Orion and Cirrus

Custom Reports with SQL Server Reporting Services

David Savell
2/5/2008 (version 1.1)

Contents

Introduction

In our organization, the existing custom report generation facilities within Orion were useful, but not flexible enough to accommodate some of our requirements.

Initially these requirements were around availability reporting (including charts) and accessing Web based Cirrus reports (something on the feature request list).

The solution we applied however resulted in far greater flexibility than providing for just these initial requirements. It also resulted in performance improvements when attempting to report on large datasets.

Fundamentally, our solution leveraged two basic principles – the open access provided by Solarwinds to the Orion and Cirrus Databases, and the availability of the Microsoft Reporting Services (either as a free Express Edition, or as part of a licensed SQL server 2005 environment).

Ingredients

SQL Server Reporting Services

We elected to utilize Microsoft SQL Server 2005 Express Edition with Advanced Services as the basis of our Orion/Cirrus report server.

It was installed on a separate server to that of either Cirrus or Orion, and it also has the advantage of being a free download (http://www.microsoft.com/Downloads/details.aspx?familyid=4C6BA9FD-319A-4887-BC75-3B02B5E48A40&displaylang=en)

The free SQL Server Express Edition does however introduce some complexities.

The primary issue is that the Express Edition only supports local tables. Hence the Orion and Cirrus tables you wish to perform reporting on, need to be copied on a regular basis to the reporting server.

As you will see below, this is easily performed using some Data Transformation tasks.

If you have non-Express Editions of SQL Server 2005, this will not be a limitation, and your reports can run directly from the real-time data (and in doing so simplify the configuration).

Be aware that the Express Edition has a database limit of 4Gig.

SQL Server Express Edition Toolkit

The SQL Server Express Toolkit contains SQL Server 2005 Management Studio Express, a graphical management tool, and Business Intelligence Development Studio, a creation and editing environment for reports using SQL Server Reporting services.

Once again, this is a free download (http://www.microsoft.com/Downloads/details.aspx?familyid=3C856B93-369F-4C6F-9357-C35384179543&displaylang=en) – you may have licensed alternatives available.

Software Installation

Many guides are available on the web to assist with the installation of the selected software.

For the Express Edition, I found http://articles.techrepublic.com.com/5100-9592-6102265.html useful.

Once the Database has been installed, Reporting Services needs to be configured.

A good initial guide to Reporting Services can be found at http://www.simple-talk.com/sql/learn-sql-server/beginning-sql-server-2005-reporting-services-part-1/

Obtaining Access to the Data

Thoughtfully Solarwinds provides open access to the SQL Server Databases that sit behind both Orion and Cirrus.

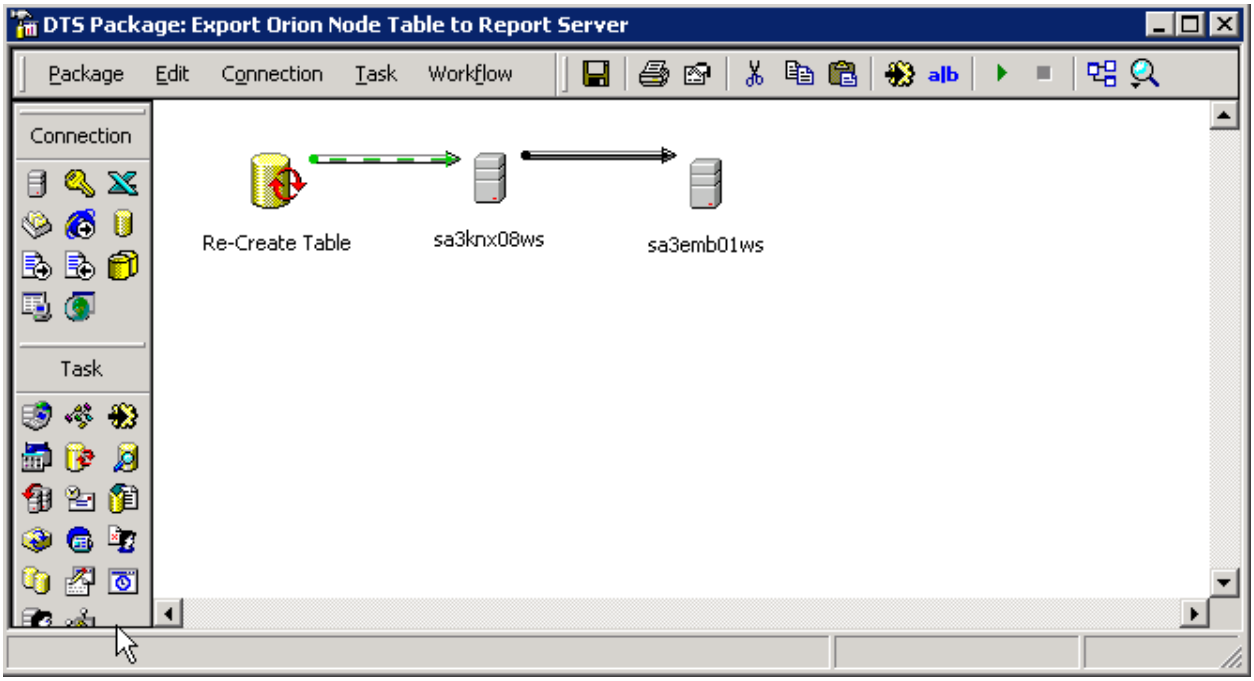
As mentioned above however, if you are using the Express Edition as the basis of your enhanced reporting, then the data will need to be copied into the Reporting Server.

There are a number of ways to achieve this; however we selected a Data Transformation script as an easy option.

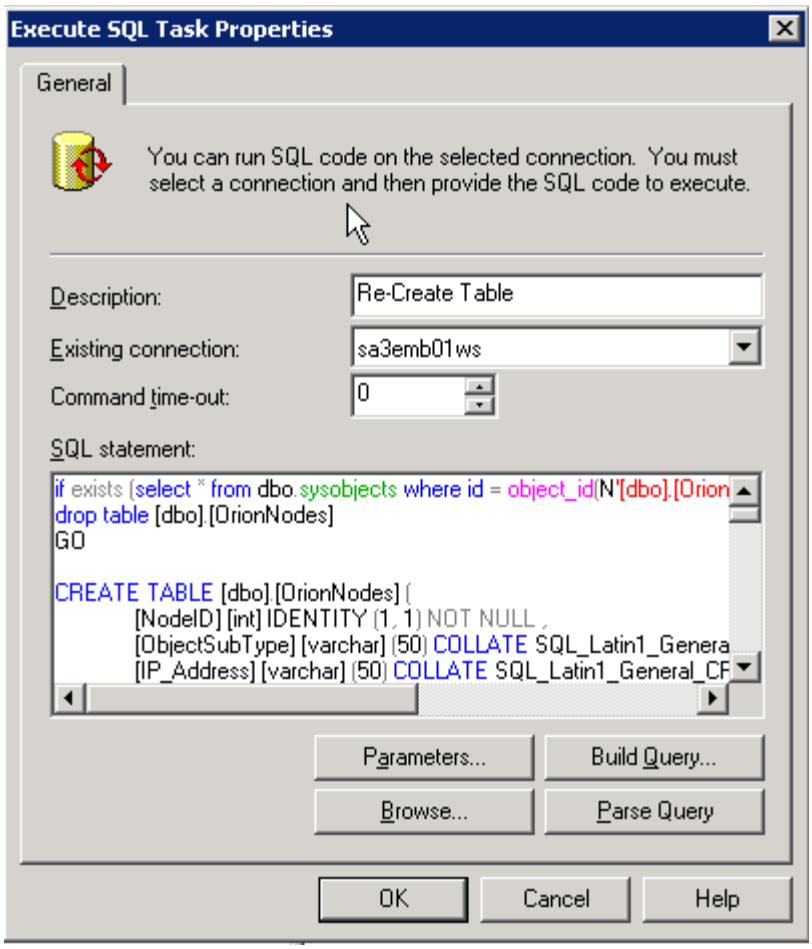
Essentially, we create a table on the Reporting server that matches the table containing the Orion or Cirrus data we wish to report on.

The data is then copied to the Reporting server database (we re-create the table each time the data is copied).

The following is a sample of the DTS script we utilize to copy the Orion Nodes table onto the Report Server.



As can be seen above, a single step is used to create the table with a single transform step (the server sa3knx08ws is the Orion Database server, and sa3emb01ws is the Report Server).



The Re-create Table task contains SQL that first drops the table on the Report Server , and then Creates the table. You could equally truncate the table.

The SQL statement to do this can easily be extracted by utilizing Enterprise Manager to generate the SQL script (right click on the table name, select All Tasks, Generate SQL script).

Please note, as in this example, we prefix the Nodes table with the string 'Orion'. This is because Cirrus and Orion share the same table names in some instances – and if you wish to produce reports on both systems, a unique table name is required.

As the two tables will be identical (except for the name), the transform step is a 1-1 column match.

Creating New Data

It's interesting to note that we are not limited to the raw data provided in the Orion/Cirrus tables – this feature can be utilized to create new or modified datasets from the originally available Orion or Cirrus stored data.

As noted above, one of our primary aims was to provide enhanced availability data and present it in a graphical format.

We also wanted to calculate the availability for groups of Network devices, and to ensure that the figures didn't change from month to month as a result of data roll-up.

To do this, a custom column was created within Orion called 'NetworkName', and each of our devices was assigned to a particular Network (based on logical groupings for our Network).

We then created a DTS script to run on the 1st of every month to calculate the availability for each Network (for the previous month) and store the results (Data Driven Query)

Our table definition looks like this:

```
CREATE TABLE [dbo].[OrionMonthlyNetworkAvailability] (
        [NetworkName] [varchar] (50) COLLATE Latin1_General_CI_AS NOT NULL ,
        [YearMonth] [datetime] NOT NULL ,
        [Availability] [float] NULL
) ON [PRIMARY]
```

And the SQL for the Data Driven DTS:

```
 SELECT NetworkName, dateadd(mm,datediff(mm,0,getdate()),0)-1 as YearMonth,
AVG(ResponseTime.Availability) AS Availability
FROM
Nodes INNER JOIN ResponseTime ON (Nodes.NodeID = ResponseTime.NodeID)
WHERE
( DateTime >= dateadd(mm,datediff(mm,0,getdate())-1,0) AND   DateTime
<dateadd(mm,datediff(mm,0,getdate()),0) )
Group by NetworkName
Having Not NetworkName Is Null
```

Of course this table is not re-created each month – we want to keep the monthly data previously inserted into this table.

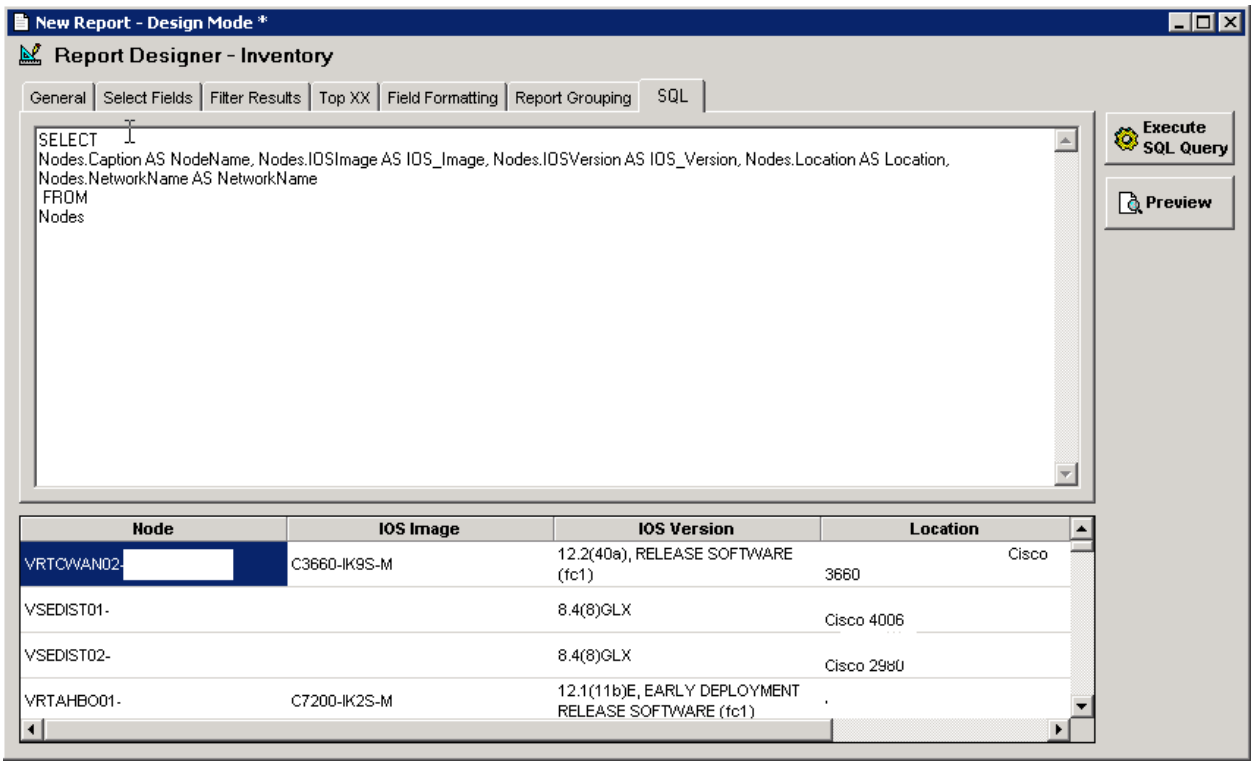
Creating a Report

Once the data is accessible to the Report server, we can then go about creating the appropriate reports. The only limit is the data available on the report server.

Hints

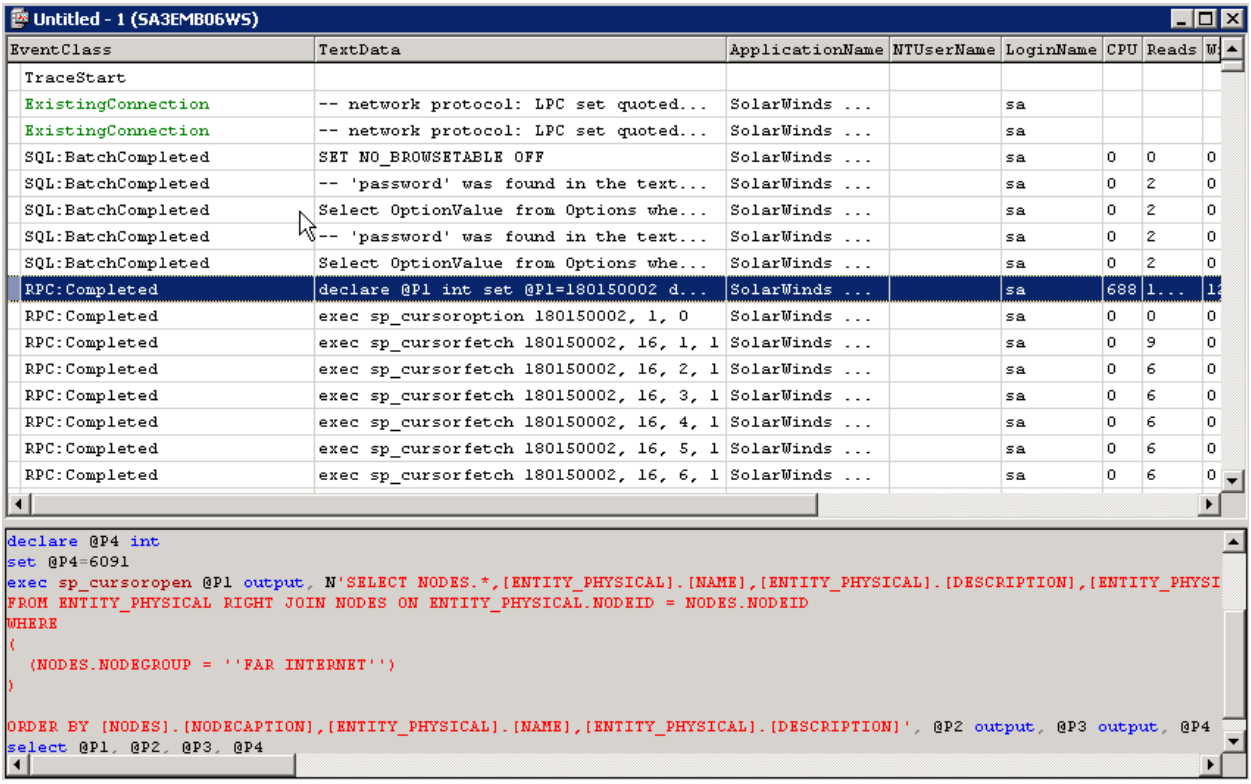
Not knowing SQL is not really a barrier to creating reports with Reporting Server. Even when knowledge fails – you can cheat.

For example, the Orion custom report writer can be used to mock up a report, and then use the SQL view as the starting point for you Reporting Services report (as below).



With Cirrus, it's not quite as easy in that the Report writer doesn't give you the SQL view.

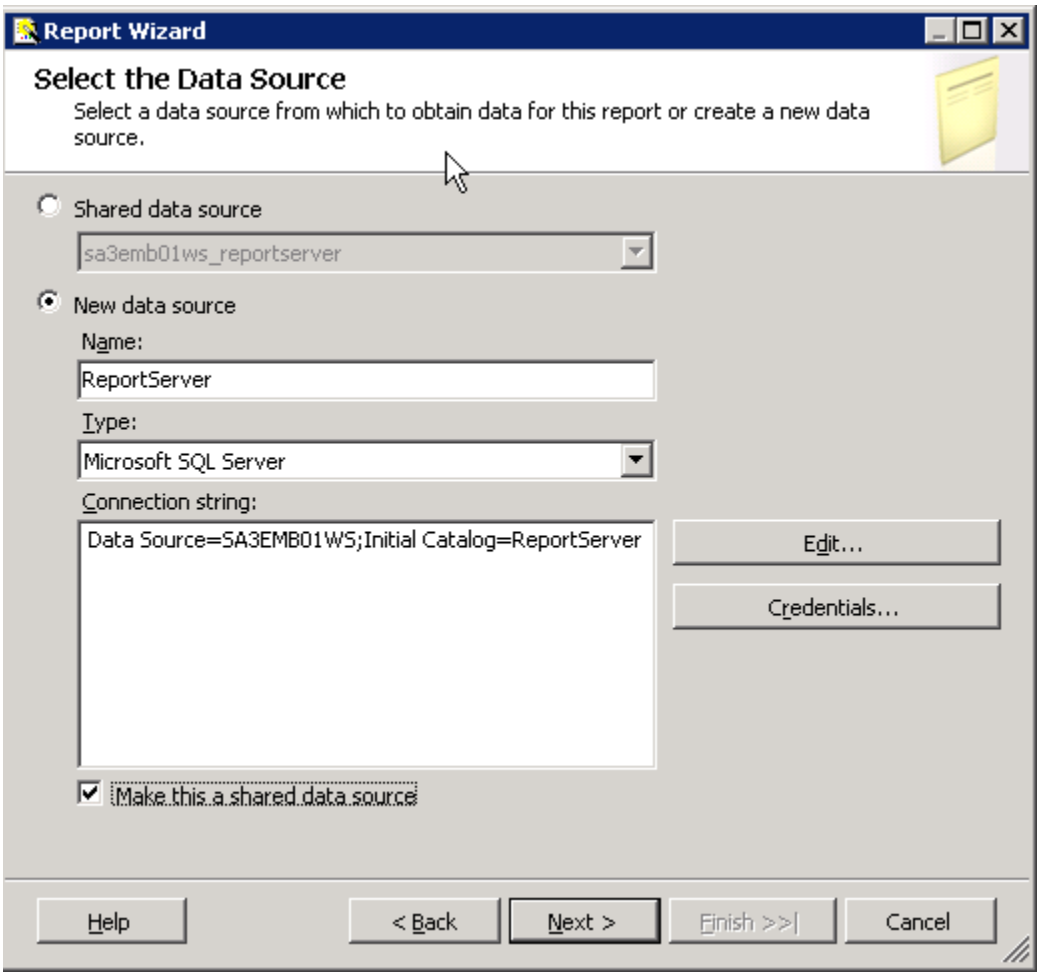
A quick SQL trace however will show the SQL statement issued to the database:



Once to have an idea of what your report is going to look like, it's time to utilize Microsoft Visual Studio 2005 to create the report.

A Simple Report

The first thing required (after you have created a project), is the creation of the datasource.

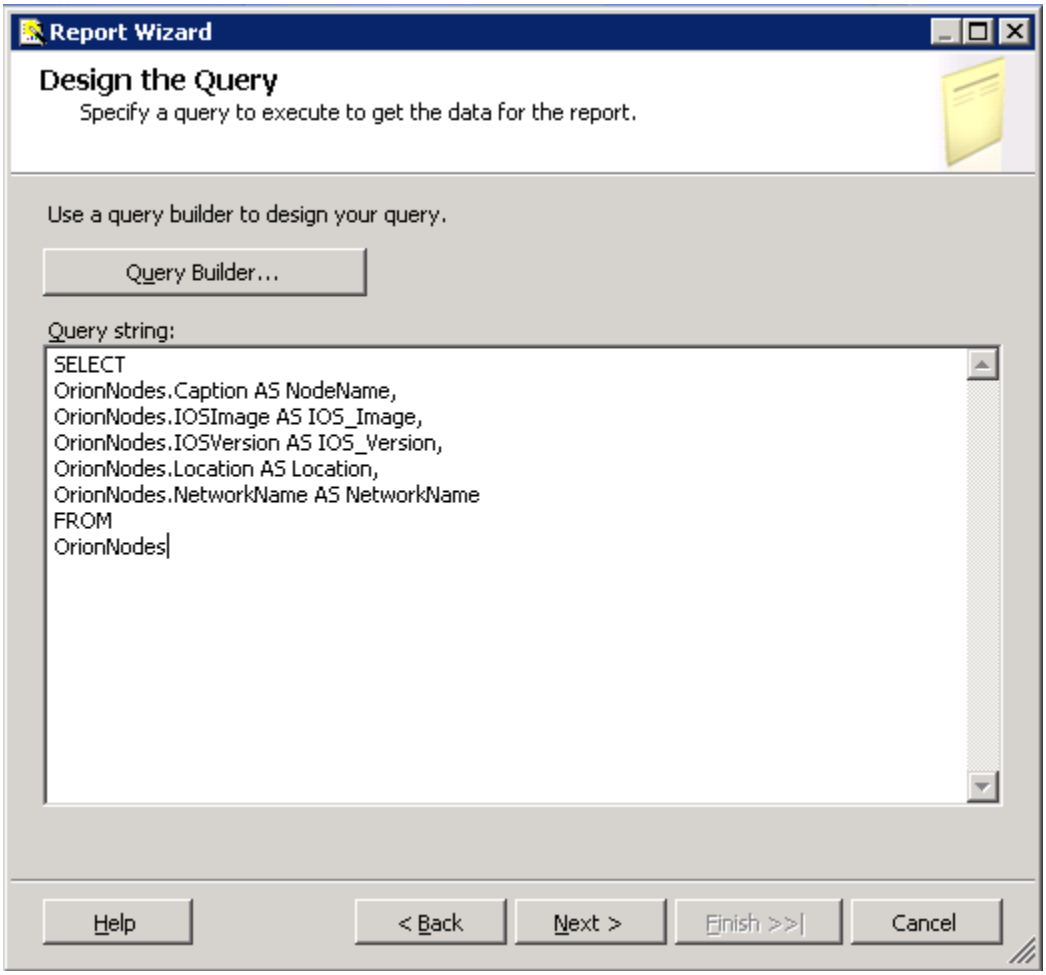


As above, make it a shared data source, so your next report can utilize it (and you don't have to create another one). You will need to edit the connection string and test the connection prior to proceeding.

Once the data source has been created, the report wizard to design the query is presented.

You can elect to enter an SQL query sourced from else-where, or create you own (using the query builder if required).

Don't forget, if you paste SQL from the Orion Report builder, that table names may not be quite the same as the tables on the report server (i.e. in our case we prefix the table names with Orion or Cirrus).

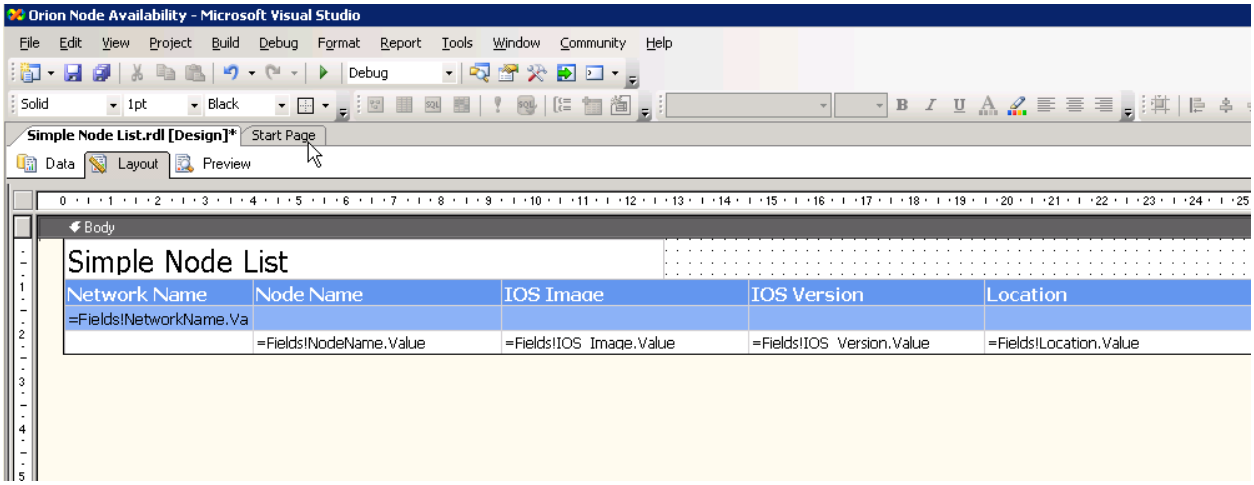


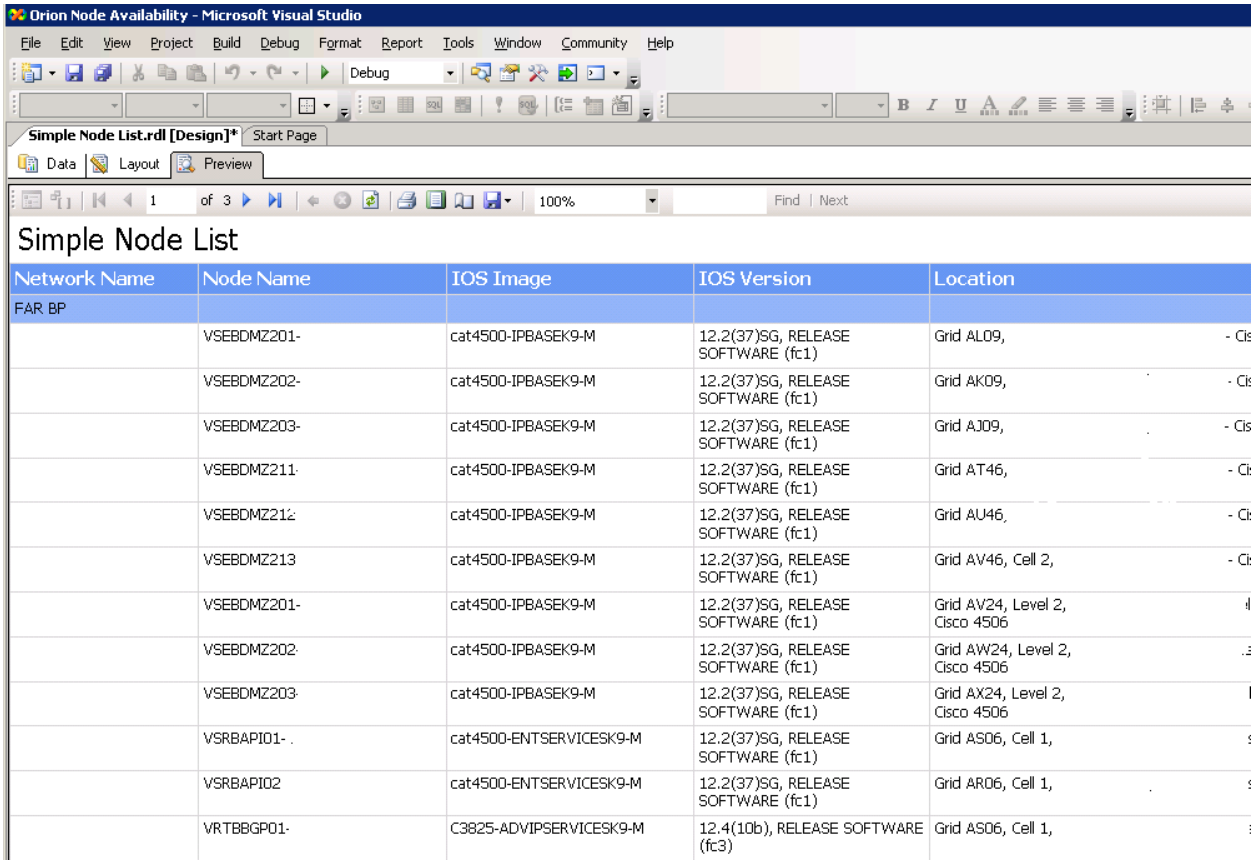
In the above example, I copied in the SQL from the Orion custom report writer (changing the table names appropriately).

A number of screens are then presented to select the way you wish the data to be presented. All of these options can be changed once the wizard has been completed (including the SQL statement).

In this example I elect to group the list by NetworkName (our custom column) and selected some formatting option.

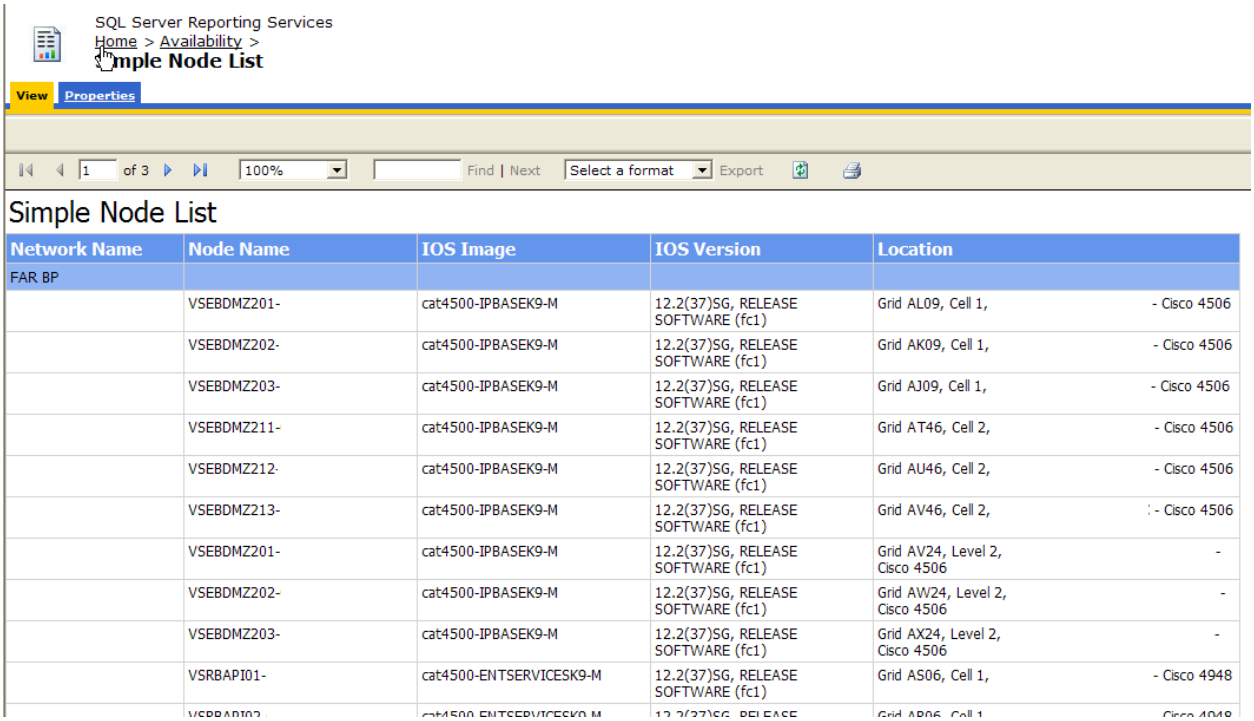
The below shows the end layout result, with the preview screen following .





Network Name	Node Name	IOS Image	IOS Version	Location
FAR BP				
	VSEBDMZ201-	cat4500-IPBASEK9-M	12.2(37)SG, RELEASE SOFTWARE (fc1)	Grid AL09,
	VSEBDMZ202-	cat4500-IPBASEK9-M	12.2(37)SG, RELEASE SOFTWARE (fc1)	Grid AK09,
	VSEBDMZ203-	cat4500-IPBASEK9-M	12.2(37)SG, RELEASE SOFTWARE (fc1)	Grid AJ09,
	VSEBDMZ211-	cat4500-IPBASEK9-M	12.2(37)SG, RELEASE SOFTWARE (fc1)	Grid AT46,
	VSEBDMZ212-	cat4500-IPBASEK9-M	12.2(37)SG, RELEASE SOFTWARE (fc1)	Grid AU46,
	VSEBDMZ213	cat4500-IPBASEK9-M	12.2(37)SG, RELEASE SOFTWARE (fc1)	Grid AV46, Cell 2,
	VSEBDMZ201-	cat4500-IPBASEK9-M	12.2(37)SG, RELEASE SOFTWARE (fc1)	Grid AV24, Level 2, Cisco 4506
	VSEBDMZ202-	cat4500-IPBASEK9-M	12.2(37)SG, RELEASE SOFTWARE (fc1)	Grid AW24, Level 2, Cisco 4506
	VSEBDMZ203-	cat4500-IPBASEK9-M	12.2(37)SG, RELEASE SOFTWARE (fc1)	Grid AX24, Level 2, Cisco 4506
	VSRBAPI01-	cat4500-ENTSERVICESK9-M	12.2(37)SG, RELEASE SOFTWARE (fc1)	Grid AS06, Cell 1,
	VSRBAPI02	cat4500-ENTSERVICESK9-M	12.2(37)SG, RELEASE SOFTWARE (fc1)	Grid AR06, Cell 1,
	VRTBBGP01-	C3825-ADVIPSERVICESK9-M	12.4(10b), RELEASE SOFTWARE (fc3)	Grid AS06, Cell 1,

Once happy with the report, it can then be deployed to the Reporting server (using the build/deploy toolbar options) for web access.



As can be seen, a number of standard Find and Export options are also available from the report page.

The non-Express versions also have additional options available (i.e. report scheduling and e-mail functions)

A Complex Report

Previously we created some monthly availability data (for each logical Network group we defined via a custom column).

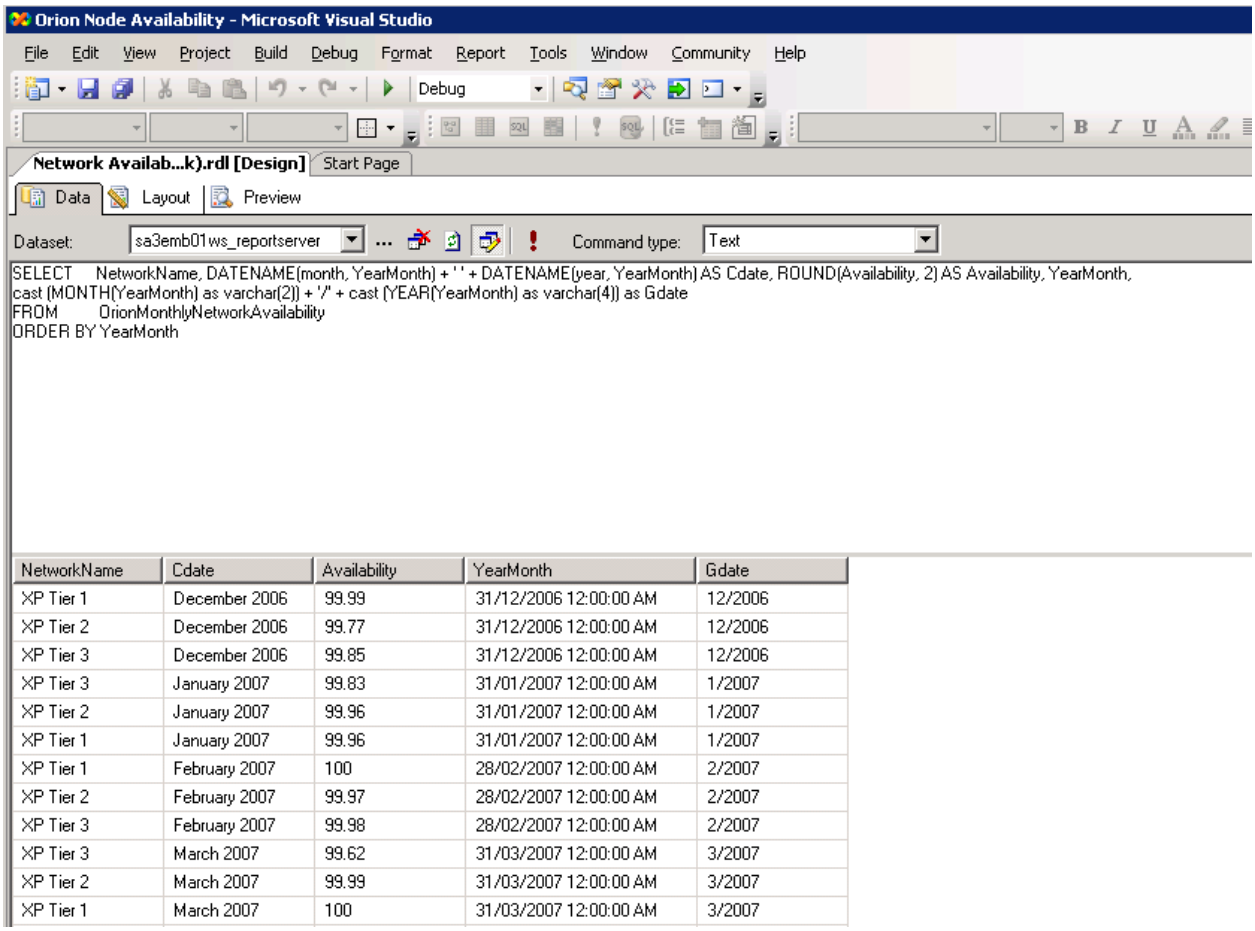
I now want to report on this data, and create a graph for each Network for its monthly availability (based upon the table and data previously described).

The SQL for this report looks like this:

```
SELECT    NetworkName, DATENAME(month, YearMonth) + ' ' + DATENAME(year, YearMonth) AS Cdate,
ROUND(Availability, 2) AS Availability, YearMonth,
cast (MONTH(YearMonth) as varchar(2)) + '/' + cast (YEAR(YearMonth) as varchar(4)) as Gdate
FROM       OrionMonthlyNetworkAvailability
ORDER BY YearMonth
```

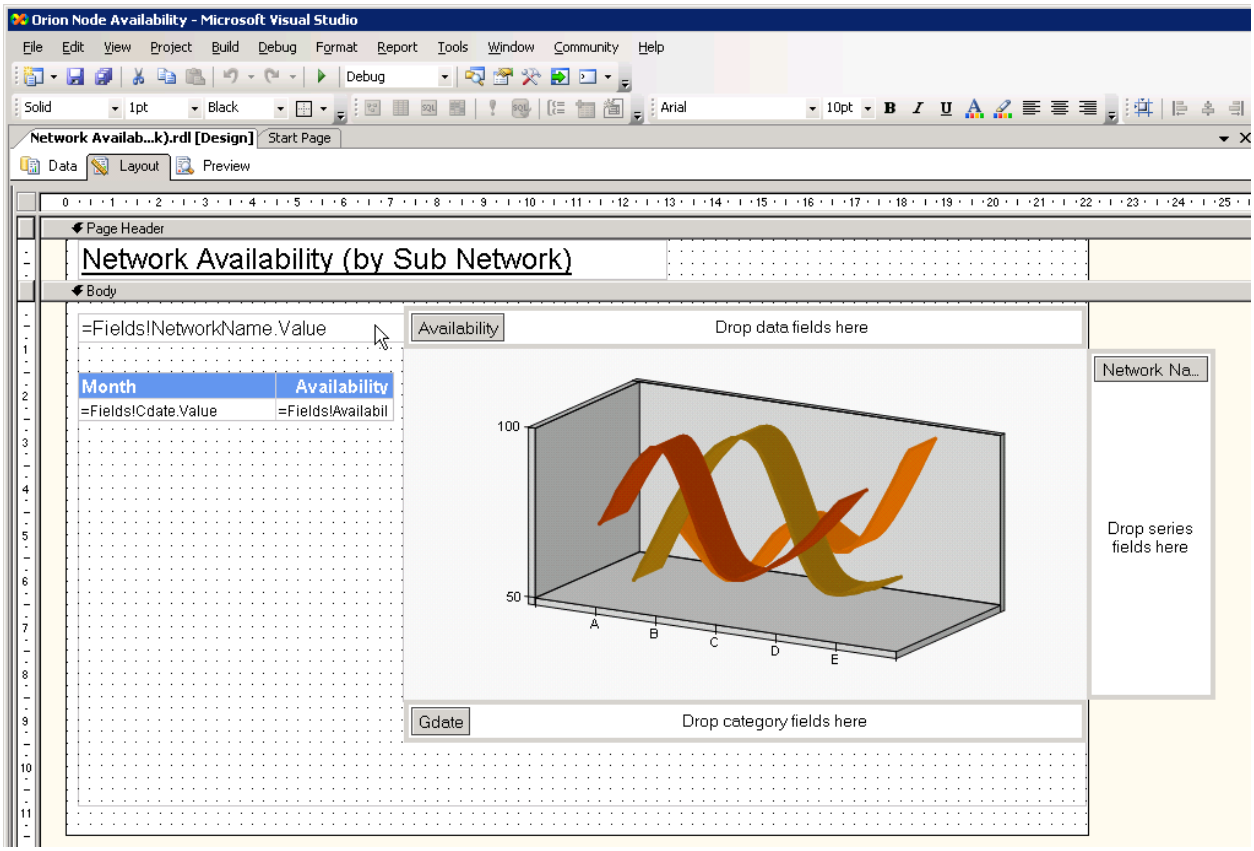
The SQL is ugly, mainly as a result of needing to manipulate the year and month values to ensure I could sort the data correctly (and label it correctly) – I'm sure this could have been done better….

The output looks like the following:



NetworkName	Cdate	Availability	YearMonth	Gdate
XP Tier 1	December 2006	99.99	31/12/2006 12:00:00 AM	12/2006
XP Tier 2	December 2006	99.77	31/12/2006 12:00:00 AM	12/2006
XP Tier 3	December 2006	99.85	31/12/2006 12:00:00 AM	12/2006
XP Tier 3	January 2007	99.83	31/01/2007 12:00:00 AM	1/2007
XP Tier 2	January 2007	99.96	31/01/2007 12:00:00 AM	1/2007
XP Tier 1	January 2007	99.96	31/01/2007 12:00:00 AM	1/2007
XP Tier 1	February 2007	100	28/02/2007 12:00:00 AM	2/2007
XP Tier 2	February 2007	99.97	28/02/2007 12:00:00 AM	2/2007
XP Tier 3	February 2007	99.98	28/02/2007 12:00:00 AM	2/2007
XP Tier 3	March 2007	99.62	31/03/2007 12:00:00 AM	3/2007
XP Tier 2	March 2007	99.99	31/03/2007 12:00:00 AM	3/2007
XP Tier 1	March 2007	100	31/03/2007 12:00:00 AM	3/2007

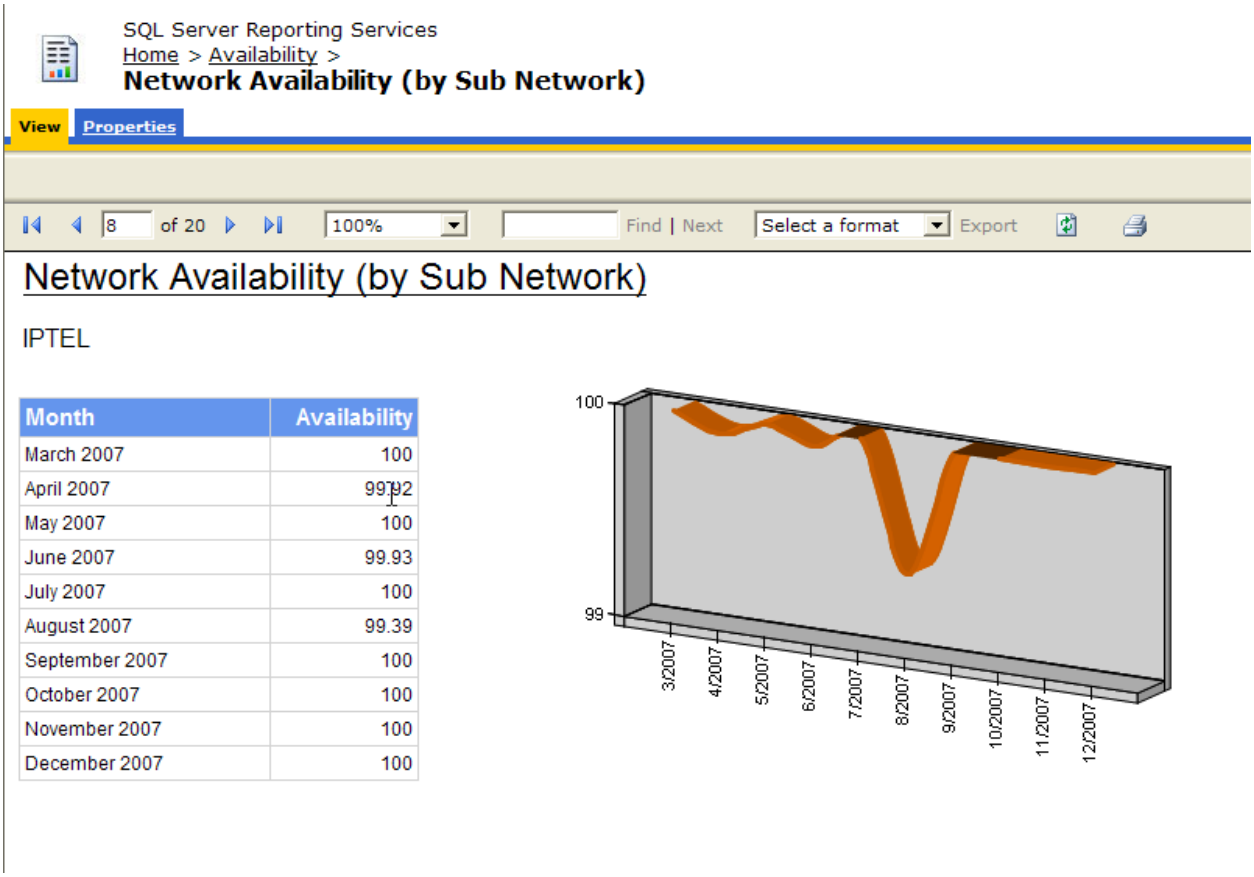
The report layout looks like the following. You will notice I have also created a graph using the same data as displayed in the table.



As previously, I once again grouped by NetworkName, so that each page would provide availability statistics for a single NetworkName.

As can be seen in subsequent examples, a drop down list could equally have been used to select the data for each unique Network (and thus only render the page requested).

The resultant published report with a page for each Network:



A More Complex Report

To date, we have just dealt with Orion reports. We also wished however to produce reports using Cirrus data (and even combine data outputs if required).

The reporting structure we have setup above allows this easily.

By mirroring Cirrus tables using DTS scripts to the report server (exactly in the fashion described for the Orion tables), we can accomplish this.

The following example shows how we can produce a combined Inventory report using information from multiple Cirrus tables. While this example contains no Orion data, it could have been included by utilizing table joins on common fields such as the Sysname (which is obtained from the devices via a snmp query by both Orion and Cirrus).

For this particular report, we wanted three tables of inventory information to be displayed on the same report (chassis information, physical entities and imageMIB information).

We also wanted to display the report based upon a device name selection.

This required four datasets to be defined for the report (the previous reports only utilized a single dataset).

Physical Entity:

```
SELECT     CirrusNodes.SysName, CirrusEntity_Physical.Name, CirrusEntity_Physical.Description,
CirrusEntity_Physical.Type,
                      CirrusEntity_Physical.ContainedIn, CirrusEntity_Physical.Class, CirrusEntity_Physical.Position,
CirrusEntity_Physical.HardwareRevision,
                      CirrusEntity_Physical.FirmwareRevision, CirrusEntity_Physical.SoftwareRevision,
CirrusEntity_Physical.Serial, CirrusEntity_Physical.Manufacturer,
                      CirrusEntity_Physical.Model, CirrusEntity_Physical.Alias, CirrusEntity_Physical.AssetID,
CirrusEntity_Physical.FieldReplaceable
FROM       CirrusEntity_Physical RIGHT OUTER JOIN
                      CirrusNodes ON CirrusEntity_Physical.NodeID = CirrusNodes.NodeID
WHERE     (CirrusNodes.SysName = @selectdevice)
ORDER BY CirrusEntity_Physical.Name, CirrusEntity_Physical.Description
```

ImageMIB:

```
SELECT    CirrusNodes.SysName, CirrusCisco_ImageMIB.Name, CirrusCisco_ImageMIB.Value
FROM       CirrusNodes INNER JOIN
                CirrusCisco_ImageMIB ON CirrusNodes.NodeID = CirrusCisco_ImageMIB.NodeID
WHERE    (CirrusNodes.SysName = @selectdevice)
AND (CirrusCisco_ImageMIB.Name NOT IN ('BEGIN', 'END', 'MAGIC'))
ORDER BY CirrusNodes.NodeCaption, CirrusNodes.MachineType, CirrusNodes.SysLocation,
CirrusCisco_ImageMIB.ImageIndex
```

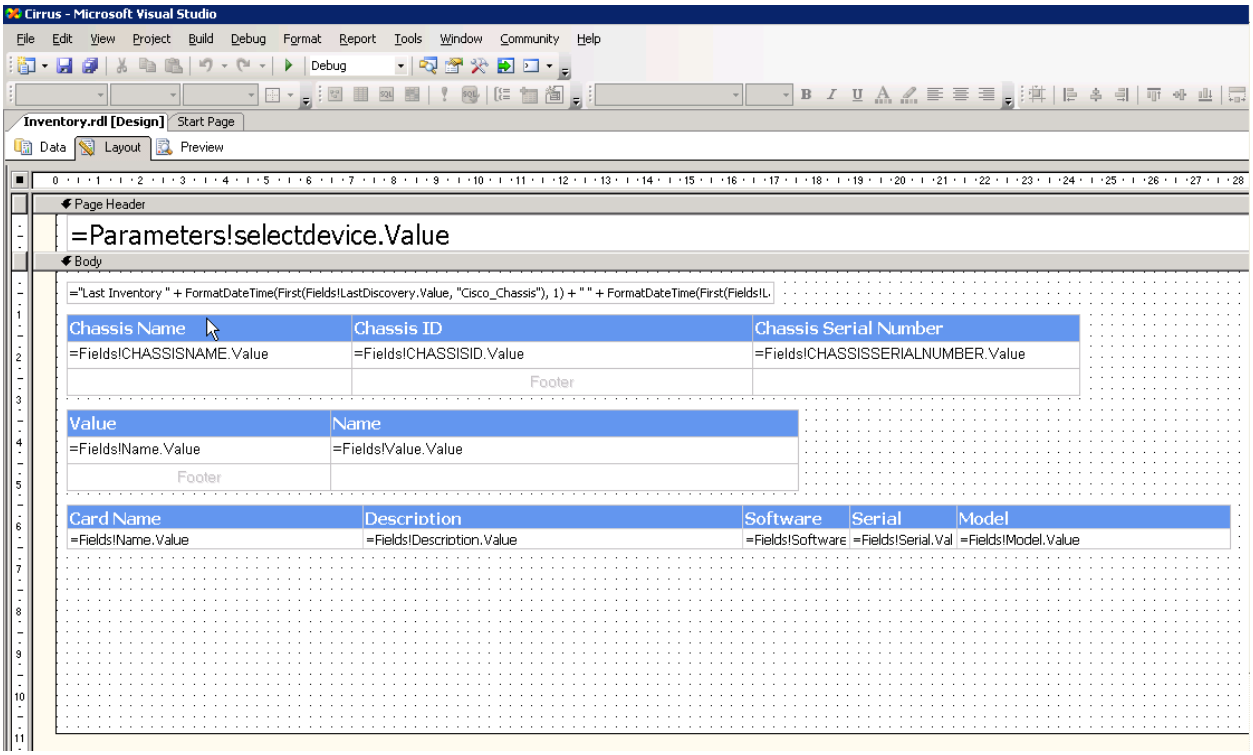
CiscoChassis:

```
SELECT
CirrusNODES.SysName,[CirrusCISCO_CHASSIS].[CHASSISNAME],[CirrusCISCO_CHASSIS].[CHASSISID],
[CirrusCISCO_CHASSIS].[CHASSISSERIALNUMBER], [CirrusCISCO_CHASSIS].[LastDiscovery]
FROM CirrusNODES LEFT JOIN CirrusCISCO_CHASSIS ON CirrusNODES.NODEID =
CirrusCISCO_CHASSIS.NODEID
WHERE  ( CirrusNODES.SysName = @selectdevice )
```

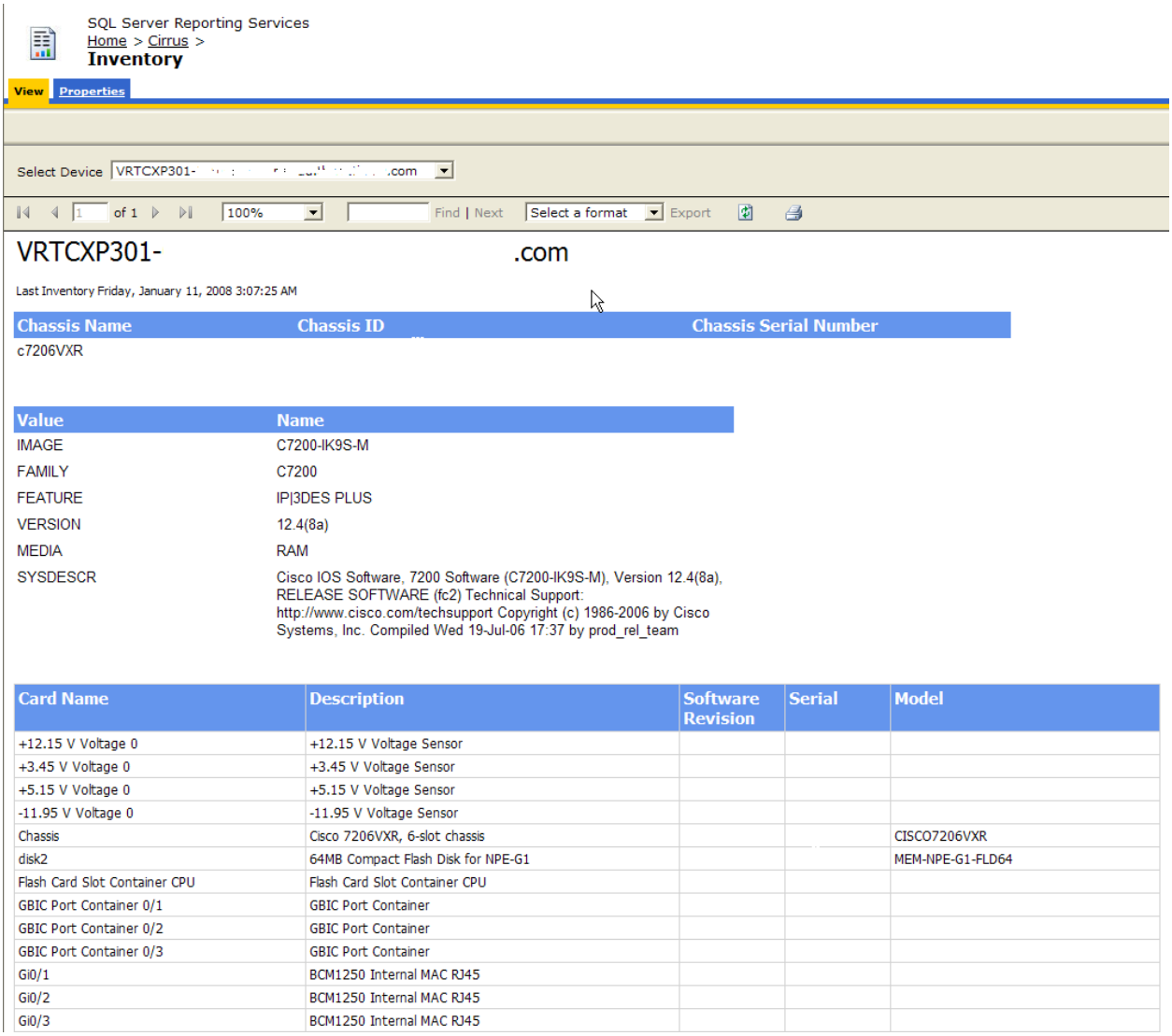
A dataset also was required to populate the report selection list:

```
SELECT    SysName AS selectdevice
FROM       CirrusNodes
WHERE    (SysName <> '')
ORDER BY selectdevice
```

Put together, a layout like that below can be created:



With the resultant report (apologies for the obscured data in some slides for confidentiality reasons):



SQL Server Reporting Services
Home > Cirrus >
Inventory

View | Properties

Select Device: VRTCXP301- [redacted] .com

1 of 1 | 100% | Find | Next | Select a format | Export

VRTCXP301- [redacted] .com

Last Inventory Friday, January 11, 2008 3:07:25 AM

Chassis Name	Chassis ID	Chassis Serial Number
c7206VXR	[redacted]	

Value	Name
IMAGE	C7200-IK9S-M
FAMILY	C7200
FEATURE	IP|3DES PLUS
VERSION	12.4(8a)
MEDIA	RAM
SYSDESCR	Cisco IOS Software, 7200 Software (C7200-IK9S-M), Version 12.4(8a), RELEASE SOFTWARE (fc2) Technical Support: http://www.cisco.com/techsupport Copyright (c) 1986-2006 by Cisco Systems, Inc. Compiled Wed 19-Jul-06 17:37 by prod_rel_team

Card Name	Description	Software Revision	Serial	Model
+12.15 V Voltage 0	+12.15 V Voltage Sensor			
+3.45 V Voltage 0	+3.45 V Voltage Sensor			
+5.15 V Voltage 0	+5.15 V Voltage Sensor			
-11.95 V Voltage 0	-11.95 V Voltage Sensor			
Chassis	Cisco 7206VXR, 6-slot chassis		[redacted]	CISCO7206VXR
disk2	64MB Compact Flash Disk for NPE-G1		[redacted]	MEM-NPE-G1-FLD64
Flash Card Slot Container CPU	Flash Card Slot Container CPU			
GBIC Port Container 0/1	GBIC Port Container			
GBIC Port Container 0/2	GBIC Port Container			
GBIC Port Container 0/3	GBIC Port Container			
Gi0/1	BCM1250 Internal MAC RJ45			
Gi0/2	BCM1250 Internal MAC RJ45			
Gi0/3	BCM1250 Internal MAC RJ45			

Note that I have also used some additional data from within the datasets to populate the Last Inventory date and the device name contained within the report title.

The device selection is also achieved via a drop down select list (using the report selection dataset), something that could also be utilized in the Network Availability reports if required.

Other Reports

Our organization utilizes SQL Reporting services for a number of web based reporting functions. No doubt there are many more possibilities.

Reconciliation

Provide a report on Devices that exist in Cirrus, but not Orion. We also added a custom column in Orion 'Exclude Cirrus' to indicate devices that should not exist in Cirrus (i.e. Firewalls that we monitor from Orion, but do not manage from Cirrus).

The SQL:

```
SELECT OrionNodes.IP_Address, OrionNodes.Caption, OrionNodes.NetworkName,
OrionNodes.ExcludeCirrus
FROM OrionNodes
LEFT JOIN CirrusNodes ON OrionNodes.[IP_Address] = CirrusNodes.[AgentIP]
WHERE (((CirrusNodes.AgentIP) Is Null)) and OrionNodes.ExcludeCirrus = 0
AND
 OrionNodes.NetworkName <> ' '
```

Notice that this also excludes Orion devices that do not have a Network Name assigned – this information is provided in another reconciliation report.

Alternatively, the reverse direction (exists in Cirrus but not Orion):

```
SELECT CirrusNodes.NodeGroup, CirrusNodes.SysName, CirrusNodes.AgentIP
FROM CirrusNodes
LEFT JOIN OrionNodes ON CirrusNodes.[AgentIP] = OrionNodes.[IP_Address]
WHERE (((OrionNodes.IP_Address) Is Null));
```

Compliance Reporting

Our organization makes extensive use of the Cirrus Compliance Reporting feature. However, we did not wish to provide access to a large number of users – rather a web based reporting function was required (something that Cirrus at v4 currently is lacking). In addition, we wanted to reference the source document (i.e. policy document) with linkage to the compliance rule.

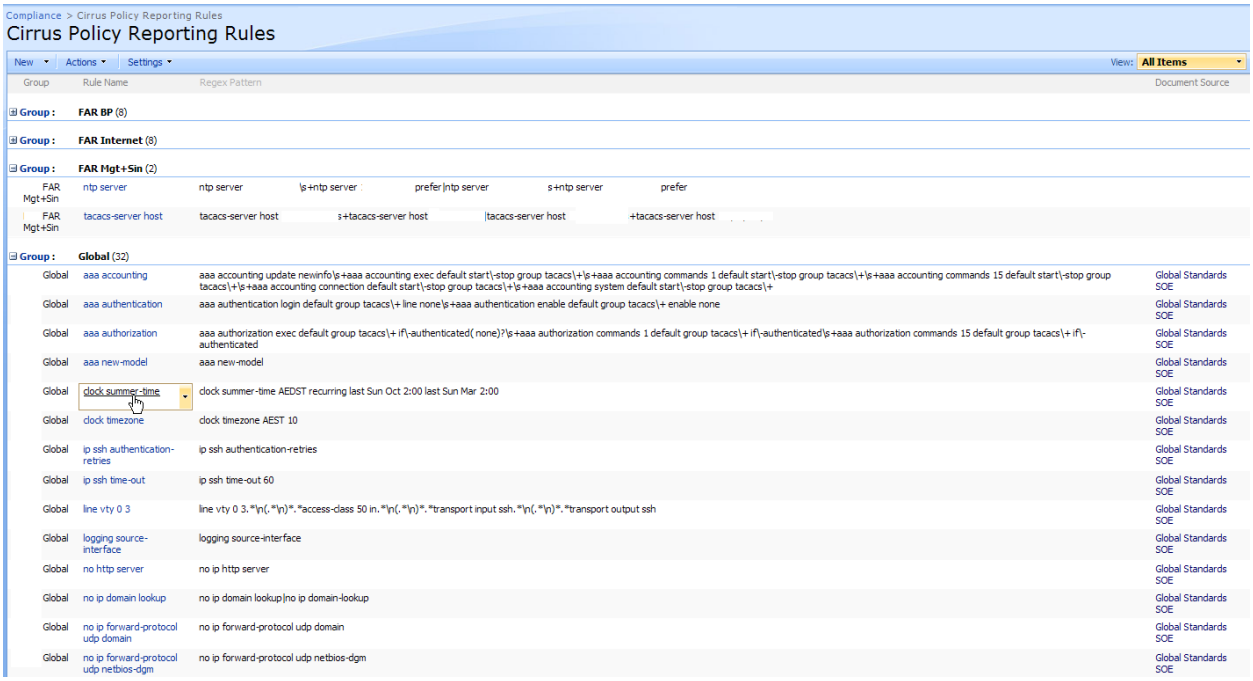
We created a Microsoft Sharepoint site that contains the source documents, linked to the policy rules. The site also receives the daily e-mailed policy reports from Cirrus.

As a result, we needed to ensure that the Policy rules on the web site, matched those contained within Cirrus.
Once again we export both the Cirrus policy rules table and the SharePoint list nightly, and produce an exception report for non-matched rules.

```
SELECT SharepointPolicyRulesList.[group], SharepointPolicyRulesList.[pattern]
FROM SharepointPolicyRulesList
LEFT JOIN CirrusPolicyRules
ON
(SharepointPolicyRulesList.[pattern] = CirrusPolicyRules.[pattern]
AND
SharepointPolicyRulesList.[group] = CirrusPolicyRules.[grouping])
WHERE (((CirrusPolicyRules.pattern) Is Null))
```

You will note that we also utilize grouping, so that we can have a Global policy rules document, underpinned with individual policy rule documents for Networks that require exemptions or exclusions from the Global policies.

Once again I've edited the screen dump to protect the innocent; however enough remains to understand the general picture.



Orion Duplicates

With Multiple pollers and 4000+ devices, there are occasions where we end up with duplicate nodes entered within Orion.

The SQL:

```
SELECT    Caption, COUNT(Caption) AS NumberOfDups
FROM      OrionNodes
GROUP BY Caption
HAVING    (COUNT(Caption) > 1)
```

IP Address and Interface Report

A very useful report from Cirrus that summarizes IP address and Interface information. While this information can be obtained from Cirrus directly (via the reports function), we found that this information in our configuration (i.e. large number of nodes and interfaces) took over 20 minutes to be displayed on the screen.
SQL Reporting services allows us to select via either device name or IP address as parameters, and takes a few seconds to display the result.
When support staff need to find an IP address- quick access is critical (and it also means that we they don't need to access Cirrus directly).

The SQL:

```
SELECT
CirrusNODES.SysName,[CirrusIPADDRESSES].[IPADDRESS],[CirrusIPADDRESSES].[REVERSEDNS],[CirrusINTERFACES].[INTERFACENAME]
FROM (CirrusIPADDRESSES RIGHT JOIN CirrusNODES ON CirrusIPADDRESSES.NODEID = CirrusNODES.NODEID) LEFT JOIN CirrusINTERFACES ON (CirrusIPADDRESSES.INTERFACEINDEX = CirrusINTERFACES.INTERFACEINDEX) AND (CirrusIPADDRESSES.NODEID = CirrusINTERFACES.NODEID)
WHERE  (  (CirrusNODES.SysName = @selectsysname or @selectsysname IS null ) or (CirrusIPADDRESSES.IPADDRESS = @specifyipaddress OR (@specifyipaddress IS null and @selectsysname IS NULL)  ) )
AND CirrusNODES.Sysname <> ''
ORDER BY
[CirrusNODES].[NODECAPTION],[CirrusIPADDRESSES].[IPADDRESS],[CirrusIPADDRESSES].[REVERSEDNS],
CirrusIPADDRESSES.INTERFACEINDEX
```

You can see the two parameters that are used on the web report (@selectsysname and @specifyipaddress).
As a sidenote, we use an interesting technique in the selectsysname dataset to ensure that a null can be presented in the list should just an IP address to entered (i.e. search all devices).

```
SELECT distinct CirrusNodes.SysName as selectsysname
FROM CirrusNODES
UNION
SELECT NULL
ORDER by SysName asc
```

Integration with Orion

The simplest method of providing access to the SQL Reporting Server is via the creation of a custom menu item within Orion.

It would be nice to retain the Solarwinds menu bars when accessing the report server (i.e. open the page within a frame), however I have not spent any time looking at this as yet.

We also link to the Report Server from our Intranet Sharepoint site – SharePoint's web part support does provide this function

Notes

SQL Reporting Services access security has not been touched upon. There are options available (i.e. domain security), however these will be site specific. Suggest that the Microsoft web site or Google is your friend here.

This document also does not attempt to teach the user how to create reports on a step by step basis. Once again there are much better guides available on the internet than I could hope to re-create here. This document merely attempts to show what is possible with SQL Reporting Services when creating custom reports based on data from Orion or Cirrus.

There are a large number of third party .NET components to make you reports both faster to create and pretty. The scope is only limited by your budget. You can go a long way however with the free tools available.