K2 SQL SERVER SERVICE BROKER

STORED PROCEDURE REQUIREMENTS

Table of Contents

OVERVIEW

When using the K2 blackpearl OOTB SQL Server Service Broker (Broker) to expose Stored Procedures (SProcs) you will sometimes find methods based on these stored procedures that do not expose the expected properties of the returned datasets. These cases are due to the dynamic nature of the result set schema, which prevents it from being properly mapped to the SmartObject (SmO) properties.

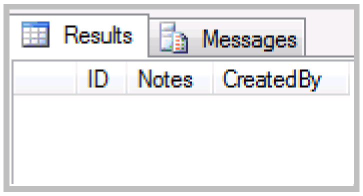
To get the column information, the stored procedure is executed in ADO.Net with System.Data.CommandBehavior.SchemaOnly. This is the same as executing it with SET FMTONLY option. Thus, to ensure that your SProcs results set will be properly exposed the SProc needs to be able to execute without error when SQL’s SET FMTONLY option is set to ON.

SET FMTONLY

When the Broker Instance is created / refreshed it needs to determine the Schema of the objects in the target SQL Database. When the target objects are SProcs it executes them with the SET FMTONLY option set to ON. This returns only metadata to the client and is typically used to test the format of the response without actually executing the query. The following SQL shows an example of executing a SProc using SET FMTONLY

```
SET FMTONLY ON
EXEC spTestDynamicSQL NULL, NULL, NULL
SET FMTONLY OFF
```

The results are shown below.



ID	Notes	CreatedBy

Note that they contain only the column headers and no rows of data. This is the behavior you want to see (no errors and a single result set) to ensure that the SProc will be properly mapped in the Service Broker Instance.

The SET FMTONLY option makes a number of changes to the execution of the procedure, which can cause the results to not be expected. Some of the changes to consider include:

- No conditional logic is execute (IF...ELSE)
- SET variable statements are ignored (SET @NUMBER = 0)
- No statements are executed that would modify data in the database. This includes UPDATE, INSERT, SELECT INTO etc.

For additional information on SET FMTONLY please see: http://technet.microsoft.com/en-us/library/ms173839.aspx.

TIPS AND TRICKS

If you encounter issues getting your Stored Procedure to execute with “SET FMTONLY” the following topics offer some ways that may help you resolve the issue.

TEMP TABLES

Temp tables are one of the most common causes of issues with “SET FMTONLY”. However, this scenario can be subdivided into two classes depending on the data needed: Single Value and Result Set.

When only a single value is required Output Parameters can be used as the current Broker (blackpearl 4.6.6) allows for stored procedures that fail with FMTONLY if they only return parameters and not result sets.

When a result set is required you can create a normal table in the Database that the Stored Procedure can write to. Then you can use two SmO calls, one to execute the stored procedure which writes to the normal table and the second to call that table’s list method to get the results.

CONDITIONAL LOGIC

As mentioned in the section on SET FMTONLY conditional logic statements are not executed when SET FMTONLY is set to on. This can result in unexpected mappings. For example in the following SProc, the lines in red are not executed.

```
CREATE PROCEDURE [GetLookupItems]
      @LookupId int,
      @ParentLookupItemId int = null
as
    if @ParentLookupItemId is null
       begin
              select ID, Value
              from [dbo].[LookupItem]
              where LookupId = @LookupId
       end
    else
       begin
              select ID, Value
              from [dbo].[LookupItem]
              where LookupId = @LookupId and
                     @ParentLookupItemId in (select * from
                     [dbo].Split(ParentLookupItemIDList, ','))
       end

       return 0
```

This results in 4 columns being returned from the two select statements which is not the intended result of two columns. To work around this limitation of SET FTMONLY it is recommended that you attempt to rewrite the SProc to use only one Select statement as shown in the following snippet.

```
ALTER PROCEDURE [GetLookupItems]
      @LookupId AS INT ,
      @ParentLookupItemId AS INT = NULL
AS
      SELECT ID, Value
      FROM [dbo].[LookupItem]
      WHERE LookupId = @LookupId
            AND ( @ParentLookupItemId IS NULL
            OR @ParentLookupItemId IN(
                  select * from [dbo].Split(ParentLookupItemIDList, ',')
            ) )

RETURN 0
```

INPUT PARAMETERS

When a stored procedure has input parameters K2 must provide some default value when executing the SProc with SET FTMONLY. While these values may not be valid runtime values it is important to make sure that the SProc will still execute in a manner that will return the correct schema using these default parameters. So when testing your SProcs with SET FMTONLY it is important to try the same parameter values that the Broker will use. The following table shows the default parameters used when testing several SProcs for this document in blackpearl 4.6.6.

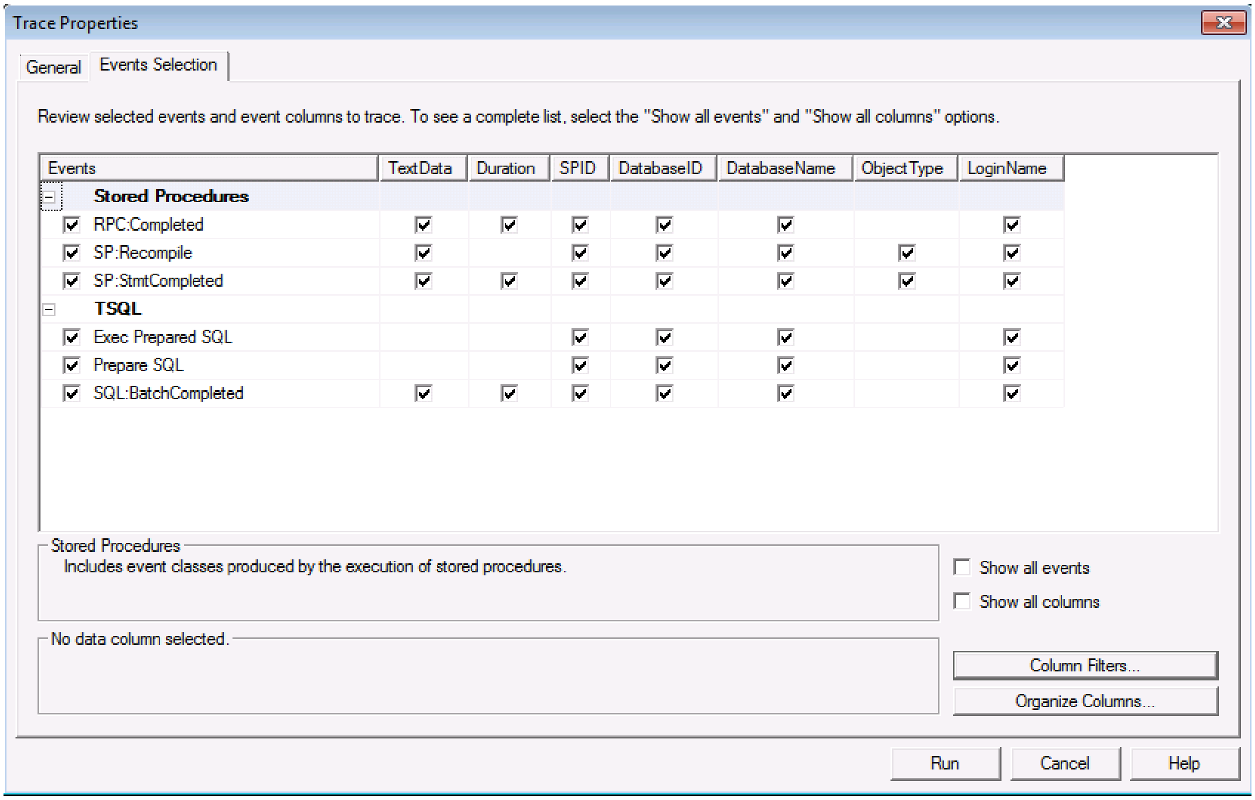
Type	K2 Value User
Int	0
VarChar	NULL

While these values can be used as a timesaver for a quick test they may not match the values used in all situations or versions of K2. As such it is recommended that you confirm the parameter values with SQL profiler as described in the next section.

In one test case the NULL value provided for a VarChar input parameter was resulting in an error. To resolve this error the Parameter was wrapped in an ISNULL statement to provide a valid value such that the execution with SET FMTONLY completed successfully. The caveat here is that you must insure that any parameter substitution does not change the intended operation of the procedure during normal execution.

SQL Profiler

To determine the exact query and parameter values the Broker uses when refreshing a given instance / database, you can use the following settings in a SQL Profiler trace.



Review selected events and event columns to trace. To see a complete list, select the "Show all events" and "Show all columns" options.

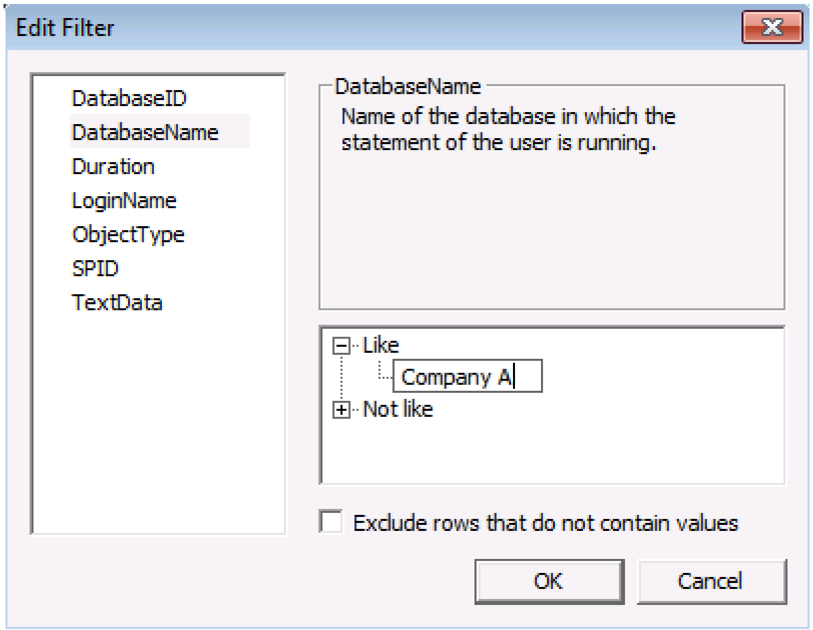
Events	TextData	Duration	SPID	DatabaseID	DatabaseName	ObjectType	LoginName
Stored Procedures							
☑ RPC:Completed	☑	☑	☑	☑	☑		☑
☑ SP:Recompile	☑		☑	☑	☑	☑	☑
☑ SP:StmtCompleted	☑	☑	☑	☑	☑	☑	☑
TSQL							
☑ Exec Prepared SQL			☑	☑	☑		☑
☑ Prepare SQL			☑	☑	☑		☑
☑ SQL:BatchCompleted	☑	☑	☑	☑	☑		☑

Stored Procedures
Includes event classes produced by the execution of stored procedures.

No data column selected.

□ Show all events
□ Show all columns

Make sure to filter on the Database Name that the broker is pointed at and not the K2 database itself. The following screenshot shows a filter used when refreshing against the Company A database.

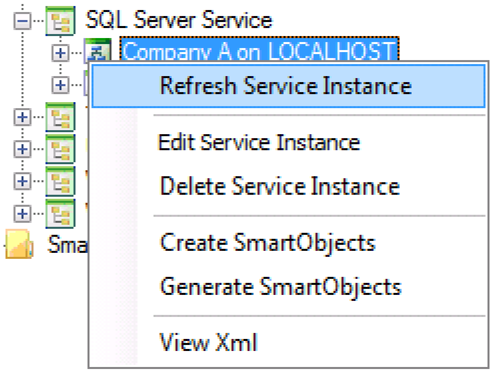


DatabaseName
Name of the database in which the statement of the user is running.

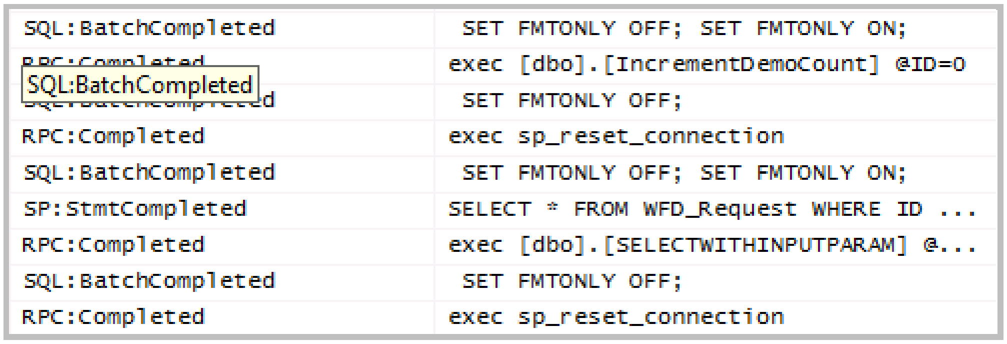
Like
Company A

□ Exclude rows that do not contain values

Once this trace is running use the SmartObject Service Tester to Refresh the Service Instance as shown in the following screenshot.

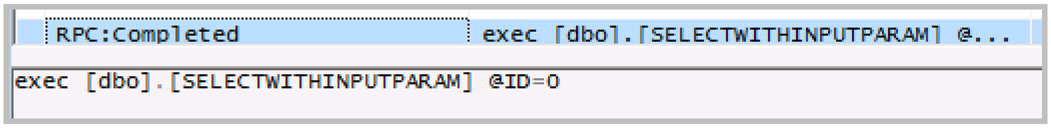


Depending on the number of objects in the Database the Service Instance has permissions to see, this can result in a rather long trace. Look for the block of entries for the stored procedures. These should start with a SET FMTONLY OFF; SET FMTONLY ON; batch. The following screenshot of a trace shows the commands representing two Stored Procedures.



SQL:BatchCompleted	SET FMTONLY OFF; SET FMTONLY ON;
RPC:Completed	exec [dbo].[IncrementDemoCount] @ID=0
SQL:BatchCompleted	SET FMTONLY OFF;
RPC:Completed	exec sp_reset_connection
SQL:BatchCompleted	SET FMTONLY OFF; SET FMTONLY ON;
SP:StmtCompleted	SELECT * FROM WFD_Request WHERE ID ...
RPC:Completed	exec [dbo].[SELECTWITHINPUTPARAM] @...
SQL:BatchCompleted	SET FMTONLY OFF;
RPC:Completed	exec sp_reset_connection

Once you find the name for the procedure you are interested in, you can click on the RPC:Completed row to see the full text of the SQL statement used which will show the parameters passed to the SProc by the Broker as shown below.



RPC:Completed	exec [dbo].[SELECTWITHINPUTPARAM] @...

```
exec [dbo].[SELECTWITHINPUTPARAM] @ID=0
```