {

```
defaultTab: C_HISTORY;
tabParentJSP: /jsp/myTabParent.jsp;
tabHeadingsJSP: /jsp/myTabHeadings.jsp;}
```

- **Tab Layout JSP (entire screen)** – This JSP is responsible for positioning the four other JSP's. Those four JSP's are all contained within the tab layout JSP. If you want to use a JSP other than the WOW default, specify your tab layout JSP in the "Details JSP" field in the tabbed operation.

Stored Procedures

A stored procedure consists of one or more SQL statements that have been precompiled on a database system. All of the SQL examples in the above chapters are dynamic SQL statements – no compilation takes place until they are run. For this reason, stored procedures tend to perform better than dynamic SQL. This chapter will discuss how to call stored procedures and display their results using WOW. Creating stored procedures is not covered in this guide. For more information on creating stored procedures check your database documentation.

Calling Basic Stored Procedures

The SQL for calling a stored procedure named MYSP located in the PLANETJTMP library is:

```
CALL PLANETJTMP.MYSP()
```

To call this stored procedure and display the results it returns in WOW, simply place type this SQL in the code section of an operation.

Operation Code:











```
call planetjtmp.mysp()
```

NOTE: When calling MS SQL SERVER stored procedures, you may need to place the stored procedure call within {}. For example: {call mySqlServer.myStoredProc() }. Consult the database documentation for details for each specific database being used.

When you run the operation, the results are displayed like a normal select statement:

All Customers (Basic SP)



	ID 	NAME 	BALANCE 
	8	Paul Holm	2361
	17	Nate Levis	4359
	77	Justin Epstein	5359
	10	John	413
	11	Sally	345
	12	Bob	782
	29	Maggy Sue	344

By default, when the results from a stored procedure call are displayed, they do not make use of any field descriptors you have created. This is because the names of the actual tables from which the results are read are not present in the SQL code that is entered in WOW;

the table names are contained inside the stored procedure code, which WOW does not have access to.

In order to have your results use the field descriptors you have created, you must add a StoredProcedure property group to the properties of the operation which calls the stored procedure. In the *tables* property of the StoredProcedure property group, you should list the names of the tables which are used in the query.

NOTE: This property group goes in the Properties section of the operation, NOT the operation code!

























For example:

```
StoredProcedure {  
  tables; planetj.customer, planetj.balancedta;  
}
```

Once you do that, your results will then use the appropriate field descriptors when they are displayed:

All Customers (Basic SP)



	▲ ID ▼ 	▲ Name ▼ 	▲ Balance ▼ 
  	8	Paul Holm	2361
  	17	Nate Levis	4359
  	77	Justin Epstein	5359
  	10	John	413
  	11	Sally	345
  	12	Bob	782
  	29	Maggy Sue	344

Passing Parameters to Stored Procedures

Many stored procedures have input parameters whose values are used at runtime to execute a query. To call a stored procedure and prompt the user at runtime to supply the values for its parameters, you must identify which field descriptors to use when generating the input prompts. For example, in the screenshot below the stored procedure is being passed two parameters, the first will use field descriptor 1135 (this is the id of the desired FD) to display the prompt to the user and the second will use field descriptor 1139.

Operation Code: `call planetjtmp.custbalbtw(?1135,?1139)`

NOTE: You cannot use a single question mark as a parameter in a stored procedure as you can with other SQL statements.

Inserting, Updating, and Deleting

Some stored procedures do not return rows from the database for display; instead they alter one or more database tables. To identify these types of stored procedures, you should specify a value of false for the *rowCollection* property of the StoredProcedure property group.

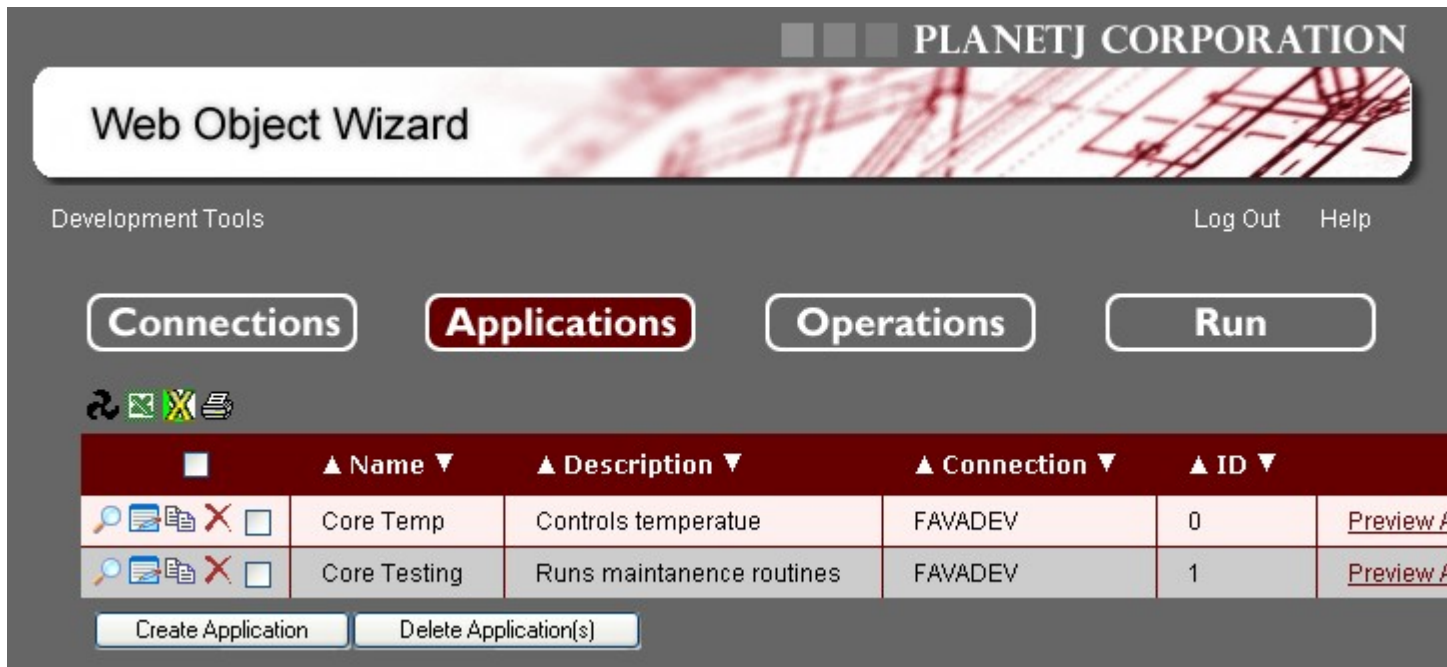
```
StoredProcedure {  
  tables: planetj.customer, planetj.balance;  
  rowCollection: false;  
}
```

This lets WOW know that it should not attempt to display a collection of rows read from the database as a result of calling the stored procedure.

Running Your Applications

Running SQL Queries and Operations

After creating the different operations as shown in the above chapters it's now time to run the application. This section will assume that you have entered your user ID and password and successfully logged on to WOW. The screen shot below is what the screen should look like after a successful login. (You may have fewer applications, and they will be named differently than the ones shown in the screen shot.)



The screenshot displays the PlanetJ Corporation Web Object Wizard interface. At the top, the title bar reads "PLANETJ CORPORATION". Below it, a banner says "Web Object Wizard". The interface includes a navigation bar with four buttons: "Connections", "Applications" (which is highlighted in red), "Operations", and "Run". Below the navigation bar, there are four icons: a refresh icon, a green checkmark, a red X, and a document icon. A table lists the applications, with columns for Name, Description, Connection, and ID. The table contains two rows: "Core Temp" and "Core Testing". Each row has a "Preview Application" link. Below the table, there are two buttons: "Create Application" and "Delete Application(s)".

	▲ Name ▼	▲ Description ▼	▲ Connection ▼	▲ ID ▼	
	Core Temp	Controls temperatue	FAVADEV	0	Preview Application
	Core Testing	Runs maintenance routines	FAVADEV	1	Preview Application

After you have successfully logged on with your user ID and password, select the application you would like to run. Then click on the Step 4 button 'Run!' on the left side of the screen. Alternatively, you can click the Preview Application link next to the application you wish to run. In either case, a new browser window will open up and execute your application. By default, your operations can be found in their respective display groups under the drop down menu bar near the top of the window.



Customer Service

SQL Stored Procedure Exe

- All Customers
- Customer Balances Between
- All Customers (User)
- New Customer
- Delete By Balance
- Update By ID

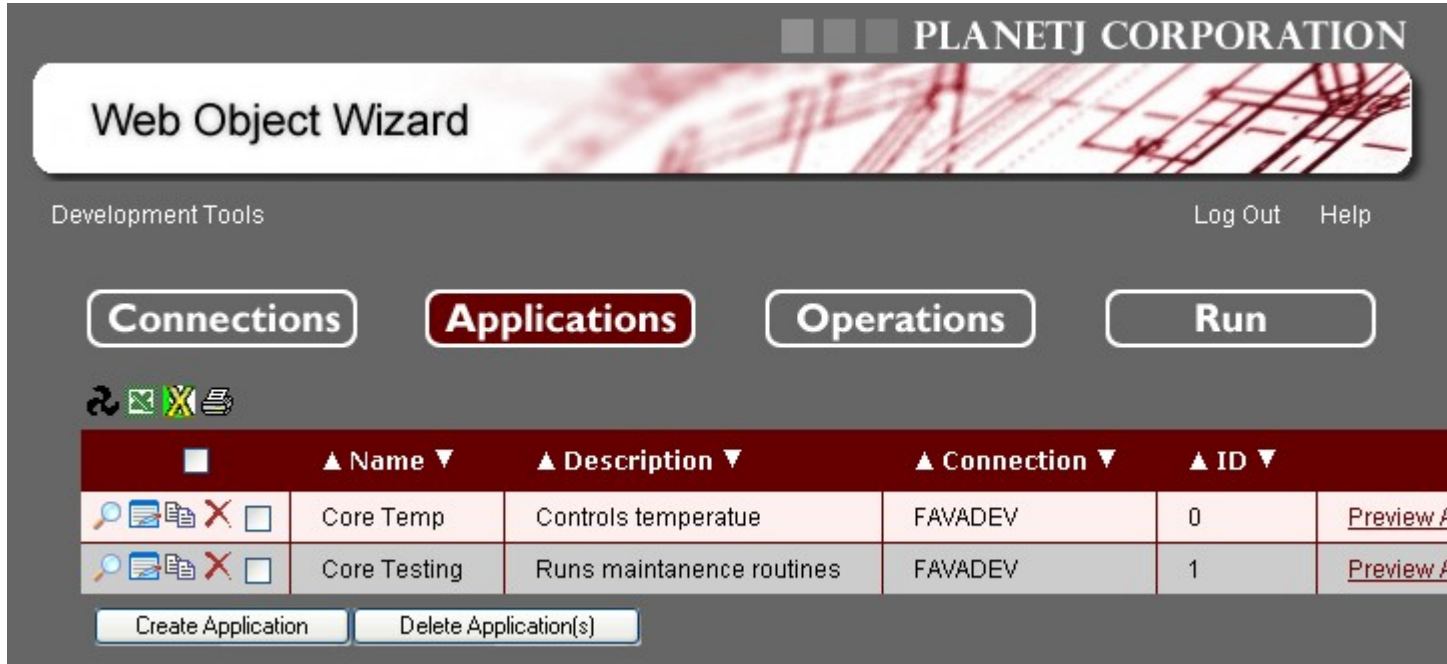
Each one of these links corresponds to a different operation. Your application may have more or fewer operations listed. Notice the different Display Groups: "SQL", "Stored Procedures", "Exe"; these different Display Groups were created by changing the "Display Group" from "default" to a more descriptive name.

Clicking on a link will execute that link's operation. Operations which query the database will display their results directly on the screen. If your operations use runtime prompting, the user will be prompted for the required values when the operation's link is clicked. Below is a screen shot of what a runtime prompt may look like:

SALARY >=

Running WOW Applications by URL

To run a WOW application directly, without going through the WOW builder you must first know the ID of the application you wish to run. The application ID is located in the second-to-last column on the "Applications" screen. For example, the applications in the screen shot below have IDs of 0 and 1:



The screenshot shows the 'Web Object Wizard' interface for PlanetJ Corporation. It has a navigation bar with 'Connections', 'Applications' (selected), 'Operations', and 'Run'. Below the navigation bar is a table of applications. The table has columns for Name, Description, Connection, and ID. Two applications are listed: 'Core Temp' with ID 0 and 'Core Testing' with ID 1. Both are connected to 'FAVADEV'. There are buttons for 'Create Application' and 'Delete Application(s)' at the bottom of the table.

	▲ Name ▼	▲ Description ▼	▲ Connection ▼	▲ ID ▼	
	Core Temp	Controls temperatue	FAVADEV	0	Preview A
	Core Testing	Runs maintenance routines	FAVADEV	1	Preview A

Buttons: Create Application, Delete Application(s)

To run an individual application (the 'Core Temp' application in this example) use the URL shown below replacing 'planetjavainc' with the URL of WOW on your machine, and 0 with the ID of the application you want to run:

<http://www.planetjavainc.com/wow/runApp?id=269>

This is the URL of WOW followed by "runApp?id=0". Make sure to capitalize the word App but keep the word run in lowercase. The screen shot below shows the web browser after the above URL has been entered in.



The URL does not have to be directly typed into a browser; you could also create a link to it from another web page that users would click on to run the application. If your application is in an application library you will need to append that onto the URL as well, as described below:

Running Applications in Application Libraries

If you want to run an application which is in an application library other than the default one, you must also specify the application library on the URL. You first take the application's URL as defined above, and append the text:

`&_pj_lib=<APPLIB>`

where <APPLIB> is the name of the application library containing the application. For example, if we wanted to run the application with ID 269 in the application library TEMPLIB, the URL would be:

```
http://www.planetjavainc.com/wow/runApp?id=269&_pj_lib=TEMPLIB
```

Of course, you will have to replace planetjavainc with the URL of your web server. Note that the above URL does not contain any spaces – just underscores. Keep in mind that it is possible for two different applications to have the same application ID if they are in different application libraries.

Directly Executing Operations

The normal way to execute an operation created with WOW is through an application. The user clicks a link, taking them to the main page of an application and the available operations for that application are then available from a left-hand TOC or a drop-down menu (default). However, it is also possible to directly execute an operation without going to any intermediate pages. To do this, you must first know the ID of the operation you are going to execute. You can find an operation's ID by choosing to edit that operation, and then scrolling to the 'Internal' section at the bottom of the page:



The screenshot shows a web interface with a dark red header bar containing a lock icon and the word "Internal". Below this is a grey form area. It contains three fields: "Operation ID*" with a text input containing the number "3", "Application" with a text input containing "Core Temp", and a dropdown menu labeled "Core Temp" with a downward arrow.

Once you know an operation's ID, you can directly execute that operation by going to a URL similar to this one:

```
http://www.planetjavainc.com/wow/runApp?opid=3
```

The above URLs would directly execute operation 3 on the PlanetJ website. You should replace "3" with the ID of the operation you want to directly execute, and "www.planetjavainc.com" with the URL of your server.

NOTE: If you choose to directly execute an operation that is part of a secured application you will have to sign on to that application before the operation is executed, unless you have already signed onto that application.

Passing Parameters

You can also pass parameters to operations which are directly executed in one of two ways: by name or by index. For example, say the operation with an ID of 10 has this code: `SELECT * FROM PJDATA.CUSTOMER WHERE NAME = ? and BALANCE = ?`. To directly execute this operation looking for customers whose name is John and whose balance is 400, you could specify the parameters by name on the URL like this:

```
http://www.planetjavainc.com/runApp?opid=10&BALANCE=400&NAME=John
```

You can simply list out the parameters using the "name=value" format, separated by ampersands.

Alternatively, you could specify the parameters by index like this:

```
http://www.planetjavainc.com/runApp?opid=10&_parm1=John&_parm2=400
```

NOTE: There are no spaces in the above URL, just underscores.

When specifying parameters by index you list them on the URL using the "_parm1=value1" format, separated by ampersands (the first parameter has an index of 1). Specifying parameters by index can be useful when two parameters in your operation have the same name (e.g. `SELECT * FROM PJDATA.CUSTOMER WHERE BALANCE > ? AND BALANCE < ?`). If you specify parameters by name, there is no way to give these two parameters (both named BALANCE) different values.

WOW Security Protocols

Securing Applications

WOW contains multiple ways of securing applications, shown in the drop down below. When creating or editing an application, you can choose which type of security it uses. All of the security options are described below.

The screenshot shows a configuration window for an application. The 'Basic' tab is active. The 'Name' field contains 'EC Tests'. The 'Connection' dropdown is set to 'EC'. The 'Initial Operation' dropdown is set to '-- None --'. The 'Sign On Type' dropdown is open, showing a list of options: 'Local Users Only' (highlighted), 'Local Users Only or Operating System Profile', 'Operating System Profile', 'Personal Connection Pool', 'SQL Operation', 'Unsecured Sign On', and 'User List Sign On'. The 'Sign On Operation' field is empty.

Local Users Only

When a user attempts to use an application, WOW examines the IP address of that user. If the IP indicates that the user is on the local network (i.e. not connecting via the Internet) the user is allowed to use the application. Otherwise the user is locked out and cannot use the application.

Local Users Only or Operating System Profile

This sign on option allows local users (as described above) to access the application without entering a user ID or password. Non-local users must enter a user ID and password recognized by the underlying operating system (or database system) before using the application.

Operating System Profile

This type of sign on can be useful to validate a user against an operating system (or more specifically, a database system). This feature prompts the user for a user ID and password that it validates against the database. Users with a valid database sign on will be allowed to use your application – those who cannot sign onto the database will be locked out of the application as well.

Many organizations choose to use Operating System Profile to secure their applications.

This presents the benefit that you do not have to create a whole new table to manage users and their credentials; you can simply utilize your database's existing user profiles.

However, in some cases, you may want to also secure specific operations or fields within your application. In this case, you can use Operating System Profile to secure the application and then SQL Authorization operations to secure individual operations and fields. See chapter 19.5 for more information on using SQL Authorization operations.

NOTE: When WOW performs Operating System authentication, it stores the Userid of a successful sign-on in the user sign-on row in the session. This userid can now be accessed from any WOW SQL statements using the following row parameter: ???USERID

For example:

```
SELECT * FROM library.table WHERE user = ???USERID
```

This statement would return all records in the specified table that have a "user" value that matches the "USERID" used to sign into the application.

Personal Connection Pool

The Personal Connection Pool sign on validates a user ID and password against the database, much like the Operating System Profile sign on. However, when an application uses the Personal Connection Pool sign on method, all database accesses by that application will be tied to the profile of whichever user has signed onto the application and requested that database access. All other sign on methods use a shared pool of database connections when accessing the database – this can significantly improve performance but means that the database cannot determine which particular user is accessing it, only which application is doing the access. This sign on type should be selected when the database needs to know which user is accessing it.

Operating System Profile Plus Operation

The Operating System Profile Plus Operation sign on is almost identical to an Operating System Profile sign on, with the exception that after the user is authenticated against the operating system or database, an SQL operation is run to retrieve a property Row.

Steps for configuring Operating System Plus Operation Authentication

1) Within the application, create an operation of type **Authentication**. Set the Operation Code to the SQL that will be used to retrieve the desired properties Row. Use a built-in parameter such as ???USERID (the user ID used to sign onto the application) to filter the results returned from the operation so they correspond with the user signing on:

Basic

Label*	Authentication for Properties Row	?	Title
Operation Type*	Authentication	?	Description
Operation Code	SELECT * FROM PJDATA.EMPLOYEE where USERID = ???USERID		

2) Edit the application and change the following:

- Sign On Type = **Operating System Plus Operation Sign On**
- Sign On Operation = the operation from step 1.

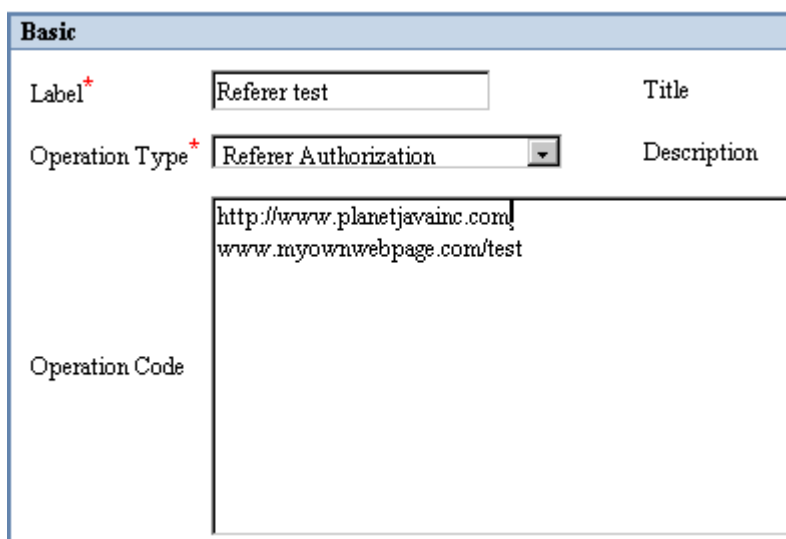


A screenshot of a configuration window with a dark grey background and a red header bar. It contains three rows of settings, each with a label, a text input or dropdown, and a help icon (a question mark in a circle). The first row is 'Description' with an empty text input. The second row is 'Sign On Type' with a dropdown menu showing 'Operating System Plus Operation Sign On'. The third row is 'Sign On Operation' with a dropdown menu showing 'Authentication for Properties Row'.

HTTP Referrer

You can use the HTTP Referrer sign on to allow only users coming from a certain web page into your application. This can be useful when combining an existing web site with WOW – when the user arrives at a WOW application from a page on the existing web application you may not want to force the user to sign on again. It is possible, though unlikely, that someone with an understanding of the HTTP protocol could trick WOW into thinking they came from a particular page.

To use HTTP Referrer sign on, you must first create an operation of type “Referrer Authorization”. In the operation code list the comma separated URL prefixes of permitted referrers. For example, if your operation looked like this:



A screenshot of the 'Basic' tab in a configuration window. It has a light blue header bar with the word 'Basic'. Below it, there are four fields: 'Label*' with the value 'Referer test', 'Title' (empty), 'Operation Type*' with a dropdown menu showing 'Referrer Authorization', and 'Description' (empty). Below these fields is a large text area for 'Operation Code' containing the text 'http://www.planetjavainc.com, www.myownwebpage.com/test'.

Users coming from the following pages would be granted access to your application:

<http://www.planetjavainc.com>

<http://www.planetjavainc.com/wow>

<http://www.myownwebpage.com/test>

Users coming from <http://www.myownwebpage.com> would not be given access.

SQL Operation

This type of authentication allows for a very versatile way to secure the sign-on of an application. SQL Operation sign on is the most powerful type of application sign on. It allows for a couple additional features to be used; Securing Operations (as described later in this chapter) and using user properties as parameters in other operations.

SQL Operation sign on requires that the given application contains an authentication operation. An authentication operation is a regular SQL Operation that has a different type (authentication) to denote that it should be used for sign on authentication. This allows any number of values to be retrieved as user properties, as well as allowing authentication against any data in the database. See the two examples below for more details.

SQL Operation sign on example 1:

To properly setup an Application with a *Sign On Type* of SQL Operation, the user must specify this setting while editing an Application. Once this is set, the user creates the SQL Operation that will be used to authenticate the Sign On. To do this, create an SQL Operation for the Application and set the required property of *Operation Type* to **Authentication**.

Basic			
Label:	User Authentication	Title:	
Operation Type:	Authentication	Description:	Authenticates the User
Operation Code:	SELECT * FROM planetj.user where username = ? and urpassword = ?		
Instructions:		Application:	View Application

The user must specify the SQL Query that contains the field or fields to be authenticated against. Save this information with the Operation and run the Application. A screen should appear that presents TextFields for the fields specified in the query and a "Login" button.

Sign On

USERNAME =

URPASSWORD =

Login

The order of the fields presented on the screen is determined by the order of the fields in the SQL Operation. In order to provide nicer values for the labels on the TextFields, be sure to specify the external name in the Field Data for that file.

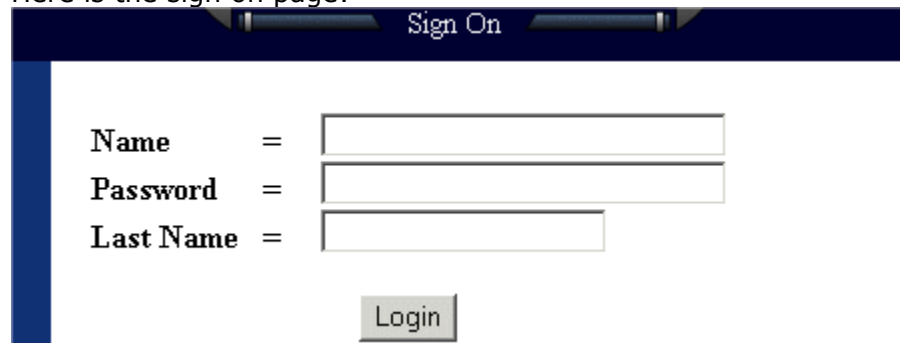
When the login button is pressed, the authentication operation is run using the values the user supplied. If the operation returns any rows from the database, the sign on is successful. If no rows are returned, the sign on fails.

SQL Operation sign on example 2:

This example is the same as example 1, but the SQL is different to give a more clear explanation of how user properties and sign on fields are generated. In this example, we will also authentic against the same table as the first example, but we will change our SQL. This time, we will authenticate against the user name, user password, and last name columns of the table.

Basic			
Label*:	Application Sign On	Title:	
Operation Type*:	Authentication	Description:	Authenticates the User
Operation Code:	<pre>select urfirstname, urlastname, urname, urseclevel from planetj.user where urname = ? and urpassword = ? and urlastname = ?</pre>		

Here is the sign on page:



You'll notice that this example selects certain fields from the user table. These fields may then be used by other operations as parameters. For example, the query

```
SELECT * FROM PLANETJ.CUST_DATA WHERE F_NAME = ???URFIRSTNAME
```

would select from the CUST_DATA table where the F_NAME field contains the same value as the URFIRSTNAME field in the current user's row of the USER table. (See section 10.1 for more information on user parameters.) In addition, the urseclevel field selected has a usage id (see Appendix A) set to denote that it's a security level field. This means that the application could contain operations that are secured by level and each user signing in would have a security level (see the next section for more information on security levels).

Multiple Sign On Operations

Most applications which use SQL for signing on do so in a single operation. However, it is possible to use multiple operations for a single sign on. For example, on the first screen you may want to have the user enter in his or her ID and check that against the list of authorized users before presenting the user with a second page for entering in a password.

To configure your application to use two sign on operations (as in the above example) follow these three steps:

1. Define both of your operations normally. Be sure to set their operation type to Authentication. This is a screenshot of the first operation (for checking the user's name):

Basic			
Label *	ID Login (Primary)	Title	
Operation Type *	Authentication	Description	Prompts for
<pre>SELECT *FROM PRMILL.CSPCM where USERID = ?</pre>			

This is the second sign on operation (for verifying a user's password):

Basic			
Label *	Pin Login (Secondary)	Title	
Operation Type *	Authentication	Description	Prompts for the
<pre>SELECT *FROM PRMILL.CSPCM where PASSWD = ? and USERID = ??USERID</pre>			

Notice that this operation refers to data returned by the first operation with the double question mark syntax (??USERID). This is necessary because we need to check not only that the password is valid, but that it is valid for the particular user ID obtained from the first operation

2. In the second sign on operation, the first sign on operation must be identified as the parent operation. This lets WOW know that after running the first operation, the sign on isn't complete until the second operation is run as well.

Advanced			
Connection Alias	-- None --	Row Count	50
Row Coll. Class		Row Class	
Parameters JSP		Caching Level *	C3
JSP File	<input checked="" type="radio"/> -- None --	Details JSP	
	<input type="radio"/>		
Parent Operation	ID Login (Primary)	Depends On	
Usage Id			

3. Finally, the application's sign on type should be set to SQL Operation, and the primary (first) sign on operation specified. Now the application is set up to use multiple sign on operations.

Basic			
Name *	<input type="text" value="Demo"/>	Description	<input type="text" value="Demo Application"/>
Connection *	<input type="text" value="PEPPERMILL"/>	Sign On Type *	<input type="text" value="SQL Operation"/>
Initial Operation	<input type="text" value="Welcome"/>	Sign On Operation	<input type="text" value="ID Login (Primary)"/>

Unsecured

This type of sign on is useful for applications showing public information. It is just as it says; unsecured, meaning the application does not require any type of authentication in order to view the application and its operations.

User List Authentication Operation

An application can be secured by creating a User List Authentication Operation which is defined by a comma separated list of user names and passwords. When the user logs on to an application with list based security they are prompted for their user name and password. This is a useful option when the WOW developer wants to quickly implement application level security for a small group of users without having set up table or user profile based security.

 **User Sign In**

User Id

Password

Here is an example of a user list authentication operation:

Basic			
Label *	<input type="text" value="User Auth List Application Test"/>	Title	<input type="text" value="User Auth List Application Test"/>
Operation Type *	<input type="text" value="User Authentication List"/>	Description	<input type="text" value="User Auth List Application Test"/>
Operation Code	<input type="text" value="user, password; user_2, password_2;"/>		
Instructions	<input type="text"/>		

The user name and password are separated by a comma. The name password pairs are separated by a semi-colon.

After the authentication operation has been created it is assigned to the application.

Basic			
Name*	User Authentication Test	Description	User Authentication Test
Connection*	Authentication Test	Sign On Type*	User List Sign On
Initial Operation	-- None --	Sign On Operation	User Auth List Application Test
			-- None --
			User Auth List Application Test

LDAP [Minimum Version: WOW 7.0]

[EE] The LDAP sign on validates a user ID and password against an existing LDAP server. Using LDAP allows the user ID and password to be centrally located. The user signs in to the application with their LDAP sign-on (user ID and password). WOW utilizes the LDAP credentials provided by the user and contacts LDAP to verify authentication. This authentication option requires that the existing LDAP system be accessible from the WOW server.

NOTICE: Because each LDAP configuration may be unique, it is not guaranteed that the default WOW implementation will be compatible with your environment. Custom Java development may be required that is not part of standard WOW product support. Customization can be done by WOW consultants.

Steps for configuring LDAP Authentication

1) Create a Referrer Operation:

To use the LDAP sign on, you must first create an operation of type "Referrer Authorization". In the operation code, specify the LDAP connection URL.

Basic	
Label*	LDAP Signon
Operation Type*	Referrer Authorization
Description	
Operation Code	ldap://myldapservers.mycompany.com:636,uid=??USERNAME,ou=people,o=mycompany

You will need to contact your LDAP administrator for the specifics of your LDAP server, but here is a basic break down of the LDAP connection URL:

```
ldap://ldapiip:636,uid=??USERNAME,ou=people,o=mycompany
```

- **ldap://** - required URL prefix.
- **ldapiip** – the host or ip address of the LDAP service directory
- **636** – the LDAP port to use. Generally port 389 is the default for unsecured (user ID and password are visible) and port 636 is the default for secured (SSL). Port unsecured port should only be used for initial testing. Enabling the use of SSL will be explained later.
- **uid=??USERNAME** – provides the user ID to the LDAP server. WOW will replace the ??USERNAME parameter with the user ID specified on the signon screen.
- **Base information** -Authenticating for individual people (e.g. "ou=people,o=mycompany")

2) Configure the Application:

Next configure the application to use the LDAP sign on. Change the *Sign On Type* to "LDAP Sign On". Then change *Sign On Operation* to the LDAP sign on operation (Referer Authorization) created in the previous step.

Testing Without SSL:

By default, WOW tries to use SSL when authenticating with LDAP. For initial testing, it is recommended to test without SSL first to ensure that the configuration is correct. To disable LDAP:

- Use the non-secure port in the LDAP URL, usually 389.
- In the application's properties, include the following:

```
LDAP{ssl:false;}
```

Enabling SSL:

To enable the use of SSL for LDAP authentication:

- 1) Install the required certification files on your WOW server.
 - Contact your LDAP administrator and acquire the necessary certificate files needed for your server.
 - Determine the Java location used by Tomcat (if needed, run Monitor Tomcat => Configure).
 - Copy those certificate files to Java's security folder: ~C:\Program Files\Java\jre6\lib\security
 - Run Window's Command Prompt

- (From the command prompt) change the current directory to correspond to Java's security folder:
`cd "C:\Program Files\Java\jre6\lib\security"`
- (From the command prompt) For each certificate file, run the keytool command to import the file:
`"C:\Program Files\Java\jre6\bin\keytool" -import -trustcacerts -alias certfile1 -file certfile1.crt -keystore ./cacerts -storepass changeit`
- (From the command prompt) If more than 1 certificate file, repeat the above step for each additional file, changing the alias and file parameters.
- Restart Tomcat (application restart will not work).

2) Change the port in the LDAP URL to the SSL port (default is 636).

Restricting Group Access to an Application

You can limit which users can sign into an application by designating a specific LDAP group or list of groups. To designate specific groups, edit the application. In the application properties, if the "LDAP" property group does not already exist, you'll need to add it.

Configure Group Search Properties

First you'll need to add search configuration properties for groups for your LDAP system (used to search for/extract groups from the LDAP system). If needed, contact your LDAP administrator for the correct configuration values:

- groupSearchBase - The search base to use for LDAP groups. E.g. "ou=people,o=mycompany". This parameter must be specified for group extraction.
- groupKey - The group attribute key. The default is "groupMembership".
- groupOrgUnit - The Group Organizational Unit. E.g. "ou=groups,o= mycompany "

```
LDAP{ groupSearchBase: ou=people,o=mycompany;
groupKey: groupMembership; groupOrgUnit: ou=groups,o= mycompany;
}
```

Add Group Properties

Next, to designate the groups to restrict access to the application, add the "groups" property.

```
LDAP{ groupSearchBase: ou=people,o=mycompany;
groupKey: groupMembership; groupOrgUnit: ou=groups,o= mycompany;
groups:group1,group2;
}
```

If more than one group is to have access, separate them with commas. The list of groups a user belongs to will be cross checked against the list of groups specified for the groups property above.

LDAP {} Property Group:

The LDAP property group should be specified in the application's properties and allows you to alter some of the LDAP configuration:

Property	Value	Description
ssl	TRUE FALSE	Should WOW use SSL when contacting LDAP? The default is true.
authentication	simple none strong	Contact your LDAP administrator for the correct setting. The default is simple.
groupSearchBase	search base string	The search base to use for LDAP groups. E.g. "ou=people,o=mycompany". This parameter must be specified for group extraction.
groupKey	group key string	The group attribute key. The default is "groupMembership".
sqlOperation	integer value	The ID of the authentication or SQL operation. Used with LDAP plus Operation.
groupOrgUnit	organizational unit string	The Group Organizational Unit. E.g. "ou=groups,o=mycompany "
groups	list of group names	A list of comma separated LDAP group names to restrict access to.

NOTE: When WOW performs LDAP authentication, LDAP user profile information is stored in the user's sign-on row in their browser session. This information will be referred to later as the LDAP signon row. These profile attributes can now be accessed from any WOW SQL statements using a sign-on row parameter ("???" + property-name). For example, if an LDAP property ID is "MAIL", then an operation's SQL might look like the following:

```
SELECT * FROM MYSCHEMA.MYTABLE WHERE emailfld = ???MAIL
```

Available property field names can be viewed from WOW's user tools (visible when application launched from WOW builder) by clicking on the "User Info" link.

LDAP Plus Operation[Minimum Version: WOW 7.0]

[EE] The LDAP Plus Operation sign on is almost identical to an LDAP sign on, with the exception that after the user is authenticated against LDAP, an SQL operation is run to retrieve a property Row.

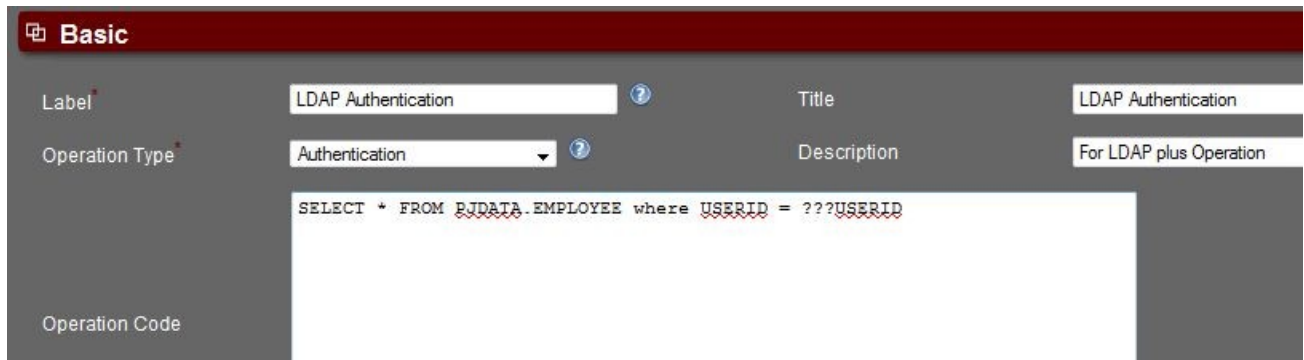
Steps for configuring LDAP Plus Operation Authentication

1) Follow the steps defined for LDAP authentication, with the exception that Sign On Type = **LDAP Plus Operation Sign On** (step: Configure the Application).



A screenshot of a configuration form for LDAP Plus Operation Sign On. The form has a dark red header bar. Below it, there are three rows of configuration fields. The first row is 'Description' with a text input field containing 'My Application' and a help icon. The second row is 'Sign On Type' with a dropdown menu showing 'LDAP Plus Operation Sign On' and a help icon. The third row is 'Sign On Operation' with a dropdown menu showing 'LDAP Signon' and a help icon.

2) Within the LDAP application, create an operation of type **Authentication**. Set the Operation Code to the SQL that will be used to retrieve the desired properties Row. Use a built-in parameter, such as ???USERID (the user ID used to sign onto the application) or any fields from the LDAP signon row (prefaced with "???", e.g ???USER), to filter the results returned from the operation so they correspond with the user signing on:



A screenshot of a configuration form for LDAP Authentication. The form has a dark red header bar with a 'Basic' tab. Below it, there are four rows of configuration fields. The first row is 'Label' with a text input field containing 'LDAP Authentication' and a help icon. The second row is 'Operation Type' with a dropdown menu showing 'Authentication' and a help icon. The third row is 'Title' with a text input field containing 'LDAP Authentication'. The fourth row is 'Description' with a text input field containing 'For LDAP plus Operation'. Below these fields is a large text area for the 'Operation Code' containing the SQL query: `SELECT * FROM PJDATA.EMPLOYEE where USERID = ???USERID`.

Optionally, the **Authentication** operation can exist in another application (or application set to shared, where the application ID = None or -1) or be set to type **SQL Operation**. In this case, you will need to use the *sqlOperation* property to specify the operation ID. For example, if the operation ID of the **Authentication/SQL** operation is 1025, then include the following property in the LDAP property group:
sqlOperation:1025;

User Groups [Minimum Version: WOW 7.0]

[EE] User groups can be used to secure applications, operations and fields. A user can be assigned to one or more groups, and the groups represent a certain level of authorization. WOW supports the follow types of User Groups:

- LDAP Signon - When using an LDAP type signon, all groups a user belongs to are automatically gathered when the user signs in and the LDAP server is contacted for signin authorization. That group list is then retained as part of the user's session and can then later be used by WOW to see if the user is authorized to an application, operation or field. See [LDAP \[Minimum Version: WOW 7.0\]](#) for more details on configuring an LDAP signon for groups.
- SQL Operation Signon - When defining an SQL Operation Signon, if one of the fields retrieved by the operation has a usage ID = -135, the string value will be parsed (comma separated) for one or more user group names. That group name list is retained as part of the user's session and can then later be used by WOW to see if the user is authorized to an application, operation or field.

Once you have your application configured to keep track of a user's groups, follow the instructions below for User Group Authorization List.

Securing Operations

[PRO] WOW has a powerful security feature that can be set from within each individual operation. The security measures within the operations are used to allow only certain users with a specific security level to view operations. Every user (anonymous or otherwise) by default has a security level of zero, unless otherwise specified. WOW uses a simple number scheme to control which users can view certain operations within a specific WOW Application. When an operation is set to a security level of 0, all users will have access to that operation. When an operation is set to another security level, only users with security level equal to or higher than the operation's security level will have access to that operation. An operation's security level can be set in that operation's detail screen. The Security options are found under the Administration section as show in the screen shot below:



The screenshot shows a web interface for the 'Administration' section. It contains two rows of settings. The first row has 'Security Type:' followed by a dropdown menu showing 'Level Security' and 'Security Level:' followed by a dropdown menu showing '0'. The second row has 'Auto Run Op.:' followed by a dropdown menu showing '- None -'.

Security Settings

Security Type: The Security type is where you set the initial security level. Security Type has two available settings. Level Security is the most powerful of the settings. It uses a numbering scheme to allow access to specific operations.

Level Security – This is the most the most powerful of the security settings that are available within WOW. When Security Type is set to Level Security it allows the application builder to give Operations a specific number which determines which users have access to that Operation.

Operation Security – This security setting allows a very versatile way to secure operations. When Security Type is set to Operation Security, it allows the application builder the flexibility to utilize any file. The Operation Security setting requires that the Execute Authority Operation be set to a User Authorization operation.

See section Securing Fields and Operations with User Authorization Operations for more details.

Unsecured – An unsecured Operation has no security measures. This means that anyone who uses WOW can use that Operation. This is similar to a security level of 0 when using the Level Security Feature. An unsecured SQL Operation can be viewed by anyone with access to that application.

Secured Operation Example

This example is can also be found under the "Level Based Security" application provided with the sample applications. This example assumes an application has already been created and has its sign on type set to SQL Operation.

First, we need to create an Authentication Operation to use for authenticating users. Follow steps provided previous in the chapter on how to create an authentication operation. For this example, the Authentication's Operation SQL will be:

```
select seclevel from planetj.user where email = ? and password = ?
```

This sign on Operation will validate the email and password given by a user and if valid, store that user's **urseclevel** Field's value as a user property. Listed below are some of the Fields and values that can be found in the planetj.user table. There are actually more columns than these, but we only need three for this example.

URNAME	URPASSWORD	URSECLEVEL
joepublic	j949	0
peterpoweruser	trustno1	6

You'll notice that Joe Public has a security level of 0 and Peter Poweruser has a security level of 6. This value will be stored as a user property when each of them sign in to the application.

Next, we create another operation, this time just a regular SQL Operation. Its operation code can be set to:

```
select * from planetj.emp
```

This operation should also be secured. To do this, set the operation's security type to **Level Security**, along with a level of **2**.

Administration

Security Type:	<input type="text" value="Level Security"/>	Security Level*:	<input type="text" value="2"/>
Auto Run Op.:	<input type="text" value="- None -"/>		

Now when the two users sign on, neither will be able to see this operation. Peter Poweruser has a security level of 6, so why can't he see it? There's one more step that needs to be done. The **SecLevel** Field is being stored as a user property, but in order for the security feature utilizing Level Based Security to pick it up, the **SecLevel's** FieldDescriptor needs to have its usage id set to -160. This usage id value denotes a **Security Level Field**.

Now when Peter Poweruser signs in, he will be allowed to see the Operation for view employees, whereas Joe Public still cannot see it because his security level field value is not greater than or equal to the Operation's security level.

Optional Sign On

In some scenarios, you may have an application with certain operations that anyone can use and other operations to which only certain users have access. In this case, you may want to let users access your application without signing on, as long as they limit themselves to the operations available to everyone. To allow users to access the application without signing on, but still allow them to sign on if they want to, check the "Optional Sign On" checkbox in the application details screen:

Advanced

Subclass:	<input type="text"/>	JSP File:	<input type="text"/>
Optional Sign On* :	<input checked="" type="checkbox"/>	Auto Run Status* :	<input type="text" value="- Choose -"/>

When a user starts an application whose sign on is optional, they are presented with a screen similar to this one (the operations will vary depending on the application):

Acme Co


Application Launch

Welcome

Optional Access

Email

Password

 LOGIN

Employee Photos

All Customers Reprise

Dakota Cities

Dakota Cities Reprise

Clients by State

Query Parameters

Balance <

Search

There is a sign on area in the top left of the screen which the user can use if he or she wants to sign on. If the user does not sign on, then that user has a security level of 0, and only has access to operations which also have a security level of 0. Once signed on, the user has a security level determined by the userid used to sign on, and will have access to all operations with a security level equal to or less than the user's security level.

NOTE: Optional Sign On is only supported when the sign on type is SQL Operation.

Table Authorization

[EE] WOW allows you to control access to the tables used by your application. By default all users are permitted to access a table in any way your operations allow. In order to restrict the ways in which users are allowed to access a table you create a user authorization operation and then specify that operation in a table descriptor, as described below.

Using Table Authorization

In this example, we will restrict the users who are allowed to delete from our table. It is also possible to restrict the users who may read, update, insert into, or alter a particular table. Here is the result of selecting rows from our table, before we restrict deletes.

User Id =



	▲ Customer Id ▼	▲ User Id ▼	▲ Balance ▼
	1	Amy	130
	2	Bradly	250
	3	Chuck	35
	4	Daynica	333
	5	Evelyn	122
	6	Frank	-60
	7	Gipper	-66
	8	Helga	-88

The first step is to create a user authorization operation. This operation will return a list of the users who have authority to delete from the table. (User authorization operations are covered in more detail earlier in this chapter.) This is what our user authorization operation looks like:

Basic




Label: Operation Type*:

Description:

The SQL in the operation may use the "??TABLE_NAME" parameter to refer to the table whose authority is being checked. This allows a single user authorization operation to be used by multiple tables.

This user authorization operation must now be specified in the table's table descriptor. To locate the table descriptor, open up Field Descriptor Manager, and go to the FDs for that table. (See the Field Descriptors Chapter for more information on Field Descriptor

Manager.) The table descriptor will have the same name as the table, except it will be prefixed with a tilde (~). In our example, we are working with the CUSTOMER table, so the table descriptor is named "~CUSTOMER".

	▲ Id ▼	▲ Field Name ▼	▲ External Name ▼
	123154	~CUSTOMER	CUSTOMER
	123155	BALANCE	Balance
	123156	ID	Customer ID

Edit the table descriptor, and in the Authorization Settings section, look for the Delete Authorization Operation field. Set the value of this field to the user authorization operation created earlier. (To restrict reads, updates, inserts, or alters set the authorization operation in the appropriate field.)

Advanced Settings

Table Class

-- None --

Field Descriptor Type*
Table Desc

Authorization Settings

Read Authorization Operation

-- None --

Update Authorization Operation

-- None --

Insert Authorization Operation

-- None --

Delete Authorization Operation

Sample Use

Now, any operation which attempts to access the CUSTOMER table will prevent unauthorized users from deleting rows from the table. Note that the table descriptor is associated with a particular connection alias, so the security restrictions will only apply when the table is accessed using this connection.

If signing on as a user without authority to delete from the table, select rows from the table will not be displayed.

User Id =

Search



	▲ Customer Id ▼	▲ User Id ▼	▲ Balance ▼
	1	Amy	130
	2	Bradly	250
	3	Chuck	35
	4	Daynica	333
	5	Evelyn	122
	6	Frank	-60
	7	Gipper	-66
	8	Heloa	-88

Securing Fields and Operations with User Authorization Operations

Overview

[PRO] User authorization operations give the developer the ability to restrict a user from viewing and/or editing any fields or operations based on sign on.

Four types of Authorization Operations are available.

- User Authorization List Operation
- User Authorization (SQL) Operation
- User Group Authorization List Operation
- User Group Authorization (SQL) Operation

Authorization Operations should only be referenced by fields/operations included within secure applications

The field/operation being secured must exist within a secure application that requires a sign on.

If the user has not signed on to the application or the application does not require a sign on then authorization will fail and the user will not have access to the field/operation.

User Authorization List

This type of operation holds a static list of user names that is defined when the authorization operation is created.

The list of user names cannot be changed except by changing the authorization operation itself.

This is authorization operation is useful when a small number of users will have restricted access to certain fields and/or operations.

Creating User Authorization List

This is the screen where the application builder creates the User Authorization List Operation. Each relevant setting is described below.

The screenshot shows a software configuration window titled "Basic" with a dark red header. The window contains several input fields and a large text area. The "Label" field is set to "Sample User Authorization List". The "Operation Type" dropdown menu is set to "User Authorization List". The "Description" field is empty. The "Operation Code" field is a large text area containing the text "jsmith, bsmith, dickson, bddley". A "Database Explorer" button is located in the bottom right corner. Each input field has a small blue question mark icon to its right.

Field	Value
Label	Sample User Authorization List
Operation Type	User Authorization List
Description	
Operation Code	jsmith, bsmith, dickson, bddley

Basic

Operation Type - User Authorization List

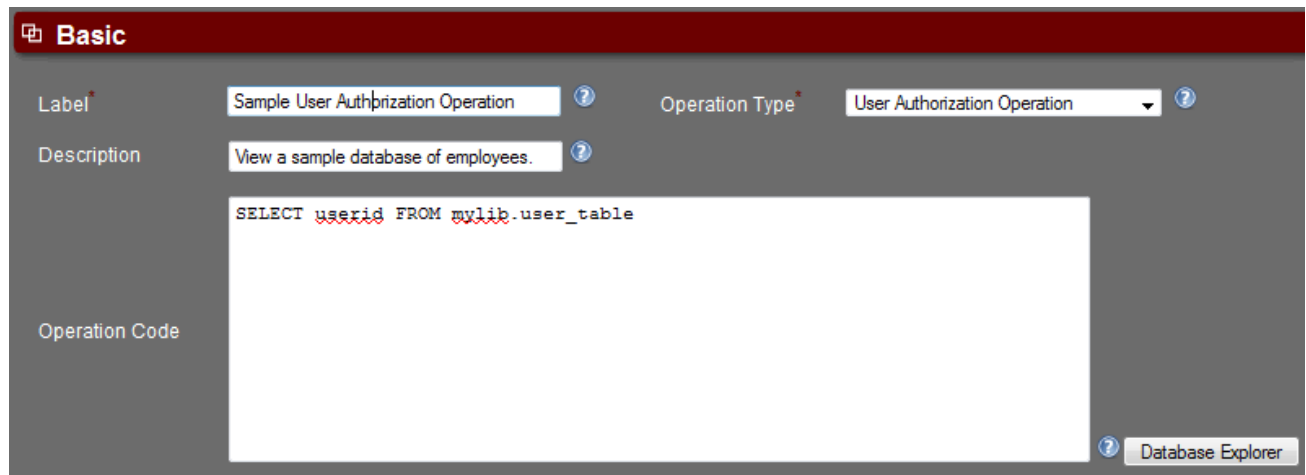
Operation Code - A static list of comma separated names

User Authorization Operation

This type of operation uses SQL to retrieve a list of users (user ID's) that are authorized to use an operation or field. The list of user names is dynamically generated, which provides greater flexibility.

Creating User Authorization Operation

This is the screen where the wow builder creates the User Authorization Operation. Each relevant setting is described below:



The screenshot shows a software interface titled "Basic" in a red header bar. Below the header, there are several configuration fields:

- Label:** A text box containing "Sample User Authprization Operation" (note the typo) with a blue question mark icon to its right.
- Operation Type:** A dropdown menu showing "User Authorization Operation" with a blue question mark icon to its right.
- Description:** A text box containing "View a sample database of employees." with a blue question mark icon to its right.
- Operation Code:** A large text area containing the SQL query: `SELECT userid FROM mylib.user_table`. The text "userid" and "mylib" are underlined with red wavy lines, indicating syntax highlighting or warnings.

In the bottom right corner, there is a button labeled "Database Explorer" with a blue question mark icon to its left.

Basic

Operation Type - User Authorization Operation

Operation Code - SQL used to retrieve a list of the desired user names (user ID's).

User Group Authorization List

This type of operation holds a static list of group names that are defined when the authorization operation is created. The list of group names cannot be changed except by changing the authorization operation itself.

This authorization operation is useful when a small number of user groups will have restricted access to certain fields and/or operations.

Creating User Group Authorization List

This is the screen where the wow builder creates the User group Authorization List Operation. Each relevant setting is described below.

The screenshot shows a 'Basic' configuration window with a dark red header. It contains the following fields:

- Label:** A text box containing 'Sample User Group Auth List' with a help icon.
- Operation Type:** A dropdown menu set to 'User Group Authorization List' with a help icon.
- Description:** An empty text box with a help icon.
- Operation Code:** A large text area containing the text 'group1, group2, group3'.
- Database Explorer:** A button with a help icon located at the bottom right of the text area.

Basic

Operation Type - User Group Authorization List

Operation Code - A static list of comma separated group names

User Group Authorization Operation

This type of operation uses SQL to retrieve a list of group names that are authorized to use an operation or field. The list of group names is dynamically generated, which provides greater flexibility.

Creating User Group Authorization Operation

This is the screen where the wow builder creates the User Group Authorization operation. Each relevant setting is described below:

The screenshot shows a 'Basic' configuration window with a dark red header. It contains the following fields:

- Label:** A text box containing 'Sample User Group Auth Operation' with a help icon.
- Operation Type:** A dropdown menu set to 'User Group Authorization Operation' with a help icon.
- Description:** An empty text box with a help icon.
- Operation Code:** A large text area containing the SQL query 'Select group from mylib.grouptable'.
- Database Explorer:** A button with a help icon located at the bottom right of the text area.

Basic

Operation Type - User Group Authorization Operation

Operation Code - SQL used to retrieve a list of the desired group names.

Field Level Authorization

[PRO] Access to a field may be restricted in two ways.

1. **Authorization to Read**- Authorized users has access to view the field but not necessarily edit the field.
2. **Authorization to Edit**- Authorized users has access to edit the field. This applies only if the field is visible to them.

A field may be secured for Read or Edit by either type of authorization operation.

Both types of access can be used together to provide field security that may be:

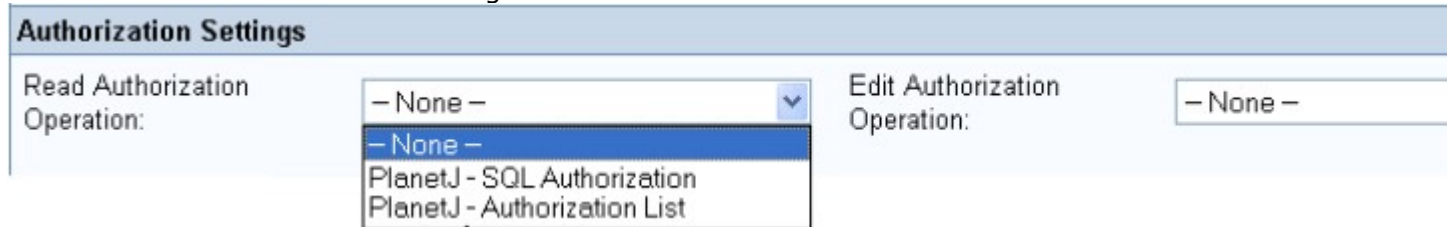
- Visible/Editable by a group of users while hidden from another group.
- Visible/Editable by a group of users while Visible/Not Editable to another group.
- Visible/Not Editable by a group of users while Not Visible from another group.

Assigning an Authorization Operation to a Field

A Field Descriptor should already have been created for the field that requires the authorization operation.

In the field descriptor:

This is the screen where the application builder sets the Authorization Operation on the Field to be secured. Each relevant setting is described below.



The screenshot shows a window titled "Authorization Settings". It has two main sections. The first section is labeled "Read Authorization Operation:" and contains a dropdown menu that is currently open, showing three options: "- None -", "PlanetJ - SQL Authorization", and "PlanetJ - Authorization List". The second section is labeled "Edit Authorization Operation:" and contains a dropdown menu that is closed, showing the option "- None -".

Authorization Settings

All Authorization Operations defined for the current application should appear in the drop down selection.

Read Authorization Operation - Only users returned by the Authorization Operation will have access to view this field. If no operation is selected all users will be authorized to view this field.

Edit Authorization Operation - Only users returned by the Authorization Operation will have access to edit this field. If no operation is selected all users will be authorized to edit this field.

Both User Authorization List Operations and User Authorization Operations apply.

Assigning Authorization Operation to an SQL Operation

Current Application - Edit Operation

This is the screen where the application builder sets the Authorization Operation on the SQL Operation to be secured. Each relevant setting is described below.

Administration			
Security Type:	<div>Operation Security</div>	Security Level:	<div>0</div>
Auto Run Op.:	<div>- None -</div>	Execute Authority Operation:	<div>- None -</div> <div>- None -</div> <div>PlanetJ - SQL Authoriza</div> <div>PlanetJ - Authorization L</div>

Administration

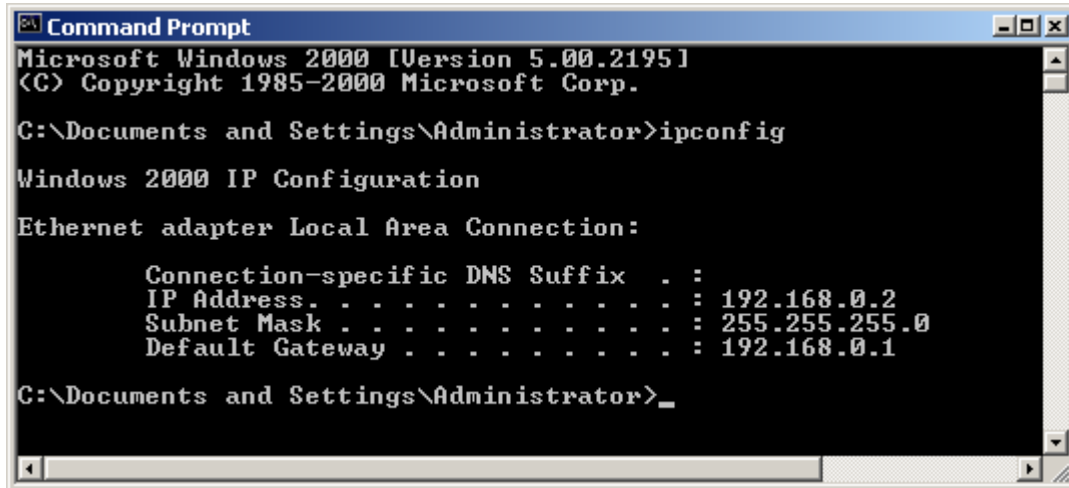
Security Type-Operation Security

Execute Authority Operation - All Authorization Operations defined for the current application should appear in the drop down selection. If no operation is selected all users will be authorized to execute this operation.

Both User Authorization List Operations and User Authorization Operations apply.

Deploying Applications

Once an application has been built, it is ready to be run by users. A WOW application can be run by any user on your network or Internet/Intranet/Extranet. WOW applications can be run by specifying a URL in the form: `http://yourIP/wow64/run?id=x` where `yourIP` is the TCP/IP address of the WOW server and `x` is the WOW application ID. The IP can be found by opening a DOS window on the WOW server and issuing an `IPCONFIG` command as shown below:



```
Command Prompt
Microsoft Windows 2000 [Version 5.00.2195]
(C) Copyright 1985-2000 Microsoft Corp.

C:\Documents and Settings\Administrator>ipconfig

Windows 2000 IP Configuration

Ethernet adapter Local Area Connection:

    Connection-specific DNS Suffix  . : 
    IP Address. . . . .               : 192.168.0.2
    Subnet Mask . . . . .             : 255.255.255.0
    Default Gateway . . . . .          : 192.168.0.1

C:\Documents and Settings\Administrator>
```

In this example, WOW applications can be run using:

`http://192.168.0.2/wow64/run?id=0`

where 0 is the WOW application ID which is found in the Application menu in the WOW Builder. Each new WOW application gets the next sequential number.

NOTE: Tomcat and WOW install by default on port 8080 which would require a URL such as:
`http://192.168.0.2:8080/wow64/run?id=0`.
Tomcats port can be configured in the default Tomcat directly (`.../config/server.xml`). See Tomcat documentation for details.

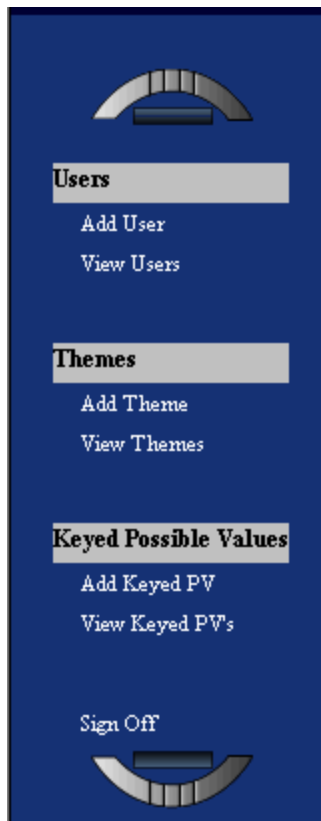


WOW Utilities

WOW is packaged with a set of utilities that helps the application builders manage different aspects of WOW. The three utilities included will help you manage different user accounts, theme management, and keyed possible values management. Themes are discussed in more depth later in this guide.

Users

The Users drop-down menu in WOW Utilities contains two operations, Add Users and View Users. User accounts are used for security purposes within WOW. If an application's security type is set to "SQL Operation", then the email address and password will be used to enter these 'secure' applications.



Add User

Adding a new user is a very straight-forward process. Simply click on the Add User Operation like shown in the screen shot above to set-up a new user. The screenshot below shows an example of the Add User Operation.

Fields marked with an asterisk (*) are required.

The screenshot shows a user creation form with two main sections: "Security Information" and "User Information". The "Security Information" section contains three fields: "Email Address*" (text input), "Password*" (text input), and "Security Level*" (dropdown menu with "0" selected). The "User Information" section contains two fields: "First Name:" (text input) and "Last Name:" (text input). The form has a blue header bar with "Insert" and "Cancel" buttons. Below the "User Information" section, there is another blue bar with "Insert" and "Cancel" buttons.

Security Information

User Name – This is the user name that will be used to sign on to secure operations within WOW.

Password – This is the password that will be used in conjunction with the User Account being created.

Security Level– This is the security level available for the user. Security levels were discussed in detail earlier.

User Information

First Name – This is the first name of the user account being created.

Last Name – This is the last name of the user account being created.

Additional Properties

Theme – ###

View User

The View User screen simply displays the user accounts that have already been created within WOW. The screen shot below will be very similar to the screen displayed after initially signing on to the WOW Utilities.

The View User screen simply displays the user accounts that have already been created within WOW. The screen shot below will be very similar to the screen displayed after initially signing on to the WOW Utilities.



Users

[Add User](#)

[View Users](#)

Themes

[Add Theme](#)

[View Themes](#)

Keyed Possible Values

[Add Keyed PV](#)

[View Keyed PV's](#)

[Sign Off](#)











































Messages

You have successfully signed on.

All Users



	▲ Email Address ▼	▲ First Name ▼	▲ Last Name ▼
   	admin@planetjavainc.com	Administrator	PlanetJ
   	jchester@planetjavainc.com	Jon	Chester
   	joe_public@planetjavainc.com	Joe	Public
   	level2@planetjavainc.com	level2	level2
   	level5@planetjavainc.com	level5	level5
   	level7@planetjavainc.com	level7	level7
   	level9@planetjavainc.com	level9	level9
   	peter_poweruser@planetjavainc.com	Peter	Poweruser
   	support@apex.com	Apex	Award
   	support@wabashnational.com	Support	Wabash

Themes

Themes are the look and feel applied to your WOW Applications. The Themes section of the WOW utilities contains two operations. Add Themes and View Themes.

Add Theme

Theme properties are contained within specific CSS files. This Chapter will not explain the process of creating a theme, when adding a theme WOW assumes the CSS file has already been created and it simply needs to be added to the list of available themes. The screenshot below shows the Add Theme operation screen. After adding the relevant information simply hit the insert button to add the new theme.

Fields marked with an asterisk (*) are required.

InsertCancel

CSS File:Description:

InsertCancel

CSS File –This is the location of the CSS file that contains the properties for the newly created theme. The location of the file will differ according to where and how you installed WOW on your system.

Description –This is simply a description of the newly created theme.

Properties – ###

View Theme

The View Theme operation is similar to the View User operation described earlier in this Chapter. This operation simply lists the themes that have been previously created. The screenshot below shows an example of the View Theme operation, listing all of the themes available within your copy of WOW.

All Themes



	▲Description ▼	▲CSS File ▼
	Blue light	/dataengine/themes/gelbase/css/gelbaselt.css
	Gelbase - Blue	/dataengine/themes/gelbase/css/gelbase.css
	Liteup - Brown	/dataengine/themes/liteup/css/liteup.css
	Brown light	/dataengine/themes/liteup/css/liteuplt.css
	Black Blue	/dataengine/themes/blackblue/css/blackblue.css
	Red Black	/dataengine/themes/redblack/css/redblack.css

Dynamic Themes with URL Parameter

There may be the case where you have added multiple themes and want certain customers to open the application with a particular theme. Well you can do this dynamically by setting the theme to be used in the URL as a parameter.

For example... let's say we have the following URL to an application:

<http://www.planetjavainc.com/wow/runApp?id=0>

We can dynamically change this application's theme by specifying a parameter on the URL: `_pj_theme=-5` or `_pj_theme=-3` or any other theme id that exists shown in view themes screen.

NOTE: A negative theme number indicates a system theme.

So for example...

http://www.planetjavainc.com/wow63/runApp?id=0&_pj_theme=-5 will open the application using the Light Brown theme.

http://www.planetjavainc.com/wow63/runApp?id=0&_pj_theme=-3 will open the application using the Light Blue theme.

The theme id is created when you add a new theme and can be seen in the view themes screen.






Keyed Values

This section allows you to manage keyed values, including possible values, configuration values, and user/error messages.

View Possible Values

With a keyed possible value, you specify a key, a value, and a display value for each PV. (The key is used to group the PV's together). Then in an FD (field descriptor) under the list of Possible Value Keys, you will see the keys of the Possible Values that you added. This is a separate PV option than using PV Operations - so there are two different ways to use PV's with WOW.

This displays a list of all of the Possible Value Keys that have been created. The size of the list will vary greatly depending on how many PV keys have already been set up on your version of WOW. The screenshot below shows a list of PV keys.

Keyed Possible Values			
 	Next		
	▲ Key ▼	▲ Value ▼	▲ Display Value ▼
   	*US_STATES*	AL	Alabama
   	*US_STATES*	AK	Alaska
   	*US_STATES*	AZ	Arizona
   	*US_STATES*	AR	Arkansas
   	*US_STATES*	CA	California
   	*US_STATES*	CO	Colorado
   	*US_STATES*	CT	Connecticut
   	*US_STATES*	DE	Delaware
   	*US_STATES*	FL	Florida
   	*US_STATES*	GA	Georgia
   	*US_STATES*	HI	Hawaii
   	*US_STATES*	ID	Idaho
   	*US_STATES*	IL	Illinois
   	*US_STATES*	IN	Indiana
   	*US_STATES*	IA	Iowa

Add Possible Value

Adding a keyed Possible Value is much like adding a user account or theme like described above. The figure below shows an example of the Add Keyed PV Operation.

Fields marked with an asterisk (*) are required.

The screenshot shows a software dialog box with a blue title bar containing 'Insert' and 'Cancel' buttons. The main area is divided into two sections. The first section, 'Possible Value Group Info', contains a label 'Key*' followed by a text input field. The second section, 'Possible Value Info', contains two labels: 'Value*' followed by a text input field, and 'Display Value*' followed by another text input field. At the bottom of the dialog is another bar with 'Insert' and 'Cancel' buttons.

Possible Value Group Information

Key – This is the Key that will be used with the Possible Values. Possible Value Keys were described in detail in Chapter 10 of this guide.

Possible Value Information

Value – This is the value given to a specific possible value. Generally numeric data is used for the value when working with PVs but this is not required.

Display Value - This is the value displayed for the PV. This should be as descriptive as possible making it easy for the user to understand the value being used.

Additional Information

Display Order – ###

View Configuration Values

###

Add Configuration Value

###

Add User Message

User Messages provides the ability for you to supply your own messages to your WOW Applications. The User Messages section of the WOW utilities contains two operations, "Add User Message" and "View User Messages"

The screen shot below shows the Add User Message screen. After adding the relevant information, simply hit the insert button to add the new user message.

Fields marked with an asterisk (*) are required.

<input type="button" value="Insert"/>	
File Information	
Connection Name*	<input type="text" value="-- Choose --"/>
Library Name*	<input type="text"/>
Table Name*	<input type="text"/>
Message Information	
Message Type*	<input type="text" value="-- Choose --"/>
Matching Error Message Text	<input type="text"/>
Replacement Message*	<input type="text"/>
<input type="button" value="Insert"/>	

Message Information

- **Message Type** – This is the message type to override (insert, update, delete, etc.). This field is required.
- **Matching Error Message Text**– For error messages, this is the matching text from the original message that will be used to identify the specific message to replace. This field does not apply to the non-error message case and will be ignored. This field is optional. If this field is not specified in the error case, the user message will be used for all error messages of the specified message type. If you specify two messages of type "Delete Error" and the second entry leaves the matching text field blank, the first entry will be used when a match is found and the second entry will be used in all other cases for that message type.
- **Replacement Message**– This is the replacement message to be displayed. The message text can contain non-prompting parameters (ex. ??FLD1 or ???USERID). This field is required.

NOTE: For performance reasons, the user message entries are read in once after the server is started. If message entries are changed after the server is started, you will most likely need to restart your server again to incorporate the new replacement messages.

File Information

- **Connection Name** – This is the connection alias used to access the table. This field is required.
- **Library Name** – This is the library that contains the table. A generic '*' (without the quotes) is allowed to represent all libraries. This field is required.



- **Table Name**– This is the table name that the message override applies to (for any insert, update and delete). This field is required.

View User Messages

This displays a list of all user messages that have been created.

All User Messages



	▲ Connection Name ▼	▲ Library Name ▼	▲ Table Name ▼
	SYS	TEST	TABLE1
	SYS	TEST	TABLE1

Interfacing WOW with Excel

Connecting WOW to an Excel File

To connect WOW to a MS Excel file the following steps need to be done:

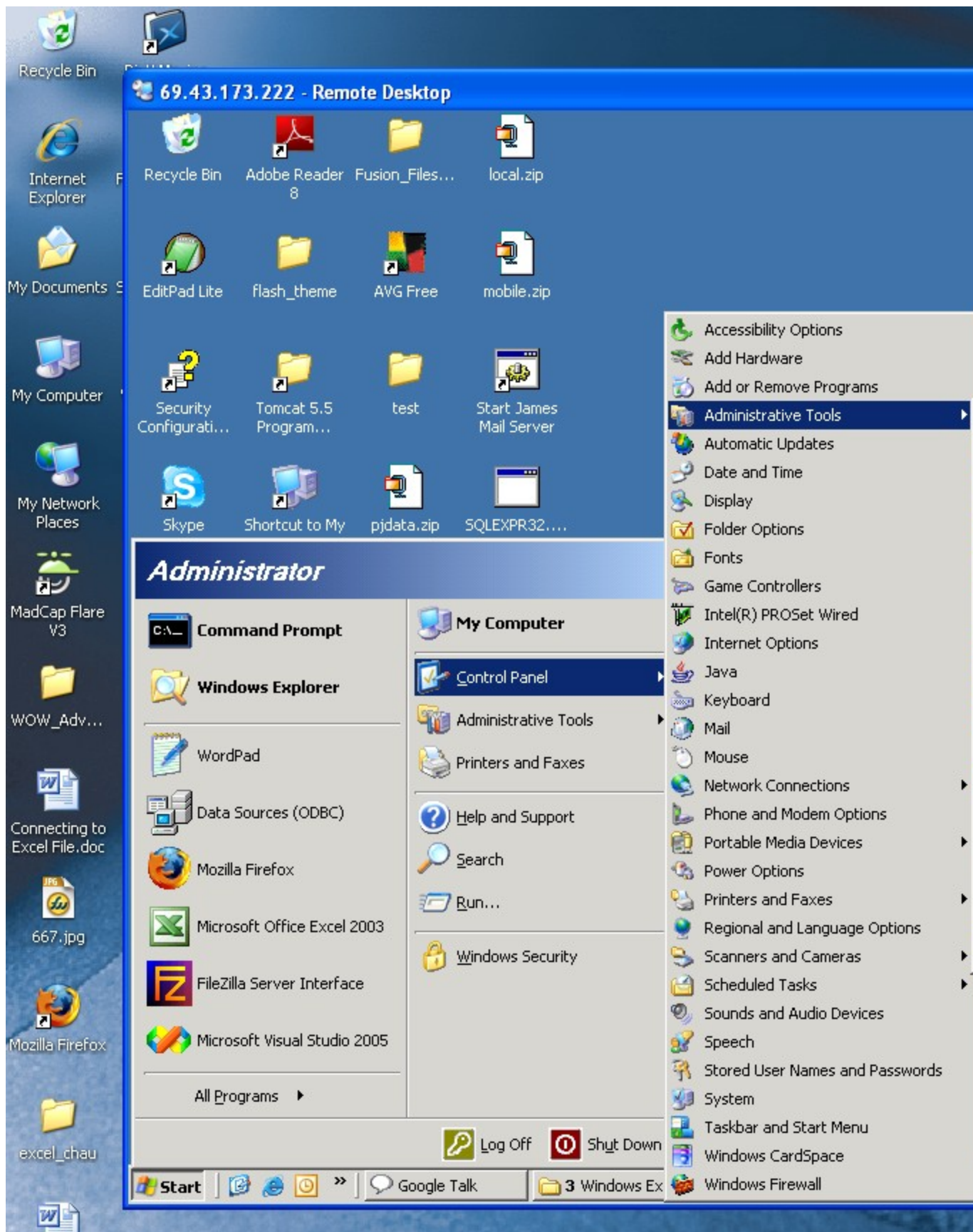
1. Create an ODBC System Data Source (DSN) on the same system where the application server is installed. (Remote access is not supported)
2. The DSN must reference the desired Excel worksheet.
3. Connect WOW to the created DSN from step 1 & 2.

NOTE:

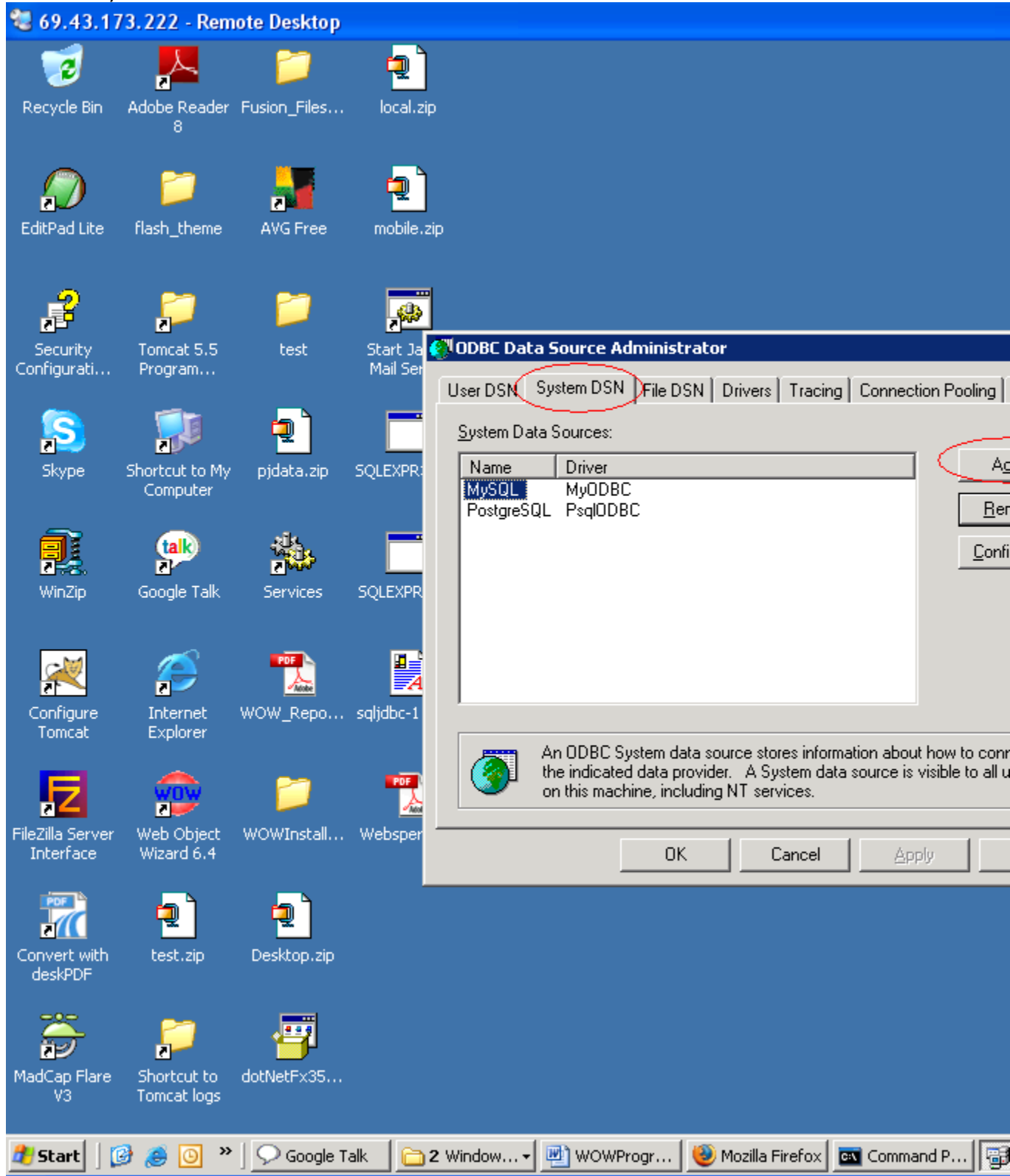
- Only read access is recommended although limited update capabilities are available.
- Support is available via JDBC ODBC Bridge.
- Excel files must reside on the same system as the WOW server although mapped drive support may be possible.

Creating system DSN (Windows Only)

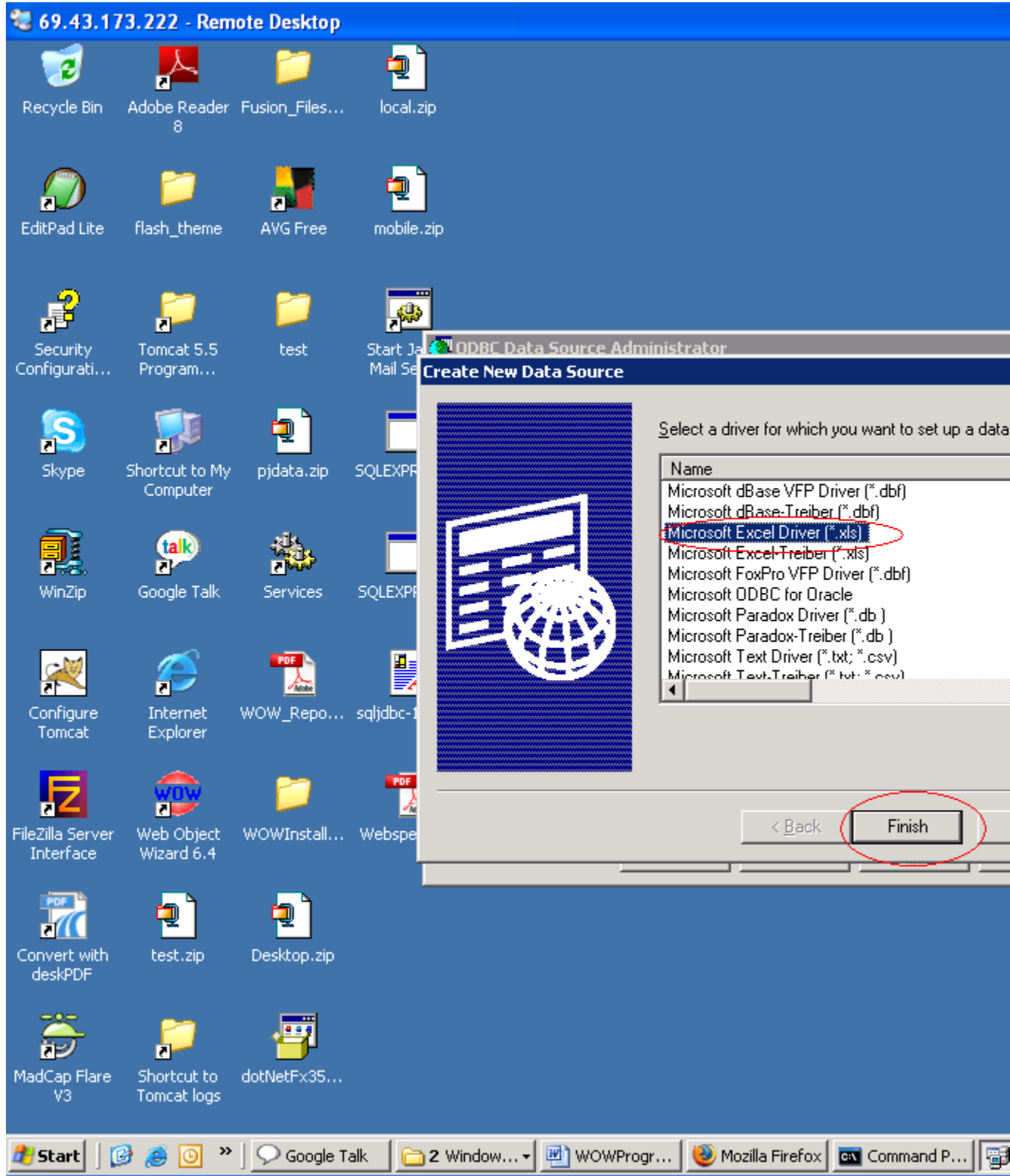
1. Go to start > Control panel > administrative tool > Data Source(ODBC) as seen in the screenshot below.



2. Choose "System DSN" and click on "Add" button.



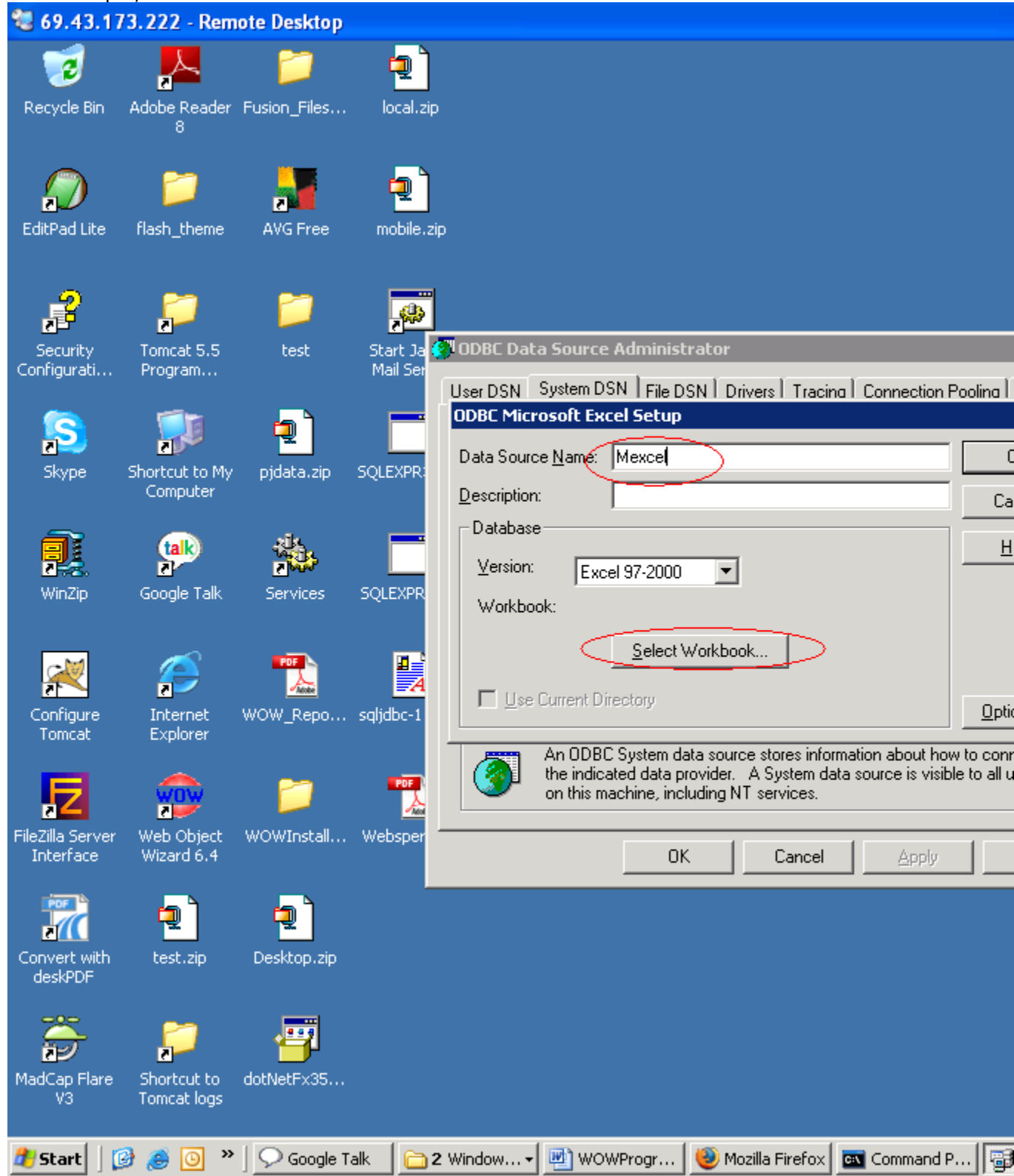
3. Under "Create New Data Source" panel, choose "Microsoft Excel Driver (*.xls)" then hit "Finish" as in the screenshot below



Pointing to desired Excel worksheet

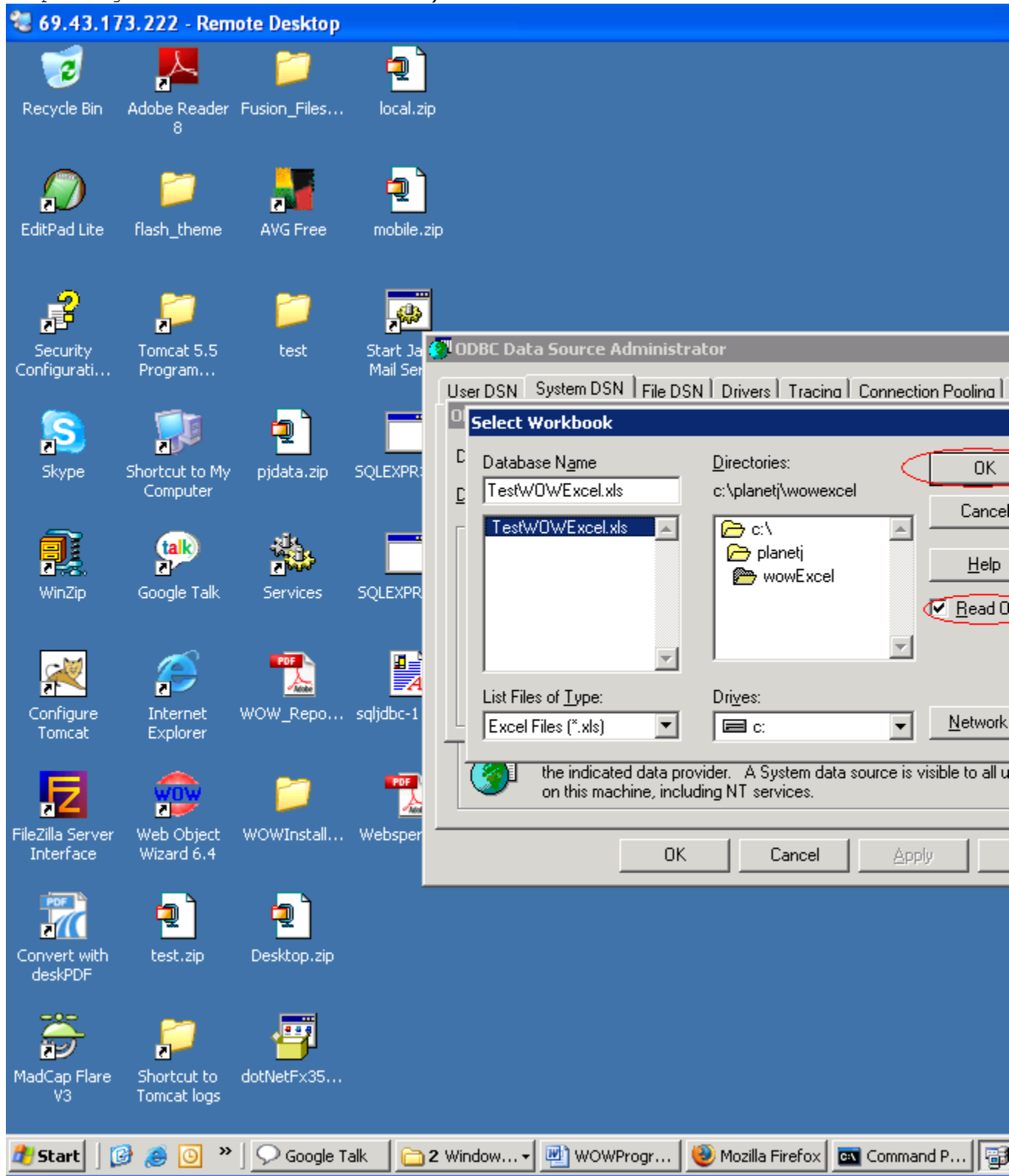
1. After clicking "finish" as the screenshot above, name the "Data Source Name" as desired and remember this name for using later then click on "Select Workbook." In

this example, the DSN is named "Mexcel."



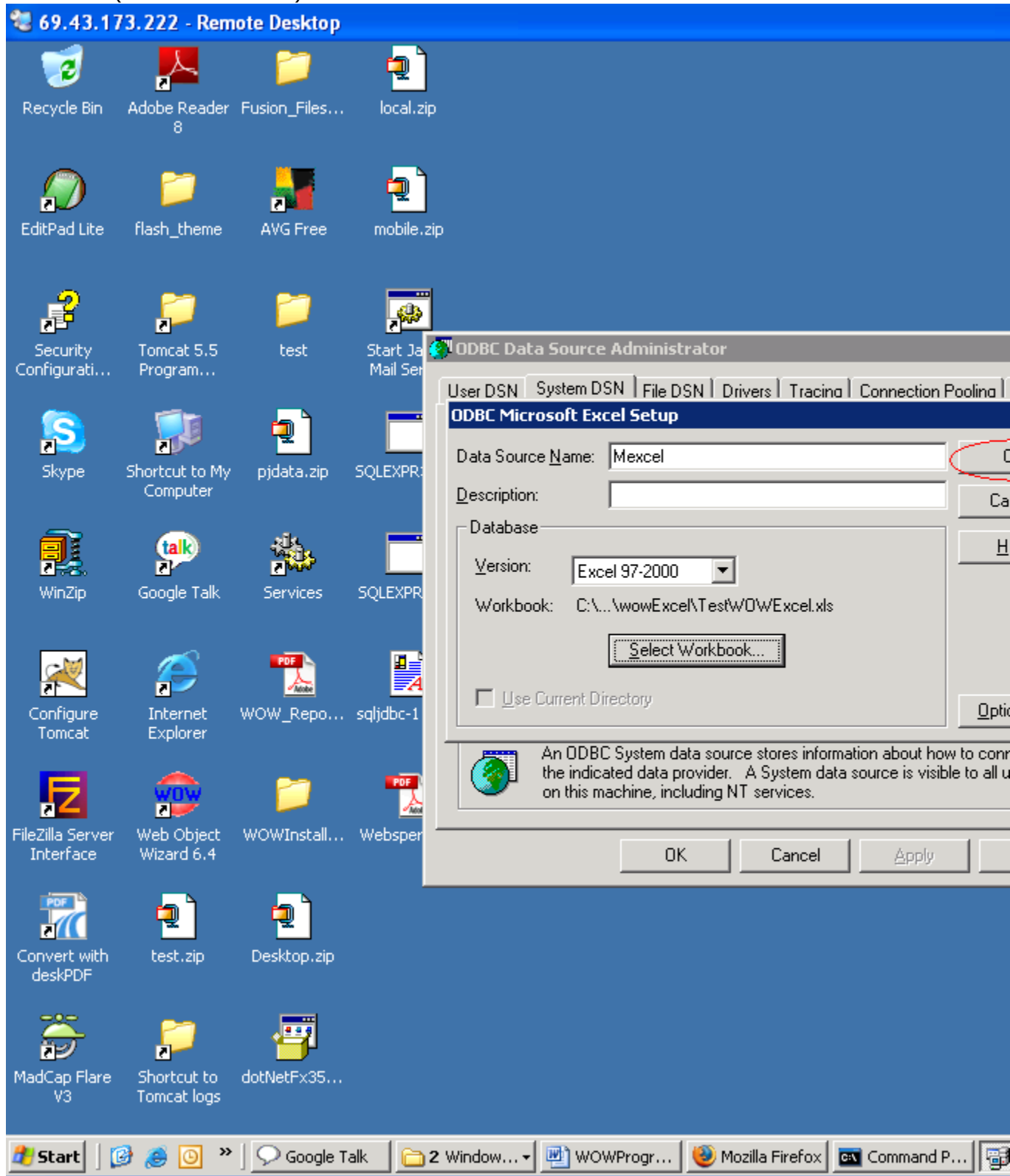
NOTE: There may be connection problems if trying to connect to a version under Microsoft Access/Excel 2000.

2. Point the DSN to the Excel file location (My file in C:\planetj\wowExcel\testWOWExcel.xls) then click "OK."

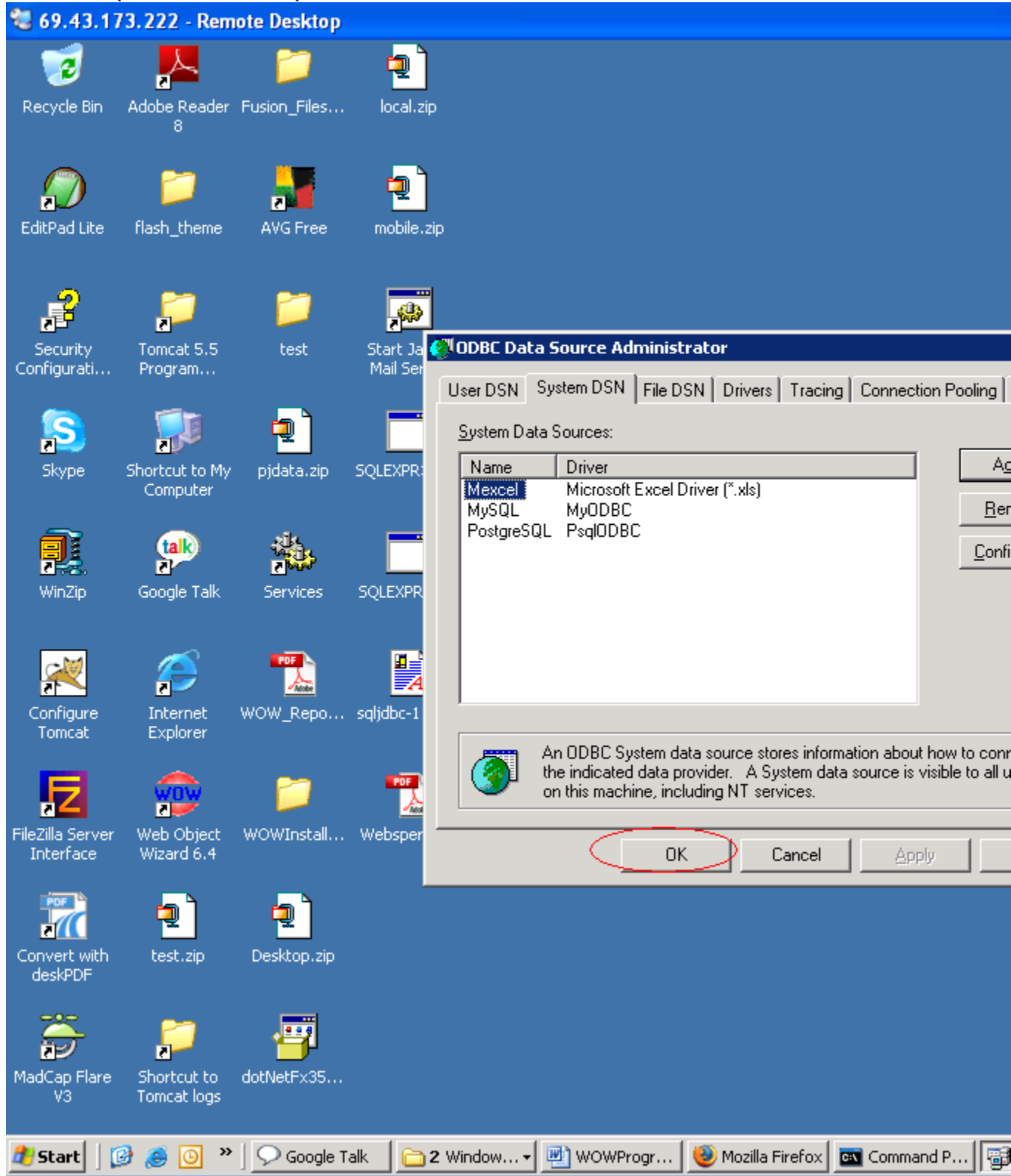


NOTE: Only Uncheck the "Read Only" box (screenshot above) when data from excel spreadsheet needs to be updated.

3. Click "OK" (screenshot below)



4. Click "OK" (screenshot below)



Connecting WOW to created DSN (screenshot below)

- **JDBC Driver:** MS Access/Excel (ODBC)
- **The IP Address:** where DSN is located (in example it is "localhost")
- **dsn:** Mexcel (this name was created earlier in step 1 of Pointing to the desired Excel worksheet)

The sample connection string:

- **JDBC Driver:** MS Access/Excel (ODBC)
- **URL:** jdbc:odbc:localhost;dsn=Mexcel;

Web Object Wizard

Hide Side Steps Development Tools

First Time User?

Check out this quick video to get started building a simple application!



Web Application Creation Steps:

1

Setup Connection(s)

2

Setup Application(s)

3

Setup Operation(s)

Connection Spec

JDBC Driver*

MS Access/Excel (ODBC)

URL*

jdbc:odbc:localhost;dsn=Mexcel;

Properties

User ID

pjsuper

Options

View Advanced Settings



Done

Syntax of SQL select, update statement

The screenshot below is an excel file that is accessed using WOW:

Recycle Bin Adobe Reader Fusion_Files... local.zip

EditPad Lite flash_theme

Security Configurati... Tomcat 5.5 Program...

Skype Shortcut to M... Computer

WinZip Google Talk

Configure Tomcat Internet Explorer

FileZilla Server Interface Web Object Wizard 6.4

Convert with deskPDF test.zip Desktop.zip

MadCap Flare V3 Shortcut to Tomcat logs dotNetFx35...

Microsoft Excel - TestWOWExcel.xls

File Edit View Insert Format Tools Data Window Help

Arial 10 B I U

A7 6

	A	B	C	D	E	F	G	H	I
	UserID	FirstName	LastName	Workdept	Age	Salary	Bonus	Job	Sex
1	1	John	Smith	A01	21	27000	9600	clerk	M
2	2	Paul	Thomas	A01	23	33000	9600	clerk	M
3	3	William	James	A02	25	31000	8800	sales	M
4	4	Philip	Smith	A03	35	105000	10800	Director	M
5	5	Betty	Silva	A03	40	55000	10800	Manager	F
6	6	Kate	Cnote	A02	18	51000	8800	sales	F
7									
8									
9									
10									
11									
12									
13									
14									
15									
16									
17									
18									
19									
20									
21									
22									
23									
24									

Sheet1 Sheet2 Sheet3

Ready

Start Google Talk 3 Window... WOWProgr... Mozilla Firefox Command P... Micros

SQL select statement syntax

Inside "testWOWExcel.xls" file, there are 3 excel worksheets by default which are sheet1, sheet2 and sheet3. In this example, the data is on sheet1.

NOTE: The use of [] around the worksheet name and the trailing \$ is a MUST while the use of [] around fields such as UserID, Firstname, etc is optional. This may be required on many systems.

Basic SQL Queries Using the SELECT Statement

```
Select * from [Sheet1$]
```

Other Queries Using the SELECT Statement

SQL allows you to select specific columns from a table.

```
SELECT Firstname, Lastname, workdept , salary FROM [Sheet1$]
```

Using a WHERE clause with the SELECT statement

```
SELECT Firstname, Lastname, workdept , salary FROM [Sheet1$] where  
FirstName like ?
```

or

```
SELECT * FROM [Sheet1$] where workdept like ?
```

NOTE: In a WOW application, the select statement written with "?" allows the end user to be prompted to generate results. The SQL statement including a "like" statement allows the end user to gather their information with a fuzzy search – or only needing to enter a portion of the query – i.e. last name/first name.

SQL update statement syntax

Basic SQL Queries Using the UPDATE Command

```
UPDATE [Sheet1$] SET [Bonus] = [Bonus] + '200'
```

Using a WHERE clause with the UPDATE statement

```
UPDATE [Sheet1$] SET [Bonus] = [Bonus] + '200' WHERE WORKDEPT =?
```

or

```
UPDATE [Sheet1$] SET [SALARY]= [SALARY] + 1000 WHERE WORKDEPT=?
```

Creating Reports and Graphs with WOW and Excel

Since Excel is widely used for data analysis and charting, we added the capability to access data from your Database and then chart that data into custom graphs and charts in Microsoft's Excel program. This is a very powerful feature because it allows you to customize the graph and charts a great deal and still bring in the data from Database.

We also have the feature to use one of WOW's Excel templates to create and run excel charts from data returned in an operation. You can get your data into excel spreadsheets and have it update dynamically by using the web query option in Excel. This means you have to set up your own Excel file.

WOW Setup for Excel Web Query

First, make sure that you have created an application and have an operation that selects from the database some information that you would like to graph. In this case, we have selected from the `pjdata.employee` table the number of employees in each department. The way we do this is select all of the departments and then sum up all the records of employees and which department they are a part of:

```
select deptno, sum(deptno) as employees_in_department from pjdata.employee
```

'employees_in_department' is a derived field meaning it is created when the select statement is run and is not actually inserted into the database.

Creating and Updating Excel Tables from WOW Web Data

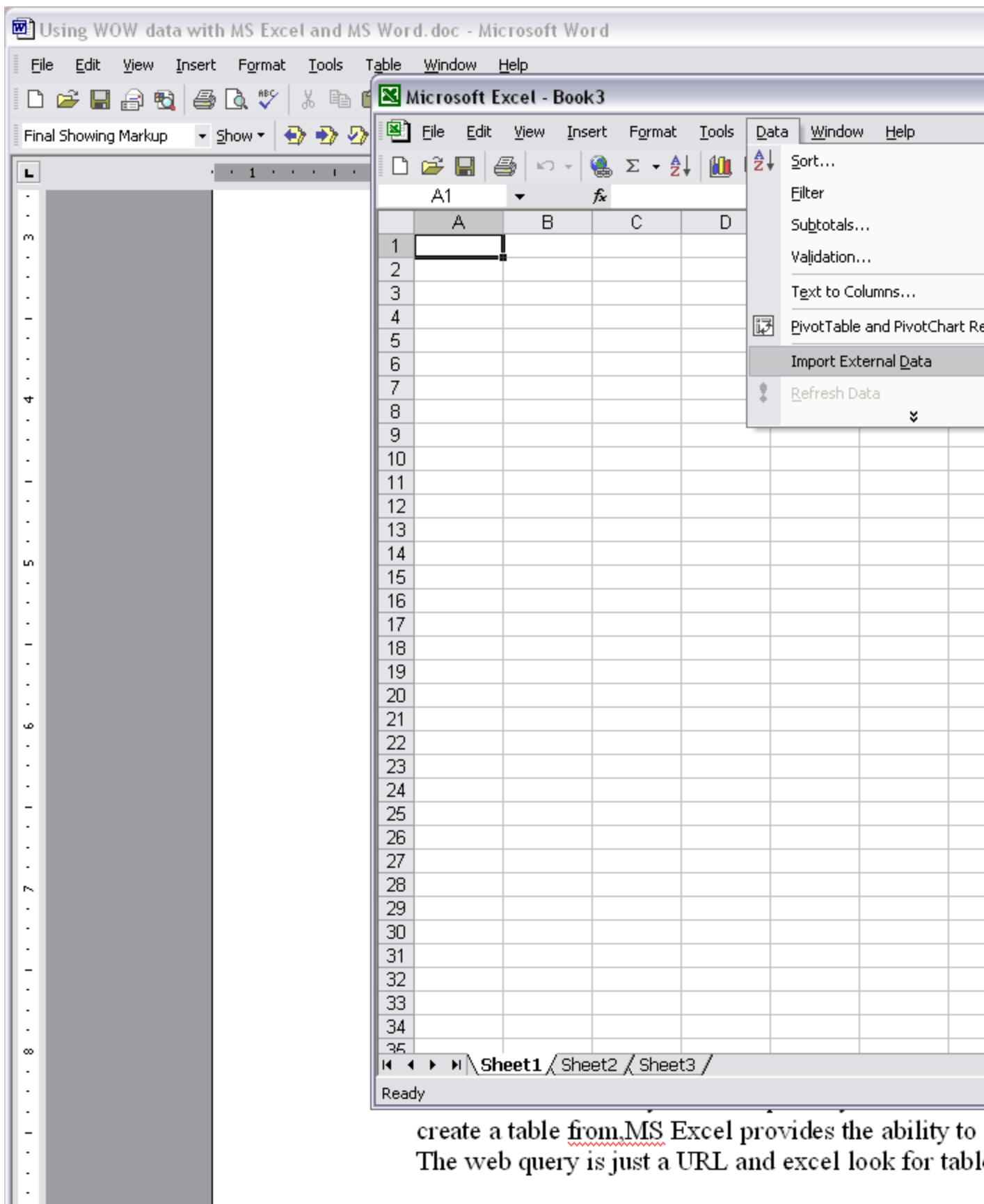
WOW excels at pulling information from any database and presenting it in a browser. This provides powerful processing but at times you may want to have an EXCEL spreadsheet setup with graphs and other formatting attributes and then merge database data into it at runtime. The feature described below, gives you that ability.

http://www.planetjavainc.com/wow/runApp - Microsoft Internet Explorer											
File Edit View Insert Format Tools Data Go To Favorites Help											
Back Forward Stop Home Search Favorites Media Print Mail Calendar Tasks Internet Options Help											
Address http://www.planetjavainc.com/wow/runApp											
K14 9/11/2002											
	A	B	C	D	E	F	G	H	I	J	K
1	EMPNO	FIRSTNAME	MIDINIT	LASTNAME	WORKDEPT	PHONE	HIREDATE	JOB	EDLEVEL	SEX	BIRTHDAY
2	10	Bob	T	TstLN	A00	3978	1/1/1965	PRES	16	M	
3	20	MICHAEL	L	THOMPSON	B01	3476	5/14/2003	MANAGER	16	M	2/2/19
4	30	SALLY	A	KWAN	C01	4738	4/5/1975	MANAGER	18	F	5/11/19
5	50	John	B	GEYER	C01	6789	8/17/1949	MANAGER	25	M	
6	60	IRVING	F	STERN	D11	6423	9/14/1973	MANAGER	25	M	7/7/19
7	70	EVA	D	PULASKI	D21	7831	9/30/1980	MANAGER	25	F	5/26/19
8	90	EILEEN	W	HENDERSON	E21	5498	8/15/1970	MANAGER	25	F	5/15/19
9	100	Paul	Q	SPENSER	F22	972	6/19/1980	MANAGER	14	M	#####
10	110	VICENZO	G	LUCCHESI	G22	3490	5/16/1958	SALESREP	10	M	
11	120	SEAN	r	O'CONNELL	H22	2167	12/5/1963	CLERK	14	M	#####
12	130	DELORES	M	QUINTANA	D11	4578	7/28/1971	OPERATOR	25	F	
13	140	HEATHER	A	NICHOLLS	I22	1793	#####	ANALYST	26	F	1/19/19
14	150	Gerald	e	ADAMSON	J22	4510	2/12/1972	DESIGNER	25	M	9/11/20
15	160	ELIZABET	R	PIANKA	R55	3782	#####	DESIGNER	26	F	4/12/19
16	152	Gerald	e	BOB	J22	4510	2/12/1972	DESIGNER	25	M	9/11/20
17	190	JAMES	H	WALKER	B01	2986	7/26/1974	DESIGNER	25	M	6/25/19
18	200	DAVID	K	BROWN	L22	4501	3/3/1966	DESIGNER	25	M	5/29/19
19	210	WILLIAM	T	JONES	F22	942	4/11/1979	DESIGNER	26	M	2/23/19
20	220	JENNIFER	K	LUTZ	I22	672	8/29/1968	DESIGNER	16	F	3/19/19
21	230	JAMES	J	JEFFERS	R55	2094	#####	CLERK	14	M	
22	240	SALVATO	M	MARINO	H22	3780	12/5/1979	CLERK	26	M	3/31/19
23	250	DANIEL	S	SMITH	L22	961	#####	CLERK	24	M	
24	260	SYBIL	P	JOHNSON	E11	8953	9/11/1975	CLERK	25	F	
25	270	MARIA	L	PEREZ	E11	9001	9/30/1980	CLERK	24	F	5/26/19
26	454545	John	Q	Public	C01	566	5/14/2003	CLERK	14	M	5/13/20
27	290	JOHN	R	PARKER	E11	4502	5/30/1980	OPERATOR	12	M	7/9/19
28	300	PHILIP	X	SMITH	E11	2095	6/19/1972	OPERATOR	14	M	
29	310	MAUDE	F	SETRIGHT	G22	3332	9/12/1964	OPERATOR	12	F	
30	320	RAMLAL	V	MEHTA	E21	9990	7/7/1965	FILEREP	25	M	
31	330	WING	J	LEE	G22	7878	2/23/1976	MANAGER	14	M	7/18/19
32	340	JASON	R	GOUNOT	R44	5698	5/5/1947	FILEREP	25	M	
33	200010	DIANE	J	HEMMING	B01	3978	1/1/1965	SALESREP	10	F	
34	200120	GREG	o	ORLANDO	F22	2167	5/5/1972	CLERK	14	M	#####
35	200140	KIM	N	NATZ	C01	1793	#####	ANALYST	10	F	1/19/19
36	200170	KIYOSHI	D	YAMAMOTO	I22	2890	9/15/1978	DESIGNER	25	M	1/5/19
37	200220	REBA	K	JOHN	B01	672	8/29/1968	DESIGNER	10	F	3/19/19
38	200240	ROBERT	M	MONTEVE	H22	3780	12/5/1979	CLERK	26	M	3/31/19
39	200280	EILEEN	R	SCHWARTZ	R55	8997	3/24/1967	OPERATOR	26	F	
40	200310	MICHELLE	F	SPRINGE	E21	3332	9/12/1964	OPERATOR	12	F	
41	200330	HELENA	g	WONG	D21	2103	2/23/1976	FIELDREP	14	F	7/18/19
42	100765	Peter	M	Potter	B01	7878	11/5/2002	Mgmt	5	M	11/6/20

Steps to create a web query

1. Use of this feature requires Microsoft Excel. Open a new Excel spreadsheet or a preexisting spreadsheet that you would like to use for graphing. With Excel opened as below and a WOW application running with your target data showing, you are ready to begin the web query process. Excel provides the ability to perform a "web query" to get data. The web query is just a URL call that returns data in an HTML format. Excel allows you to specify an HTML table to use as a data source. In Excel go to Data -> Import External Data -> New Web Query to get data from a WOW HTML table.

2.



create a table from MS Excel provides the ability to
The web query is just a URL and excel look for table

This enables Excel users to code a web query that invokes a WOW operation and puts the results into a "PREBUILT" Excel worksheet. Data can be refreshed on demand. Before creating the Web Query, there is some information that needs to be noted and changed about the application for it to be properly shown in Excel. The application like shown below has many queries

EmployeeTest - Welcome - Microsoft Internet Explorer

File Edit View Favorites Tools Help

Back Forward Stop Reload Home Search Favorites Media

Address http://www.planetjavainc.com/wow/runApp

Search for Search

EmployeeTest

Welcome

Default

- Employees
- departments
- add employees
- update bonus
- Update Salary by Dept No

Sample Employees

	▲ EMPNO ▼	▲ First Name ▼	▲ MIDINIT ▼
	000010	Bob	T
	000020	MICHAEL	L
	000030	SALLY	A
	000050	John	B
	000060	IRVING	F
	000070	EVA	D
	000090	EILEEN	W
	000100	Paul	Q
	000110	VICENZO	G
	000120	SEAN	r
	000130	DELORES	M
	000140	HEATHER	A
	000150	Gerald	e
	000160	ELIZABETH	R
	000152	Gerald	e
	000190	JAMES	H
	000200	DAVID	K
	000210	WILLIAM	T
	000220	JENNIFER	K

3. A common problem is that when the table data is imported or refreshed, copy and delete icons will show up on the excel spreadsheet, which may be undesirable. Simply adjust the operation to disable those features. Refer to the appropriate areas

of this manual to adjust these features.

http://www.planetjavainc.com/wow/WOWBuilder - Microsoft Internet Explorer

File Edit View Favorites Tools Help



Address http://www.planetjavainc.com/wow/WOWBuilder

Search for Search

PlanetJ

Web Object Wizard 5

WOW 5.06 ENTERPRISE EDITION



Applications

[View Applications](#)

Current Application

[Create Operation](#)

[Edit Operation](#)

[Delete Operation](#)

[Preview Application](#)

Connections

[Create Connection](#)

[View Connections](#)

Development Tools

[WOW Utilities](#)

[Sign Off](#)



Web Application Creation Steps

- 1 Setup Connection(s)
- 2 Setup Application(s)
- 3 Setup Operation(s)
- 4

Operations for EmployeeTest



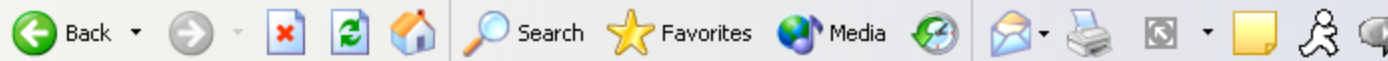
	▲ Label ▼	▲ Type ▼	▲ Descrip
	Employees	SQL	View a samp
	dartments	SQL	View a samp
	add employees	SQL	
	update bonus	SQL	View a samp
	Update Salary by Dept No	SQL	View a samp
	WORKDEPTASSOC	Association 1-MANY	View a samp

In the display menu for the Operation you may want to disable the options for Details, Inserts, Deletes, and Updates so they won't be shown on the table. You may also remove the "copy row" feature in the display properties by setting it to false as shown below. Now that all the extra options are turned off, make sure you write down the Operation ID located in the Internal section of properties. Excel will u

4.

http://www.planetjavainc.com/wow/WOWBuilder - Microsoft Internet Explorer

File Edit View Favorites Tools Help



Address http://www.planetjavainc.com/wow/WOWBuilder

Search for Search

Connections

Create Connection

View Connections

Development Tools

WOW Utilities

Sign Off



Label*:	<input type="text" value="ViewEmployees"/>	Title:	<input type="text" value="Sample Employees"/>
Type*:	<input type="text" value="SQL"/>	Description:	<input type="text" value="View a sample databa"/>
Operation Code:	<input type="text" value="SELECT * FROM pjdata.employee"/>		
Instructions:	<input type="text"/>		

Display

Allow Details:	<input type="checkbox"/>	Display Group:	<input type="text" value="Default"/>
Allow Inserts:	<input type="checkbox"/>	Display Order:	<input type="text" value="0"/>
Allow Updates:	<input type="checkbox"/>	Display Columns:	<input type="text"/>
Allow Deletes:	<input type="checkbox"/>		
Properties:	<input type="text" value="selectionType:single; refresh:true; chart:true; excel:true; msWord:true; xml:true; editFD:false; print:true; sorting:true; drawGrid:true; rowCopy:false; updateable:false; deleteAll:false; nextPrevious:true; }"/>		

Advanced

Connection Alias:	<input type="text" value="-- None --"/>	Row Count:	<input type="text" value="50"/>
Row Coll. Class:	<input type="text"/>	Row Class:	<input type="text"/>
Usage Id:	<input type="text"/>	Caching Level*:	<input type="text" value="Check cache and cache"/>
JSP File:	<input type="text"/>	Details JSP:	<input type="text"/>

Administration

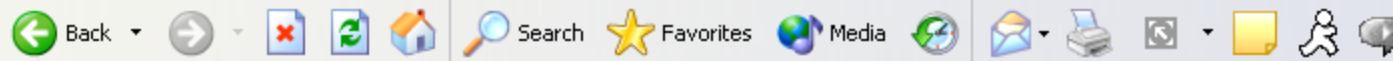
Security Type:	<input type="text" value="-- None --"/>	Security Level*:	<input type="text" value="0"/>
Auto Run Op.:	<input type="text" value="-- None --"/>		

If you only want certain columns of a table to display in Excel you could write those columns in the Display Columns field with commas between columns. Then only those columns would show in WOW and be updated in Excel. After changing all e

5.

http://www.planetjavainc.com/wow/WOWBuilder?Next=/dataengine/applicationbuilder/jsp/wab_view_ap - Mi

File Edit View Favorites Tools Help



Address  http://www.planetjavainc.com/wow/WOWBuilder?Next=/dataengine/applicationbuilder/jsp/wab_view_apps.jsp&PJ_SESSIC

Search for  Search



Applications

Create Application
View Applications
Edit Application
Delete Application
Work with Operations

Connections

Create Connection
View Connections

Development Tools

WOW Utilities

Sign Off



Web Application Creation Steps

1 Setup Connection(s) 2 Setup Application(s) 3 Setup Operation(s) 4 Run

Applications



	Name	Description	Connection	ID
   	EmployeeTest		COLINSDB2	286

Insert

The name of the Application

Click on the name of the application to run it in a new browser window, then copy the generated URL or write it down. This is the URL that will be used by Excel Web

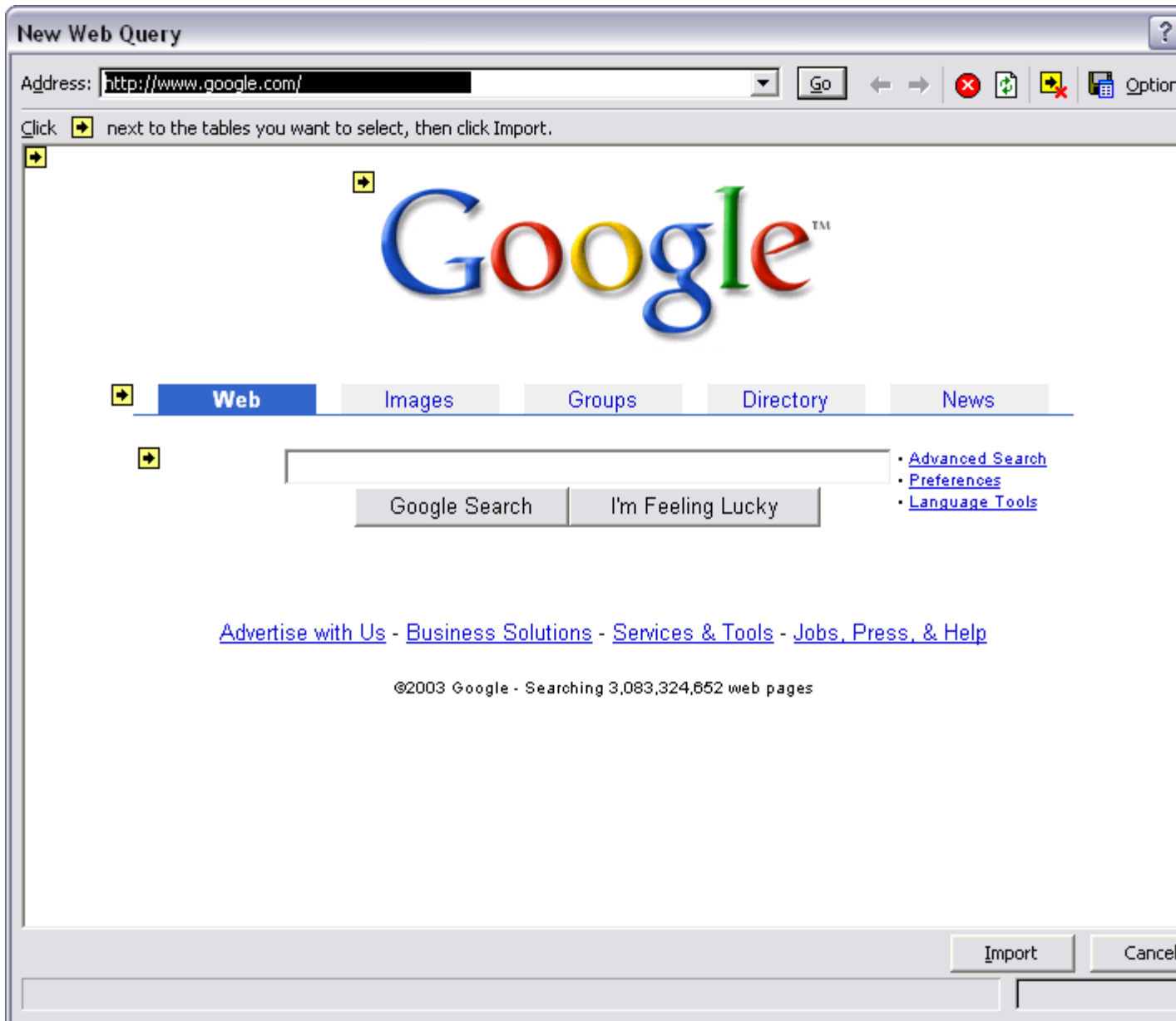
ery. It should look similar to this: <http://www.planetjavainc.com/wow/>

nApp?id=286 Now we are ready to go through the Excel import external data process.

Setting up a New Web Query in Excel 2002

6. After opening the Web Query dialog box shown below, you will see a browser window.

7.



This is the web page that Excel is going to look for to open up HTML tables and update the Excel tables. Now, either type in the URL of the application or use copy and paste.

This will open up the application but without the operation or table that you want. Now append the Operation ID recorded earlier to the URL.

8. The web query should now be of the following form:
www.planetjavainc.com/wow/runApp?id=xx&opid=yy. xx is the id of the application while yy is the id of the operation. For this example we use <http://www.planetjavainc.com/wow/runApp?id=286&lvid=1242>. Now press enter and the operation should show up in the web query screen. There will be little yellow arrows around the page representing the location of various html tables. Select the one containing your desired data. A green checkmark will appear to indicate you have selected a particular table. Press import to proceed.

9.

New Web Query

Address: <http://www.planetjavainc.com/wow/runApp?id=286&lvid=1242>

Click  next to the tables you want to select, then click Import.



































EmployeeTest

Results






























Sample Employees

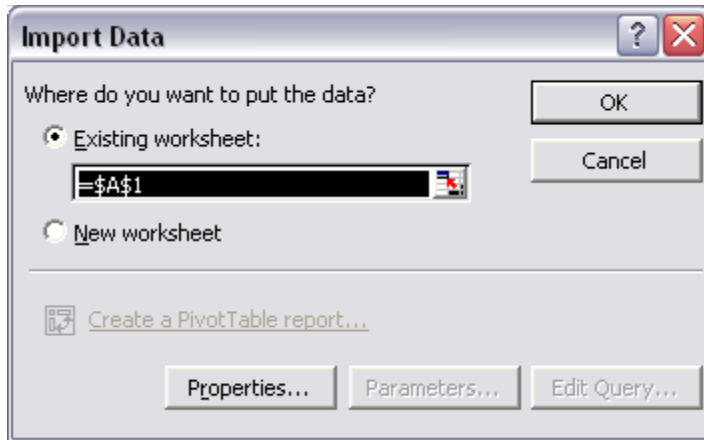







▲ EMPNO ▼	▲ First Name ▼	▲ MIDINIT ▼	▲ Last Name ▼
000010	Bob	T	TstLNM
000020	MICHAEL	L	THOMPSON
000030	SALLY	A	KWAN
000050	John	B	GEYER
000060	IRVING	F	STERN
000070	EVA	D	PULASKI
000090	EILEEN	W	HENDERSON
000100	Paul	Q	SPENSER
000110	VICENZO	G	LUCCHESI
000120	SEAN	r	O'CONNELL
000130	DELORES	M	QUINTANA
000140	HEATHER	A	NICHOLLS
000150	Gerald	e	ADAMSON
000160	ELIZABETH	R	PIANKA

Excel will then return an import data screen with the option to either create a new worksheet with the data from WOW or put it in current worksheet. Pick the option suitable for your situation.



After selecting the data location, Excel will momentarily say "getting data" and then display the WOW table information in an Excel table like shown.

Microsoft Excel - Book1

File Edit View Insert Format Tools Data Window Help

100% Arial

	A	B	C	D	E	F	G	H	
1	EMPNO	First Name	MIDINIT	Last Name	WORKDEPT	PHONENO	HIREDATE	JOB	
2	10	Bob	T	TstLNM	A00	3978	1/1/1965	PRES	
3	20	MICHAEL	L	THOMPSON	B01	3476	5/14/2003	MANAGER	
4	30	SALLY	A	KWAN	C01	4738	4/5/1975	MANAGER	
5	50	John	B	GEYER	C01	6789	8/17/1949	MANAGER	
6	60	IRVING	F	STERN	D11	6423	9/14/1973	MANAGER	
7	70	EVA	D	PULASKI	D21	7831	9/30/1980	MANAGER	
8	90	EILEEN	W	HENDERSON	E21	5498	8/15/1970	MANAGER	
9	100	Paul	Q	SPENSER	F22	972	6/19/1980	MANAGER	
10	110	VICENZO	G	LUCCHESSI	G22	3490	5/16/1958	SALESREP	
11	120	SEAN	r	O'CONNELL	H22	2167	12/5/1963	CLERK	
12	130	DELORES	M	QUINTANA	D11	4578	7/28/1971	OPERATOR	
13	140	HEATHER	A	NICHOLLS	I22	1793	12/15/1976	ANALYST	
14	150	Gerald	e	ADAMSON	J22	4510	2/12/1972	DESIGNE	
15	160	ELIZABETH	R	PIANKA	R55	3782	10/11/1977	DESIGNE	
16	152	Gerald	e	BOB	J22	4510	2/12/1972	DESIGNE	
17	190	JAMES	H	WALKER	B01	2986	7/26/1974	DESIGNE	
18	200	DAVID	K	BROWN	L22	4501	3/3/1966	DESIGNE	
19	210	WILLIAM	T	JONES	F22	942	4/11/1979	DESIGNE	
20	220	JENNIFER	K	LUTZ	I22	672	8/29/1968	DESIGNE	
21	230	JAMES	J	JEFFERSON	R55	2094	11/21/1966	CLERK	
22	240	SALVATORE	M	MARINO	H22	3780	12/5/1979	CLERK	
23	250	DANIEL	S	SMITH	L22	961	10/30/1969	CLERK	
24	260	SYBIL	P	JOHNSON	E11	8953	9/11/1975	CLERK	
25	270	MARIA	L	PEREZ	E11	9001	9/30/1980	CLERK	
26	454545	John	Q	Public	C01	566	5/14/2003	CLERK	
27	290	JOHN	R	PARKER	E11	4502	5/30/1980	OPERATO	
28	300	PHILIP	X	SMITH	E11	2095	6/19/1972	OPERATO	
29	310	MAUDE	F	SETRIGHT	G22	3332	9/12/1964	OPERATO	
30	320	RAMLAL	V	MEHTA	E21	9990	7/7/1965	FILEREP	
31	330	WING	J	LEE	G22	7878	2/23/1976	MANAGE	
32	340	JASON	R	GOUNOT	R44	5698	5/5/1947	FILEREP	
33	200010	DIANE	J	HEMMINGER	B01	3978	1/1/1965	SALESREP	
34	200120	GREG	o	ORLANDO	F22	2167	5/5/1972	CLERK	
35	200140	KIM	N	NATZ	C01	1793	12/15/1976	ANALYST	
36	200170	KIYOSHI	D	YAMAMOTO	I22	2890	9/15/1978	DESIGNER	
37	200220	REBA	K	JOHN	B01	672	8/29/1968	DESIGNER	
38	200240	ROBERT	M	MONTEVERDE	H22	3780	12/5/1979	CLERK	
39	200280	EILEEN	R	SCHWARTZ	R55	8997	3/24/1967	OPERATOR	
40	200310	MICHELLE	F	SPRINGER	E21	3332	9/12/1964	OPERATOR	
41	200330	HELENA	g	WONG	D21	2103	2/23/1976	FIELDREP	
42	100765	Peter	M	Potter	B01	7878	11/5/2002	Mgmt	
43	102579	Lewis	S	Libman	G22	2118	12/6/1999	MIC	
44	654	Tony	W	Tiger	H22	2399	5/28/2003	CLERK	

Setting up a New Web Query in Excel 2000 or earlier

6. Setting up the Web Query is a little different in earlier versions of Excel. Below is the New Web Query screen. Option 1 is where you enter the application and operation's URL. After setting the URL you need to set the second option to use only specific HTML tables. Use the table identifier "WOW" which is the standard HTML table id that WOW will generate for data tables. Then after setting the third option to none, the web query should be ready to run.

New Web Query

1. Enter the address for the Web page that contains the data you want. If browsing, switch back to Excel once you have located the Web page in your browser.

2. Choose the part of the Web page that contains the data you want. Note that pre-formatted sections are treated as tables.

☐ The entire page
☐ Only the tables
☒ One or more specific tables on the page.

Enter table name(s) or number(s) separated by commas:


3. Choose how much formatting from the Web page you want your data to keep:

☒ None
☐ Rich text formatting only
☐ Full HTML formatting


7. You will get the import data screen asking where you want to put the table data from WOW. Select a location and click OK.

Import Data

Where do you want to put the data?

☒ Existing worksheet:
 

☐ New worksheet

 [Create a PivotTable report...](#)

8. After selecting where to put data, Excel should say "getting data" and display the WOW data in an Excel table as shown below.

9.

Microsoft Excel - Book1

File Edit View Insert Format Tools Data Window Help

A1 fx

	A	B	C	D	E	F	G	H	I
1	EMPNO	First Name	MIDINIT	Last Name	WORKDEPT	PHONENO	HIREDATE	JOB	EDLE
2	10	Bob	T	TstLNM	A00	3978	1/1/1965	PRES	
3	20	MICHAEL	L	THOMPSON	B01	3476	5/14/2003	MANAGER	
4	30	SALLY	A	KWAN	C01	4738	4/5/1975	MANAGER	
5	50	John	B	GEYER	C01	6789	8/17/1949	MANAGER	
6	60	IRVING	F	STERN	D11	6423	9/14/1973	MANAGER	
7	70	EVA	D	PULASKI	D21	7831	9/30/1980	MANAGER	
8	90	EILEEN	W	HENDERSON	E21	5498	8/15/1970	MANAGER	
9	100	Paul	Q	SPENSER	F22	972	6/19/1980	MANAGER	
10	110	VICENZO	G	LUCCHESSI	G22	3490	5/16/1958	SALESREP	
11	120	SEAN	r	O'CONNELL	H22	2167	12/5/1963	CLERK	
12	130	DELORES	M	QUINTANA	D11	4578	7/28/1971	OPERATOR	
13	140	HEATHER	A	NICHOLLS	I22	1793	12/15/1976	ANALYST	
14	150	Gerald	e	ADAMSON	J22	4510	2/12/1972	DESIGNER	
15	160	ELIZABETH	R	PIANKA	R55	3782	10/11/1977	DESIGNER	
16	152	Gerald	e	BOB	J22	4510	2/12/1972	DESIGNER	
17	190	JAMES	H	WALKER	B01	2986	7/26/1974	DESIGNER	
18	200	DAVID	K	BROWN	L22	4501	3/3/1966	DESIGNER	
19	210	WILLIAM	T	JONES	F22	942	4/11/1979	DESIGNER	
20	220	JENNIFER	K	LUTZ	I22	672	8/29/1968	DESIGNER	
21	230	JAMES	J	JEFFERSON	R55	2094	11/21/1966	CLERK	
22	240	SALVATORE	M	MARINO	H22	3780	12/5/1979	CLERK	
23	250	DANIEL	S	SMITH	L22	961	10/30/1969	CLERK	
24	260	SYBIL	P	JOHNSON	E11	8953	9/11/1975	CLERK	
25	270	MARIA	L	PEREZ	E11	9001	9/30/1980	CLERK	
26	454545	John	Q	Public	C01	566	5/14/2003	CLERK	
27	290	JOHN	R	PARKER	E11	4502	5/30/1980	OPERATOR	
28	300	PHILIP	X	SMITH	E11	2095	6/19/1972	OPERATOR	
29	310	MAUDE	F	SETRIGHT	G22	3332	9/12/1964	OPERATOR	
30	320	RAMLAL	V	MEHTA	E21	9990	7/7/1965	FILEREP	
31	330	WING	J	LEE	G22	7878	2/23/1976	MANAGER	
32	340	JASON	R	GOUNOT	R44	5698	5/5/1947	FILEREP	
33	200010	DIANE	J	HEMMINGER	B01	3978	1/1/1965	SALESREP	
34	200120	GREG	o	ORLANDO	F22	2167	5/5/1972	CLERK	
35	200140	KIM	N	NAT7	C01	1793	12/15/1976	ANALYST	

Sheet1 Sheet2 Sheet3

Ready

After the table has been created, you can right click anywhere on the data for various web query options (refresh values from web, edit query information, or edit range options).

Integrating WOW with Existing Excel Files

WOW provides the ability to generate a new spreadsheet from any existing data. It may be desirable to have a preformatted spreadsheet, which contains titles, business charts, etc and have WOW update the data dynamically from your database. This is possible with new support in WOW 7.0. A spreadsheet is placed on the web server at specific location. WOW can be configured to read in the spreadsheet, update the data, and send the merged Excel file to the users browser.

Setting WOW Operations to use Existing Excel Templates

An application and operation must be created that provides access to your data. For example, the following operation does a simple select of states and the balance due by each state. Next, the operation needs to have some properties set so that it knows which Excel file to use. Shown below is the operation with the SQL that returns the states and balances due for each state.

PlanetJ

Web Object

WOW 6.3.5 ENTER



Applications

View Applications

Connections

Create Connection

View Connections

Development Tools

WOW Utilities

Sign Off



Fields marked with an asterisk (*) are required.

Previous

Update and Previous

Update

Basic

Label*

Balance Due By State

Title

Operation Type*

SQL

Description

SELECT state, sum(baldue) as baldue FROM qiws.qcustcdt where state between ? and ? group by state order by state asc

Operation Code

Instructions

For this example, use states MN and WY. Click on the state in the table to drill down into state specific data. Click on the chart icon to chart the report! This dynamically updates a pre created MS EXCEL spreadsheet with runtime data. Upon open a MS EXCEL macro updates the graph to display real time graph results. Once you have EXCEL spreadsheet, click on the tab at the bottom of the page to display the graph.<hr/>

Output Connection Alias

Display

After the operation is open for edit there are some properties that need to be added. In the Properties field of the operation there are property groups like DisplayColumns, DetailDisplay and TableDisplay. These are the standard property groups but there may be others. In the group, TableDisplay, add the following property: `excelXls:true;`

NOTE: Properties and PropertyGroups **ARE** case-sensitive.

This property tells the operation to display the chart icon when the operation runs.

View Connections

Development Tools

WOW Utilities

Sign Off



Operation Code

SELECT state, sum(baldue) as baldue FROM qiws.qcustcdt
where state between ? and ? group by state order by state asc

Instructions

For this example, use states MN and WY. Click on the state in the table to drill down into state specific data. Click on the chart icon to chart the report! This dynamically updates a pre created MS EXCEL spreadsheet with runtime data. Upon open a MS EXCEL macro updates the graph to display real time graph results. Once you have EXCEL spreadsheet, click on the tab at the bottom of the page to display the graph.<hr></hr>

Output Connection Alias

Display

Allow Details	<input checked="" type="checkbox"/>	Display Group
Allow Inserts	<input checked="" type="checkbox"/>	Display Order
Allow Updates	<input checked="" type="checkbox"/>	Display Columns
Allow Deletes	<input type="checkbox"/>	

Properties

```

}
TableDisplay{
  selectionType:single; refresh:true;      chart:true;
  excel:true;           msWord:true;      xml:true;
  editFD:false;         print:true;       sorting:true;
  drawGrid:true;        rowCopy:true;     updateable:false;
  deleteAll:false;      nextPrevious:true;
  excelXls:true;
}
    
```

Now that WOW knows to show the Excel Chart icon, it needs to know which Excel file to open when the chart icon is pressed. The specifics of the property need to be set, such as what excel file to open, and what worksheet inside of that excel file to write the data to.

To set the specifics of the **excel/Xls** property we need to insert a Property Group called **XLS**, as shown below. XLS has three properties that need to be set, including:

```
XLS {  
fileName:excel_reports.xls;  
sheetIndex:2;  
directToFile:false;  
}
```


Instructions

you have EXCEL spreadsheet, click on the tab at the bottom of the page to display the graph.<hr></hr>

Output Connection Alias

Display

Allow Details	<input checked="" type="checkbox"/>	Display Group
Allow Inserts	<input checked="" type="checkbox"/>	Display Order
Allow Updates	<input checked="" type="checkbox"/>	Display Columns
Allow Deletes	<input type="checkbox"/>	

Properties

```
TableDisplay{
  selectionType:single; refresh:true;      chart:true;
  excel:true;           msWord:true;      xml:true;
  editFD:false;         print:true;       sorting:true;
  drawGrid:true;        rowCopy:true;     updateable:false;
  deleteAll:false;      nextPrevious:true;
  excelXls:true;
}
XLS{ fileName:excel_reports.xls; sheetIndex:2; directToFile:false; }
```


Advanced

Connection Alias	-- None --	Operation Class
Row Count	50	Row Coll. Class
Row Class		Parameters JSP
Caching Level*	Check cache and cache results	JSP File

filename - This is the Excel template file that will be opened when the chart icon is pressed.

sheetIndex - This is the worksheet in the Excel file where the data will be written.

directToFile - true: go directly to Excel chart when the operation runs. false: have to click on chart icon to open Excel chart.

Now that the operation display properties have been set, update the operation. After execution of the operation, a graph icon () shows up alongside the Microsoft Word and Excel icons.



WOW Reports WOW Range Reports WOW Excel Graphs WOW Drill Down


For this example, use states MN and WY. Click on the state in the table to drill down data. Click on the chart icon to chart the report! This dynamically updates a pre created spreadsheet with runtime data. Upon open a MS EXCEL macro updates the graph to current graph results. Once you have EXCEL spreadsheet, click on the tab at the bottom of the excel spreadsheet.

State BETWEEN AND

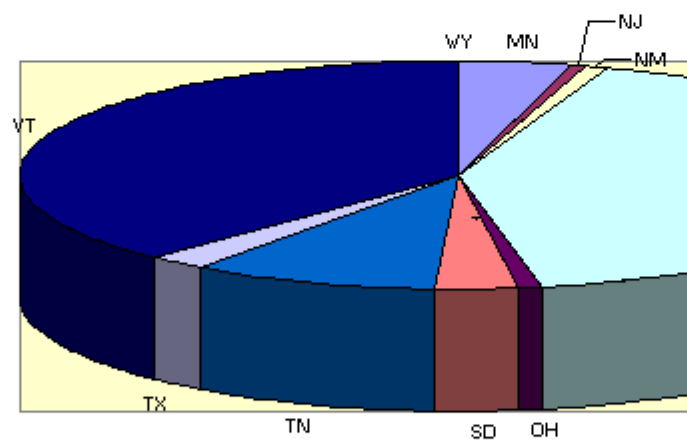
Balance Due By State Range



▲ State ▼	▲ Balance due ▼
MN	51.00
NJ	5.00
NM	10.00
NY	492.50
OH	10.00
SD	37.00
TN	109.00
TX	25.00
VT	-451.00
WY	0.00

After the data has been returned, click on the graph icon  to open the data in an Excel graph and/or spreadsheet. The data will be automatically imported and shown on an Excel chart, there is also a tab called data that has the returned results in table format.

PlanetJ Excel Integration



Microsoft Excel - excel_reports.xls

File Edit View Insert Format Tools Data FlashPaper Window Help

	A	B	C	D	E	F	G	H	I	J	K
1	MN	\$51.00									
2	NJ	\$5.00									
3	NM	\$10.00									
4	NY	\$492.50									
5	OH	\$10.00									
6	SD	\$37.00									
7	TN	\$109.00									
8	TX	\$25.00									
9	VT	-\$451.00									
10	WY	\$0.00									
11											
12											
13											
14											
15											
16											
17											
18											
19											
20											
21											
22											
23											
24											
25											
26											
27											
28											
29											
30											
31											
32											
33											
34											
35											
36											
37											
38											
39											

An Excel macro updates the chart after the data has been updated. Users may see a security warning that a macro is being run. They can safely grant permission.

NOTE: It is necessary to use a template provided by PlanetJ because of the included macros. However, a template can be changed to any graph or display and can be made to handle any number of columns or rows.

The PlanetJ template is located in the 'wowexcel' folder inside your application server. For example:

```
C:\Program Files\Apache Software Foundation\Tomcat 5\webapps\wow64\web-inf\wowexcel\
```

There are a few samples included that you can use as is or copy, rename, and change to make custom chart or reports. When using the XLS property filename just specify the file name and WOW will automatically look in the wowexcel folder.

Creating Reports from Data Imported from WOW into Excel

After creating an Excel spreadsheet, it is possible to create reports, order forms, etc using the power of Excel along with WOW. This WOW feature allows the full power of Excel to be combined with live database data. Sophisticated Excel applications can be created complete with macros, graphs, and pivot tables.

Restrictions

- The Excel macro requires that data be written to a spreadsheet called "data" and the chart be in a sheet called "chart". Experienced Excel developers may update the macro or create their own to meet specific needs. WOW professional services may also be used for customization needs.
- Browsers may handle downloads differently. Browser support and specific browser behavior are beyond WOW support and control.
- Users may be required to have MS Excel or the MS Excel viewer to view spreadsheets.

Utilizing Existing RPG Applications

To utilize existing legacy code, such as an RPG or Cobol programs, you'll need to create an external stored procedure for each program to be called. Once a procedure exists, WOW can call the procedure, which in turn calls the program. Creating a procedure registers the program so that SQL can locate it, as well as defining properties needed to correctly call the program, such as the program name, the parameters required, the language the program was written in, etc.

Calling an RPG Program That Returns a Result Set

Add Code to Return a Result Set

To have an RPG program return result sets, you'll need to add code similar to the following in your RPG program:

```
C*****
C* Opens the cursor for Stored Proc                                     *
C*****
*
C/EXEC SQL
C+ DECLARE C1 CURSOR FOR SELECT * FROM MYLIB/FILE1
C/END-EXEC
C/EXEC SQL
C+ Open C1
C/END-EXEC
C/EXEC SQL
C+ set result sets cursor c1
C/END-EXEC
```

In the example above, all of the data from file FILE1 in MYLIB will be returned to the calling procedure. When SQL runs a select statement, the resulting rows comprise the result table. A cursor (C1) provides a way to access a result table. It is used within a program to maintain a position in the result table. SQL uses a cursor to work with the rows in the result table and to make them available to your program.

Defining the Stored Procedure

Next you'll need to define an external stored procedure for the RPG program above:

```
CREATE PROCEDURE MYLIB.MYPROC ( )
    DYNAMIC RESULT SETS 1
    LANGUAGE RPGLE
    SPECIFIC MYLIB/MYPROC
    NOT DETERMINISTIC
    MODIFIES SQL DATA
    CALLED ON NULL INPUT
    EXTERNAL NAME 'MYLIB/RPGPGM'
    PARAMETER STYLE GENERAL;
```

In the example above, there are no parameters, you're returning one result set, the program language is RPGLE and the program name is RPGPGM in MYLIB. Run the SQL statement using one of the available SQL interfaces such as STRSQL, iSeries Navigator (Run SQL Scripts) or MySQL Query Browser.

AUTHORITY TIPS: Make sure the WOW user ID has proper authority to the program, as well as any files accessed by the program. You can grant authority to the program by running SQL similar to the following:

```
GRANT EXECUTE ON SPECIFIC PROCEDURE MYLIB.MYPROC TO WOW
```

NOTE: The above example assumes the WOW user ID is WOW.

Defining the WOW Operation

Defining an operation to call the external stored procedure is very similar to defining any other operation. The primary difference is what's specified for the "Operation Code". Instead of specifying an SELECT statement, you'll specify the procedure call:

```
CALL MYLIB.MYPROC()
```

The above example has no parameters in the procedure call. For more details on using stored procedures, including the use of field descriptors, see the chapter entitled Stored Procedures in the first section of the Builders Guide.

More Than One User Running the Operation at the Same Time

After the RPG program is called, the table MYLIB/FILE1 is still open and having more than one user call the same operation could result in an "In Use" error for the other users. One possible solution is to have the original RPG program only populate the result table. Then define a second "wrapper" program to call the first program, copy the table to QTEMP and then return the results set from the QTEMP version of the table. Below is an example of a wrapper program written in RPG Free:

```
HDFTACTGRP(*NO)
DReportPgm          Pr                      Extpgm('MYPGM')
*
D Run               Pr                      ExtPgm('QCMDEXC')
D Cmd               200A  CONST
D len               15P  5  CONST
*
/Free
```

//Create the file in QTEMP

```
Monitor;
  run('DLTF FILE(QTEMP/FILE1)':200);
On-Error;
EndMon;

//Create empty file in
QTEMP
run('CRTDUPOBJ OBJ(FILE1) FROMLIB(MYLIB) +
    OBJTYPE(*FILE) TOLIB(QTEMP)':200);
run('OVRDBF FILE(FILE1) TOFILE(QTEMP/FILE1)':200);
run('CLRPFM emppfbk':200);

//Call the program that populates FILE1.
// The override ensure that the QTEMP file is populated.

ReportPgm();
```

```

        run('DLTOVR
*ALL':200);

        //Open the cursor for Stored
Proc
        Exsr
Resultset;

        return;
/End-Free
C*****
C* Opens the cursor for Stored Proc *
C*****
C        Resultset        Begsr
*
C/EXEC SQL
C+ DECLARE C1 CURSOR FOR SELECT * FROM QTEMP/EMPPFBK
C/END-EXEC
C/EXEC SQL
C+ Open C1
C/END-EXEC
C/EXEC SQL
C+ set result sets cursor c1
C/END-EXEC
C        Resultset        endsr

```

Calling an RPG Program That Returns a MODS (Array) in RPG Free

Add Code to Return an Array

The following code shows a simple RPG Free program that receives an integer and loops through a customer master file (CSTMSTPF) for the number of times specified in the pRows parameter:

```

h dftactgrp(*no)
fcstmst1  if a e          k disk    prefix('CS.')
d cs          e ds          qualified extname(CSTMSTPF)
d CustRS          pr
d pRows          10i 0
d CustRS          pi
d pRows          10i 0
d CustList        ds          occurs(100)
d CSTMST          481a
d i          s          10i 0
/free
i = 0;
setll *Loval CstMst1;
read CstMst1;
dow not %eof(CstMst1);
    if i >= pRows;          leave;
        endif;i = i + 1;
        %occur(CustList) = i;
        CSTMST = cs;
        read CstMst1;enddo;
exsr setResult;

```

```

*inlr = *On;

begsr setResult;
/end-free
C/EXEC SQL
C+ SET RESULT SETS ARRAY:CustList FOR:I ROWS
C/END-EXEC
/free
endsr;

/end-free

```

Any file could have been used here and any number of parms could have been passed in to dictate the criteria for the end result set. The key is that each time a record is read, that entire record is added to a **multiple occurrence data structure (array CustList)**. Once the looping has completed, an SQL result set is created based on the **multiple occurrence data structure** and is sized based on the value in variable I, as shown in subroutine setResult.

Defining the Stored Procedure

Next you'll need to define an external stored procedure for the RPG program above:

```

CREATE PROCEDURE MYLIB.MYPROC (IN LOOPCT INTEGER)
  DYNAMIC RESULT SETS 1
  LANGUAGE RPGLE
  SPECIFIC MYLIB/MYPROC
  NOT DETERMINISTIC
  MODIFIES SQL DATA
  CALLED ON NULL INPUT
  EXTERNAL NAME 'MYLIB/RPGPGM'

  PARAMETER STYLE GENERAL;

```

In the example above, there is one integer parameter, you're returning 1 result set, the program language is RPGLE and the program name is RPGPGM in MYLIB. Run the SQL statement using one of the available SQL interfaces such as STRSQL, iSeries Navigator (Run SQL Scripts) or MySQL Query Browser.

AUTHORITY TIPS: Make sure the WOW user ID has proper authority to the program, as well as any files accessed by the program. You can grant authority to the program by running SQL similar to the following:

```
GRANT EXECUTE ON SPECIFIC PROCEDURE MYLIB.MYPROC TO WOW
```

NOTE: The above example assumes the WOW user ID is WOW.

Defining the WOW Operation

Defining an operation to call the external stored procedure is very similar to defining any other operation. The primary difference is what's specified for the "Operation Code". Instead of specifying a SELECT statement, you'll specify the procedure call:

```
CALL MYLIB.MYPROC(10)
```

The above example has one parameter in the procedure call. For more details on using stored procedures, including the use of field descriptors, see the chapter entitled Stored Procedures in the first section of the Builders Guide.

Calling an RPG Program That Returns a MODS (Array) in RPG IV

Add Code to Return an Array

The following code shows a simple RPG IV (RPGLE) program that has a char(2) state input parameter and returns records from file (QIWS/QCUSTCDT) that match the passed in state value:

```

FQCUSTCDT  IF    E                K DISK
D* Multi-occurrence data structure for returning rows to calling procedure
D CUSTLIST          DS                      OCCURS(100)
D  DCUSNUM              6S 0
D  DLSTNAM              8A
D  DINIT                3A
D  DSTREET             13A
D  DCITY                6A
D  DSTATE              2A
D  DZIPCOD             5S 0
D  DCDTLMT             4S 0
D  DCHGCOD             1S 0
D  DBALDUE             4S 0
D  DCDTDUE             1S 0
D*
D ROWCOUNT          S                10I 0
C*
C*-----*
C* Inputs
C   *ENTRY          PLIST
C                   PARM                      PSTATE          2
C*-----*
C                   MOVE      '0'          OFF          1
C                   MOVE      '1'          ON           1
C                   Z-ADD      0            ROWCOUNT
C                   READ      QCUSTCDT
C** Read Loop
C   *IN99           DOWEQ      OFF
C*   Only add records where the passed in state matches
C   STATE          IFEQ      PSTATE
C                   ADD        1            ROWCOUNT
C**   Write record to CUSTLIST at occurrence ROWCOUNT
C   ROWCOUNT      OCCUR      CUSTLIST
C                   Z-ADD      CUSNUM      DCUSNUM
C                   MOVE      LSTNAM      DLSTNAM
C                   MOVE      INIT        DINIT
C                   MOVE      STREET      DSTREET
C                   MOVE      CITY        DCITY
C                   MOVE      STATE      DSTATE
C                   Z-ADD      ZIPCOD      DZIPCOD
C                   Z-ADD      CDTLMT      DCDTLMT
C                   Z-ADD      CHGCOD      DCHGCOD
C                   Z-ADD      BALDUE      DBALDUE
C                   Z-ADD      CDTDUE      DCDTDUE
C                   ENDIF
C**   Read next record
C                   READ      QCUSTCDT

```

99

99

```

C      ROWCOUNT      IFEQ      100
C      MOVE      ON      *IN99
C      ENDIF
C      ENDDO
C*
C/EXEC SQL
C+ SET RESULT SETS ARRAY:CUSTLIST FOR:ROWCOUNT ROWS
C/END-EXEC
C*
C      MOVE      ON      *INLR

```

Any file could have been used here and any number of parameters could have been passed in to dictate the criteria for the end result set. The key is that each time a record is read, that entire record is added to a multiple occurrence data structure (array CustList). Once the looping has completed, an SQL result set is created based on the multiple occurrence data structure and is sized based on the value in variable ROWCOUNT, as shown in the SET RESULT SET statement.

Defining the Stored Procedure

Next you'll need to define an external stored procedure for the RPG program above:

```

CREATE PROCEDURE WOWRPG63.MYPROC (IN STATE CHAR(2))
  LANGUAGE RPGLE
  NOT DETERMINISTIC
  READS SQL DATA
  CALLED ON NULL INPUT
  EXTERNAL NAME 'MYLIB/RPGPGM'
  PARAMETER STYLE GENERAL;

```

In the example above, there is one char(2) parameter, you're returning a result set, the program language is RPGLE and the program name is RPGPGM in MYLIB. Run the SQL statement using one of the available SQL interfaces such as STRSQL, iSeries Navigator (Run SQL Scripts) or MySQL Query Browser.

AUTHORITY TIPS: Make sure the WOW user ID has proper authority to the program, as well as any files accessed by the program. You can grant authority to the program by running SQL similar to the following:

```
GRANT EXECUTE ON SPECIFIC PROCEDURE WOWRPG63.MYPROC TO WOW
```

NOTE: The above example assumes the WOW user ID is WOW.

Defining the WOW Operation

Defining an operation to call the external stored procedure is very similar to defining any other operation. The primary difference is what's specified for the "Operation Code". Instead of specifying a SELECT statement, you'll specify the procedure call:

```
CALL MYLIB.MYPROC (?100841)
```

The above example has one parameter in the procedure call. The parameter is a prompt parameter using field descriptor 100841. 100841 is assigned to the state field for file QIWS.QCUSTCDT and the number will vary for each system and connection.

The properties of the operation need to contain a StoredProcedure property group to designate which field descriptors are to be used for the input parameter and the results set:

```

StoredProcedure {
  tables: qiws.qcustcdt;
}

```

For more details on using stored procedures, including the use of field descriptors, see the chapter entitled Stored Procedures in the first section of the Builders Guide.

Calling an RPG Program That Returns Parameters

Generally WOW does not utilize the data returned from a procedure call (OUTPUT parameters), unless the procedure call is included in a Java custom Row class. For more details on calling procedures from a custom Row class, see the WOW Programmer's Guide, chapter Rows, section Example of Overriding the insert Method in a Row Subclass.

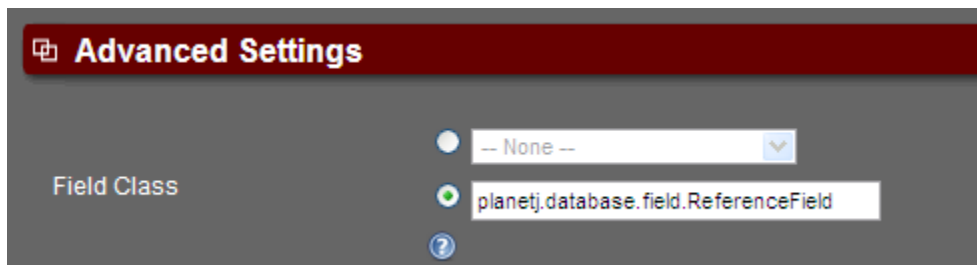
Advanced Development Techniques

Multi Value Reference Fields: [PRO]

In scenarios where you may want to reduce the number of columns shown to reduce horizontal scrolling, using a "ReferenceField" may provide value. A ReferenceField is a derived field that refers to other fields to get its value and field descriptor attributes. A ReferenceField would allow a single field to have different associations as well. Consider a scenario where a Shipment record had 2 columns, one that held the UPS tracking number and another that held the FEDEX tracking number. Using a ReferenceField would allow you to condense both columns into 1 and also handle the scenario where the WOW http reference association is different for FEDEX versus UPS.

Using the ReferenceField:

1. Create the initial SQL operation such as:
"SELECT SHIPMENT_ID, FEDEX#, UPS#, CASE WHEN FEDEX# is NULL THEN 'UPS#' ELSE 'FEDEX#' END as D_TrackingLink " Notice that if FEDEX# column is NULL then we return a String that represents the column name to use to show the actual value and FD attributes otherwise we return the column 'UPS#'. The result returned is used to find ANOTHER field in that same row to use for the actual value.
2. Create a derived FD (field descriptor) for "D_TrackingLink". In this field set the FieldClass to "planetj.database.field.ReferenceField".



3. Run your operation and test.

Here is an example screen where the last column is a reference field. For this example, the operations SQL is:

SELECT LASTNAME, JOB, BONUS, COMM, CASE WHEN BONUS > COMM then 'Bonus' else 'COMM' END as d_actual_pay FROM PJDATA.EMPLOYEE where comm > 0 and bonus > 0

	▲ Last ▼	▲ Job ▼	▲ Bonus ▼	▲ Commission ▼	▲ Bonus ▼
	Rexster	MANAGER	7589.00	1384.00	7589.00
	Holm	Pres	7976.00	1387.00	7976.00
	Buck	MANAGER	200.00	43160.00	43160.00
	Tim	MANAGER	200.00	3314.00	3314.00
	Cen	MANAGER	200.00	1680.00	1680.00

Considerations

1. The ReferenceField class takes the column heading of the 1st column that was produced.
2. All column names returned to ReferenceField must exist in the Row instance.
3. You can "hide" the source fields by setting their FD to hide or by not including them in the operations columns to display.

WOW Performance

WOW utilizes the JDBC database servers that are specified in the connection definitions therefore, typically, performance is nearly all outside of WOW's control.

What happens is:

- WOW passes an SQL statement to the server (AS400).
- The server (AS400) executes the statement using the best known optimization, as determined by database code. This includes such factors as file size and database indexes.
- After execution, WOW reads the data returned from the server (AS400), generates HTML, and then sends the HTML to the browser. Thus, nearly all performance is dependent on the database server (AS400).

However, WOW does have facilities to control and enhance performance as shown in the sections that follow.

WOW's Built In High Performance Cache

WOW allows the user to set a caching level for each operation:

Advanced	
Connection Alias	-- None -- ▾
Row Count	50
Row Class	
Caching Level*	Check cache and cache results ▾

The Caching options let you control how WOW stores data so it can be used later. By allowing WOW to retrieve the data from the cache, a call to the database server can be avoided when the operation is run. For more details on the caching level, see chapter "Create User Operations", section "User Operations" in the [WOW Builder's Guide p.1](#).

Connection Properties

Each JDBC database driver allows properties to be passed to the database server. You can specify these properties in the connection definition:

Connection Spec			
Connection Alias*	<input type="text" value="YOUR SYSTEM NAME"/>	URL*	<input type="text" value="jdbc:as4"/>
JDBC Driver*	<input type="text" value="AS/400 Remote"/>	Properties	<input type="text" value="prompt=f"/>

For the IBM AS/400 Remote driver, these can be found in the [IBM i Infocenter](#).

By default, WOW sets the following two AS/400 properties on a new connection:

- prompt=false - Specifies whether the user is prompted if a user name or password is needed to connect to the server. If a connection cannot be made without prompting the user, and this property is set to "false", then an attempt to connect will fail.
- trace=false - Specifies whether trace messages are logged. Trace messages are useful for debugging programs that call JDBC. However, there is a performance penalty associated with logging trace messages, so this property should only set to "true" for debugging. Trace messages are logged to System.out.

NOTE: Each property begins with a ';'.

You can also affect performance by setting the number of maximum connections allowed:

Advanced			
Min. Connections	<input type="text" value="3"/>	Max. Connections	<input type="text" value="10"/>

The default is ten. The maximum number of connections used can have a significant affect on your performance. This number will vary based on the power of the system.

Controlling the Number of Records Returned

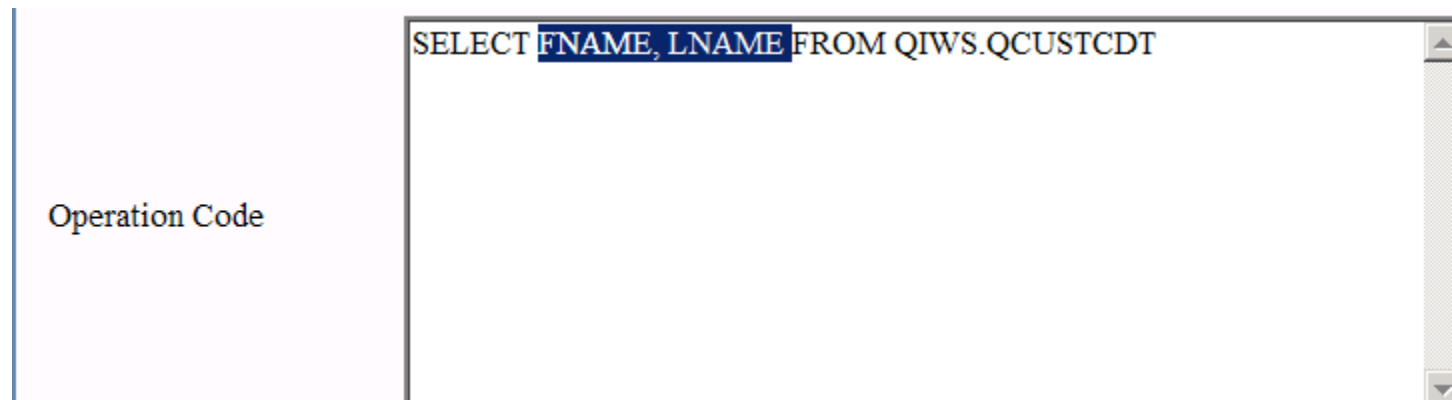
Setting the number of records returned in an SQL operation can improve performance.

Advanced			
Connection Alias	<input type="text" value="-- None --"/>	Operation Class	<input type="text"/>
Row Count	<input type="text" value="50"/>	Row Coll. Class	<input type="text"/>

The Row Count value controls how many rows are displayed in the results table. The smaller the number, the quicker each read from the database is. The default is 50. If the number of results is greater than the row count, links are generated on the results table allowing the user to page through it. This field should be adjusted based on your system performance and connection speed.

Controlling the Number of Fields Read

Reducing the number of fields read from an SQL SELECT statement to only those required can improve performance since less data will need to be read from the database. For example, instead of specifying all fields ('*'), specify only the specific fields necessary:



Optimizing SQL Performance for AS400 (iSeries)

There are a variety of methods or tools to determine how the performance of an SQL statement can be improved on an AS/400.

Find a reference(s) to SQL statement optimization techniques for AS400.

Compare SQL Performance Against Non-WOW Methods

Try taking the SQL statement from WOW and use one of these other methods to compare the results:

- AS400's STRSQL
- Or
- Put the SQL into iSeries Navigator (Run SQL Scripts)

If the SQL statement performs poorly by methods other than WOW, most likely the problem lies with the SQL statement or the file itself.

Using STRDBG

You can use debug messages for performance hints. On the AS/400:

- Turn debug on for your session è STRDBG *ALL .
- Run the SQL statement (to be run in the operation) using STRSQL.
- After successfully running the statement, look at the job log for debug messages from the query optimizer

Using iSeries Navigator (STRDBMON)

The iSeries Navigator version of the STRDBMON is called a detailed SQL performance monitor. You can start this monitor by right-clicking SQL Performance Monitors under the database portion of the iSeries Navigator tree and selecting New-> Detailed. You can monitor a single query or all queries. Once the monitor is started, it appears in the SQL performance monitors list.

For more details on iSeries optimization, see the "Optimizing Query Performance Using Query Optimization Tools" section in the "[Performance Optimization](#)" Reference.

Controlling How the Data is Accessed

On the iSeries, how the data from a file is accessed is determined by the query optimizer. If an index exists, the index is used. Otherwise, that decision is left to the query optimizer, which may not be the most efficient means. For more information on indexes (iSeries), see the "[Performance Optimization](#)" Reference.

Tomcat Server Performance

How you have your server configured can also affect your overall performance. For example, the amount of memory allocated to your server can have quite an effect on how the WOW application performs. For more information on Tomcat configuration, see the [Optimizing Tomcat](#) reference.

SQL Fragments [PRO]

Overview

WOW Enterprise Edition 7.0 and above include a feature that can significantly enhance performance of SQL. Consider a table that holds tax payer information and includes first name, last name, and social security number. If an optional search is created allowing entry of any of the three columns:

SELECT * from x.y where firstname = COALESCE(?, firstname) and lastname = COALESCE(?, lastname) and SSN = COALESCE(?, SSN)

If "Jones" is entered for lastname, the database processor must still process selection on firstname and SSN which can affect performance. SQL Fragment will translate this query to:

SELECT * from x.y where lastname = 'Jones'

Using SQL fragments, the WOW operation would be coded as: (This assumes lastname is required on search)

SELECT * from x.y where lastname = ? [[and firstname = ?]] [[and SSN = ?]]

Which will perform much faster.

This document describes how to work with SQL fragments in WOW. A SQL fragment is a portion of an SQL statement which can be dynamically removed or included in the statement at runtime. Whether or not a fragment is included is based on which of the statement's parameters the user has entered values into. Fragments are removed from the SQL only when it is sent to the database; fragments are never removed when generating the onscreen prompts for a SQL statement.

Defining a fragment

The sample SQL statement below selects fields from table A; the user can search on fields from table A or table B:

```
SELECT A1, A2, A3, A4 FROM LIB.A INNER JOIN LIB.B ON A1 = B1
WHERE (A1 = ? OR CAST(??1 AS INTEGER) IS NULL) AND
(A2 = ? OR CAST(??3 AS CHAR(10)) IS NULL) AND
(B2 = ? OR CAST(??5 AS CHAR(10)) IS NULL) AND
(B3 = ? OR CAST(??7 AS CHAR(10)) IS NULL)
ORDER BY A1
```

To define a fragment of code, enclose the fragment between two left brackets and two right brackets:

```
SELECT A1, A2, A3, A4 FROM LIB.A INNER JOIN LIB.B ON A1 = B1
WHERE (A1 = ? OR CAST(??1 AS INTEGER) IS NULL) AND
(A2 = ? OR CAST(??3 AS CHAR(10)) IS NULL) [[ AND
(B2 = ? OR CAST(??5 AS CHAR(10)) IS NULL) AND
(B3 = ? OR CAST(??7 AS CHAR(10)) IS NULL) ]]
```

This causes WOW to create a SQL fragment which contains the following code:

```
AND
(B2 = ? OR CAST(??5 AS CHAR(10)) IS NULL) AND
(B3 = ? OR CAST(??7 AS CHAR(10)) IS NULL)
```

By default, WOW will automatically remove any fragments which contain display parameters when none of those parameters have values. So if the user ran that SQL statement and left both the B2 and B3 prompts blank, then the SQL which WOW would actually run would be:

```
SELECT A1, A2, A3, A4 FROM LIB.A INNER JOIN LIB.B ON A1 = B1
WHERE (A1 = ? OR CAST(??1 AS INTEGER) IS NULL) AND
(A2 = ? OR CAST(??3 AS CHAR(10)) IS NULL)
ORDER BY A1
```

Fragment groups

SQL fragments can be associated together in a fragment group. Putting multiple fragments together in the same group tells WOW what it should do with fragments which do not contain display parameters (fragments with display parameters are removed when all of those parameters do not have values). Using the above example, we could define two fragments which belong to the same group.

```
SELECT A1, A2, A3, A4 FROM LIB.A [{group1} INNER JOIN LIB.B ON A1 = B1  
]]  
WHERE (A1 = ? OR CAST(??1 AS INTEGER) IS NULL) AND  
(A2 = ? OR CAST(??3 AS CHAR(10)) IS NULL) [{group1} AND  
(B2 = ? OR CAST(??5 AS CHAR(10)) IS NULL) AND  
(B3 = ? OR CAST(??7 AS CHAR(10)) IS NULL) ]]  
ORDER BY A1
```

Adding a group name surrounded by curly braces at the beginning of the fragments tells WOW that both fragments belong to the same group; in this case the group is named "group1". Any name could be used to identify the group; as long as both fragments use the same name they will be in the same group. In this case the first fragment does not contain any parameters (a "non-parameter fragment"), so it will be removed when the other fragment in its group (the "parameter fragment") is removed. As noted before, parameter fragments are automatically removed by WOW if none of the display parameters contain values. By using a parameter group, WOW can totally remove the join from the query when the user is not searching on any of the fields in the second table.

In the next example the SQL contains multiple fragment groups:

```
SELECT A1, A2, A3, A4 FROM LIB.A [{groupB} INNER JOIN LIB.B ON A1 = B1  
]]  
[{groupC} INNER JOIN LIB.C ON A1 = C1 ]]  
WHERE (A1 = ? OR CAST(??1 AS INTEGER) IS NULL) AND  
(A2 = ? OR CAST(??3 AS CHAR(10)) IS NULL) [{groupB} AND  
(B2 = ? OR CAST(??5 AS CHAR(10)) IS NULL) AND  
(B3 = ? OR CAST(??7 AS CHAR(10)) IS NULL) ]] [{groupC} AND  
C2 = ?]]  
ORDER BY A1
```

The fragments in groupB are only included when either B2 or B3 is given a value. The fragments in groupC are only included when the C2 parameter has a value.

SQL Adjustments

In some cases, adding fragments to your query allows you to remove redundant SQL from the statement. For example:

```
SELECT A1, A2, A3, A4 FROM LIB.A [{groupB} INNER JOIN LIB.B ON A1 = B1  
][  
[{groupC} INNER JOIN LIB.C ON A1 = C1 ]]  
WHERE (A1 = ? OR CAST(??1 AS INTEGER) IS NULL)  
[[AND (A2 = ? OR CAST(??3 AS CHAR(10)) IS NULL)]]  
[{groupB} AND (B2 = ? OR CAST(??5 AS CHAR(10)) IS NULL)]]  
[{groupB} AND (B3 = ? OR CAST(??7 AS CHAR(10)) IS NULL)]]  
[{groupC} AND (C2 = ? OR CAST(??9 AS CHAR(10)) IS NULL)]]  
ORDER BY A1
```

In this query, we can safely remove most of the checks which compare a parameter's value to null. The SQL uses those checks so that when the user doesn't enter a value into a parameter, then that parameter effectively matches all values in the database (in other words, that parameter isn't used in the search). However, enclosing a parameter in a fragment achieves the same thing, so the null checks are no longer needed. This statement will act the same as the one above:

```
SELECT A1, A2, A3, A4 FROM LIB.A [{groupB} INNER JOIN LIB.B ON A1 = B1  
][  
[{groupC} INNER JOIN LIB.C ON A1 = C1 ]]  
WHERE (A1 = ? OR CAST(??1 AS INTEGER) IS NULL)  
[[AND A2 = ? ]]  
[{groupB} AND B2 = ? ]]  
[{groupB} AND B3 = ? ]]  
[{groupC} AND C2 = ? ]]  
ORDER BY A1
```

Note that for this technique to work correctly each fragment needs to only contain a single display parameter, since a fragment is included as long as any one of its display parameters has a value.

Extra care needs to be taken when all of a statement's display parameters are in fragments. For example:

```
SELECT A1, A2, A3, A4 FROM LIB.A [{groupB} INNER JOIN LIB.B ON A1 = B1  
][  
[{groupC} INNER JOIN LIB.C ON A1 = C1 ]]  
WHERE 1 = 1  
[[AND A1 = ? ]]  
[[AND A2 = ? ]]  
[{groupB} AND B2 = ? ]]  
[{groupB} AND B3 = ? ]]  
[{groupC} AND C2 = ? ]]  
ORDER BY A1
```

Here it is important to include the "1 = 1" following the WHERE. If the user chooses to leave all of the parameters blank, then none of the fragments will be included in the SQL. In that case, the SQL will not be correct unless there is a comparison of some sort following the WHERE.

Groups with multiple parameter fragments

In all of the above examples, the fragment groups have all contained two fragments. In the next example, the fragment group contains three fragments, two of which have parameters and one with no parameters:

```
SELECT A1, A2, A3, A4 FROM LIB.A [{group1} INNER JOIN LIB.B ON A1 = B1
]]
WHERE (A1 = ? OR CAST(??1 AS INTEGER) IS NULL) AND
(A2 = ? OR CAST(??3 AS CHAR(10)) IS NULL) [{group1} AND
(B2 = ? OR CAST(??5 AS CHAR(10)) IS NULL) AND
(B3 = ? OR CAST(??7 AS CHAR(10)) IS NULL) ]] AND
(A9 = ? OR CAST(??9 AS INTEGER) IS NULL) [{group1} AND
B4 = ? ]]
ORDER BY A1
```

In order for the first fragment (the JOIN) to be included, one of the display parameters in either of the other two fragments must contain a value. So if B2 or B3 or B4 contains a value, the first fragment will be included. Only if none of those parameters have a value is the first fragment removed. The second and third fragments are removed as usual, when none of the display parameters in that particular fragment has a value.

To summarize, when a group contains multiple fragments with display parameters, if **any** of those fragments have values in their display parameters, then the group's fragments without display parameters will be included.

Note that in the above example, WOW will properly adjust the parameter numbering when fragments are removed. So if the second fragment is removed, then the ??9 parameter would be changed to ??5 since there are four fewer parameters preceding it in the SQL.

Inner fragments

In some cases such as the following example, you may need to include fragments within other fragments:

```
SELECT A1, A2, A3, A4 FROM LIB.A [{groupB} INNER JOIN LIB.B ON A1 = B1
]]
```



```

[[{groupC} INNER JOIN LIB.C ON A1 = C1 [{groupB} AND C6 = B6 ]] ]]
WHERE (A1 = ? OR CAST(??1 AS INTEGER) IS NULL) AND
(A2 = ? OR CAST(??3 AS CHAR(10)) IS NULL) [{groupB} AND
(B2 = ? OR CAST(??5 AS CHAR(10)) IS NULL) AND
(B3 = ? OR CAST(??7 AS CHAR(10)) IS NULL) ]] [{groupC} AND
C2 = ?]]
ORDER BY A1

```

The fragment in purple is only included when groupB and groupC are both included. That fragment describes the join between tables B and C, and so it only should be included in the SQL if both of those tables are in the SQL. By directly assigning the purple fragment to groupB and then embedding it within a fragment from groupC, the purple fragment becomes part of both groups.

Fragments with multiple groups

A fragment can belong to multiple groups. In the above example this was done by embedded one fragment within another. An alternative way to assign multiple groups to a fragment is by listing out the various groups it belongs to:

```

SELECT A1, A2, A3, A4 FROM LIB.A [{groupB} INNER JOIN LIB.B ON A1 = B1
]]
[[{groupC} INNER JOIN LIB.C ON A1 = C1 ]] [{groupB,groupC} AND C6 = B6
]]
WHERE (A1 = ? OR CAST(??1 AS INTEGER) IS NULL) AND
(A2 = ? OR CAST(??3 AS CHAR(10)) IS NULL) [{groupB} AND
(B2 = ? OR CAST(??5 AS CHAR(10)) IS NULL) AND
(B3 = ? OR CAST(??7 AS CHAR(10)) IS NULL) ]] [{groupC} AND
(C2 = ? OR CAST(??9 AS CHAR(10)) IS NULL) ]]
ORDER BY A1

```

Here, the third fragment belongs to both groupB and groupC. This example will behave exactly like the previous one; the third fragment is only included when both groupB and groupC contain display parameters with values. (For example, if parameters B3 and C2 were given values by the user then the purple fragment would be included, but if B3 were the only parameter with a value then the fragment would be removed.) To summarize, when a fragment without display parameters belongs to multiple groups, then that fragment is only included when **all** of those groups are included.

Consider the following example, where the first fragment contains the A2 column:

```

SELECT A1, [{groupA} A2,]] A3, A4 FROM LIB.A [{groupB} INNER JOIN LIB.B
ON A1 = B1 ]]
[[{groupC} INNER JOIN LIB.C ON A1 = C1 ]] [{groupB,groupC} AND C6 = B6
]]

```

```

WHERE (A1 = ? OR CAST(??1 AS INTEGER) IS NULL) AND
(A2 = ? OR CAST(??3 AS CHAR(10)) IS NULL) [[{groupA,groupB} AND
(B2 = ? OR CAST(??5 AS CHAR(10)) IS NULL) AND
(B3 = ? OR CAST(??7 AS CHAR(10)) IS NULL) ]] [[{groupA,groupC} AND
C2 = ? ]]
ORDER BY A1

```

In this example, that fragment will be included when one or more of the B2, B3, or C2 parameters is given a value by the user. Unlike the purple fragment which requires parameters from all of its groups to have values before it is included, the A2 fragment is included when any one (or more) of the B2, B3, or C2 parameters has a value. This is because it only belongs to a single group whereas the purple fragment belongs to multiple groups.

Fragment properties

In the previous examples curly braces were used to list out the groups to which a fragment belongs. However, the curly braces can also be used to directly assign fragment properties using the normal property group syntax. For example:

```

SELECT A1, A2, A3, A4 FROM LIB.A [[{id: fragment1; groups: groupB;} INNER
JOIN LIB.B ON A1 = B1 ]]
[[{groupC} INNER JOIN LIB.C ON A1 = C1 ]] [[{id: fragment3; groups:
groupB,groupC;} AND C6 = B6 ]]
WHERE (A1 = ? OR CAST(??1 AS INTEGER) IS NULL) AND
(A2 = ? OR CAST(??3 AS CHAR(10)) IS NULL) [[{groupB} AND
(B2 = ? OR CAST(??5 AS CHAR(10)) IS NULL) AND
(B3 = ? OR CAST(??7 AS CHAR(10)) IS NULL) ]] [[{groupC} AND
C2 = ? ]]
ORDER BY A1

```

Here, two properties are defined for the first fragment, the *id* property and the *groups* property. The following fragment properties can be assigned:

- **class** – The name of the Java class which should be used for the fragment. The class must implement `IFragment` and provide a default constructor
- **groups** – The names of the groups to which the fragment belongs
- **id** – A unique identifier for the fragment. The ID is useful when working with the fragments in code; otherwise there is no need to assign an ID
- **include** – This property describes when WOW should include the fragment in the SQL statement. The possible values for this property are:
 - **auto** – WOW should decide when to include the fragment and when not to (this is the default value)

- **false** – The fragment should never be included in the SQL. This setting can be overridden in code, but otherwise the fragment won't be included
 - **true** – The fragment should always be included in the SQL. This setting can be overridden in code, but otherwise the fragment will be included
- **remove values** – The list of values which will cause WOW to remove the fragment from the SQL. If the value of a display parameter in the fragment is equal to one of the values in the list, then that display parameter is treated as though it did not have a value (but only for the purposes of WOW's fragment logic). So a parameter whose value is in the *remove values* property can cause its containing fragment (and other fragments in its fragment group) to be removed from the SQL. If the value "null" is present in the *remove values* property then display parameters without values can cause their fragment to be removed as usual; if the *remove values* property is present but does not contain "null" then display parameters without values will not cause their fragment to be removed.

When the only property that is being assigned is the *groups* property, then you can simply list out the fragment's groups, as is done in the initial examples. Only when you want to assign other properties do you have to use the property group format.

In this example, the A3 prompt will never be sent to the database:

```
SELECT A1, A2, A3, A4 FROM LIB.A [[{group1} INNER JOIN LIB.B ON A1 = B1
]]
WHERE (A1 = ? OR CAST(??1 AS INTEGER) IS NULL) AND
(A2 = ? OR CAST(??3 AS CHAR(10)) IS NULL) [[{group1} AND
(B2 = ? OR CAST(??5 AS CHAR(10)) IS NULL) AND
(B3 = ? OR CAST(??7 AS CHAR(10)) IS NULL) ]]
[[{groups:group2; include:false;} AND A3 = ?]]
ORDER BY [[{group2}A3,]] A1
```

However, if the user enters a value for the A3 prompt, then the results would be sorted by the A3 column. In other words, the fact that the A3 parameter is never sent to the DB does not prevent the last fragment from being sent to the DB if the user enters a value for the A3 parameter.

Working with fragments in code

Internally each fragment defined in the SQL statement is represented by an IFragment object. (That interface, like most of the fragment code is located in the planetj.database.sql.fragment package.) The IFragment objects are contained in the SQLContext object. One important thing to keep in mind is that fragments are never actually removed from a SQLContext's internally stored code. Instead, they are stripped out of the code which is sent to the database, but remain in the code stored in the SQLContext. For this reason, in code the terms "included" and "removed" are not

used to describe the fragments; instead “active” and “inactive” are used to refer to fragments which are sent to the DB, or not.

There are several methods which can be used to retrieve the code for a SQLContext, depending on how much processing you want WOW to do with the code:

- **getOriginalCode()** – Returns the code as it was entered in the builder
- **getCode()** – Returns the internal code stored in the SQLContext. This will include all fragments regardless of whether or not they are active, but does not include the fragment control characters (i.e. the opening “[{group1, group2}” and closing “]” are not included).
- **getCode(true)** – Returns the internal code stored in the SQLContext, after removing any fragments which are not active. All fragment control characters are also removed.

Below is a listing of some of the methods which can be used when working with fragments:

- **SQLContext.getFragments** – Returns a SQLFragmentCollection object which contains the individual fragments
- **SQLFragmentCollection.getFragment** – Gets a fragment by its ID. A fragment’s ID is assigned with the *id* property in the property group.
- **SQLFragmentCollection.getFragmentGroup** – Gets a group of fragments by the group name
- **IFragment.isActive** – Returns true/false depending on whether or not the fragment is active
- **IFragment.setActive** – Sets a fragment as either being active or not active (if a fragment is not active it is not sent to the database)
- **IFragment.getDisplayParameters** – Returns a List containing all of the fragment’s display parameters
- **IFragment.getDisplayParameterStatus** – Examines all of the display parameters in the fragment and returns one of the following constants:
 - **PARAMETERS_NONE** – The fragment does not contain any display parameters
 - **PARAMETERS_VALUELESS** – None of the fragment’s display parameters have values
 - **PARAMETERS_SOME_WITH_VALUES** – The fragment contains multiple display parameters, some of those parameters have values and others do not
 - **PARAMETERS_ALL_WITH_VALUES** – All of the fragment’s display parameters contain a value

Actions/Events [PRO]

From the WOW Builder, run Development Tools > Actions. This menu item lets the user define a “table driven” action or event. Table driven actions and events are normally defined completely as an entry within the WOW event/action table, without the need for coding or special property group configuration.

Action

An action is a visible option that can be defined for a Row or RowCollection (Table). An action, when clicked on, can run either a pre-defined operation or a java class. An action can be visible as a button, icon, link, etc.

Creating an Action

To create an action:

1. Bring up the Actions and Events (Other > Actions/Events)
2. Click on Add Action/Event

Entry Information:

- **Entry Type** - Leave as Action.
- **Entry Subtype** - defines what type of action it is. A Row action is an action that is associated with a Row. For example, it can be an action to the left of each row (Table view), it can be a button from the row detail screen, it can be a right click option when you hover over a row (Table view), etc. A Rowcollection action is associated with a results display (Table view) and can be visible as a button on the top or bottom of the results, in the tool bar (same location as refresh, print, etc.), etc.
- **Action Name** - Unique name assigned to action.
- **Active** - Provides ability to turn off action without removing it.
- **Description** - A detailed description of the action.

Operation Inheritance:

- **Source Application** - Lets the user limit the visibility of the action to a specific application.
- **Source Operation** - Lets the user limit the visibility of the action to a specific operation.

Action/Event Execution Information:

This defines what gets run by the action. You need to specify either an operation or a Java class.

- **Operation** - Select an operation to be called. If the operation is an association operation, then the current Row is passed as the associated Row (for parameter substitution).
- **Java Class** - Specifies a Java class file that is a subclass of `planetj.wow.action.AbstractAction` and implements the `handleAction` method:

```
@Override
public Object handleAction(Object o, IActionProperties props, ExecutingContext ec) throws CMException {
    TableDrivenActionRow actionRow = (TableDrivenActionRow) props;
    Row associatedRow = (Row) o;
    // TODO - Perform action.
    if (ec instanceof HttpExecutingContext) {
        String message = actionRow.getMessage();
        if ((message != null && message.trim().length() > 0)) {
            DataEngineServlet.setUserMessage(message, ((HttpExecutingContext) ec).getRequest());
        } else {
            DataEngineServlet.setUserMessage(null, ((HttpExecutingContext) ec).getRequest());
        }
    }
    return null;
}
```

Java Classes Provided by WOW:

- `planetj.wow.repository.SaveSnapshotOfObjectAction` - Action that saves a snap shot (copy of current Row or RowCollection) into the WOW Object Repository table.

NOTE: Class files written as a subclass of `AbstractAction` are compatible for both Actions and events.

Action Location (for Row action):

- **Detail View:**
 - Bottom left, Bottom right, Top left, Top right
 - Left of (existing) buttons, Right of (existing) buttons.
- **Results (Table) View:**
 - **Left side of each row** - action similar to edit, display, etc.
 - **Hide on results display** - only show action on details display
 - **Drop down** - Shows the action as a drop down choice on each Row from a derived field configured for running drop down choices. See the

section “Implementing the Drop Down Location” for more details. Normally, the Display type is set to *Text* for this type of action.

- **Right Click** - show option when hovering over a row and right clicking. Normally, the Display type is set to *Text* for this type of action.

Action Location (for Row Collection action):

- Bottom left, Bottom right, Top left, Top right
- **Left of Search Fields** - Left side of operation’s search fields.
- **Right of Search Fields** - Right side of operation’s search fields.
- **Toolbar** - Action is location in toolbar above the results (same location as refresh or printer icon). Normally, the Display type is set to *Link/Image*.

Action Properties:

- **Display Type** - How is the action to be displayed.
 - **Link/Image** - If an image is specified, displays as an image (jpg, png, etc.). Otherwise it’s displayed as a link.
 - **Button** - Show action as a button
 - **Checkbox** - Show action as a check box
 - **Text** - Show action as text (used mostly for location “Right Click” or “Drop Down”).
- **Image** - Used with the display type Link/Image.
- **Action Group** - Group name to assign to action. Actions with the same group are grouped together.
- **Action Display Order** - If more than 1 action is assigned to an operation in the same location, this dictates which action is displayed first (lower order #).
- **Style Class** - Style class (CSS) to use when rendering this action on the screen.
- **Label** - By default, the action name is used for the label (e.g. button text, link text, etc.). Specify an alternate label to use.
- **Please Wait JSP** - When the action is running, you may want to display a please wait jsp. Specify the jsp to use, or specify “TRUE” to use the default please wait JSP.
- **Browser Target** - Specify a target for the browser window to be used, such as `_blank` (renders in a new window). Default is `_self` (same window).
- **Start Navigation Group** - Should this action start a new navigation group (add divider for right click or space between button groups)?
- **End Navigation Group** - Should this action end the current navigation group?
- **New Window Properties** - Lets you specify new window properties, such as the window size. E.g.
"toolbar=no,location=no,directories=no,status=no,menubar=no,copyhistory=no,width=700,height=800".

Action Messages:

- **No Rows Selected Error Text** - On a Results (Table) screen, lets you specify a message to display when no rows are selected when an action is run. May be useful when TableDisplay property selectionType is not set to "none". If coding a custom action class, you will need to add code to extract the message and display it:

```
TableDrivenActionRow actionRow =  
    (TableDrivenActionRow) props;  
throw new  
    CMException(actionRow.getNoSelectionErrorText());
```

- **Confirmation Text** - Text to display (warning - do you want to run this action?) before the action runs. Default is to provide no warning before running the action.
- **Action Completion Text** - Alternate completion text to display after the action runs successfully. If coding a custom action class, you will need to add code to extract the message and display it:

```
TableDrivenActionRow actionRow =  
    (TableDrivenActionRow) props;  
if (ec instanceof HttpExecutingContext) {  
    String message = actionRow.getMessage();  
    if ((message != null && message.trim().length() >  
0)) {  
        DataEngineServlet.setUserMessage(message,  
            ((HttpExecutingContext) ec).getRequest());  
    } else {  
        DataEngineServlet.setUserMessage(null,  
            ((HttpExecutingContext)  
                ec).getRequest());  
    }  
}
```

Authorization:

Authorization works the same way as operation authorization. It controls which users are able to see/use and action. This is only applicable when an application is secured (signon required).

- **Security Type** - The type of security measures to use.
- **Security Level** - Security level is used in conjunction with the *Security Type* feature.

- **Auto Run Op.** - Allows you to specify an operation that will automatically run on a set schedule. The pull-down for this field displays any available Auto Run operations created within the application.
- **Execute Authority Operation** - Used to limit which users can view and run the operation. All Authorization Operations defined for the current application should appear in the drop down selection. If no operation is selected, all users will be authorized to execute this operation.

Implementing the Drop Down Location:

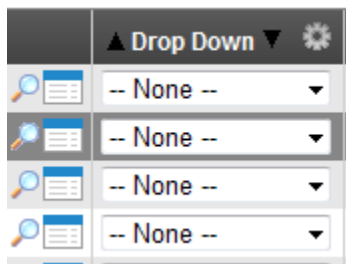
To fully implement a Row action using the drop down location, it must be configured as follows:

1. Define 1 or more actions with a location of Drop Down and Operation Inheritance (scope) pointing to the desired operation to contain the drop down options.
2. For the operation (step a), the SQL results must include a derived field to be used for showing the drop down option(s) associated with each row. For example => `Select a.*, '' as D_DROPDOWN From xxx.yyy` where field D_DROPDOWN is the derived field in this example.
3. In the same operation's properties, include this DisplayColumns property to make the drop down field editable from the table (results) display:

```
DisplayColumns { editableResults:D_DROPDOWN; }
```

4. Create a derived field descriptor for derived field D_DROPDOWN and set the following in Advanced Settings:
 1. **Field Class:** planetj.wow.action.ActionField
 2. **Field Descriptor Type:** Derived

Once configured, the derived field should look like this, with a drop down choice for each drop down action created:



Event

An event is very similar to a trigger, where an event is fired automatically when a change is made to a table.

Creating an Event

To create an event:

1. Bring up the Actions and Events (Other > Actions/Events)
2. Click on Add Action/Event

Entry Information:

- **Entry Type** - Change to Event.
- **Entry Subtype** - defines when the event is run or triggered (when a row changes). Choices are Pre-Insert, Post-Insert, Pre-Update, Post-Update, Pre-Delete or Post-Delete.
- **Event Name** - Unique name assigned to event.
- **Active** - Provides ability to turn off event without removing it.
- **Description** - A detailed description of the event.

Event Scope:

- **Source Application** - Lets the user limit when the event is to run to a specific application.
- **Source Operation** - Lets the user limit when the event is to run to a specific operation.

Table That Triggers Event:

You need to identify which table (when changed) triggers the event.

- **Source Connection** - Defines which connection is used to change the table. Choose between any of the existing connections. Connection choice "***INTERNAL CONNECTION ***" refers to the metadata connection used by WOW. This connection should only be used when utilizing a built-in WOW class such as `planetj.wow.repository.SaveSnapshotOfWowObjectAction` .
- **Source Table** - Identifies the table to attach the event or trigger to.

Action/Event Execution Information:

This defines what gets run by the event. You need to specify either an operation or a Java class.

- **Operation** - Select an operation to be called. If the operation is an association operation, then the current Row is passed as the associated Row (for parameter substitution).
- **Java Class** - Specifies a Java class file that is a subclass of `planetj.wow.action.AbstractAction` and implements the `handleAction` method:

```

@Override
public Object handleAction(Object o, IActionProperties props, ExecutingContext ec) throws CMException {
    TableDrivenActionRow actionRow = (TableDrivenActionRow) props;
    Row associatedRow = (Row) o;
    // TODO - Perform action.
    if (ec instanceof HttpExecutingContext) {
        String message = actionRow.getMessage();
        if ((message != null && message.trim().length() > 0)) {
            DataEngineServlet.setUserMessage(message, ((HttpExecutingContext) ec).getRequest());
        } else {
            DataEngineServlet.setUserMessage(null, ((HttpExecutingContext) ec).getRequest());
        }
    }
    return null;
}

```

Java Classes Provided by WOW:

- planetj.wow.repository.SaveSnapshotOfWowObjectAction - Used to save a snapshot of WOW objects (Operations, Field Descriptors) when an operation or field descriptor is changed.
- planetj.dataengine.email.SendEmailForLogEntryEvent - Used to send an email automatically when a table is changed. A user configures an email log entry and specifies the entry ID as a class parameter. e.g. planetj.dataengine.email.SendEmailForLogEntryEvent,id=1 .
Parameters supported:
 - id - log ID from EMAILLOG.
 - mlib - optional WOW email library.

NOTE: This class is only available with the WOW Mail add-on.
More details available with WOW Email documentation.

NOTE: Class files written as a subclass of AbstractAction are compatible for both Actions and events.

Example Uses:

Adding Action to save a copy of a single Row or Table Results:

To add an action that will provide the ability to save the current Row or table results from an operation, use the following settings:

This action adds an action to the toolbar to every operation for the specified application. The action is displayed as the icon “dataengine/images/camera-icon.png”.



The action utilizes the built-in action “planetj.wow.repository.SaveSnapshotOfObjectAction”, which saves table results as a blob object inside WOW’s object repository table “WOBJECTREP”. See “Object Repository Data” for more details.

Adding Event to automatically save changes to WOW Operations:

To add an event that saves a copy of the operation after each time it is updated, use the following settings:

This feature lets the developer see what changes have occurred to an operation and by which developer. The event utilizes the built-in event “planetj.wow.repository.SaveSnapshotOfWowObjectAction”, which saves the operation as a blob object inside WOW’s object repository table “WOBJECTREP”. See “WOW Metadata Changes” for more details.

You can also set the Source Table to FIELDDDATA (*), which lets you track field descriptor changes.

Adding Action to Run 2nd Operation That Remembers Rows Selected

To add an action that calls another operation and displays rows selected from the 1st operation, use the following steps:

Create 1st Operation:

Create 1st operation that allows for selection:

Set Key Fields to Global:

For any field from operation 1 that you need WOW to remember for future operations, change the field's FD (Field Descriptor) so that its usage ID = -3 (Global). In our case, we want the employee ID.

Create 2nd Operation:

Create a 2nd operation that filters the results based on previously selected rows (1st operation).

Notice the use of the global parameter (?!fieldname) within the IN clause. When the 2nd operation runs, WOW will replace ?!EMPNO with something like:

```
EMPNO IN (101,135,1002)
```

The values 101, 135, and 1002 are the EMPNO values from 3 selected rows.

Create Action on 1st operation to Run 2nd Operation:

When a table driven RowCollection action is defined to run a 2nd operation and 1 or more fields from the operation are set to global, WOW will save the selected row information for each global field. In order to provide an action button on the 1st operation that runs the 2nd operation, set something similar to the following:

- Entry Type: Action
- Entry SubType: RowCollection (Table)
- Source Operation: Operation where action is rendered.
- Operation: Operation the action runs
- Display Type: Button
- Label: Button text

This action adds a button at the bottom of the results (table) generated by operation Employees:

So now if operation 1 (Employees) is run and a user selects the last 2 rows and then clicks on the Show Selected Rows button, WOW:

1) Saves the selected rows and stores them for global field EMPNO

2) Prepares the 2nd operation (Previously Selected Employees)

3) Sees the operation's SQL referencing a global variable in an IN clause (utilizes multiple values):

```
SELECT EMPNO,FIRSTNME, MIDINIT, LASTNAME, JOB, WORKDEPT, mgrnum, BIRTHDATE  
FROM PJDATA.EMPLOYEE WHERE EMPNO IN ??!EMPNO
```

4) Finds a RowCollection for EMPNO and substitutes in all empno values:

```
SELECT EMPNO,FIRSTNME, MIDINIT, LASTNAME, JOB, WORKDEPT, mgrnum, BIRTHDATE  
FROM PJDATA.EMPLOYEE WHERE EMPNO IN (22231,33333)
```

5) Runs the altered SQL

Troubleshooting and Debugging WOW

My iSeries DB2 files are locked by WOW which is affecting my saves and other programs!

OS/400 may apply a READ lock on files that are accessed more than once from the same connection. This is a function of OS/400 and is done for performance and is not caused by WOW. If you need to use a file that OS/400 has locked use the following command:

```
ALCOBJ OBJ((QIWS/QCUSTCDT *FILE *SHRRD *FIRST)) CONFLICT(*QSRSL)
```

NOTE: Adjust the library and file name as well as the locking level to suit your needs.

WOW is unable to connect to the IBM iSeries Metadata Server because of restricted ports

WOW can work with any metadata server. Every organization handles network controls and restrictions differently. If WOW is not able to connect to your server it may be caused by blocked ports. In the example of an iSeries machine there are numerous ports that need to be opened for full JDBC support that WOW uses, yet maybe your network administrator or firewall is blocking some of these ports.

At IBM's website you can go to this link <http://www-1.ibm.com/servers/eserver/iseries/toolbox/faqports.htm> shown below and see what ports need to be opened for full JDBC and server controls. Do the same for your system and have your network administrator allow these ports so that WOW can connect. On the iSeries WOW requires the following ports to be open (other databases use different ports, check your database documentation):

- 449
- 8470
- 8471
- 8476

My Documents Mozilla Firefox tomcat_WOW ~WRL0817....

Inbox - Microsoft Outlook

IBM Toolbox for Java - Microsoft Internet Explorer

File Edit View Favorites Tools Help

Back Forward Stop Reload Home Search Favorites Media Print Mail News RSS Feeds

Address http://www-1.ibm.com/servers/eserver/iseries/toolbox/faqports.htm Go



Home | Products & services | Support & downloads | My account

Select a country

Servers > Midrange servers > Toolbox for Java & JOpen >

Toolbox for Java™ & JOpen

Server Ports Used

Service	as-svrmap	drda	as-central	as-database	as-dtaq	as-file	as-netprt	as-rmtcmd	as-signon
Port	449	446	8470	8471	8472	8473	8474	8475	8476
SSL Port	-	448	9470	9471	9472	9473	9474	9475	9476
Notes	(1)		(2)(3)	(2)	(2)	(2)	(2)	(2)	(2)
Command Call	x		x					x	x
Data Area	x		x					x	x
Data Conversion	x		x						x
Data Queues	x		x		x				x
Integrated File System	x		x			x			x
JDBC	x		x	x					x
Jobs	x		x					x	x
Messages	x		x					x	x
Network Print	x		x				x		x
Program Call	x		x					x	x
Record Level Access	x	x	x						x
System Status	x		x					x	x
System Value	x		x					x	x
Users and Groups	x		x					x	x
User Space	x		x			x		x	x

Notes:

If the Toolbox's **Proxy Server** feature is selected, and is enabled on the iSeries or AS/400, only port 2470 is needed.

Additional Information

→ [Toolbox Home](#)

→ [Downloads](#)

→ [FAQs](#)

→ [Forum](#)

→ [JDBC FAQ](#)

→ [Performance](#)

→ [Troubleshooting](#)

→ [Useful links](#)

News

→ [Articles in the press](#)

→ [News Archive](#)

Change User Name and Password of WOW for new metadata system

Switching metadata servers or types of servers will return some kind of error. If changing the Login that connects you to the metadata machine or server such as MySQL or iSeries, it is necessary to change the web.xml document located under <Apache Path>/Tomcat/webapps/wow60/web-inf/web.xml. Right click on web.xml and open with notepad. Shown below is the XML Tags that need to be changed.

Inbox - Microsoft Outlook

web - Notepad

```

        <display-name>DataEngineApplicationBuilderServlet</display-name>
<servlet-class>planetj.dataengine.application.DataEngineApplicationBuilderServlet</serv

    <!-- Below are the initial parameters for the WOW metadata system. Please not
        System URL must begin with a prefix such as jdbc:as400: or jdbc:db2: foll
    the IP address or an alias if you have the alias mapping set up in your hosts

    <init-param id="WOW_Metadata_System_URL">
        <param-name>PJ_SYSTEM_URL</param-name>
        <param-value>jdbc:as400:167.206.73.188</param-value>
        <description>Replace YOUR_SYSTEM_HERE with the IP or alias of the
            WOW metadata system</description>
    </init-param>
    <!-- we default the User ID to WOW-->
    <init-param id="WOW_Metadata_System_User_ID">
        <param-name>PJ_USER_ID</param-name>
        <param-value>wow</param-value>
    </init-param>
    <init-param id="WOW_Metadata_System_Password">
        <param-name>PJ_PASSWORD</param-name>
        <param-value>maui1a</param-value>
        <description>Replace PASSWORD_HERE with your password.</description>
    </init-param>

    <!-- Below is your driver to get to the WOW metadata system. If you are going
        AS400, the driver will be com.ibm.as400.access.AS400JDBCdriver, and the S
        above should start with jdbc:as400:-->

    <init-param id="WOW_Metadata_System_JDBC_Driver">
        <param-name>PJ_JDBC_DRIVER</param-name>
        <param-value>com.ibm.as400.access.AS400JDBCdriver</param-value>
    </init-param>
    <init-param id="WOW_Metadata_System_JDBC_optimization_Parameters">
        <param-name>PJ_JDBC_OPTIMIZATIONS</param-name>
        <param-value>;naming=sql;date format=iso;errors=full;extended
dynamic=false;package=dtaxplr;package library=qqpl;package cache=true;package clear=tru
binary=true;trace=false</param-value>
    </init-param>

    <!-- Below are the initial parameters for the WOW metadata system connectino p
    <init-param id="WOW_Metadata_System_Connection_Pool_Clean_Up_Time">
        <param-name>PJ_CLEAN_UP_TIME</param-name>
        <param-value>9800</param-value>
    </init-param>
    <init-param id="WOW_Metadata_System_Connection_Pool_Maximum_Number_of_Connecti
        <param-name>PJ_MAX_CONNECTIONS</param-name>

```

It is necessary to change the PJ_SYSTEM_URL, PJ_USER_ID and the PJ_PASSWORD initial parameter values.

As an example, the new login to the metadata system might be wow60 instead of wow. To access the metadata for WOW, change the init-param PJ_USER_ID value to wow60 instead of wow.

```
<init-param id="WOW_Metadata_System_User_ID">  
  <param-name>PJ_USER_ID</param-name>  
  <param-value>wow60</param-value>  
</init-param>
```

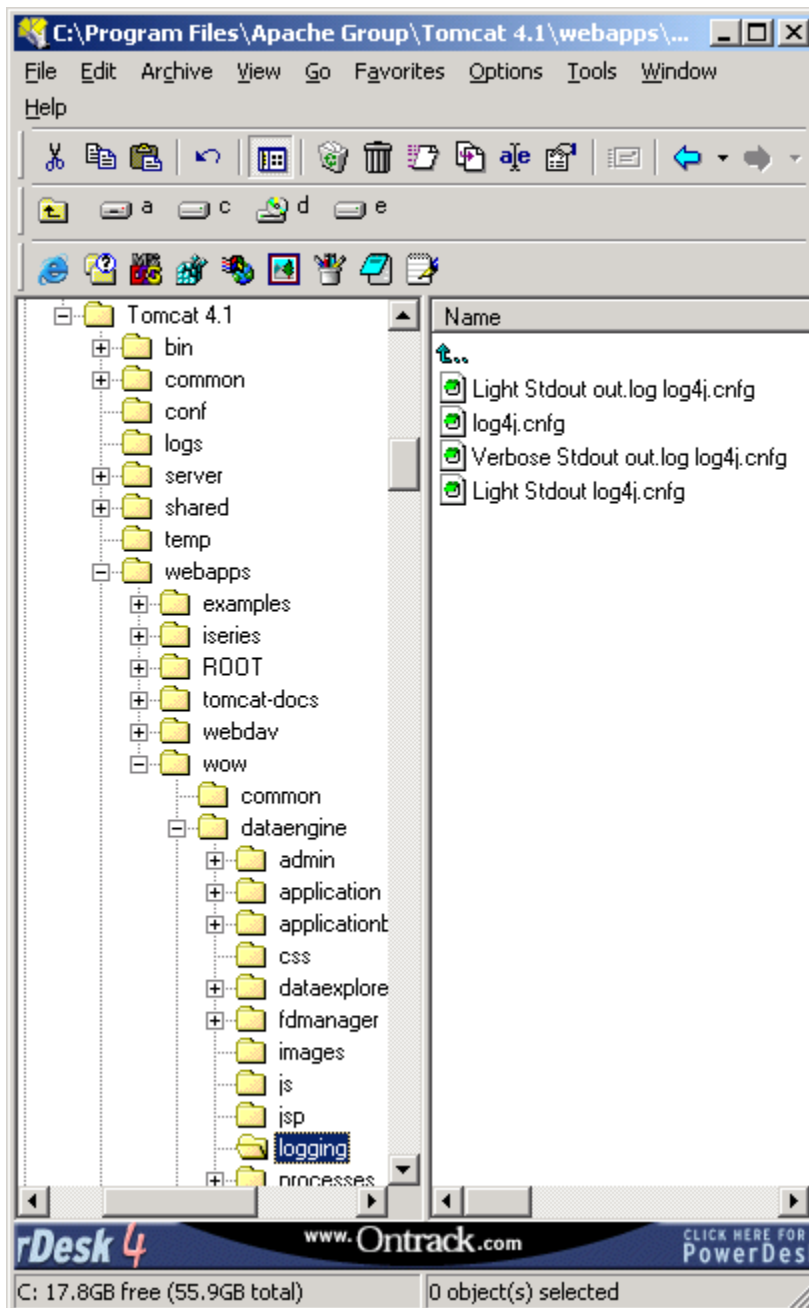
Now, change the metadata system settings and logins without reinstalling WOW. If there are errors thrown about connection to the WOW metadata system then the web.xml document is a good place to look first.

Configuring Logging (Log4j)

As WOW runs, it logs events that are important. Some events are more important than others, so we made this logging configurable by modifying a property file in the wow folder. You may want to change the level of logging to a more verbose mode to see detailed or debugging information. In most cases, you will just want to log exceptions or events that cause WOW to error. By default, only events of levels WARN, ERROR, and FATAL are logged.

Log4J Configurations

Currently events can be logged at the following levels (listed in order of increasing severity): DEBUG, INFO, WARN, ERROR, and FATAL. You can also control where you want the log messages outputted to. You can direct them to Standard Out (stdout), or a file (output.log) or both. By default we direct output to both stdout and output.log. The master configuration file is called log4j.cnfg. You can find this file in the following directory: <TOMCAT INSTALL ROOT>\webapps\wow\dataengine\logging\.



For more verbose logging, try renaming the Verbose Stdout out.log log4j.cnfg file to log4j.cnfg. Likewise, if you would like a less verbose logging level, rename one of the Light Stdout... files to log4j.cnfg. Make sure you keep the original log4j.cnfg file so you can always change it back.

NOTE: You will have to restart the Web server for the changes to take effect.

For more advanced logging techniques, see the following link:

<http://jakarta.apache.org/log4j/docs/manual.html>.

WOW Log File (output.log)

If your log4j.cnfg is directing output to a log file, you can find that log file by searching for the file name set in the config file. By default, the file is named output.log and it can be viewed by looking in the Start Button shortcut where you click Start Tomcat. For example: Start Button => Programs => Apache Tomcat 4.1 =>output.log.

Running a SQL Statement with Period in the name of a Database Table

WOW 6.0 does not support a period in any of the names of a table that are being modified by a SQL statement. For example: `SELECT * FROM PJDATA.EMPL.NM` – where all the rows of EMPL.NM are being selected will not work. We suggest that you create a logical over “EMPL.NM” and give it a normal name such as “EMPLNM”.

When Running WOW off of a Linux or Unix machine and with MySQL, some operations don't work

When running operations in on any UNIX machine and using MySQL as Metadata Server some of the operations are throwing errors or not working. This may be because MySQL running on a Unix System requires all library (schema names) and table name to be in UPPER CASE. Check to make sure all your files being accessed are in UPPER CASE and restart the MySQL Server.

When creating a row, the Current Date -CURRENT returns the wrong date

When inserting a new row, a default timestamp can be obtained by putting *CURRENT in the FD of a Date Field. Unfortunately, the time might be off by either a time zone or just by hours, minutes or seconds. This is normally a configuration issue from your OS/400 or JDK.

Instructions on how to correctly setup the time zone offsets are in [IBM Redbooks](#).

Changing Database Tables and Views:

The JDBC driver provider may cache information about tables and views for performance reasons. If you change the definition of a table or view, you may need to restart the associated WOW connection for the database change to be enforced.

WOW Administration and Support

Backing Up WOW Metadata

All of WOW's critical information is found within its metadata. This makes backing up your data very crucial. Backing up important data within WOW or any other program for that

matter is one of the most important ways you can assure that your organization will run smoothly. The backup process will be different on most machines.

Backing Up WOW Metadata from AS/400

If your installation stores WOW metadata on the AS/400 you should regularly backup the libraries PJUSER64 and PJSYS64. PJUSER64 contains user metadata while PJSYS64 contains WOW metadata. By default, the PJUSER64 library contains all operations, applications, and field descriptors that you will have created. In the event of disaster recovery, restore these libraries to your AS/400. (If you are using an application library other than the default, then you should backup that library).

Backing Up WOW Application Code

You should also regularly backup WOW's application code and any customer Java classes or other web files you may have created during your development. All this code is contained and web files are contained with in your application server's ../webapps/wow64 folder. This folder should be backed up on a regular basis.

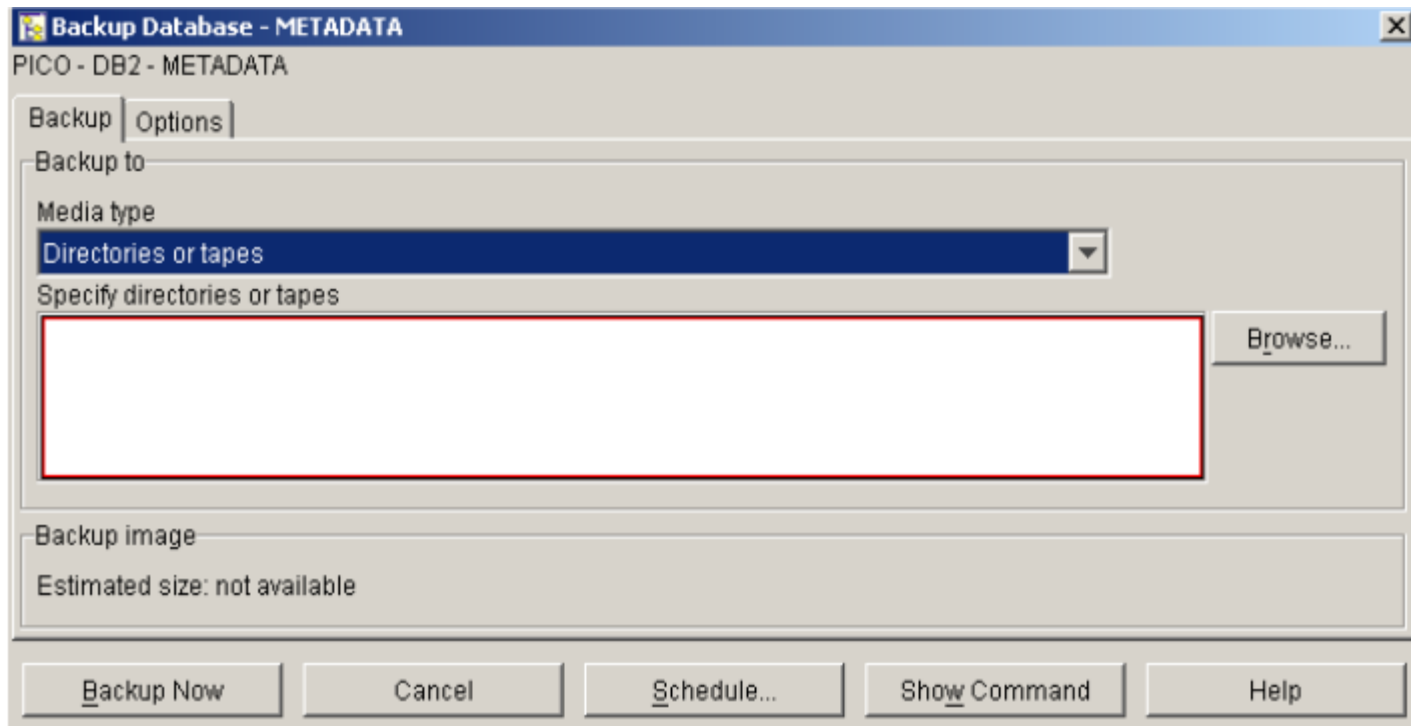
Backing Up WOW Metadata Using DB2 on Windows

NOTE: You do not need to backup DB2 on Windows for WOW 6.4. Storing WOW metadata on DB2 on Windows is not formally supported in WOW 6.4.

This section will give an overview of backing up metadata using DB2 on Windows (DB2 is IBM's Database program).

To back-up metadata with IBM WebSphere first goto:
Programs-> IBM DB2 ->Control Center

Once the DB2 Control Center is opened locate the database where your metadata is located. If there are any other databases that are important you can back them up here also. To backup a database simply right click on the database and go to backup-> database. After clicking database another window will open which will be similar to the screenshot below:



Next click browse, enter the location where you want your metadata backup saved to and click backup now. You will get two messages the first telling you the backup has began and the next telling you the backup has been completed.

Disaster Recovery

Disaster recovery procedures involve the following steps.

1. Restore the application server on which WOW is running. Normally, this will be either Tomcat or WebSphere. The WOW 6.4 download bundles all needed components plus Tomcat. This can be used to restore the application server in the event it is needed.
2. Restore or replace the ../webapps/wow64 folder in your environment. This will restore all WOW's application code plus any user specific files. It is recommended to back up your webapps occasionally, as all your custom resources reside here.
3. Restore AS/400 libraries PJSYS64 and PJUSER64 from your backups. These libraries contain metadata used to execute WOW based applications.

Version Control

Managing WOW application from a version control perspective involves the following steps.

Web Resources (Java, JSPs, HTML, etc)

These resources are PC based files residing in your IDE of choice and ultimately in the webapps/wow64 folder of your application server. These files can integrate with any PC version control package such as CVS. These web resources residing in folder wow64 can be saved at a version boundary using a simple windows copy folder operation and archived along with the metadata to provide version control and backup support.

Metadata

Metadata can be achieved at version boundaries with the associated web resources from above. Simply copying the library or schema at a version boundary also provides for version control.

NOTE: Using the application library support described in this manual can enable multiple version support.

Web Application Server Information

WOW has the ability to run on different Web Applications Servers. Web Application Servers are used to put dynamic information on the Web. WOW runs on three major Web Applications Servers. These are IBM WebSphere, Apache Tomcat, and Web Logic. Tomcat is the only free Web Application Server of the three. The following links will direct you to detailed information on the various application servers.

IBM WebSphere

<http://www.ibm.com/websphere>

<http://www.ibm.com/software/webserver>

Apache Tomcat

<http://jakarta.apache.org/tomcat>

<http://jakarta.apache.org>

Web Logic

<http://www.bea.com>

<http://www.bea.com/products/weblogic/server>

Getting Support for WOW

Terms and conditions of WOW support vary based on your individual license agreement. In general technical defect support is provided as part of your purchase while product usage, design, and consulting support are chargeable items.

NOTE: WebSphere, Tomcat, and Weblogic are separate and independent products. Technical support for usage and defects in these products is NOT covered by your WOW license.

Below is the contact information that can be used to get any information that you need concerning WOW.

Email Address:

support@planetjavainc.com

Technical Support by Phone:

760.432.0600

WOW Copyright Information

The Web Object Wizard (WOW) is owned by PlanetJ Corporation, copyright 2003.
This product includes software developed by the Apache Software Foundation
(<http://www.apache.org/>)