Table of Contents

ActiveReports 10 Server User Guide

ActiveReports Server allows you to provide a browser-based report designer for end users and to serve up both end user reports and your more sophisticated developer reports. The Class Library contains documentation and code samples for the ActiveReports 10 Server API.

In This Documentation

URL, Audit settings and server diagnostics for your site.

How To

Quickly learn how to perform specific tasks with ActiveReports 10 Server.

Samples and Walkthroughs

Learn to create and access Web services using the ActiveReports 10 Server.

Troubleshooting

Learn how to resolve some common issues with ActiveReports 10 Server.

Class Library (on-line documentation)

View the API documentation on the report list and designer controls.

Web Service Library

View the API documentation for the ActiveReports 10 Server ReportService.

Section 508 Compliance

Learn about Section 508 Compliance.

Licensing Agreement

Please see the ActiveReports 10 Server Licensing Agreement on our web site for full details about licensing for each edition.

Acknowledgements

Microsoft, Windows, Visual Studio, and Microsoft SQL Server are either registered trademarks or trademarks of Microsoft Corporation in the United States and/or other countries.

Features and Benefits

Here are the most important ActiveReports Server features that make an administrator's life easier.

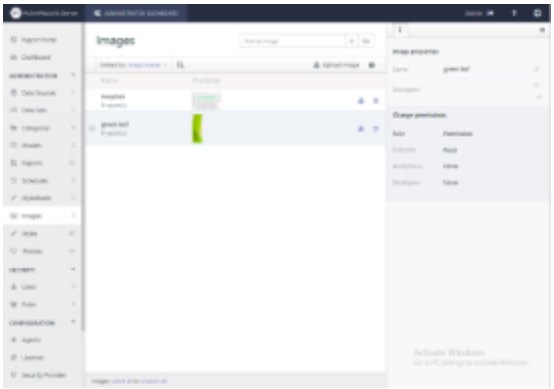
Feature	Benefit
A report designer designed especially for end users that don't use SQL.	End users can create and modify reports without depending on IT.
Completely browser-based designer. No installation required.	Fast and easy deployment in production. No updates to users' desktops.
A logical data model from your database that users understand.	End users can query, sort, and filter data using familiar terms. No knowledge of database schema or SQL required.
Automatic generation of logical data models from your database.	IT can generate and then customize and extend the data models shown to end users.
Schedule & distribute reports to specific people at specific times.	Timely distribution of information to business users when they need it most.
Self-managed clustering & load-balancing.	Cost-effective and easy to manage scalability for a few or a hundred users.

Role-based security for reports and data models.	IT makes sure that only authorized users can create and view different reports.
Single sign-on security provider.	Allow users to sign in with their existing user name and password.
A set of ASP.NET controls for embedding reports and report designer.	Programmers can rapidly integrate reporting features into ASP.NET applications.
Flexible web service access to the reporting platform.	Programmers can exercise greater control integrating reporting into their applications.
Compatibility with ActiveReports.	ActiveReports Server supports all types of reports provided by ActiveReports. Publish existing ActiveReports report files for centralized storage, distribution, security, and scalability.
Install and configure in less than 10 minutes.	Install and configure a working server in your own environment to evaluate in minutes - not days.
Fully responsive HTML5 Report Portal and Admin Dashboard	ActiveReports Server provides fully responsive HTML5 Mobile user interface which makes it possible to access the Report Portal and Admin Dashboard from any touch enabled device.

What's New

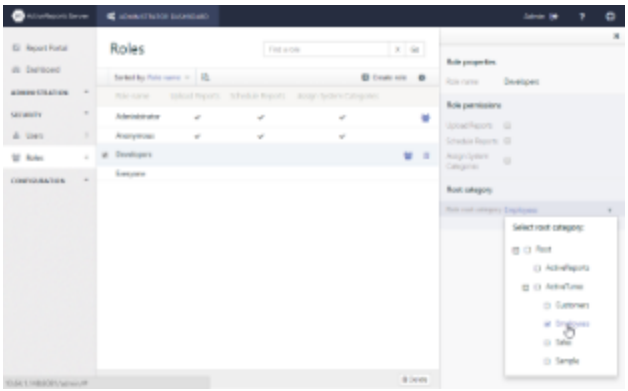
We have made a number of changes since the last version of ActiveReports Server. Here are the major ones:

Images



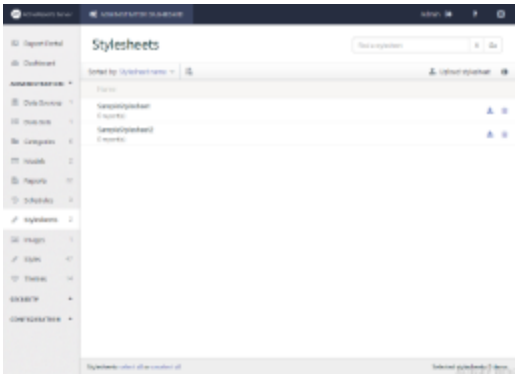
ActiveReports Server now provides a new Images page in the Administrator Dashboard where you can upload, download or delete images. These images enable the report designers to use them in multiple reports. The administrator can also provide permissions to access the images for designing reports in ActiveReports. For more information, see Managing Images and Working with Images in ActiveReports help.

Role Based Root System Category



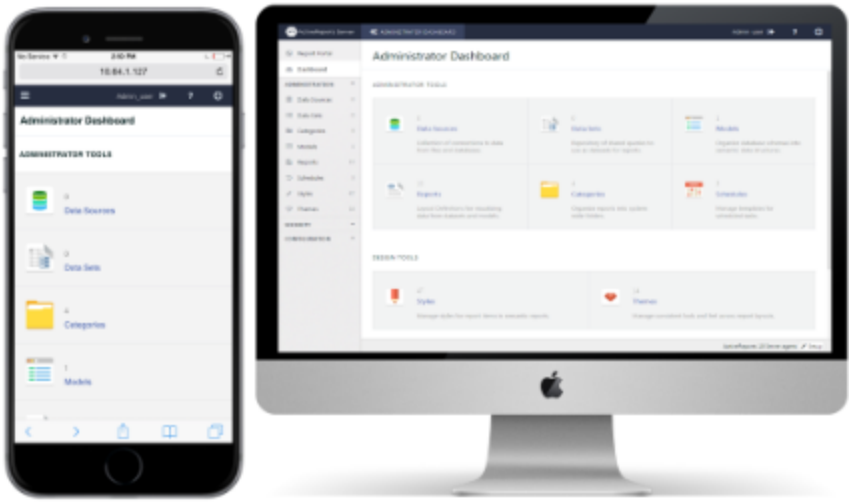
ActiveReports Server now provides multi-tenant support for System Categories by introducing the **Role Root Category** feature. This feature enables administrators to assign any system category as the top level category for a user role. By setting the role root category, only the sub-categories under the selected category level are available to the users in a user role. You can also **Hide Empty Categories in Report Portal** from the Site Settings page of the Administrator Dashboard. For more information, see Report Categories and Managing Report Categories.

Style Sheets



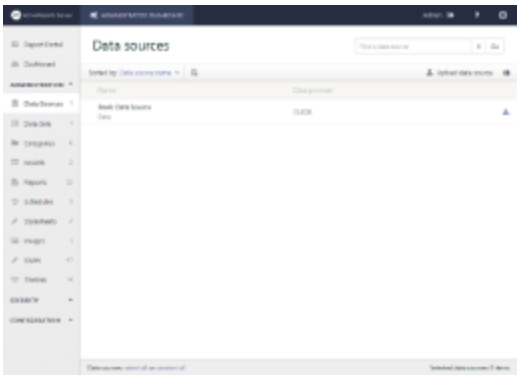
ActiveReports Server has a new **Stylesheets** page in the Administrator Dashboard where you can upload, download or delete style sheets. These style sheets enable multiple report authors to design consistently themed reports using the developer version of ActiveReports (These are not for use in the add-on Designer). The administrator can provide permissions to report authors so that they can access the style sheets. For more information, see Styles and Working with Styles in the ActiveReports help file.

Fully Responsive HTML5 Admin Dashboard



ActiveReports 10 Server now comes with a fully responsive HTML5 Administrator Dashboard that provides an improved user experience across a wide range of devices from desktop computers to handheld devices. The responsive layout of the dashboard adapts itself to the device display.

Data Sources

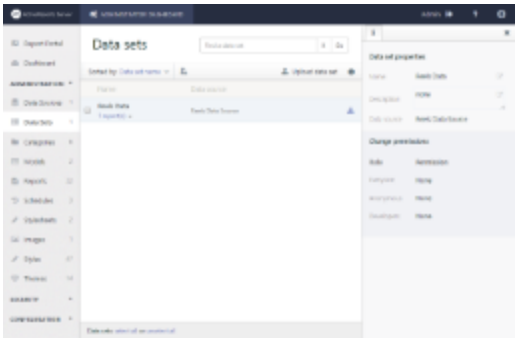


ActiveReports Server now provides a new **Data Sources** page in the Administrator Dashboard where you can upload, download or delete data sources. You can easily access data sources on this page and use them to design reports and create shared data sets. For more information, see Managing Data Sources and Server Shared Data Sources**.**

Developers using ActiveReports always had the option to use file-based shared data sources to define the data sources for multiple reports. ActiveReports 10 adds that functionality to the Server. Server data sources are managed by the server administrators, and developers can use them independently to access data. Built-in multi-tenant support in server data sources means that they are ready for large SaaS applications as well as enterprises where data isolation and governance is imperative.

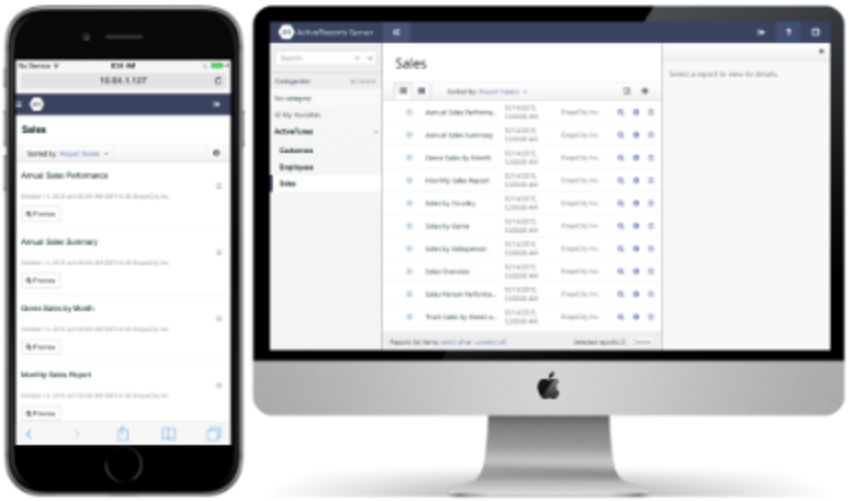
This will help the IT administrators and DBAs to control data access and to keep access credentials secure. This also ensures that developers don't have to track and manage database servers or other data sources to create their reports. They can just focus on designing reports, secure in the knowledge that the data they need is available from the server data sources. Reports that use these data sources can be scheduled or previewed in ActiveReports Server. For more information, see Managing Data Sources and Server Shared Data Sources**.**

Data Sets



ActiveReports Server introduces a new **Data Sets** page in the Administrator Dashboard where you can upload, download or delete data sets. The administrator can provide permissions to access data sets for designing reports in ActiveReports. For more information, see Managing Data Sets and Server Shared Data Sets**.**

New HTML5 Report Portal



Along with the new responsive Administrator Dashboard, we have also given the ActiveReports 10 Server Report Portal a facelift. We have introduced a new color scheme that is more pleasing to the eye and provides a better viewing experience. You can easily integrate the new HTML5 Report Portal into your application with pass-through authentication and more. For more information, see Report Portal.

Public Web API

The new RESTful API provides developers with the building blocks to use the server as a service endpoint. Also, we use the same API to provide all the functionality in the built-in report portal. You can use specific parts of the report portal, or if you want to add a feature that is not available out of the box, now you can do that with our comprehensive API. For more information, see Public WebAPI.

Breaking Changes

When you upgrade reports from previous versions of ActiveReports Server, there are a few breaking changes which are as

follows:

- Flash based Report Portal and Mobile Portal are not available with ActiveReports 9 Server Service Pack 1 onwards. Both of these report portals have been replaced by the fully responsive HTML5 Report Portal.
- Private and Public Access settings for Schedule History have been hidden from the Managed Schedules. These options are planned to be redesigned to offer more comprehensive support for sharing Schedule History across end users for the future release of ActiveReports Server. During this time, the Schedule History will be shared using the Role based permissions set for each Managed Schedule. For more information, see Manage permissions to a shared schedule.

ActiveReports Limitations

If your developers use ActiveReports, they will come across the following limitations when working with ActiveReports in ActiveReports Server.

Drill-Down in Page Reports

The drill-down in ActiveReports Page reports is not supported in ActiveReports Server, so you can view such reports without using this interactive feature.

Subreport Absolute Path

ActiveReports Server does not support using the absolute path for subreports because ActiveReports Server saves reports in the internal storage. In ActiveReports Server you have to specify the subreport name to locate the subreport.

Master Reports

Master reports are not supported and cannot be uploaded to the Report Portal of ActiveReports Server.

Migration from Previous Versions

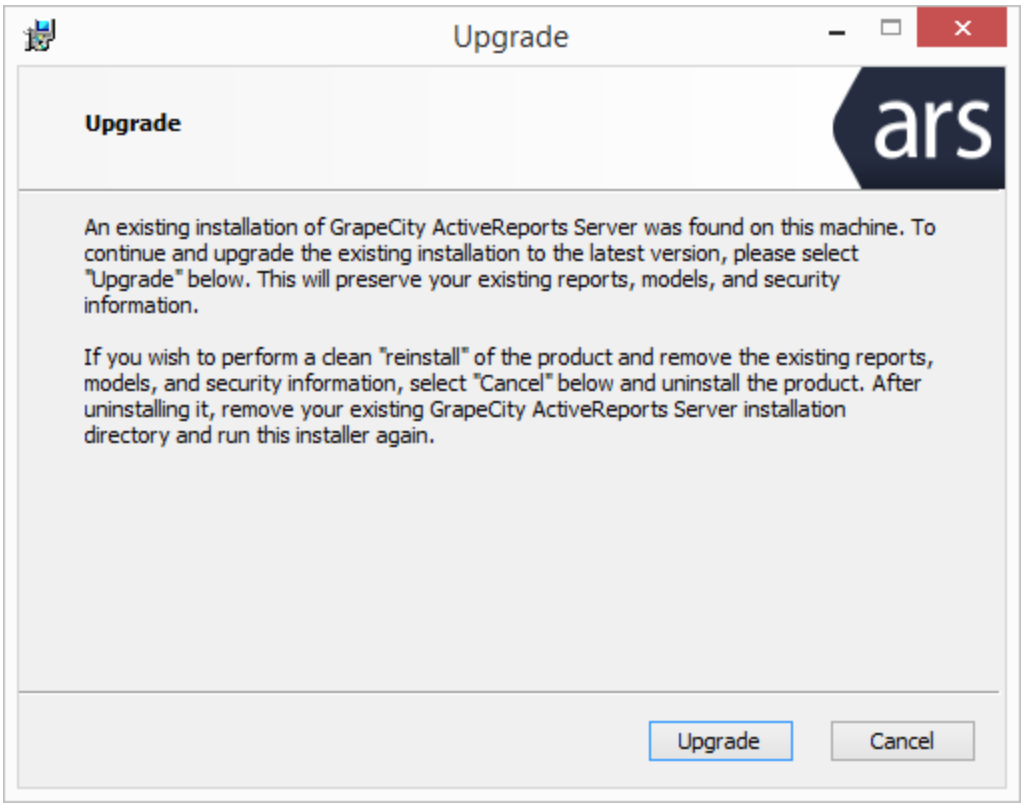
ActiveReports Server includes an easy path for migrating from previous versions.

> **Important:** Before following the upgrade steps in a live environment, we suggest that you test the upgrade in a test environment.

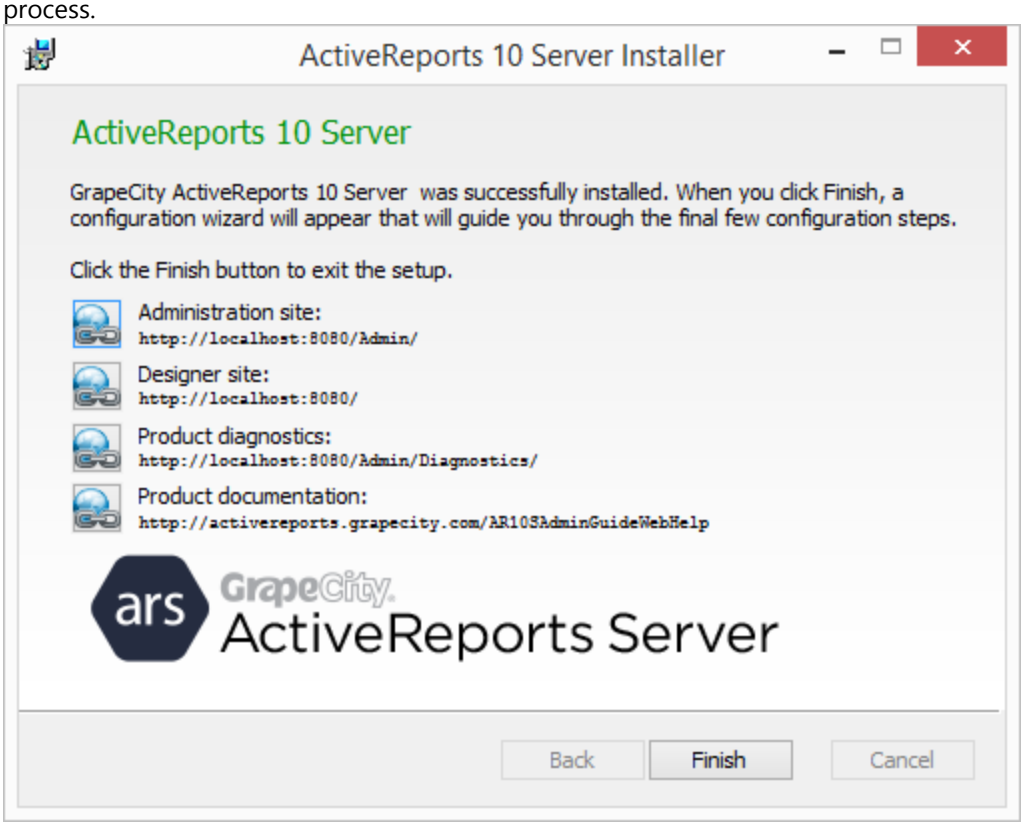
The new version of ActiveReports 10 Server installs in the same folder over the previous version. The installer detects the previous version and suggests upgrading it, and then adapts any existing agents, models, roles, groups, users, schedules, permissions, themes, reports, styles, audit settings and SMTP settings to work with the new version.

> **Note**: The minimum supported version for migration to ActiveReports 10 Server is ActiveReports 9 Server Service Pack 1.

1. Run the ActiveReports 10 Server installation file.
2. In the **Upgrade** dialog that appears, click **Upgrade** to upgrade the existing version of the product to the latest version.



3. Once the installation finishes, a dialog notifies you. Click **Finish** to close the window and complete the installation process.



Once the installation process is complete, the Administrator site opens in your browser so that you can complete the configuration of ActiveReports 10 Server. Activate a new license, because the ActiveReports 9 Server product keys do not unlock ActiveReports 10 Server. For details, see Activate a License.

If you need a new product key for ActiveReports 10 Server, contact our sales team at activereports.sales@grapecity.com.

For information on the ActiveReports 10 Server installation process, see Installation.

Important Points

- IIS configurations, custom security provider settings, and any custom settings in the Web.config of the server web site may have to be reapplied after the upgrade process is complete.
- Recompile the custom security providers with the latest version of the ActiveReports.Server.Extensibility assembly.
- Clearing the cache is recommended on completing the upgrade.
- Rebuild the code-based reports from any previous version using ActiveReports 10 and republish the reports to preview in ActiveReports 10 Server.

License Types

There are three types of licenses for ActiveReports Server:

ActiveReports Server Trial Edition

This is a 30 day free trial version of the product that you can use to determine how ActiveReports Server can fulfill your needs. The Trial Edition includes all of the features in the Core Edition plus the Designer Add-On.

If the initial trial expires before you have had a chance to fully evaluate the product, you may contact activereports.sales@grapecity.com to request an extension of the trial period.

ActiveReports Server Core Edition

This license covers use of the server, the report portal, the HTML5 report viewer, and as many managed agents as you need. Once you purchase the Core Edition, there is no need to purchase additional licenses as you scale up to handle increased server loads.

With this license, you can schedule and distribute reports, take advantage of self-managed clustering and load-balancing, and provide role-based security for reports and data models. You can even allow users to sign in with their existing user name and password to access centrally stored ActiveReports report files.

ActiveReports Server Designer Add-On

In addition to serving up existing reports, if you want to offer your users the ability to create their own ad hoc reports right in the browser, you can purchase the optional Designer Add-On.

The Designer Add-On is a completely browser-based report designer for end users who do not use SQL. End users can create and modify reports and even query, sort, and filter data with no special knowledge of database schema or SQL.

Installation

ActiveReports Server allows you to provide a browser-based report designer for end users and to serve up both end user reports and your more sophisticated developer reports. This section helps you to get started.

> **Important**: During installation, if you choose to create new users for the Web App Pool and Service, be sure to update the user properties to match your password policy, as the default setting is to use an expiring password.

In This Section

> **Important**: If you use a virtual directory, you must copy the crossdomain.xml file (from C:\ActiveReports 10 Server\Site) to the full root of the parent application. If you have it only in the virtual directory, a Security sandbox violation occurs when you try to preview a report.

System Requirements

To install and use ActiveReports Server, you must have compatible software installed on your server. If you have multiple servers set up for load balancing, the Managed Agent machines have fewer requirements than the Master Server machine. If you have a single-server deployment, the Site consists of a Master Server and a Managed Agent on the same machine.

Operating Systems

- **Server**: Windows Server 2008 and above
- **Managed Agent**: Windows Server 2008 and above; .NET Framework 4.5 and above

> **Note**: Server machine should not be an ActiveDirectory Domain Controller. ActiveDirectory Domain Controller may cause issues with the user/group level security in ActiveReports Server.

Windows Server 2008, 2012

The following features must be turned on in Windows Server 2008, 2012 to run ActiveReports Server on your server machine. From Control Panel, Programs and Features, open **Turn Windows features on or off** to ensure that these features are turned on. All are on by default.

- Internet Information Services / World Wide Web Services / Application Development Features / ASP.NET 4.5
- .NET Framework 4.5 Advanced Services / WCF Services / HTTP Activation

Web Servers

- IIS 7, IIS 7.5 with Windows Server 2008 (needs ASP.NET role service)
- IIS 8 and IIS 8.5 IIS 10
- IIS Management Console must be turned on.
- .NET Framework: 4.5 or higher

Internet Information Service (IIS) Ports

TCP/IP

- ActiveReports Server 4040
- ActiveReports Server Managed Agent 5050

UDP

- ActiveReports Server 7923, 26637, or 57323
- ActiveReports Server installer 7979, 37637, and 15377

> **Note**: You can open and access the above mentioned port numbers on the server machine using the network or firewall configuration.

Database

- Microsoft SQL Server 2005, 2008, 2008R2, 2012 and 2014
- MySQL
- Oracle 11g (32-bit and 64-bit)
- PostgreSQL
- C-TreeACE 10 and above

> **Note**: For Oracle, the Oracle client must be installed on the server machine and on all agent machines.

Browsers

- Internet Explorer 9 and above
- Google Chrome 30 and above
- Mozilla Firefox 20 and above
- Mobile Safari iOS6 and above

Flash Player

- Adobe Flash Player 11, 12 and 13

SDK (Designer and ReportList control)

- Operating System: Windows ™ Vista, Windows 7, Windows 8 and 8.1, Windows Server 2008 and above
- .NET Framework: 4.0 or higher
- Microsoft Visual Studio: 2010 and above

AppPool Permissions (SQL Server)

In order to generate data models using Windows authentication with Microsoft SQL Server data, the AppPool used for ActiveReports Server needs proper credentials to access the database.

This is set during installation with the user that you assign for the Web Application Pool.

If the user you assign for the application pool does not have permission to access the database, when you try to create a new model using Windows Authentication, you get an error like:

> "The connection test failed. Login failed for user 'YourUserName'."

You can avoid this error by setting the AppPool account with a user who has permissions to the data you want to use, or by opting to use Server Authentication when you create a data model. See Creating a New Model for more information.

Installed Files

When you install ActiveReports Server and use all of the default settings, files are installed in the following folders:

▸ C:\ActiveReports 10 Server

File (or Folder) Name	Description
Agent	Contains the files that you use to deploy your solution to separate machines that host your Managed Agents if you do a multi-server deployment for load balancing.
Data	Contains user information and reports and models. This is the folder that you need to back up to prevent loss of data. For more information, see Backup and Recovery.
DataProviders	Contains data provider assemblies.
SDK	Contains server report controls and a class library for the software developer kit.
SecurityProviders	Contains the ActiveDirectorySecurityProvider assembly used to create a security provider.
Server	Contains the files used to run the server.
Site	Contains the files used to run both the Report Portal and the Admin sites, as well as all associated help files.
ActiveReports 10 Server License Agreement.rtf	The license agreement that dictates the terms under which you may use this software.

▸ C:\ActiveReports 10 Server\SDK

File Name	Description
ActiveTunes.SecurityProvider Sample	Shows how to provide proprietary or single-sign-on authentication instead of the built-in ActiveReports 10 Server security and use row-level security.
HTML5 Viewer.Source	Includes Cascading Style Sheets and JavaScript files for HTML5.
ActiveReports.Server.ReportControls.dll	Assembly that supports the ReportDesigner and ReportList control for use in

	your ASP.NET solutions.
ActiveReports.Server.ReportControls.xml	Intellisense for the ReportControls assembly.
zh-CN	Includes resource files for the Chinese environment.
Samples	Contains sample Visual Studio projects that you can use to evaluate the software, or customize and extend to use in your own solutions. For more information on samples, see the SDK help.

C:\ActiveReports 10 Server\Site

Folder Name	Description
Admin	Contains the files and folders used to run the Admin site, as well as the associated help file.
Bin	Contains ActiveReports and Data Dynamics Reports assemblies as well as SQLite and DotNetZip assemblies for use on the site.
i18n	Contains the files and folders having properties of different locales.
portal	Contains the files and folders used to run the Admin site, as well as the associated help file.
crossdomain.xml	Cross-domain policy file that make data available to WSF files in different domain.
Default.aspx	The default web file that shows the introductory text and links.
Default.aspx.cs	The default cs that shows the introductory text and links.
Designerservice.svc	The service host file for DesignerService.
Error.aspx	The file that contains error information for the Designer Web application.
Generic.master	The file that contains information regarding log-in page configuration.
Global.asax	The default class that sets global URL routing values for the Designer Web application.
index.html	The index file that contains html code for appearance and functioning of the main page.
ReportService.svc	The service host file for ReportService.
web.config	The main file that contains configuration information for the Designer Web application.

List of open source software used in ActiveReports Server

- **Report Portal**
  Bootstrap v3.1.1
  Bootstrap Datetimepicker
  Bootstrap Multiselect
  jQuery Cookie Plugin v1.4.1
  jQuery JavaScript Library v2.1.0
  Sizzle.js
  knockout-handlebars.js v0.0.6
  Knockout JavaScript library v3.1.0
  Less - Leaner CSS v2.0.0
  Modernizr 2.7.1
  Placeholders.js

parseUri 1.2.2

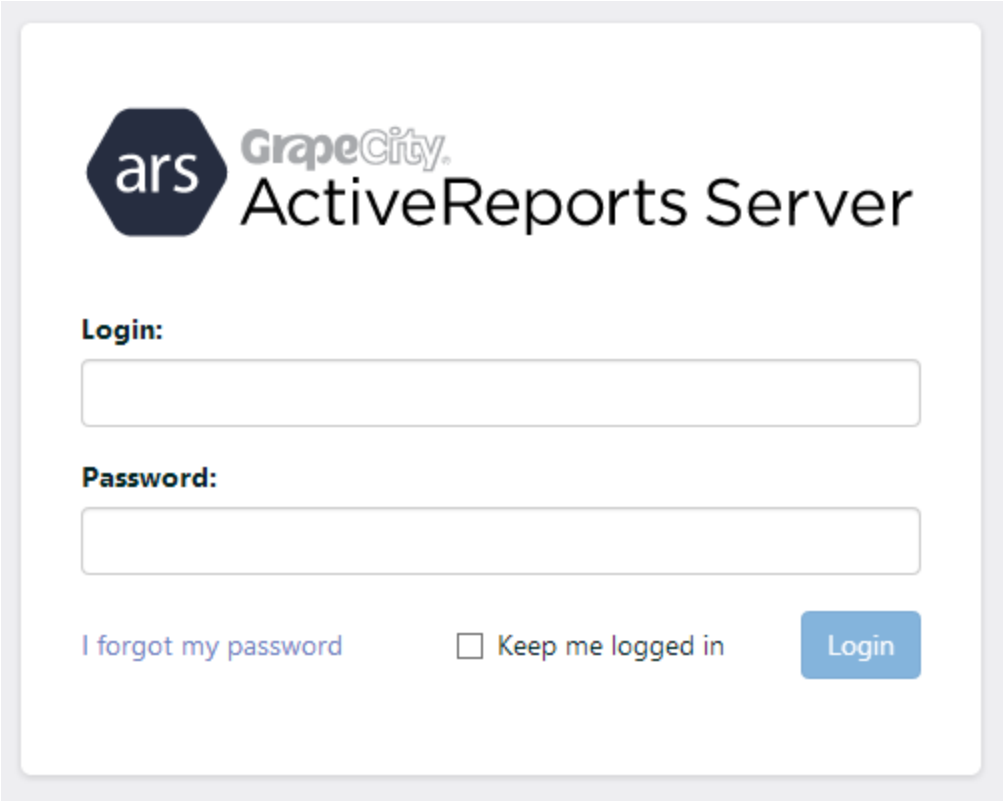
- **Administrator Dashboard**
  jQuery.data.js
  jQuery UI Datepicker 1.8.10
  jQuery.deferred.js
  jQuery.editable-select
  jQuery customfileinput
  jQuery v1.4.4
  Sizzle.js
  jquery.meio.mask.js 1.1.3
  jQuery UI Tabs 1.8.10
  jQuery Tooltip plugin 1.3
  jquery.ui.i18n.datepicker.js
  jquery.ui.i18n.timepicker.js
  jQuery UI 1.8.10
  jQuery UI Slider 1.8.10
  jQuery timepicker add-on
  Watermark plugin for jQuery 3.1.4
  json2.js
  Knockout JavaScript library v3.1.0
  jQuery UI Spinner 1.20

- **Designer**
  Adobe Flex framework

- **Others**
  Microsoft.Owin
  Json.NET
  ESENT Managed
  Owin.Routing
  Owin.Hosting
  SQLite and SQLite LINQ

Logging On to the Administrator Site

1. From the Start menu, in All Programs, select **GrapeCity**, then **ActiveReports 10 Server**, **Administration**.
2. In the Administration Login dialog that appears in your browser, enter your registered **User Name** and **Password**.



3. Click **Log In**. The Administrator Dashboard appears.

Backup and Recovery

It is always a good idea to back up your data so that you have a safe copy of it in the event of a disaster.

To back up and recover your data

1. Copy the contents of the `C:\ActiveReports 10 Server\Data` folder to a safe location.
2. Install a fresh instance of ActiveReports Server with the same version as the one from which you backed up your data.
3. Copy the contents of the Data folder into the new `C:\ActiveReports 10 Server\Data` folder. The data is recovered.

Configuring ActiveReports 10 Server on SSL and HTTPS Web Sites

To use SSL and HTTPS on the ActiveReports Server Web site, you need to make some configuration changes.

> **Note:** If you want to use the HTTPS communication protocol, you need to specify the URL as "https://server:443/mobile/" or "https://server:443/m/".

To use the HTTPS protocol, you must create an SSL certificate. For help with this, see Microsoft's article How to: Configure

an IIS-hosted WCF service with SSL.

To configure for HTTPS only

1. Open Internet Information Services (IIS) Manager and expand the **Sites** node.
2. Select the **ActiveReports 10 Server** site, and on the right, in the Edit Site section, click **Bindings..**.
3. In the **Site Bindings** dialog that appears, click **Add.**.
4. In the **Add Site Binding** dialog that appears, select https under **Type** and select an SSL certificate under **SSL certificate**, then click **OK** to close the dialog.
5. In the **Site Bindings** dialog, select the default http binding, and then click **Remove**. This will remove the default http binding.
6. Open the web.config file of the Site and add the following code to the <system.serviceModel> section, between the <service> and </service>tags, to change the server address and port.

   Paste in the web.config file BETWEEN the <service> and </service> tags

   ```
   <endpoint address="https://servername:443/ReportService.svc/json"
   binding="webHttpBinding"
   contract="ActiveReports.Server.ReportServices.Servicing.IReportService"
   bindingConfiguration="ReportServiceJsonBindingSecured"
   behaviorConfiguration="JsonBehavior" />
   ```

7. In the <system.serviceModel> section, between the <service> and </service> tags, remove the default code between the <endpoint> tags where the **endpoint address="json"**.
8. In the same <system.serviceModel> section, between the <service> and </service> tags, change the default code between the <endpoint> tags from "mexHttpBinding" to "mexHttpsBinding".
9. In the same <system.serviceModel> section, under <behavior>, remove the default code `httpGetEnabled="true` from <serviceMetadata> tag and add the following code.

   Paste in the web.config file BETWEEN the <serviceMetadata> and </serviceMetadata> tags

   ```
   httpsGetEnabled="true"
   ```

10. In the <bindings> section, between the <webHttpBinding> tags, remove the default section <binding name="ReportServiceJsonBinding"> and add the following code.

    Paste in the web.config file BETWEEN the <webHttpBinding> and </webHttpBinding> tags

    ```
    <binding name="ReportServiceJsonBindingSecured" maxReceivedMessageSize="0x3000000">
    <readerQuotas maxStringContentLength="0x3000000" />
    <security mode="Transport" />

    </binding>
    ```

11. In the <bindings> section, between the <wsHttpBinding> tags, change the default security mode code from

`<security mode="None /">` to `<security mode="TransportWithMessageCredential" />`.

12. Restart the Internet Information Services (IIS) Manager.

To configure for HTTP/HTTPS

1. Open Internet Information Services (IIS) Manager and expand the **Sites** node.
2. Select the **ActiveReports 10 Server** site, and on the right, in the Edit Site section, click **Bindings..**.
3. In the **Site Bindings** dialog that appears, click **Add.**.
4. In the **Add Site Binding** that appears, select https under **Type** and select an SSL certificate under **SSL certificate**, then click **OK**.
5. Open the web.config file and add the following code to the <system.serviceModel> section, between the <service> and </service>tags, changing the Server address and port.

   Paste in the web.config file BETWEEN the <service> and </service> tags

   ```
   <endpoint address="https://servername:443/ReportService.svc/json"
   binding="webHttpBinding"
   contract="ActiveReports.Server.ReportServices.Servicing.IReportService"
   bindingConfiguration="ReportServiceJsonBindingSecured"
   behaviorConfiguration="JsonBehavior" />
   ```

6. In the same <system.serviceModel> section, under <behavior>, between the <serviceMetadata> tags, add the following code.

   Paste in the web.config file BETWEEN the <serviceMetadata> and </serviceMetadata> tags

   ```
   httpsGetEnabled="true"
   ```

7. In the <bindings> section, between the <webHttpBinding> tags, remove the default section <binding name="ReportServiceJsonBinding" and add the following code.

   Paste in the web.config file BETWEEN the and </webHttpBinding> tags

   ```
   <binding name="ReportServiceJsonBindingSecured" maxReceivedMessageSize="0x3000000">
   <readerQuotas maxStringContentLength="0x3000000" />
   <security mode="Transport" />
   </binding>
   ```

8. Restart the Internet Information Services (IIS) Manager.

Once you set up the certificate and start using https:// to access the Web site on which ActiveReports 10 Server is installed, it begins working in secure mode.

For more help with this, see Microsoft's article on Configuring SSL on a Web Server or Web Site (IIS 6.0) or How to Set Up SSL on IIS 7.

Concepts

This section reviews the most common concepts to help you understand ActiveReports Server and allow you to create and serve reports more efficiently.

Logical Data Models

A **logical data model** is an abstract model that organizes data into separate entities with defined relationships between entities. Each entity has its own set of attributes. A user selects a data model from a list of available data models, and then uses it as a basis for creating reports.

Entities

Entities represent business objects (e.g. Contact, Product, etc.) that are included in a data model. Entities are represented to the end user in a multi-level tree. Each entity has an associated set of attributes.

You can create Ad Hoc Entities by writing SQL queries and binding them to a logical table, but certain restrictions apply.

Attributes

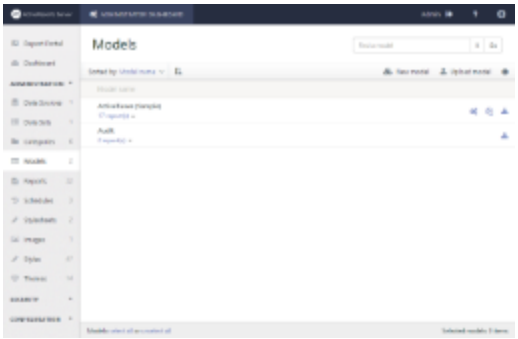
Attributes are associated with an entity and represent values that are used for creating a report. A **calculated field** is a type of attribute which calculates a value based on other attributes.

Relations

A relation is an element of a data model that represents relationships between entities.

Relation Cardinality dictates how relations are created.

As administrator, you can create new models, delete models, and edit models to provide what your end users need in order to create meaningful reports. See the Managing Models section for information on how to perform these tasks.



Ad Hoc Entities

An ad hoc entity is one that is bound to a dynamic SQL query, for example, a stored procedure or a view, rather than a query built by ActiveReports Server. You can use an ad hoc entity in the same ways as any other entity, except that you cannot add a logical column, a calculation, to the base table of an ad hoc entity. You can, however, add calculated attributes in the SQL query itself. You can also add attributes that bind to the existing columns, and you can add relations to other entities.

An example of a query that creates an ad hoc entity from the included ActiveTunes (Sample) model is as follows.

SQL Query
SELECT Address, City, State FROM Customer

Creating Ad Hoc Entities

Any limitations that apply to the underlying data also apply to your ad hoc entity.

- In SQL subqueries, you can have up to 32 levels of nesting (or less depending on memory resources and the complexity of other expressions in the query).
- Microsoft SQL Server does not allow the TOP function in nested queries.

If you try to create an entity using a query that exceeds your underlying data's limits, ActiveReports Server displays an appropriate error.

> **Tip**: When editing queries for ad hoc entities, provide all attributes that are currently used by the entity. This way, if you want to exclude an attribute from the SQL query of the ad hoc entity at run time, you can delete it before the edit operation.

Take care when changing these queries, because you can change the actual data for attributes without changing the semantic model (except the entity edits). This could cause data in some reports to change dramatically.

Relation Cardinality

Relation cardinality dictates how relations are generated when you create a logical model, and how they are used to generate underlying SQL JOIN queries when you run reports. Generally, the cardinality is set in the underlying database, but if it is not set up properly there, you can set the cardinality of a relation from the Administrator Dashboard in the Model Editor. For more information, see Modifying a Relation.

Relation Types

When you add relations between entities, ActiveReports Server uses rules to determine cardinality. There are four ways that entities can relate:

- **One**: One-to-one
- **Many**: Many-to-many
- **OptionalOne**: One-to-zero or one
- **OptionalMany**: Many-to-zero or more

If you add relations between entity A (from table A) and entity B (from table B) using the corresponding sets of tables in AC and BC, the rules for determining cardinality are as follows.

Adding relation AB to entity A

- If there is a unique constraint on table B that is nullable and exactly matches the columns in BC: **OptionalOne**.
- If there is a unique constraint on table B that is not nullable and exactly matches the columns in BC: **One**.
- Otherwise: **OptionalMany**. (ActiveReports Server does not use **Many** in creating relations, but you can set it explicitly.)

Adding relation BA to entity B

- If there is **no** unique constraint on table A that matches the columns in AC: **OptionalMany**.
- If there is a unique constraint on table B that matches the columns in BC, and one of those columns is nullable: **OptionalOne**.
- If there is a unique constraint on table B that matches the columns in BC, and **none** of those columns is nullable: **One**.

Example

We can add a relation between the Album and Artist tables using the ArtistID column. Assuming this relation does not exist in the database yet, if we add the relation from the Album entity, we get an Album-to-Artist relation with OptionalOne cardinality and an Artist-to-Album relation with OptionalMany cardinality.

SQL JOIN Types

ActiveReports Server does not take role cardinality into account in determining what JOIN type to use. The relation in the data source controls the behavior.

- If the relation belongs to the table that the query joins to, then it uses a LEFT JOIN.
- If there is at least one column in the relation that accepts a null value, then it uses a LEFT OUTER JOIN.
- Otherwise, it uses an INNER JOIN.

Basically, if it accepts null values, it uses outer joins, otherwise it uses inner joins.

Examples

The Album table has a foreign key constraint linked with the Artist table, there is a relation created for it, and it belongs to the Album table. So long as Album.ArtistID does not accept null, then for queries joining the Album table to the Artist table, we use an INNER JOIN, but for queries joining the Artist table to the Album table, we use a LEFT JOIN. However, if Album.ArtistID is nullable, then even in the case of queries joining the Album table to the Artist table, we use a LEFT OUTER JOIN.

Consider relations based on foreign constraints between Playlist, Track and PlaylistTrack tables via PlaylistID and TrackID. The relations belong to the PlaylistTrack table as they refer to Playlist and Track. Assume for this purpose that CollapseInRelations is not set for PlaylistTrack.

If you drop Playlist.Name and Track.Name to create a table, then the query tree looks like this:

```
PlaylistTrack
> Playlists
> Tracks
```

In both cases, it uses an INNER JOIN, as both relations belong to the table (PlaylistTrack) from which the query pulls.

In the case of the Album and Artist tables, the relation connects an Album to an Artist by the ArtistID. So you can drop the Artist.Name attribute and a "Total Albums" aggregate to create a table. If you drop the Name attribute first, then the query looks like this:

```
Artist
> Album
```

In this case, it uses a LEFT OUTER JOIN because the relation belongs to the table (Album) from which the query pulls.

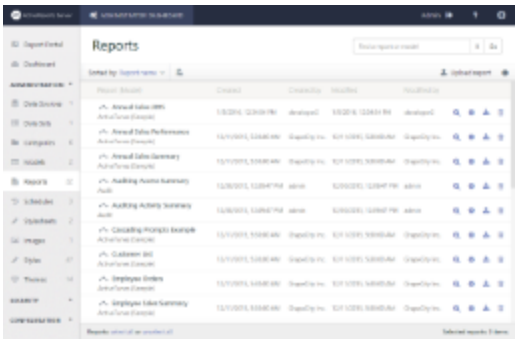
Reports

End users can create reports in the ActiveReports Server Report Portal. When they save them in the Report Portal, you can see them listed on the **Reports** page of the Administrator Dashboard, where you can modify, delete, download, or set permissions on them. Reports that are hosted on ActiveReports Server can be previewed in ActiveReports. See Viewing Reports from ActiveReports Server, for more information.

End users can also serve more complex developer reports created with ActiveReports. See Managing Reports for information on adding developer reports.

Note

- Developer reports are visible to end users, but they can only download or preview them, and cannot open them in Design view.
- ActiveReports Server Designer Add-On license is required to design a report.



For more information on report options in the Administrator Site, refer to Managing Reports.

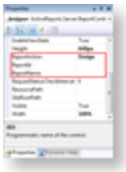
Report Loading

When you use the Designer Web control, you can load reports into it by specifying a value for one of two properties:

- **ReportName** loads the first report with the specified name.

  > **IMPORTANT**: Multiple reports can share the same name, so using the ReportName property does not guarantee that the same report is loaded every time.

- **ReportID** loads the unique report with the specified ID. When this is specified, the ReportName property is ignored.



A third property, **ReportAction**, tells the designer how to open the report. ReportAction has two enumerated values:

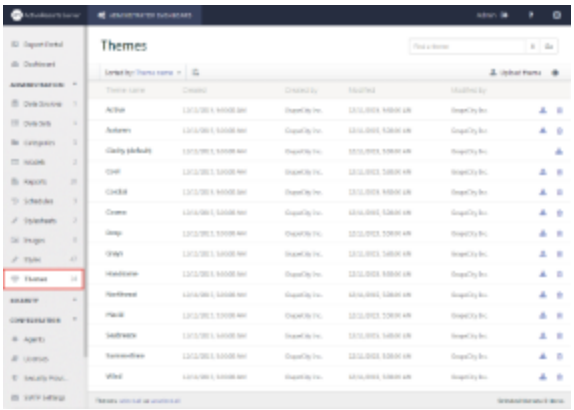
- **Design** is the default value for ReportAction, and tells the designer to open the report in Design view.
- **Preview** tells the designer to open the report in the Preview tab.

> **Note**: ActiveReports Server Designer Add-On license is required to design a report.

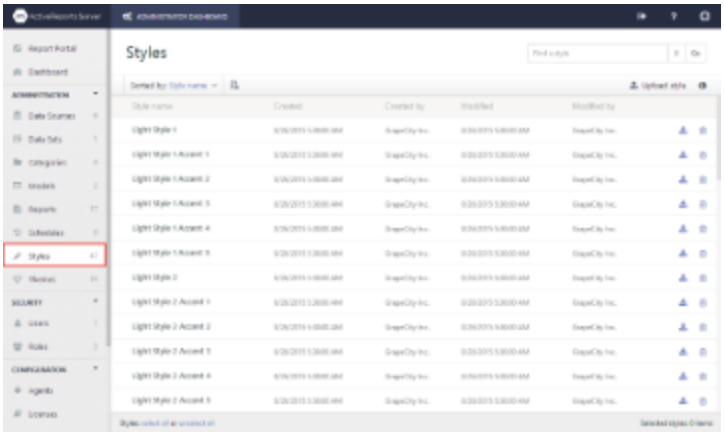
Styles and Themes

Styles and themes allow your end users to control the visual presentation of the reports they create. You can control which styles and themes are available to your users.

Themes determine the overall look of a report, and change the appearance of all report items: tables, charts, etc. The end user accesses themes on the Report tab of the Report Portal.



Styles can be applied to individual report items. Some styles apply only to the Table report item, and the end user accesses them on the Table Design Tools tab. Others apply only to the Chart report item, and the end user accesses them on the Chart Design Tools tab.

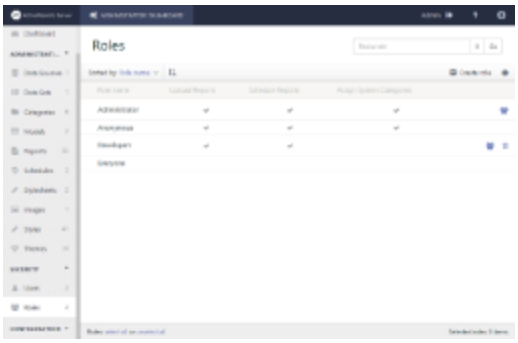


Note: ActiveReports Server Designer Add-On license is required to apply Themes and Styles to a report.

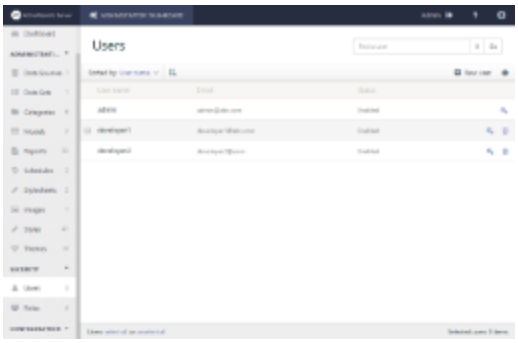
You can upload, download, modify, and delete styles and themes, and control which are used by default. For more information, see Managing Themes and Styles.

Security: Users and Roles

ActiveReports Server employs role-based security, which means that permissions are assigned to roles, rather than to individual users. To allow a user to access specific report options, you must include them in a role group that has the proper set of permissions. In the Security section of the dashboard, click **Roles** to administer roles.



As Administrator, you can create roles and users. You then add users to each role and control which roles are allowed to upload ActiveReports and which roles have Read access to each model. In the Security section of the dashboard, click **Users** to administer users. For more information, see Managing Security.



Agents and Scalability

Agents allow you to serve reports to more concurrent users, and respond to higher report execution loads.

Scale-Out Deployment

Using multiple report server instances that share a single report server database is called scale-out deployment. You can add Managed Agents to your deployment that the central controller server uses to automatically load-balance report execution tasks.

Licensing

When you purchase the ActiveReports Server Edition, you can use as many agents as you need.

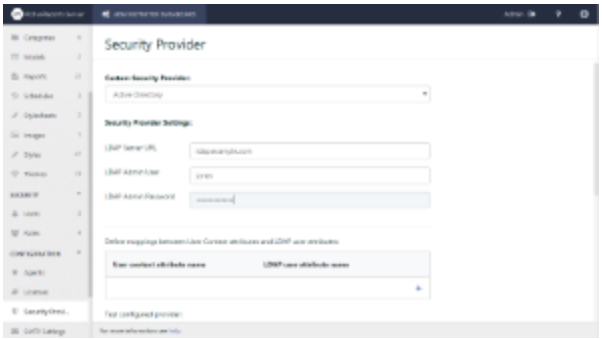
Deployment Scenarios and Licensing Requirements

For optimal scalability, it is best practice to run agents on separate machines from the main host. But if your focus is on having less hardware to manage and keeping costs down, you can keep the agent on the same machine as the host.

Single Sign-On with LDAP or Active Directory

If your user information is already stored in a corporate directory that uses Lightweight Directory Access Protocol (LDAP) or Active Directory®, you can connect your ActiveReports Server users and roles to the directory by configuring it on the

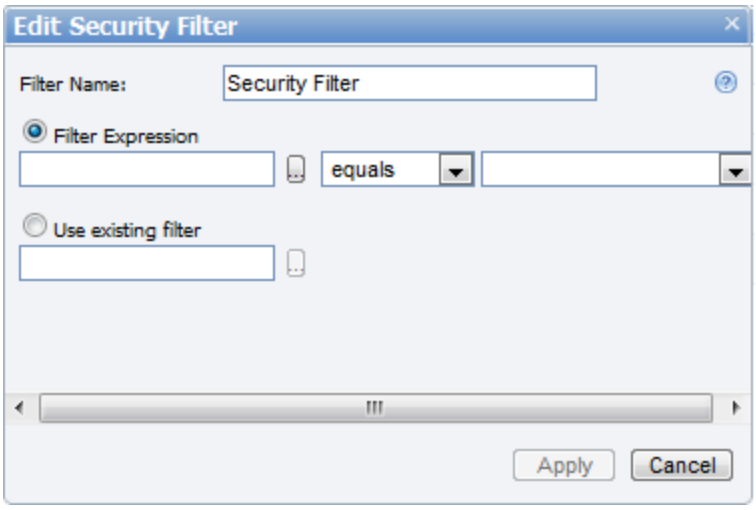
Security Provider page.



For information on how to set up your LDAP or Active Directory in ActiveReports Server, see the Managing Security Providers and Using LDAP and Active Directory topics.

> **Note**: ActiveReports Server supports OpenLDAP v2.3 or later. Earlier versions are not supported because they do not have a MemberOf attribute.

Once you have the settings and mappings in place, when you edit a model, you can add a security filter to each model entity. (See the Modifying an Entity topic for more information.)



In the Filter Expression, the value on the right side gives you a list of the attributes that are mapped to your LDAP or Active Directory.

Web Services

When you install ActiveReports Server, it runs an ASP.NET Web site that gives you access to the Report Portal and the Administrator Dashboard. The root of that Web site is located in:

C:\ActiveReports 10 Server\Site\

In the Site folder, you will find the **ReportService.svc** file. This allows you to call and access the report service.

> **Important**: If you use a virtual directory, you must copy the crossdomain.xml file (from C:\ActiveReports 10 Server\Site) to the full root of the parent application. If you have it only in the virtual directory, a Security sandbox

> violation occurs when you try to preview a report.

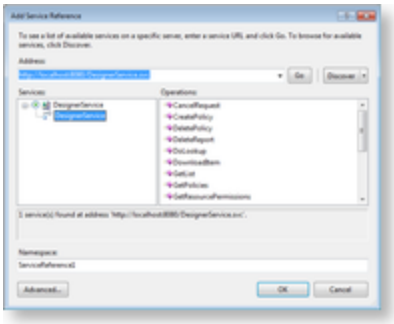
There is code in each of these Web service files that tells ASP.NET that when it encounters one of our file types, it can direct the call to ActiveReports Server for handling.

To use one of the ActiveReports Server Web services, you create a client of some sort, for example a **console application ('Accessing the Web Service' in the on-line documentation)**, and add a Service Reference that points to the service address, by default:

http://localhost/ReportService.svc

> **Note**: If you entered a different site port, enter that value after localhost. For example, `http://localhost:8080/ReportService.svc`

In the Add Service Reference dialog, once you enter the correct address and click **Go**, you can select the service and view the available operations in the pane to the right.



ASP.NET responds differently depending on how you access the Web service URL.

- If you access it using a raw Web service client, it initiates an exchange of XML.
- If you access it with a browser it displays a Web page with information about using and testing the service.

When you add a service reference, Visual Studio calls the URL and appends ?WSDL. For example,

http://localhost/ReportService.svc?wsdl

WSDL, or Web Service Description Language, is XML that describes the methods and arguments that are available on the Web service, which is how Visual Studio generates the classes that you can access from code.

To see the Web services in action, check out the included sample located in:

C:\ActiveReports 10 Server\SDK\Samples

Configuration Section Handlers

You can use the ReportServiceProxy class to provide a handler for the ResolveRemoteEndPoint event using code in the web.config file.

Syntax

In web.config file, you can either provide a handler for the ResolveRemoteEndPoint or you can provide a

remoteReportServicePath with a username and password. ASP.NET code for both the implementations are given below.

▸ To provide a handler for the ResolveRemoteEndPoint event

ASP.NET code. Paste in the web.config file BETWEEN the <configSections> and </configSections> tags.

```
<sectionGroup name="activereports.server">
  <section name="reportServiceProxy"
type="ActiveReports.Server.ReportControls.Configuration.ActiveReportsServerSection,
  ActiveReports.Server.ReportControls, Version=x.x.xxxx.x, Culture=neutral,
PublicKeyToken=d557f2f30a260da2" allowDefinition="Everywhere" />
</sectionGroup>
```

▸ To provide a remoteReportServicePath with a username and password

ASP.NET code. Paste in the web.config file BELOW the </configSections> tag.

```
<activereports.server>
  <reportServiceProxy remoteReportServicePath="http://localhost:8080/" username="Admin"
password="1" />
</activereports.server>
```

Remarks

Default configuration:

- You must specify a value for the **RemoteReportServicePath** to use the section handlers.
- The **username** and **password** values are empty by default, and are optional.

Configurable Location

Use these handlers in the application-level web.config file.

> **Note**: While other locations may work, we do not support them.

URL Access to Reports and Designer

You can use a uniform resource locator (URL) request to preview or edit reports, or even design new ones from a specified model.

> **Important**: The logged-in user must have permission to perform the specified action on the report.

Syntax

http://*server***/?***param***="***string***"&ReportAction=***enum*

Arguments

Argument	Description
server	Replace this with the name or IP address of the computer on which you run the report server.
?	The question mark indicates to ActiveReports Server that the rest of the URL contains parameters and an action to perform.
param	Replace this with the parameter that you want to use. Select from **ModelName**, or **ReportID**.
string	Replace this with the model name or ID, or report ID, depending on the parameter you use.
&	The ampersand indicates to ActiveReports Server that the parameter is complete and the action to perform follows.
ReportAction	This is the designer property that indicates to ActiveReports Server that an enumerated action value is to follow.
enum	Replace this with the enumerated action value that you want the designer to perform. Select from **Create**, **Design**, or **Preview**.

Examples

Create a New Report

To create a new report from a model that you specify without opening the Models page, replace the italicized portions of the following URL with your server name and model.

URL request syntax for creating a new report.
http://*MyServer*/?ModelName="*MyModel*"&ReportAction=Create

For the **ModelName** parameter, you can use the model name, the model ID, or the GUID. If you use the model name, and there is more than one model with the same name, ActiveReports Server uses the first model of that name that it encounters.

> **Tip**: To create a quick link to a new report, from the Administrator Dashboard, open the Models list. To the right of the model you want to edit, right-click the **Create report** link and select **Copy link address**.

Edit a Report

To edit an existing report in the designer without opening the report list, replace the italicized portions of the following URL with your server name and report ID.

URL request syntax for editing a report.
http://*MyServer*/?ReportId="*1*"&ReportAction=Design

> **Tip**: To create a quick link to a report, from the Administrator Dashboard, open the Reports list. To the right of the report you want to edit, right-click the **Design** link and select **Copy link address**.

Preview a Report

To preview an existing report without opening the report list, replace the italicized portions of the following URL with your server name and report ID.

URL request syntax for previewing a report.
http://*MyServer*/?ReportID="*1*"&ReportAction=Preview

Tip: To create a quick link to a report, from the Administrator Dashboard, open the Reports list. To the right of the report you want to preview, right-click the **Preview** link and select **Copy link address**.

Managing Models

This section contains information that helps Administrators with data management tasks.

Model List Overview
: This section describes how to maintain the list of data models from which Report Portal users create their reports.

Model Editor Overview
: Refer to this section to learn about the ways in which you can modify individual data models.

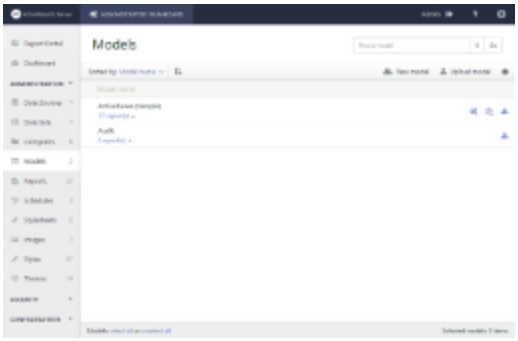
Note

- You can create row-level security by creating custom security providers. See the Managing Single Sign-On Security Providers topic for more information.
- ActiveReports Server Designer Add-On license is required to work with Models.

Model List Overview

You can view and modify all of the data models you have available for creating reports on the Administrator Dashboard Models page.

Note: ActiveReports Server Designer Add-On license is required to create new Model.



Creating a New Model
: Learn to create a new data model with the wizard.

Cloning a Model
: Learn to copy an existing data model.

Working with Model Versions

Learn to use versioning to track changes to your data model. This allows you to revert to previous versions.

Deleting a Model

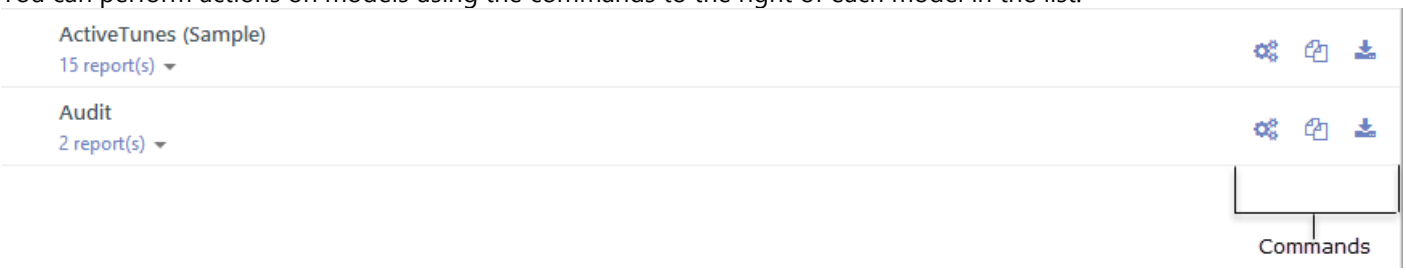
Learn to delete data models.

Uploading and Downloading Models

Learn to upload and download data models for backing up or transferring them.

Model Commands

You can perform actions on models using the commands to the right of each model in the list.



Command Descriptions

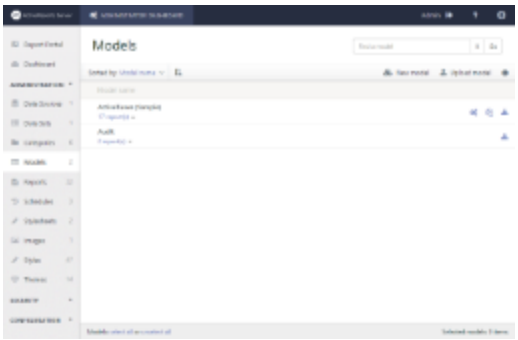
Command	Description
Create report	Opens the report designer with the data model already selected.
Edit	Opens the Model Editor so that you can modify the model's properties. For more information, see Model Editor Overview.
Rename	Allows you to change the model's name.
Permissions	Opens the **Change Permissions** dialog where you can select which roles have access to the model.
History	Displays a list of modifications by date. If you do not supply comments when you modify it, a description of the modification is automatically generated.
Download	Downloads and saves a model to the local file system.
Clone	Creates a copy of the latest version of the model, without any of the reports.
Delete	Deletes the model. This command is not available for the sample model, ensuring that there is always at least one model available.

Creating a New Model

Create a new data model using the model generation wizard.

To create a new model

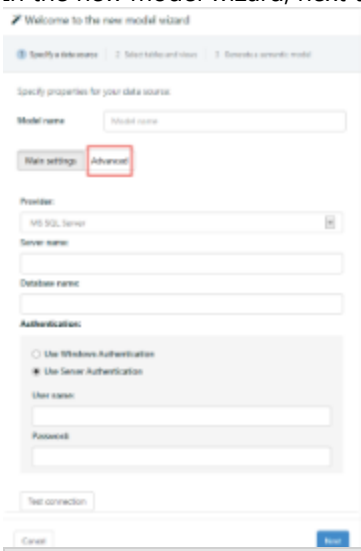
1. From the Administrator Dashboard, select **Models**.
2. On the **Models** page, click **New Model**.



3. In the new model wizard that appears, enter a **Name** for this new data source and the database server information in the provided fields.

▶ To enter the connection string directly in the Connection String Editor dialog

1. In the new model wizard, next to **Main settings**, click the **Advanced** button.



Sample connection string for SQL server
Provider=SQLCLIENT;Data Source=HQ;Initial Catalog=PUBS;User Id=myUsername;Password=myPassword;
Sample connection string for Oracle
Provider=ORACLE;Data Source=MyServiceName;User Id=myUsername;Password=myPassword;

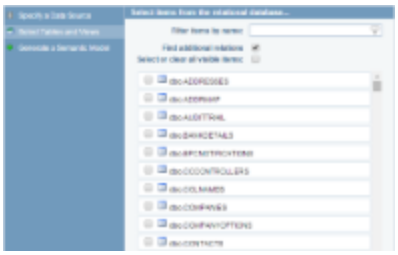
2. Click **Test connection** to test the connection string for the new model.

> **Tip**: You can use any UserContext attribute in the connection string by putting the attribute name between percent signs. For example,
>
> `Provider=SQLOLEDB.1;Data Source=HQ;Initial Catalog=`**`%TenantDatabase%`**`;`
> `User Id=myUsername;Password=myPassword;`

4. Click **Next** to proceed to the **Select Tables and Views** page of the wizard.
5. Click check boxes to select items from the list of available tables and views.

> **Tip**: You can use the **Find additional relations** check box to determine whether to create relations between like-named columns (selected by default) or not. If this checkbox is not selected, a model is generated with predefined relations from the data source, which helps to avoid the generation of extra relations.

You can also select all visible items at once by selecting the **Select/Deselect All Visible Items** check box.



> **Notes:** If you have a lot of items, you can filter what is shown by entering a value in the **Filter Items By Name** box.
>
> If you do not select any items in this step, you can generate an empty model without any data, and later add data to the model manually. See Adding a New Entity, Adding a New Logical Column (Attribute) and Adding a New Logical Relation for details.

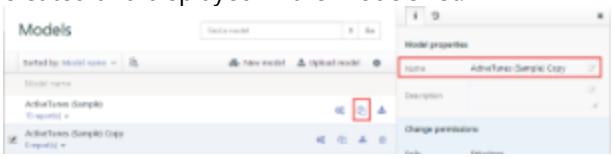
6. Click **Next** to finish the model generation wizard. A new model is generated and appears in the **Models** list, available to users for creating reports in the Report Portal.

Cloning a Model

Another way to create a new model is by cloning. In this case, we choose a model from the list of current models and create its precise copy as an independent model that we can edit.

To clone a model

1. From the Administrator Dashboard, select **Models**.
2. In the Models list, select the model that you want to copy and then click **Clone model** button. A new model is created and displayed in the Models list.



3. To rename the newly created model, click the **clone model** properties button to display the model properties.
4. In the **Name** box, enter the name to update.

> **Note:** When you clone a model, the reports remain bound to the initial model. A newly cloned model is created without any bound reports.

Working with Model Versions

Model versioning allows you to keep and manage various versions of a model, so that you can track modifications and easily revert to any version of the model at any time. ActiveReports Server always uses the latest version of the model to create reports in the Report Portal.

To save a model version

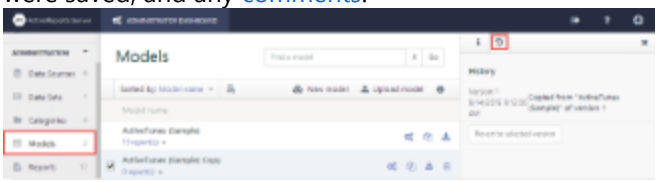
When you upload an existing model, ActiveReports Server saves it in the version history the same as when you modify it in the model editor.

1. After you finish Saving a Modified Model, click the **Save & Publish** button. This creates a new version of the model. The old version, as well as all other previous model versions, is kept in the **History** section of the model.
2. All reports associated with the model automatically bind to the latest model version. Model changes, especially deletions, may break some reports, so compatibility is checked automatically.

   > **Note**: See Managing Model Breaking Changes for more information on how broken reports are handled.

To view the model history

1. In the Administration section of the Administrator Dashboard, click **Models**.
2. In the Models list, next to the model whose history you want to view, click **History**.
3. The **History** section of the model drops down to display a numbered list of model versions, the dates the changes were saved, and any comments.

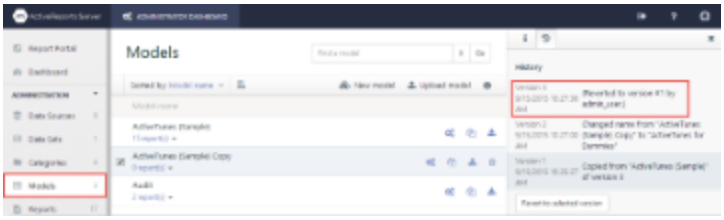


4. To hide the expanded History section, click **History** again.

To revert to an historic model version

1. Click **History** to expand the History section of the model you want to revert.
2. In the list of model versions, next to the version that you want, click **Revert to**.
3. A copy of the selected version of the model becomes the latest version, and a compatibility check runs for all associated reports.

   > **Note**: If a report is not compatible with the selected model version, the compatibility check continues until a compatible model version is found.



Deleting a Model

You can delete any model that does not have any associated reports.

To delete a model

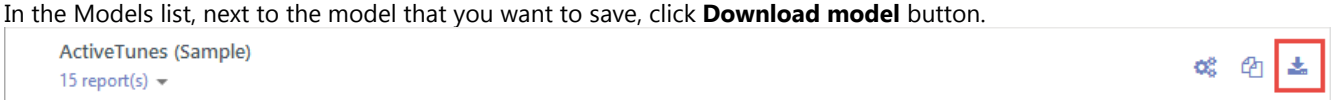
1. Delete any reports associated with the model to enable the delete function. (See Managing Reports for more information.)
2. In the Models list, next to the model that you want to delete, click the **Delete** button.

Uploading and Downloading Models

The upload and download model operations offer an additional means of backing up or transferring models between servers without requiring access to the server console. To streamline upload and download times, models are saved as specially formatted and compressed xml with a file extension of .datamodel.

To download a model

1. In the Models list, next to the model that you want to save, click **Download model** button.



2. The compressed xml model file is copied onto your machine in .datamodel format.

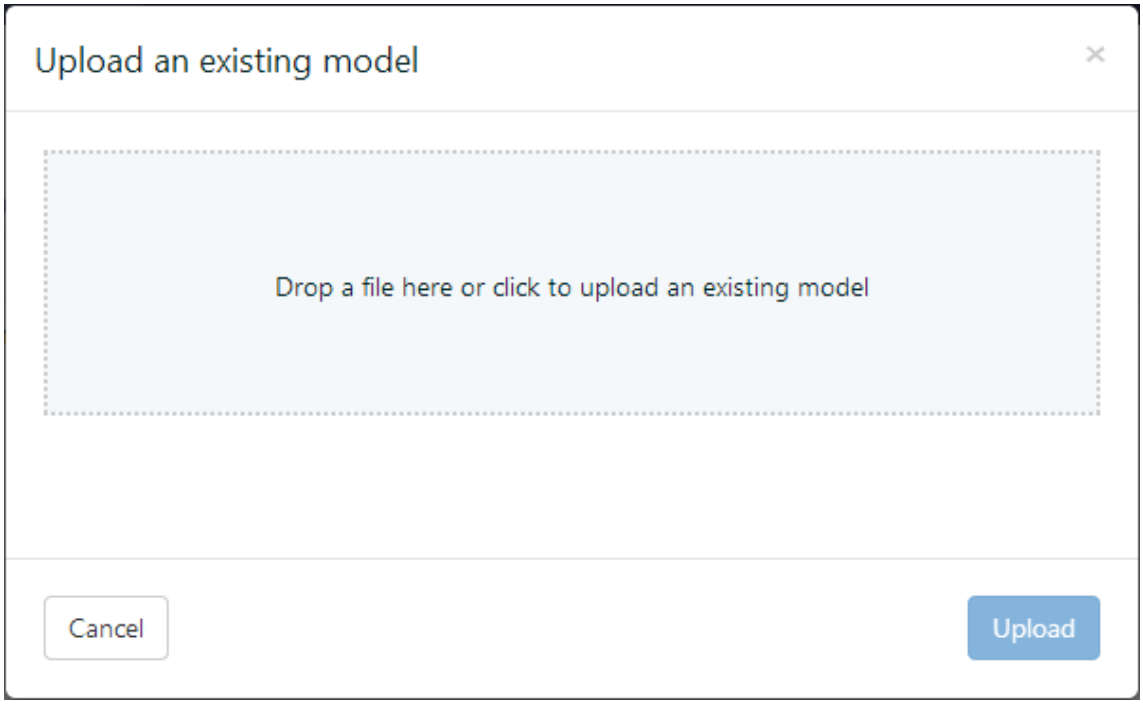
> **Tip**: If your browser does not automatically offer you a choice of where to save the file, you can right-click **Download** and select **Save As**.

To upload a model

1. On the Models page, click the **Upload model** button next to **Upload Existing Model**, click **Browse** and navigate to the location of a locally saved compressed xml model file.
2. In the **Upload an existing model** dialog that appears, you can either drop a model file or click inside the dotted box to browse to a model file.

> **Tip**: Supported file extensions are .datamodel, .xml, and .smdl, so long as the file uses valid semantic model data.

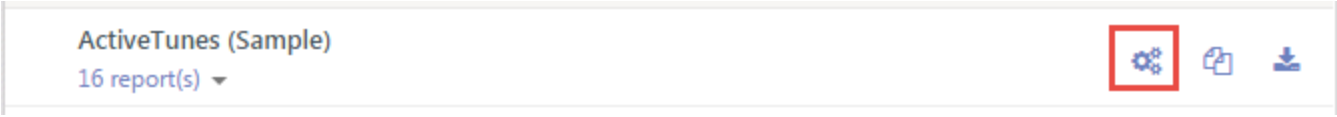
> **Note:** ActiveReports 10 Server does not support the SSRS data model.



3. In the **Open** dialog that appears, navigate to the model file that you want to upload and click **Open**. The dialog closes and the file name appears in the dotted box.
4. Click **Upload** to add the model to the server.

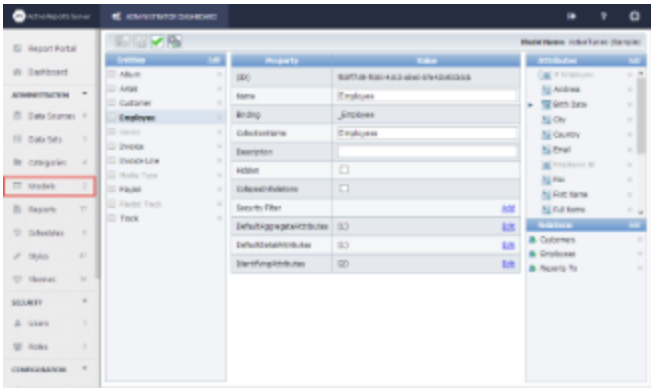
Model Editor Overview

To access the Model Editor, next to a model in the Models list, click **Edit**.



The Model Editor allows you to modify a model's entities, attributes and relations. An asterisk (*) by the model name in edit mode indicates that the model has been modified. Select an entity from the **Entities** list on the left to edit it.

Note: ActiveReports Server Designer Add-On license is required to edit Models.



Changing the Connection String

Learn to modify the Connection String of an existing model.

Modifying an Entity

Learn which properties you can modify on each entity.

Adding a New Logical Table (Entity)

Learn to add new logical tables and to base new tables on existing entities.

Modifying an Attribute

Learn which properties you can modify on an entity's attributes.

Adding a New Logical Column (Attribute)

Learn to add new logical columns and to base new columns on existing attributes.

Modifying a Relation

Learn which properties you can modify on an entity's relations.

Adding a New Logical Relation

Learn to add new logical relations and to base new relations on existing ones.

Deleting an Item

Learn to delete any entity, attribute, or relation.

Saving a Modified Model

Learn to save a working draft of a model and to publish model changes.

Entering Modification Comments

Learn to add comments that help you to identify versions in the model history.

Managing Model Breaking Changes

Learn about changes that may break reports, and how ActiveReports Server handles them.

Model Editor Toolbar Buttons

- **Save & Publish** validates and, if no errors are found, saves modifications as a new version of the model.
- **Save Draft** saves modifications as a local draft copy available only to the user making the modifications. (Use this button to save a draft of a model that does not pass validation.)
- **Validate** checks modified elements for errors.
- **Properties** displays model properties such as the model name and connection string information.

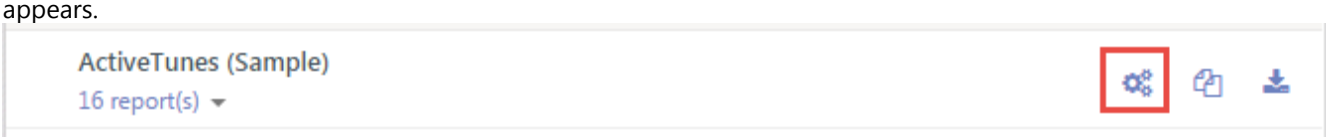
Changing the Connection String

You can modify the connection string of an existing model. This is useful if you want to substitute a UserContext attribute for a value in the connection string when you use LDAP and ActiveDirectory or a custom security provider.

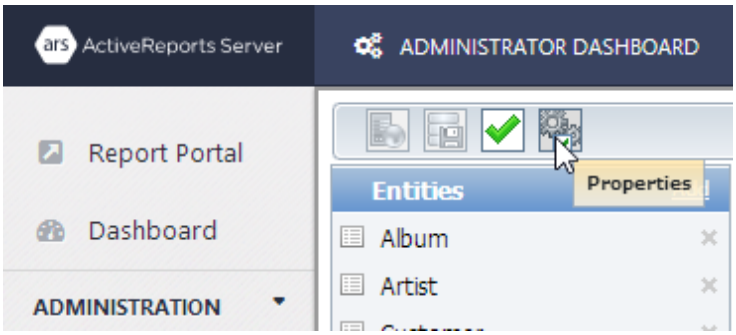
You can also modify the name and command timeout value of the model.

To change the connection string

1. In the Models list, select the model that you want to edit and then click **Edit model** button. The model editor appears.



2. In the Model Editor toolbar, click the **Properties** button.



3. In the workspace, change properties as necessary. The table below details the properties that you can change.

▸ Model Properties

Property	Description
Model Name	Changes the name of the model that appears in the Models list. Note that if you change this name, you must update it in any reports based on the model.
Command Timeout	Specifies the wait time before terminating an attempt to execute a command and generating an error. Leave this value blank to use the provider's default timeout.
Connection String	Specifies the connection string used to access data for the model. This value is disabled for the sample model, but with an actual model, you can edit the connection string here. You can put a UserContext attribute name between percent signs in place of any value in the connection string.

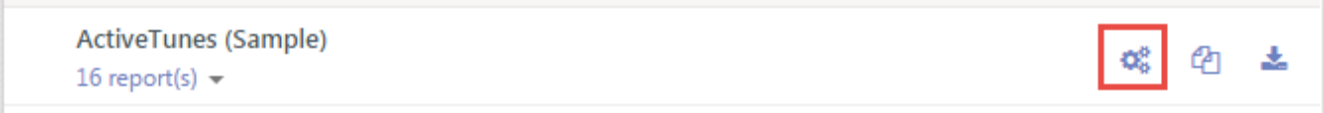
Modifying an Entity

You can modify the properties of an existing entity within a model.

> ⚠ **Caution**: Modifying a model may break reports attached to it. See Managing Model Breaking Changes for more information.

To modify an entity

1. In the Models list, select the model that you want to edit and then click **Edit model** button. The model editor appears.



2. In the model editor that appears, in the **Entities** list on the left, select the entity that you want to modify. The properties appear in the workspace and the Attributes and Relations lists populate with any associated attributes and relations.
3. In the workspace, change properties as necessary. The table below details the properties.

▸ Entity Properties

Property Name	Type	Description
(Id)	ID	A unique identifier for the entity. Used internally by the system.

Name	String	The name of the entity.
Binding	ID	The database object that the entity represents.
CollectionName	String	The name to be used for a collection of instances of the entity. Use this property to ensure that plural nouns are spelled correctly.
Description	String	A description of the entity that appears as tooltip text in the Report Portal when a user hovers the pointer over the entity.
Hidden	Boolean	Indicates whether the entity is hidden from end users.
CollapseInRelations	Boolean	Indicates whether to collapse the entity when it is displayed within a relation.
Security Filter	Expression	Allows you to use custom security providers to filter the data displayed for the entity. See Managing Single Sign-On Security Providers for more information.
DefaultAggregateAttributes	Object	Determines the attributes to display by default when the entity appears in an aggregate.
DefaultDetailAttributes	Object	Determines the attributes to display by default. For example, when you drag the entity onto the designer surface and create a table or chart, these attributes display by default.
IdentifyingAttributes	Object	The set of attributes to use to identify a unique instance of the entity. This is used instead of DefaultDetailAttributes to build a table or chart if that collection is empty, since this one is never empty. This is also used to create the grouping label when you group an entity.

Adding a New Logical Table (Entity)

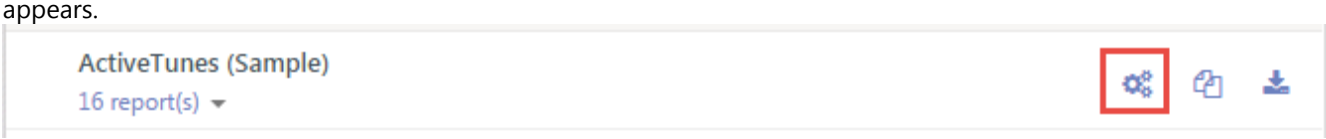
You can create and add a new logical table to a model, or base a new one on an existing entity from the list.

To create a new logical table (entity)

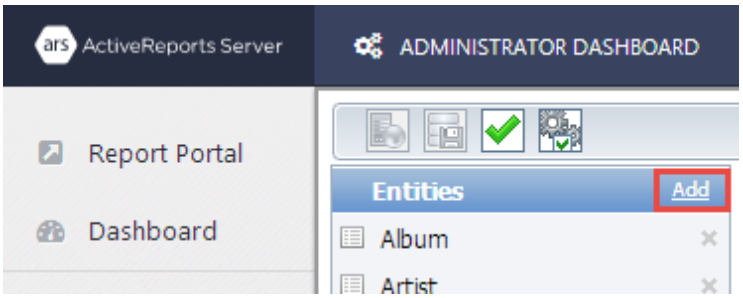
Follow these steps to create a new entity using a SQL query.

Create a new entity from a query

1. In the Models list, select the model that you want to edit and then click **Edit model** button. The model editor appears.



2. In the model editor that appears, in the **Entities** list on the left, click **Add**. The New Entity dialog appears.

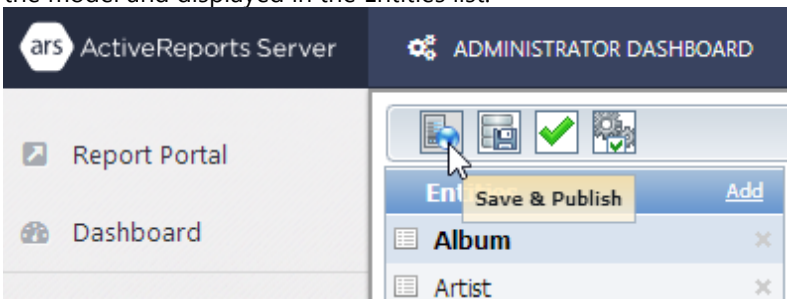


3. In the **New Entity** dialog, enter a **Name** for the new entity. This value appears in the list of entities for the model.
4. In the **Description** box, enter a description that provides users with additional information about the entity, such as when it is appropriate to use it.
5. In the **Query** box, define the entity with a SQL query that pulls the data that you want.
6. Check your query using the commands above the preview box.
   - **Validate** verifies the query and displays any errors it finds in red at the top of the dialog.
   - **Preview Schema** populates a table with information pulled by the query (column name, size, data type, and is nullable).
   - **Preview Data** populates a table with a sampling of the data pulled by the query.

7. Click **OK** to add the new logical table to the model as a new entity.

> **Note**: In the Entities pane, when you select an entity that you created, an **Edit** command appears next to the **Add** command at the top of the pane. Click **Edit** to re-open the New Entity dialog and make changes to the name, description, or query.

8. In the Model Editor toolbar, click the **Save & Publish** button to save the changes. The new logical table is added to the model and displayed in the Entities list.

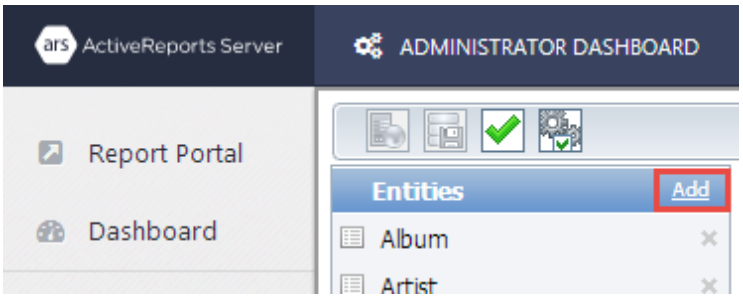


Base a new entity on an existing logical table

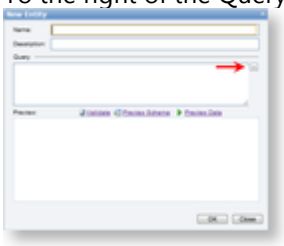
Follow these steps to select an existing logical table to use as a new entity.

▶ Base a new entity on an existing logical table

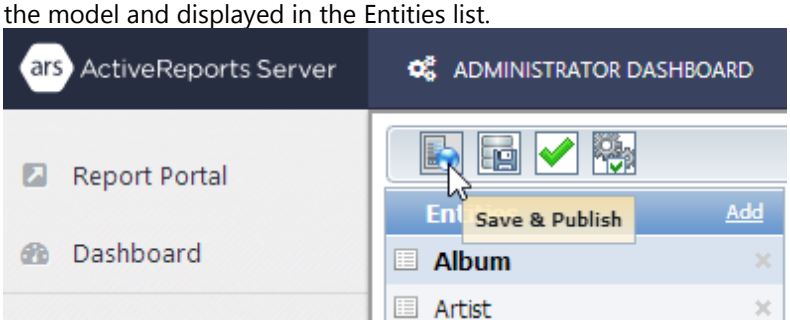
1. In the Entities pane to add an existing entity, click **Add**. The New Entity dialog appears.



2. To the right of the Query box, click the ellipsis button.



3. In the **Select an Existing Table** dialog that appears, select the table that you want and click **OK**. The Query box in the New Entity dialog is populated with the table name, for example **_Expenses**, and the Name box is populated with the user friendly table name, for example **Expenses**.
4. You can check your results using the commands above the preview box.
   - **Validate** verifies the query and displays any errors it finds in red at the top of the dialog.
   - **Preview Schema** populates a table with information pulled by the query (column name, size, data type, and is nullable).
   - **Preview Data** populates a table with a sampling of the data pulled by the query.
5. In the **Description** box, enter a description that provides users with additional information about the entity, such as when it is appropriate to use it.
6. Click **OK** to add the new logical table to the model as a new entity.
7. In the Model Editor toolbar, click the **Save & Publish** button to save the changes. The new logical table is added to the model and displayed in the Entities list.



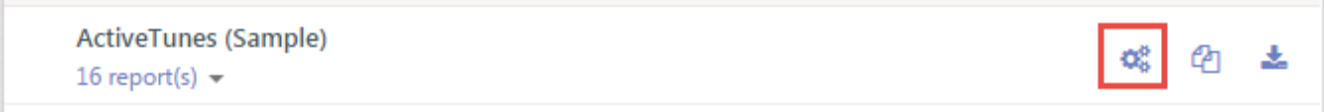
Modifying an Attribute

You can modify the properties of an existing attribute within a model.

> ⚠ **Caution**: Modifying a model may break reports attached to it. See Managing Model Breaking Changes for more information.

To modify an attribute

1. In the Models list, select the model that you want to edit and then click the **Edit model** button.



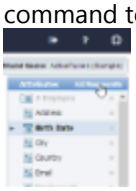
2. In the model editor that appears, in the **Entities** list on the left, select the entity whose attribute you want to modify. The properties appear in the workspace and the Attributes and Relations lists populate with any associated attributes and relations.
3. In the **Attributes** list on the right, select the attribute whose properties you want to modify. The properties appear in the workspace.
4. In the workspace, change properties as necessary. The table below details the properties that you can change.

Attribute properties

Property Name	Type	Description
(Id)	ID	A unique identifier for the attribute. Used internally by the system.
Name	String	The name for the attribute.
Binding	ID	The database object that the attribute represents.
DataType	Enum	The data type for the attribute.
DiscourageGrouping	Boolean	Indicates whether data is grouped by entity instead of by attribute.
Format	String	The default format for the attribute.
Hidden	Boolean	Indicates whether the attribute is hidden from end users.
IsAggregate	Boolean	Indicates whether the attribute is an aggregate.
Nullable	Boolean	Indicates whether the attribute can have a null value and is used in filter and parameter editors to provide special values.
ValueSelection	Enum	Determines what control is used to display attribute values in the filter and parameters editors. ○ **Dropdown** shows lookup values in a dropdown control. ○ **List** shows more values at once than Dropdown. ○ **None** disables the entire lookup process. By default, ActiveReports Server selects this value based on the UAVC (unique attribute values count): ○ Less than 200 - Dropdown ○ Between 200 and 1,000 - List ○ Otherwise - None
Supports Null Values	Boolean	Indicates whether the attribute supports null values.
Supports Blank Values	Boolean	Indicates whether the attribute supports blank values. This property only applies to attributes with a string DataType.

5. If the attribute that you are modifying has a list of variations under it such as date parts (e.g. Expense Day, Expense

Month) or aggregations (e.g. Total Amount, Avg Amount), that you have modified with format changes or deletions, you can click the **Regenerate** command at the top right of the Attributes list to generate all of the available variations. Note that the attribute for which you want to regenerate variations must have focus for the Regenerate command to appear.



Adding a New Logical Column (Attribute)

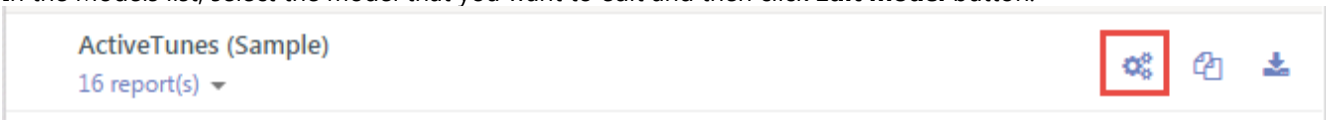
You can create and add a new logical column to an entity, or base a new attribute on an existing logical column from the list.

To base a new attribute on a related entity

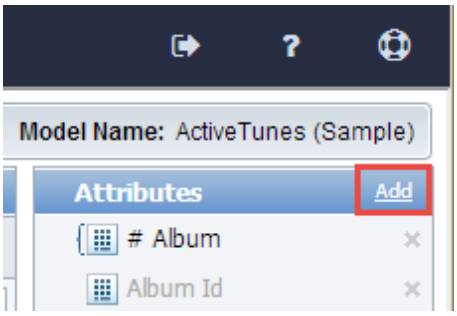
Follow these steps to select an existing attribute on a related entity.

Base a new attribute on an existing logical column

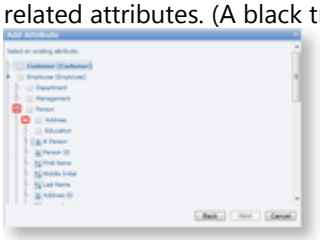
1. In the Models list, select the model that you want to edit and then click **Edit model** button.



2. In the model editor that appears, in the **Entities** list on the left, select the entity to which you want to add an attribute.
3. In the **Attributes** list on the right, click **Add**. The Add Attribute wizard appears.

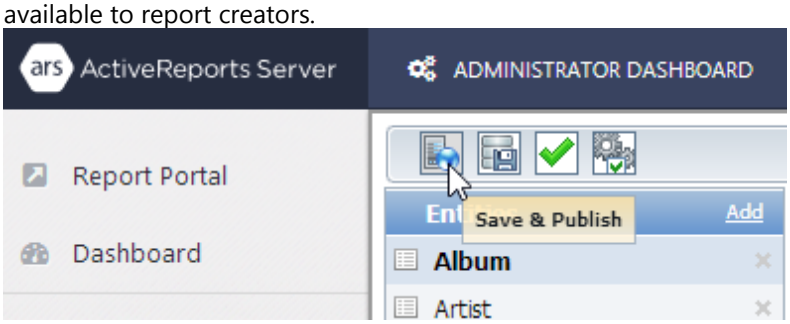


4. In the **Add Attribute** wizard, click the **Related Entity** selection.
5. In the list of existing attributes that appears, a white triangle indicates that you can expand the node to reveal related attributes. (A black triangle indicates that a node is expanded.)



6. Select the one that you want to use and click **Next**.
7. In the **Name** and **Description** boxes that appear, enter the name and description for the new attribute.

8. Click **Finish**. The new attribute appears in the Attributes list.
9. In the Model Editor toolbar, click the **Save & Publish** button to save the changes. The new logical column is available to report creators.

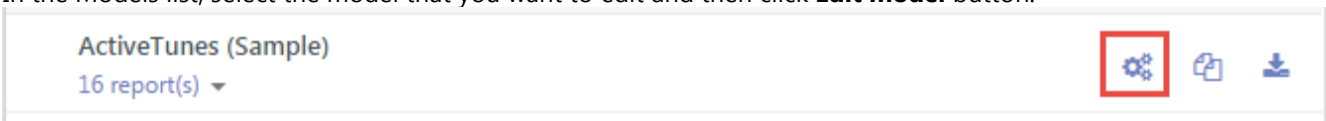


To base a new attribute on a database field

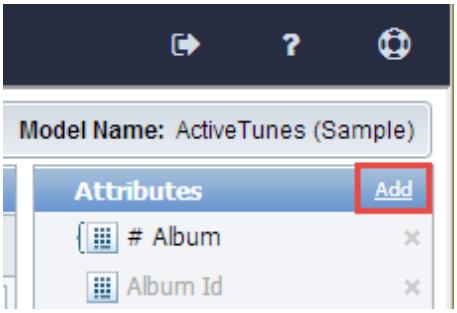
Follow these steps to select a database field to use as an attribute.

Base a new attribute on a database field

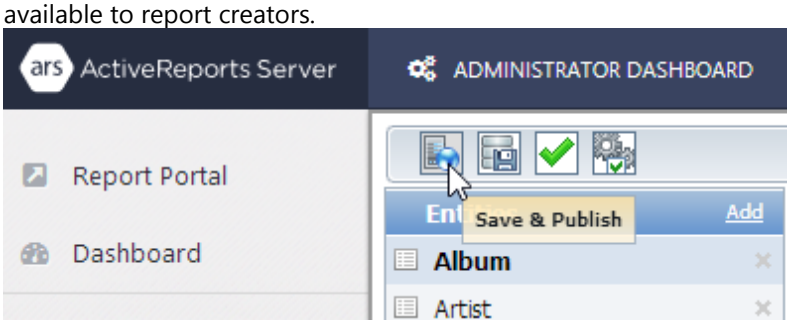
1. In the Models list, select the model that you want to edit and then click **Edit model** button.



2. In the model editor that appears, in the **Entities** list on the left, select the entity to which you want to add an attribute.
3. In the **Attributes** list on the right, click **Add**. The Add Attribute wizard appears.



4. In the **Add Attribute** wizard, click the **Database Field** selection.
5. In the list of existing columns that appears, select the one that you want to use and click **Next**.
6. In the **Name** and **Description** boxes that appear, enter the name and description for the new attribute.
7. Click **Finish**. The new attribute appears in the Attributes list.
8. In the Model Editor toolbar, click the **Save & Publish** button to save the changes. The new logical column is available to report creators.

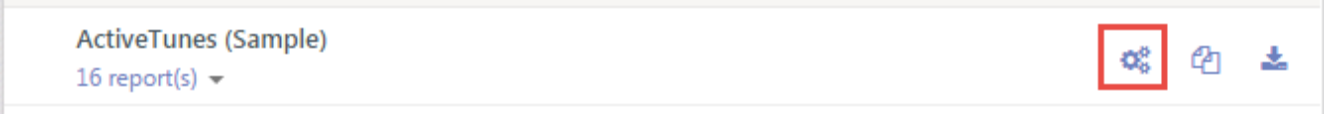


To create a new logical column (attribute)

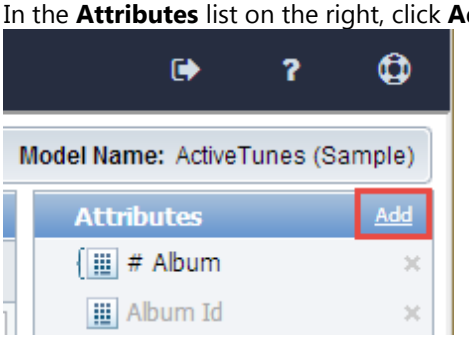
Follow these steps to create a new attribute using a SQL expression definition.

Creating a new attribute from an expression

1. In the Models list, select the model that you want to edit and then click **Edit model** button.



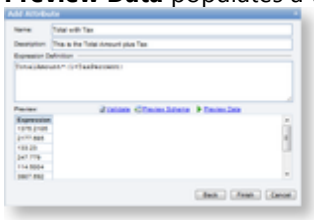
2. In the model editor that appears, in the **Entities** list on the left, select the entity to which you want to add an attribute.
3. In the **Attributes** list on the right, click **Add**. The Add Attribute wizard appears.



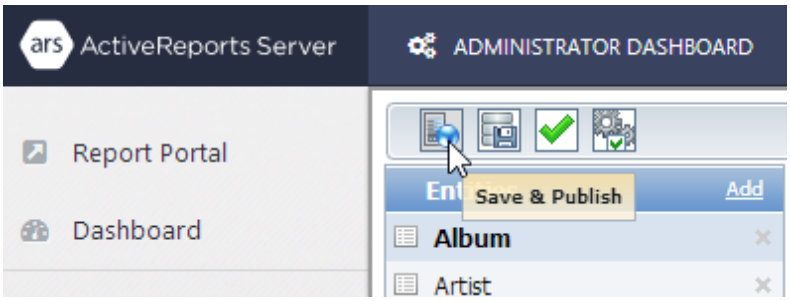
4. In the **Add Attribute** wizard, click the **SQL Expression** selection.
5. In the **Name** box that appears, enter the name for the new attribute. This value appears in the list of entities for the model.
6. In the **Description** box, enter a description that provides users with additional information about the attribute, such as when it is appropriate to use it.
7. In the **Expression Definition** box, define the expression for the logical column with a SQL expression that pulls the data that you want.

   Note: The syntax to use for expressions is T-SQL for SQL data sources or PL-SQL for Oracle data sources.

8. Check your query using the commands above the preview box.
   - **Validate** verifies the expression and displays any errors it finds in red at the top of the dialog.
   - **Preview Schema** populates a table with information pulled by the expression (column name, size, data type, and is nullable).
   - **Preview Data** populates a table with a sampling of the data pulled by the expression.



9. Click **Finish**. The new attribute appears in the Attributes list.
10. In the Model Editor toolbar, click the **Save & Publish** button to save the changes. The new logical column is available to report creators.



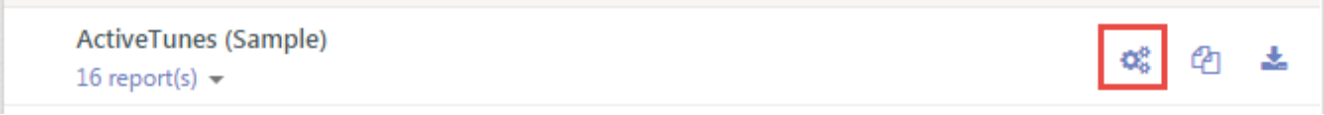
Modifying a Relation

A relation item in a model describes the relationships between entities. You can modify the properties of an existing relation within a model.

> **Caution**: Modifying a model may break reports attached to it. See Managing Model Breaking Changes for more information.

To modify a relation

1. In the Models list, select the model that you want to edit and then click **Edit model** button.



2. In the model editor that appears, in the **Entities** list on the left, select the entity whose relation you want to modify. The properties appear in the workspace and the Attributes and Relations lists populate with any associated attributes and relations.
3. In the **Relations** list on the lower right, select the relation whose properties you want to modify. The properties appear in the workspace.
4. In the workspace, change properties as necessary. The table below gives details about the properties.

▸ Relation properties

Property Name	Type	Description
(Id)	ID	A unique identifier for the relation. Used internally by the system.
Name	String	The name for the relation.
Binding	ID	The database object that the relation represents.
Cardinality	Enum	Defines the rules and restrictions for the relation's behavior in the entity tree and in semantic query building. Possible values are One, Many, OptionalOne, or OptionalMany. This property determines whether the Recursive property has an effect on the relation.
Description	String	A description of the relation that appears as tooltip text in the Report Portal when a user hovers the pointer over the relation.

Hidden	Boolean	Indicates whether the entity reached by the relation is hidden from end users.
Recursive	Boolean	Indicates whether the relation can be added more than once to the same path. This applies only to relations with a Cardinality of One or OptionalOne. For relations with a Cardinality of Many or OptionalMany, this property is ignored.
Include self	Boolean	Indicates whether to allow the relation to be accessible to itself in the same recurring path. This applies only in the second level of an entity.
HiddenFields	Object	Defines the set of entity fields that is hidden from end users when they reach the entity through the relation.
RelatedRelation	Object	Indicates the backward relation (the relation for the related entity) of the relation. You cannot edit this value, but you can use the navigate link to go to the related relation.
RelatedEntity	Object	The entity to which the relation is bound. You cannot edit this value, but you can use the navigate link to go to the related entity.

Adding a New Logical Relation

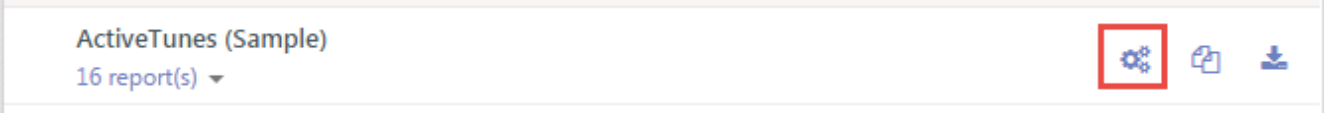
You can create and add a new logical relation to an entity, or base a new relation on an existing logical relation from the list.

To create a new relation

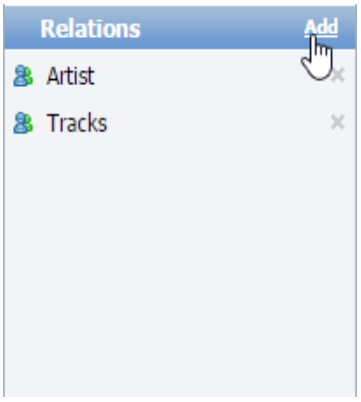
Follow these steps to create a new logical relation by specifying a target entity, and source and target attributes.

Creating a new logical relation

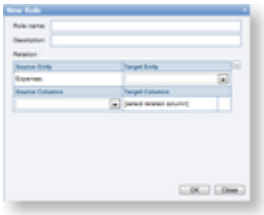
1. In the Models list, select the model that you want to edit and then click **Edit model** button.



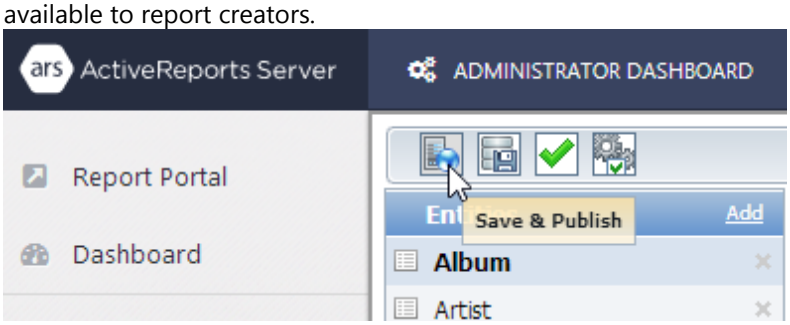
2. In the model editor that appears, in the **Entities** list on the left, select the entity to which you want to add a relation.
3. In the **Relations** list on the lower right, click **Add**. The New Relation dialog appears.



4. In the **New Relation** dialog, in the **Name** box, enter the name for the new relation. This value appears in the list of relations for the model.



5. In the **Description** box, enter a description that provides users with additional information about the attribute, such as when it is appropriate to use it.
6. In the **Relation** section, the **Source Entity** is already set (based on the entity you selected to which to add a relation). Drop down the **Target Entity** list and select a target for the relation. This enables the Target Columns drop-down list below.
7. Drop down the **Source Columns** list and select a source attribute for the relation. This adds a new pair of drop-down lists for additional sources and targets below.
8. Drop down the **Target Columns** list and select a target attribute for the relation.
9. Click **OK**. The new relation appears in the Relations list.
10. In the Model Editor toolbar, click the **Save & Publish** button to save the changes. The new logical relation is available to report creators.

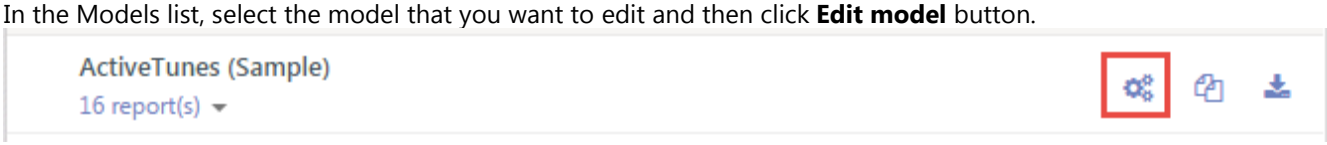


To base a new logical relation on an existing relation

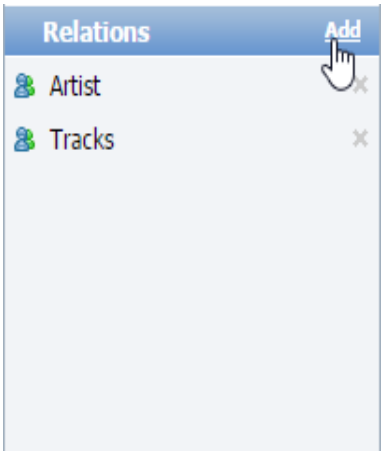
Follow these steps to select a relation to use as a new logical relation.

Base a new attribute on an existing logical column

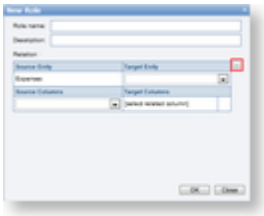
1. In the Models list, select the model that you want to edit and then click **Edit model** button.



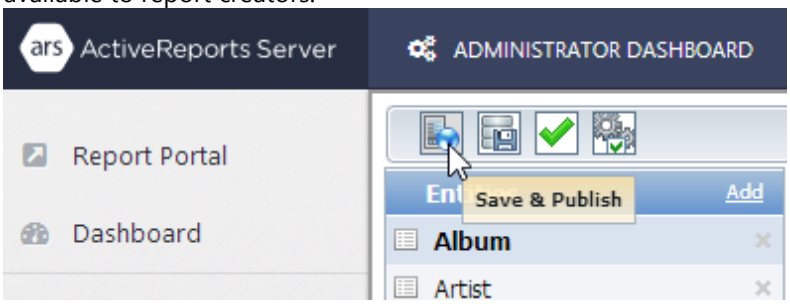
2. In the model editor that appears, in the **Entities** list on the left, select the entity to which you want to add a relation.
3. In the **Relations** list on the lower right, click **Add**. The New Relation dialog appears.



4. In the **New Relation** dialog, in the **Relation name** box, enter the name for the new relation. This value appears in the list of relations for the model.
5. In the **Description** box, enter a description that provides users with additional information about the attribute, such as when it is appropriate to use it.
6. Next to the Relation section, click the ellipsis button.



7. In the Select an Existing Relation dialog that appears, select the one that you want to use and click **OK**.
8. In the New Relation dialog, click **OK**. The new relation appears in the Relations list.
9. In the Model Editor toolbar, click the **Save & Publish** button to save the changes. The new logical relation is available to report creators.



Deleting an Item

You can delete any model item, whether entity, attribute or relation, by clicking the X button to the right each item in the Model Editor.

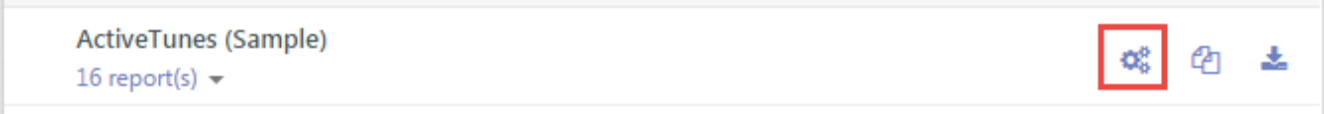
> ⚠ **Caution**: Modifying a model may break reports attached to it. See Managing Model Breaking Changes for more information.

To delete an entity

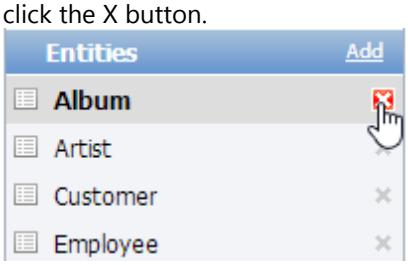
Follow these steps to delete an entity from a model.

Delete an entity

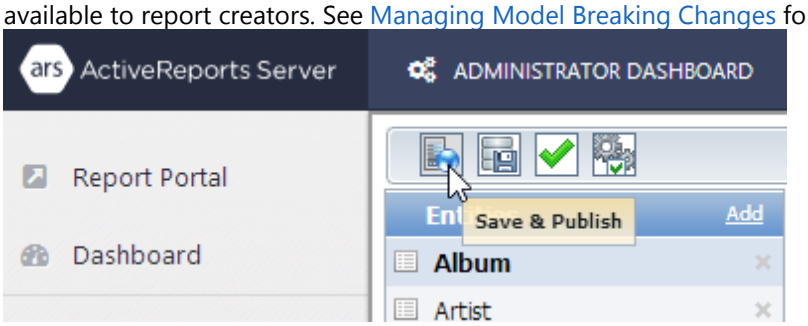
1. In the Models list, select the model that you want to edit and then click **Edit model** button.



2. In the model editor that appears, in the **Entities** list on the left, to the right of the entity that you want to delete, click the X button.



3. In the Model Editor toolbar, click the **Save & Publish** button to save the changes. The deleted entity is no longer available to report creators. See Managing Model Breaking Changes for more information.



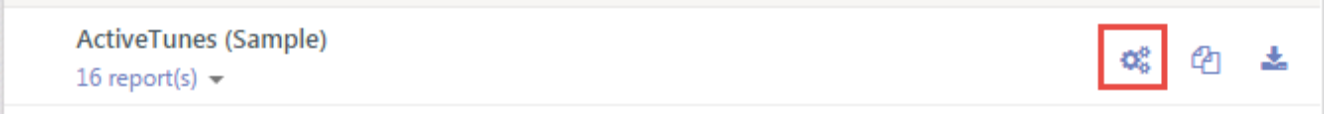
Note: It is recommended that you hide an entity or an attribute of the model instead of deleting it. This way, the entity or attribute is not available for use in new reports, but existing reports are not broken.

To delete an attribute or relation

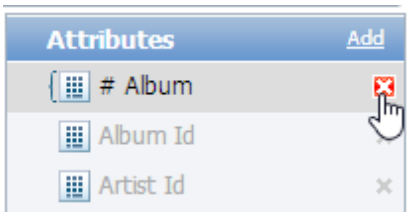
Follow these steps to delete an attribute or relation from an entity.

Delete an attribute or relation

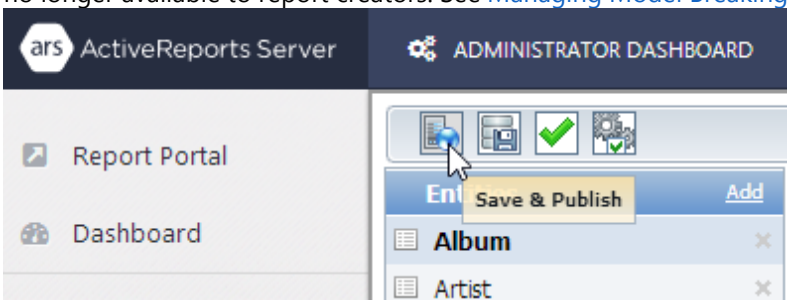
1. In the Models list, select the model that you want to edit and then click **Edit model** button.



2. In the model editor that appears, in the **Entities** list on the left, select the entity that contains the attribute or relation that you want to delete. The Attributes and Relations lists populate with any associated attributes and relations.
3. In the **Attributes** list on the right, or the **Relations** list on the lower right, to the right of the attribute or relation that you want to delete, click the X button.



4. In the Model Editor toolbar, click the **Save & Publish** button to save the changes. The deleted attribute or relation is no longer available to report creators. See Managing Model Breaking Changes for more information.



Saving a Modified Model

When you modify a model in the Model Editor, you can save the modifications and make them available to users right away, or if you have many changes to make, or a specific roll-out date, you can save a draft until you are ready to make your changes public.

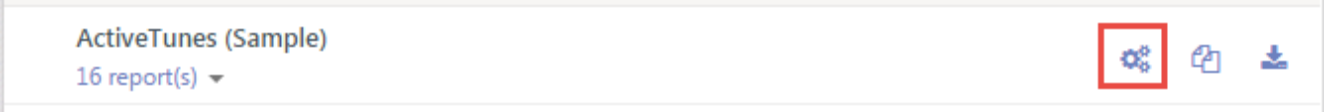
> ⚠ **Caution**: Modifying a model may break reports attached to it. See Managing Model Breaking Changes for more information.

To save changes and make them public

Follow these steps to publish your changes so that report creators can use the updated model.

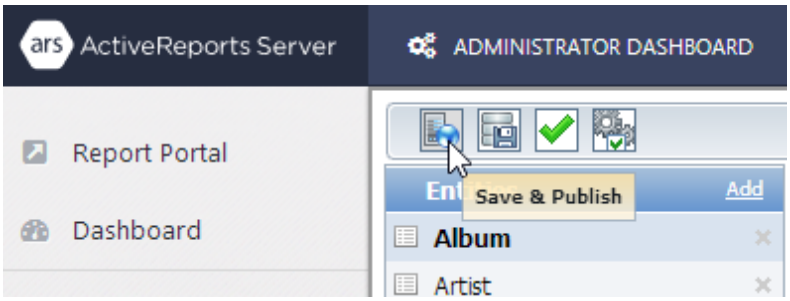
▶ Publishing changes

1. In the Models list, select the model that you want to edit and then click **Edit model** button.



2. Modify, add, or delete any entities, attributes, or relations that you need to modify. (See Model Editor Overview for topics relating to these actions.)

3. In the Model Editor toolbar, click the **Save & Publish** button to save the changes. This creates a new version of the model. The old version, as well as all other previous model versions, is kept in the **History** section of the model. (See Working with Model Versions for more information.)



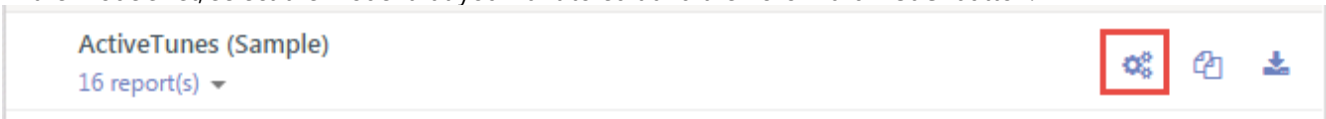
4. All reports associated with the model automatically bind to the latest model version. Model changes, especially deletions, may break some reports, so compatibility is checked automatically. (See Managing Model Breaking Changes for more information.)

To save a draft of your changes

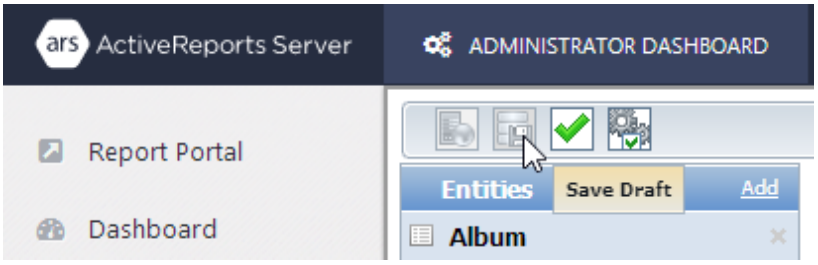
Follow these steps to save a working draft that you can return to later, without releasing the changes to report creators. ActiveReports Server automatically saves a draft while you are working, but you can save the draft any time.

▸ Saving a working draft

1. In the Models list, select the model that you want to edit and then click **Edit model** button.



2. Modify, add, or delete any entities, attributes, or relations that you need to modify. (See Model Editor Overview for topics relating to these actions.)
3. In the Model Editor toolbar, click the **Save Draft** button to save a draft copy of the model with the changes you have made. This does not create a new version of the model. The old version is still what report creators use.



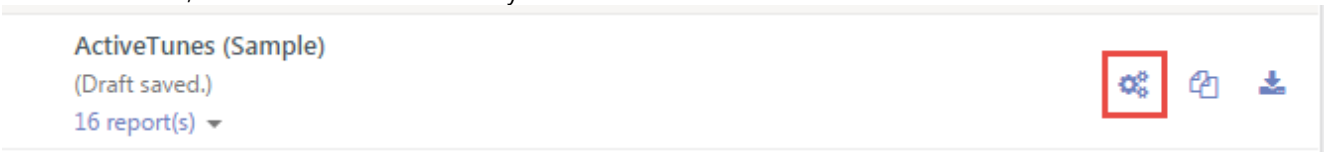
4. The draft copy is available only to the user who made the modifications, and is marked in the Models list with (Draft Saved).

To discard a draft

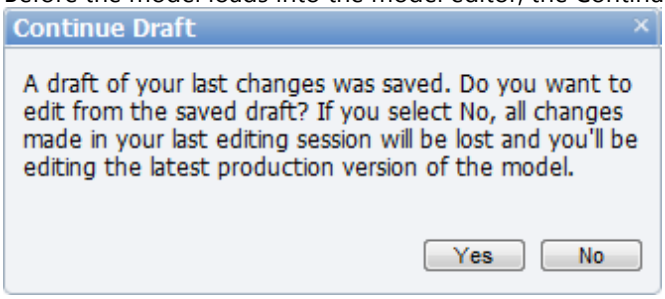
ActiveReports 10 Server automatically saves a draft while you are working, and you can save it as you make changes, but if you do not want to keep the changes, you can discard them.

▸ Discarding a draft

1. In the Model list, select the model for which you have saved the draft and then click **Edit model** button.



2. Before the model loads into the model editor, the Continue Draft message box appears.



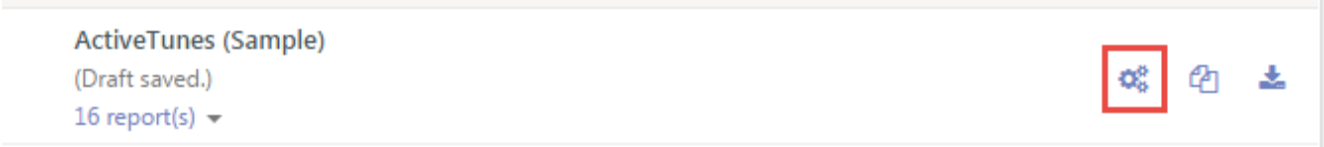
3. Click **No**. The draft changes are discarded and the unchanged version of the model loads into the model editor.

To apply the modifications of the draft copy to the model

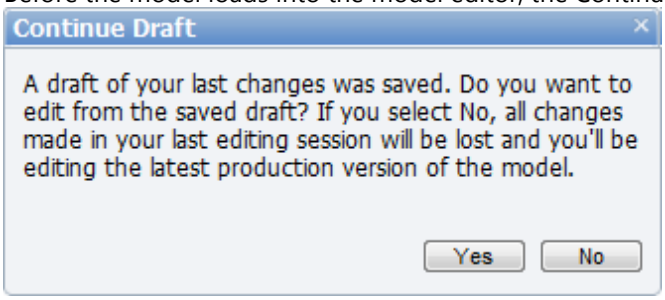
Follow these steps to keep changes saved to a working draft, and publish the changes for report creators.

▶ Publishing a draft

1. In the Model list, select the model for which you have saved the draft and then click **Edit model** button.

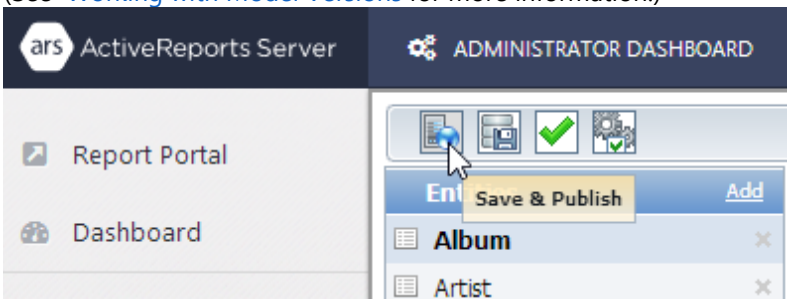


2. Before the model loads into the model editor, the Continue Draft message box appears.



3. Click **Yes**. The changed draft version of the model loads into the model editor.

4. In the Model Editor toolbar, click the **Save & Publish** button to save the changes. This creates a new version of the model. The old version, as well as all other previous model versions, is kept in the **History** section of the model. (See Working with Model Versions for more information.)



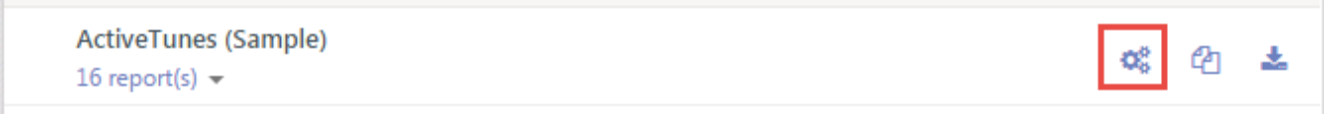
5. All reports associated with the model automatically bind to the latest model version. Model changes, especially deletions, may break some reports, so compatibility is checked automatically. (See Managing Model Breaking Changes for more information.)

Entering Modification Comments

ActiveReports Server automatically adds a comment with the name of the administrator who published a change, or you can enter your own comments after you modify and save a model. These comments appear in the history of model versions so that you can easily find the version you want. See Working with Model Versions for more information.

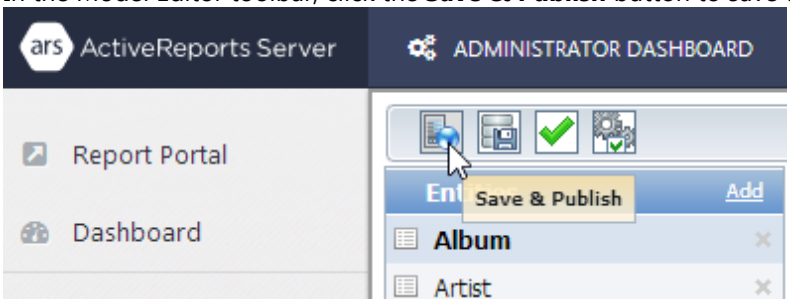
To enter modification comments

1. In the Models list, select the model that you want to edit and then click **Edit model** button.

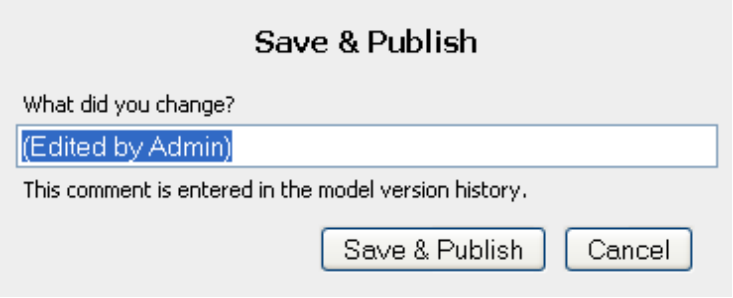


2. Modify, add, or delete any entities, attributes, or relations that you need to modify. (See Model Editor Overview for topics relating to these actions.)

3. In the Model Editor toolbar, click the **Save & Publish** button to save the changes.



4. In the **Save & Publish** dialog that appears, the automatically generated comments are highlighted.



5. Enter your comments and click **Save & Publish**. This creates a new version of the model. The old version, as well as all other previous model versions, is kept in the **History** section of the model.

> **Note:** If you do not wish to enter your own comments, click **Save & Publish** anyway to save the automatically generated comments.

6. The comments are displayed in the model history. (See Working with Model Versions for more information.)

Managing Model Breaking Changes

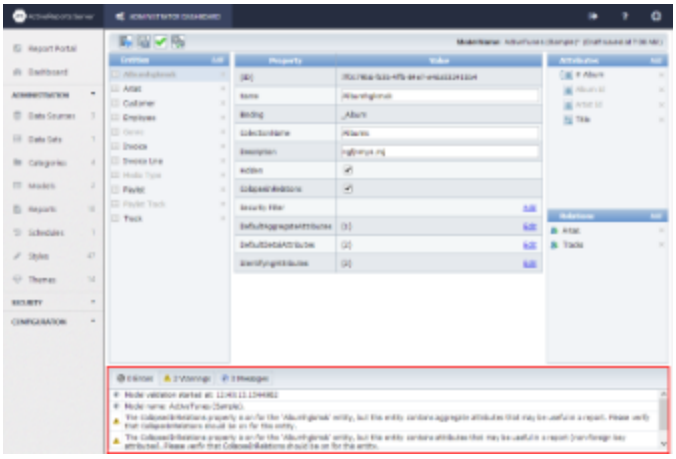
When you modify a model, certain changes may break reports that are associated with the model. Such model changes are:

1. Deleting an entity, attribute or relation.
2. Modifying the Cardinality property of a model's relation.

> **Note:** It is recommended that you hide an entity or an attribute of the model instead of deleting it. This way, the entity or attribute is not available for use in new reports, but existing reports are not broken.

When you save a change that breaks a report, a warning appears below the work area for each broken report.

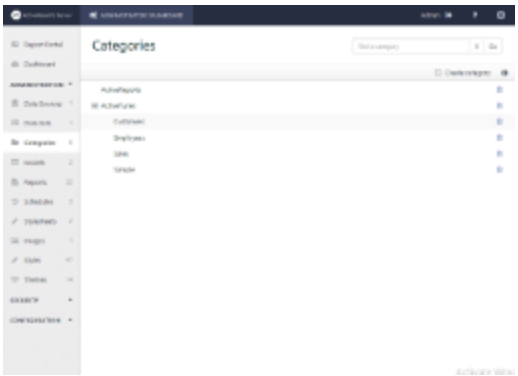
> *The "ReportName" report was broken by your changes, therefore it is still bound to the old version of this model so that users can view this report without errors. For more information about this warning, see the "Model Breaking Changes" topic in the help.*



Report Categories

ActiveReports Server provides the ability to create categories and sub-categories to help manage your reports better. The purpose of this feature is to allow users to organize their reports into categories to make report navigation easier.

You can create, edit, rename, move or delete a category or create sub-categories within your category.



There are two types of categories available in ActiveReports Server.

System Categories

The **Administrator** creates and manages **System Categories** from the Administrator Dashboard. Both the Administrator

and end users can view these categories but only the administrator can create, rename, edit, move or delete a system category.

The **System Categories** are visible to the End Users in the **Categories** tab of the **Report Portal** but the end users can only assign these System Categories if the Administrator gives them permission to do so. The System category names are highlighted in bold in the **Categories** panel of the Report Portal. See Managing Reports, for further information. The Administrator can assign multiple System Categories to a report, and can also move multiple reports to a single category at the same time.

Role Root Category

Role Root Category is a system category that is set as the top level category for each user role. All the sub-categories under this level are available to the users in that particular user role. If more than one role is assigned to any user, then the user can view the category hierarchy for both the roles on the Report Portal. The Administrator can set a role root category for each role using the Role Properties option. Each user is assigned a root category based on their role.

For example, an administrator wants to create two system categories, namely Sales and Marketing, that are only visible to the users in their respective roles. To do this, set the Sales system category as the role root category for the Sales role, and the Marketing system category as the role root category for the Marketing role. For further details, see Managing System Categories.

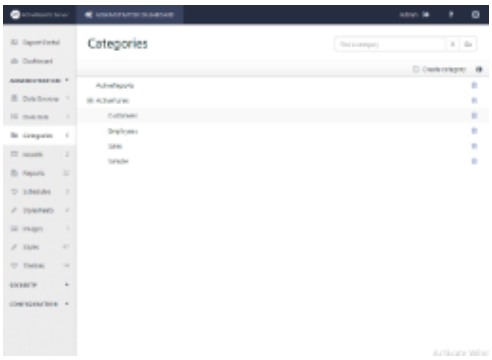
> **Note**: By default, the **Root** category is assigned for a role if no role root category specified. This allows the role to access all categories.

Personal Categories

End users create and manage **Personal Categories** from the **Report Portal**. These categories are unique to each end user and are not visible on the Administrator Dashboard. End users can create, rename, edit, move or delete these **Personal Categories** and can assign one or multiple **Personal Categories** to a report. or even move multiple reports to a single category at the same time. See Personal Categories for further information.

Managing System Categories

The Administrator can add, delete, edit, move and modify existing System Categories from the Administrator Dashboard. These System Categories are available to the Administrator and End Users. See Report Categories for further details on System Categories.

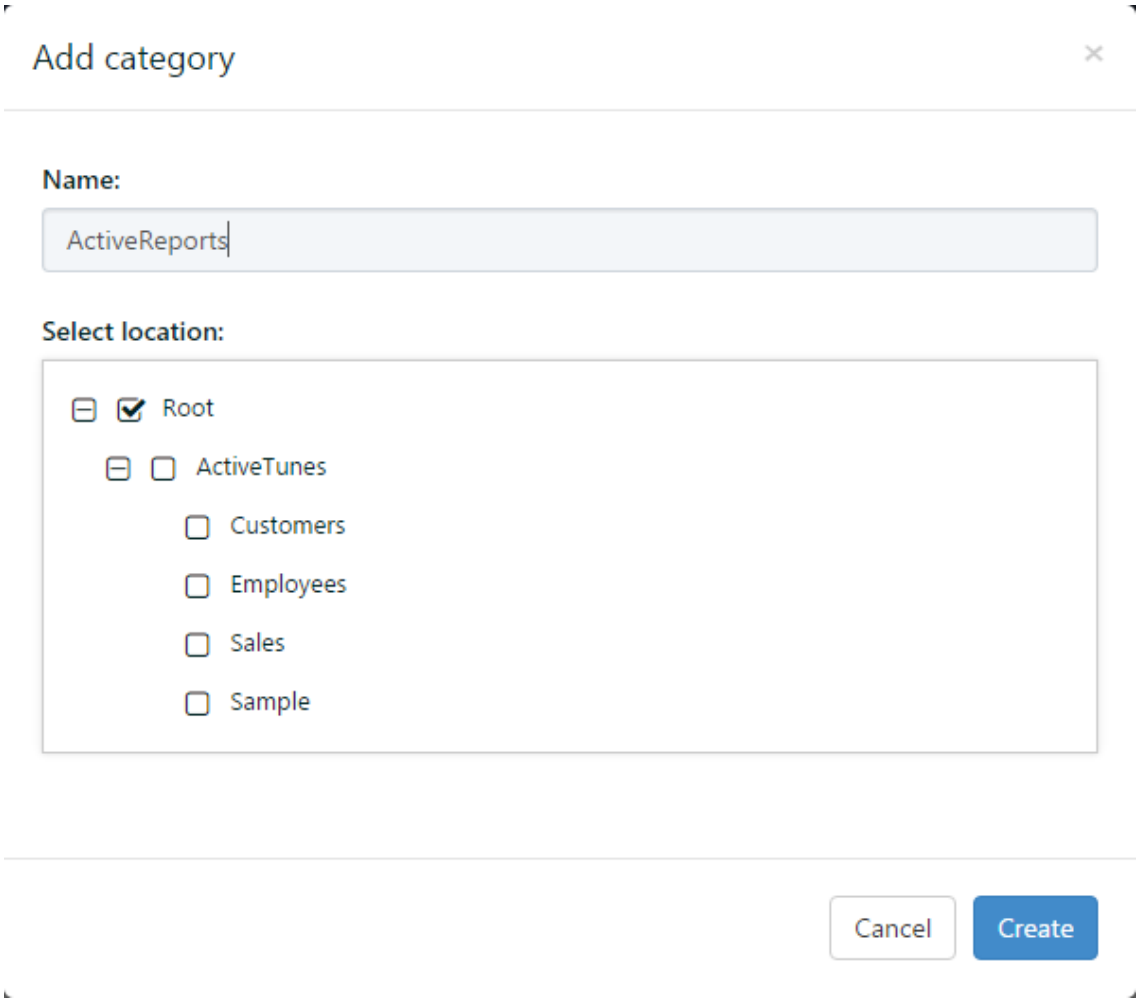


- **Add a New System Category**
- **Edit an Existing System Category**
- **Delete a System Category**

- **Assign Multiple System Categories to a Report**
- **Assign a Single System Category to Multiple Reports**
- **Change the System Category of a Report**
- **Set a Role Root Category**
- **Hide Empty System Categories**

Add a New System Category

1. In the Administration section of the Administrator Dashboard, click **Categories**.
2. In the list of categories that appear, click the **Add Category** button to open the **Add category** dialog.



3. In the **Add category** dialog, enter the Category **Name** and select a **Location** where you want to create a System category.

   > **Note**: System category location can either be root (Parent node) or you can create a sub-category under any existing category.

4. Click **Create** to add the new System category to the list of categories.

Edit an Existing System Category

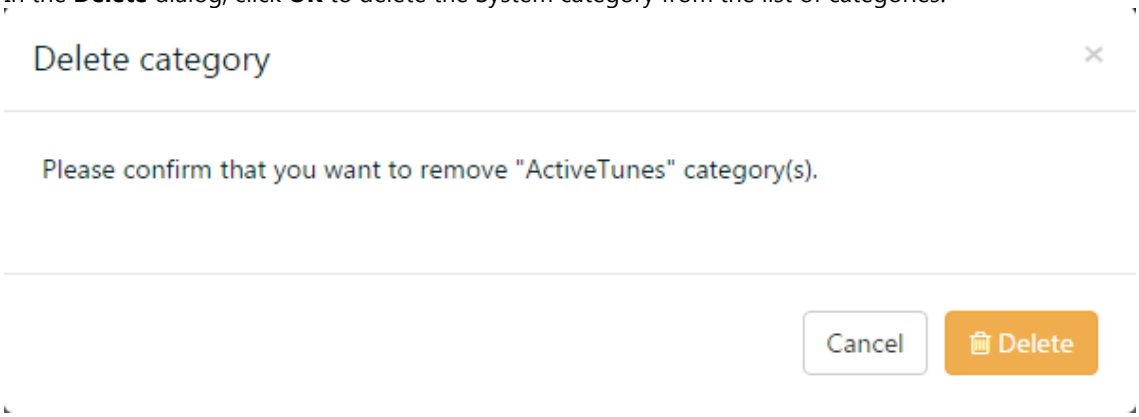
1. In the Administration section of the Administrator Dashboard, click **Categories**.
2. On the **Categories** page, select a category from the list, and then click the **category properties** button to display the

category properties.

3. From the **Category properties** pane, modify the Category **Name** or **Location** of the System category.

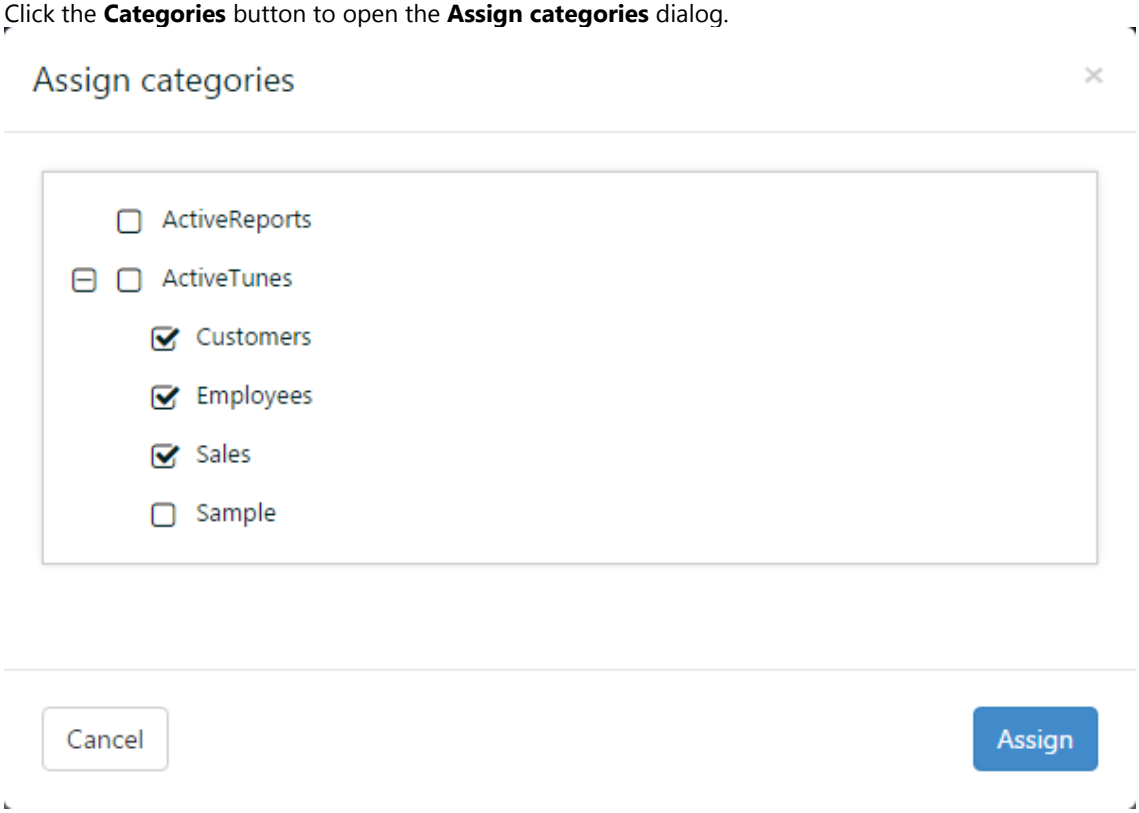
Delete a System Category

1. In the Administration section of the Administrator Dashboard, click **Categories**.
2. On the **Categories** page, click the **Delete tag** button next to the category.
3. In the **Delete** dialog, click **OK** to delete the System category from the list of categories.



Assign Multiple System Categories to a Report

1. In the Administration section of the Administrator Dashboard, click **Reports**.
2. In the report list that appears, select the check-box to the left of the report.
3. Click the **Categories** button to open the **Assign categories** dialog.

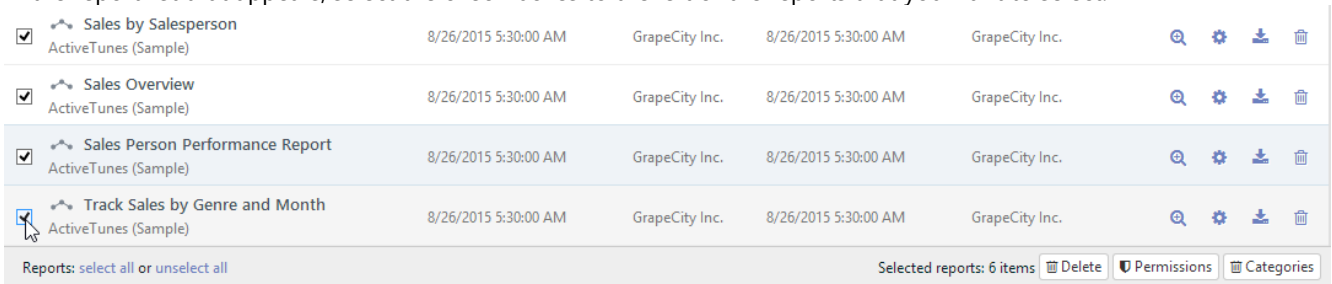


4. In the **Assign categories** dialog, select check boxes to assign your report to multiple System Categories.

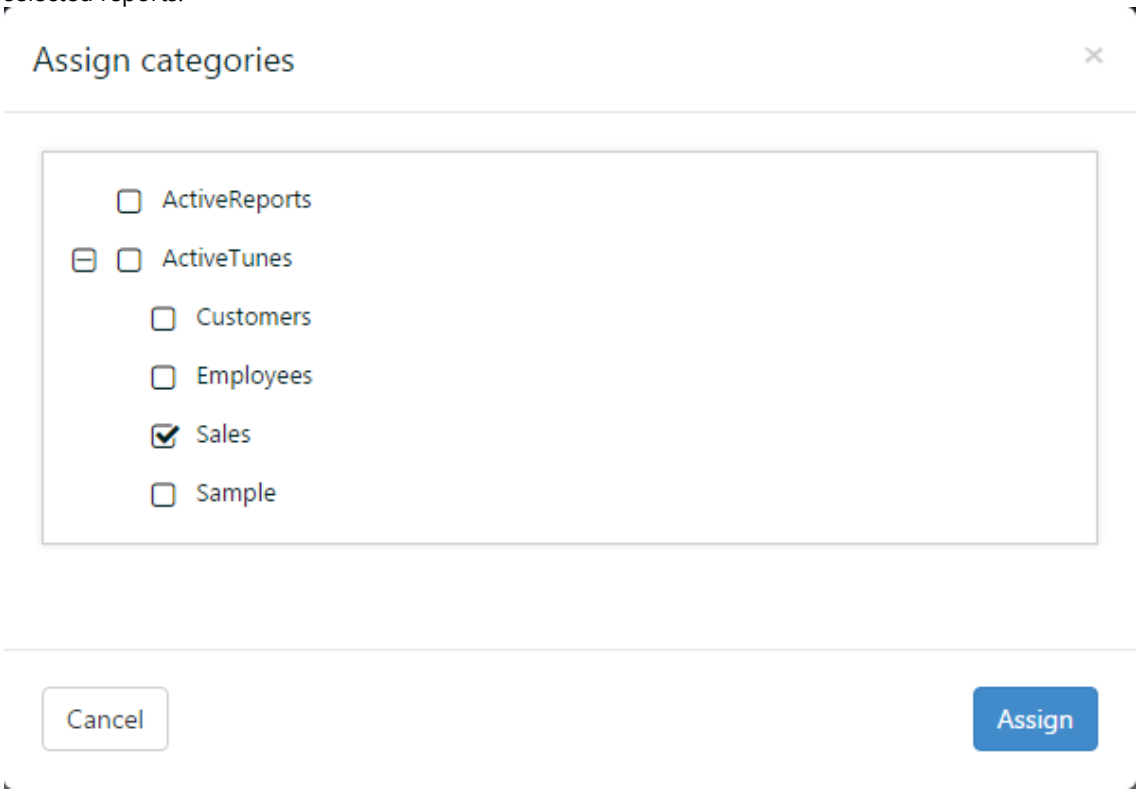
5. Click **Assign** to assign multiple System Categories to your report.

Assign a Single System Category to Multiple Reports

1. In the Administration section of the Administrator Dashboard, click **Reports**.
2. In the report list that appears, select the check-boxes to the left of the reports that you want to select.



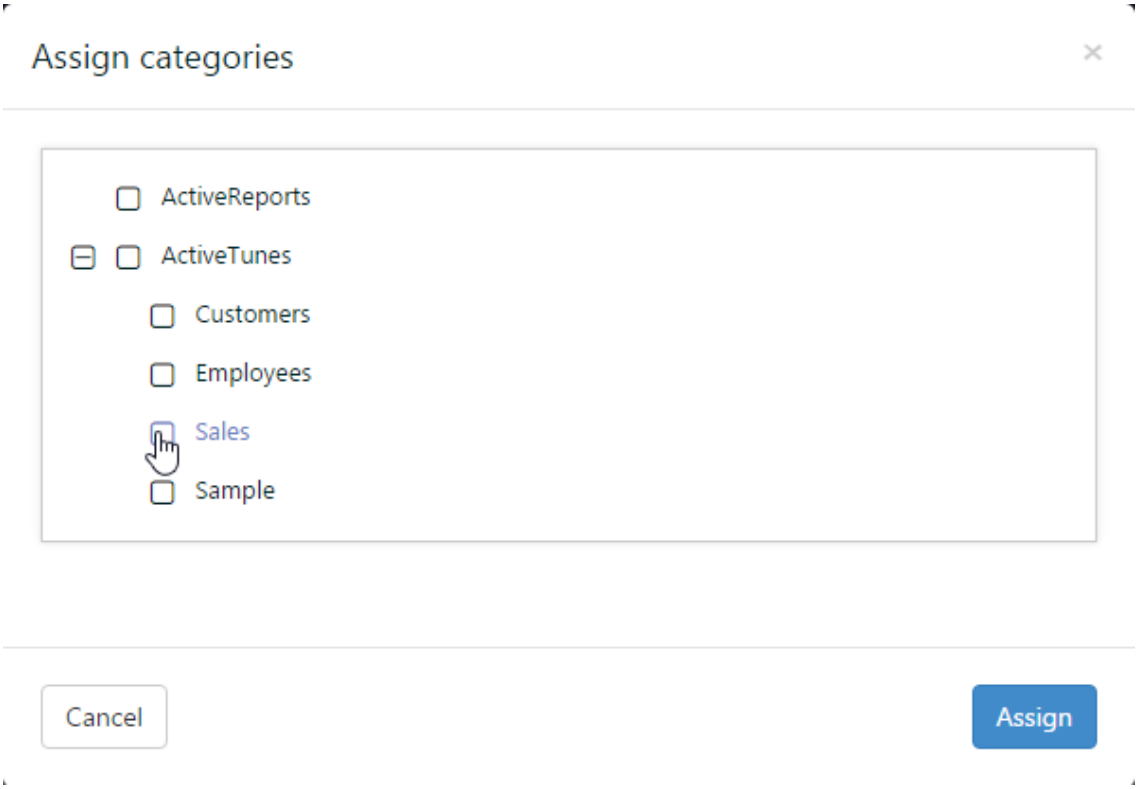
3. Click the **Categories** button to open the **Assign categories** dialog.
4. In the **Assign categories** dialog, check any check-box to select the System category to which you want to add all the selected reports.



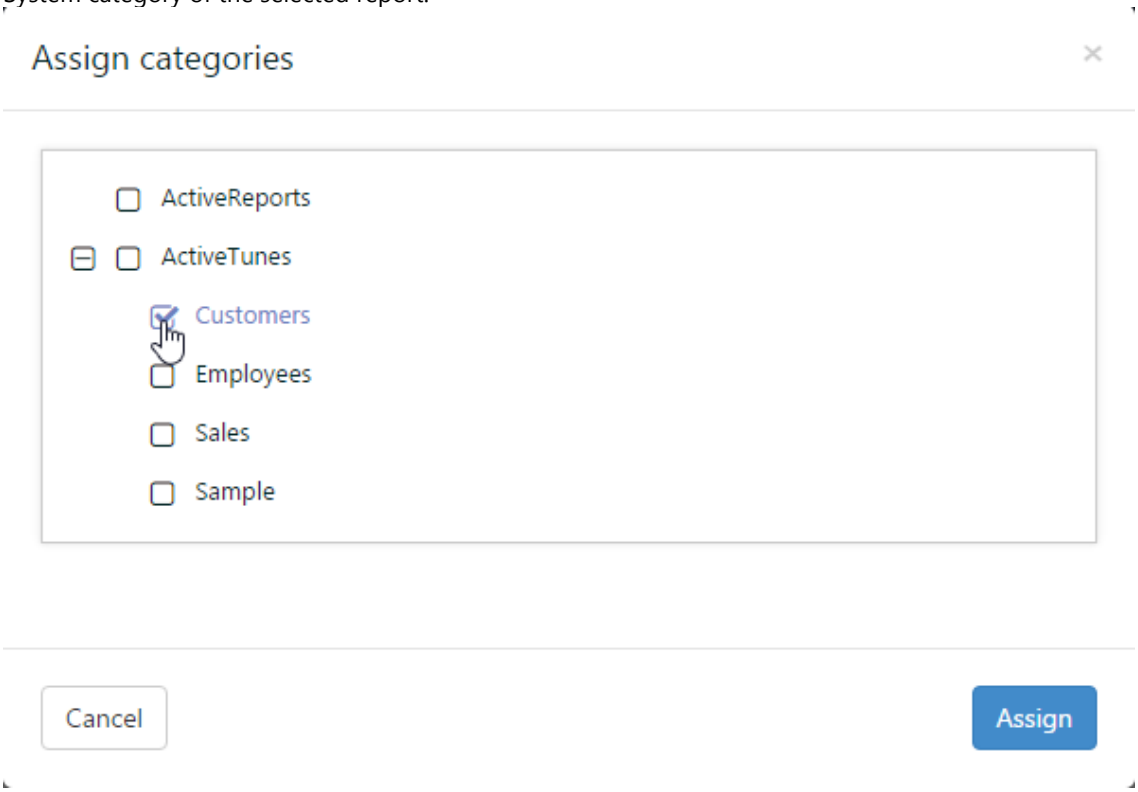
5. Click **Assign** to assign a System category to multiple reports.

Change the System Category of a Report

1. In the Administration section of the Administrator Dashboard, click **Reports**.
2. In the report list that appears, select the check-box to the left of the report to select the report.
3. Click **Categories** button to open the **Assign categories** dialog.
4. In the **Assign categories** dialog, clear the check box for the assigned System category.

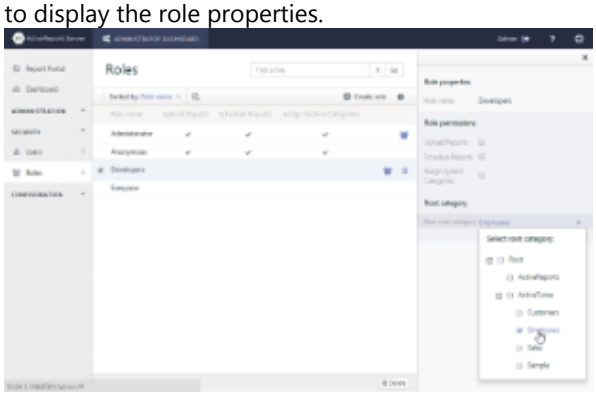


5. Select the check box for the new System category that you want to assign to your report. Click **Assign** to change the System category of the selected report.



Set a Role Root Category

1. In the Security section of the Administrator Dashboard, click **Roles.**
2. On the **Roles** page of the Administrator Dashboard, select a role from the list, and then click the **role properties** button to display the role properties.



3. In the **Root category** section, select any role root category from the drop-down list box. The role root category is set for the selected role.

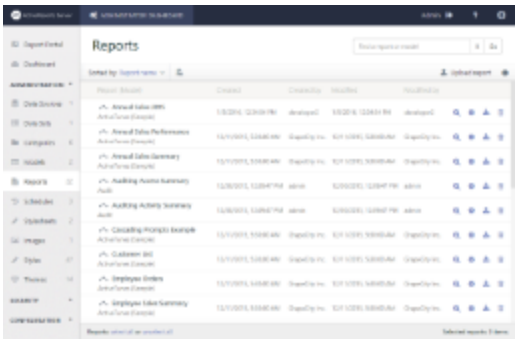
> **Note:** Setting up a root category does not impact the report permissions. Reports that are not visible under the System Category hierarchy for the role, but for which the role has the required permissions, appear under the **No category** section of the report Portal.

Hide Empty System Categories

1. In the Configuration section of the Administrator Dashboard, click **Site Settings**.
2. On the **Site Settings** page of the Administrator Dashboard, check the **Hide Empty Categories in Report Portal** option. This will hide the empty system categories from the Report Portal.

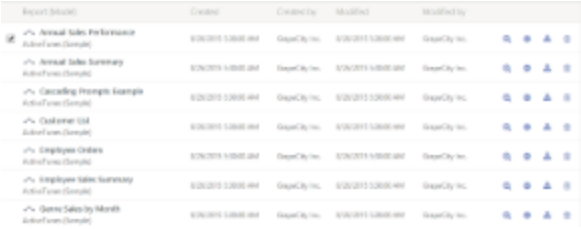
Managing Reports

Reports saved in the Report Portal are displayed on the Administrator Dashboard in the **Reports** list. Administrators can assign categories, rename, set permissions, download, preview, delete, or upload them. Reports can also be opened and saved directly from ActiveReports Designer or Visual Studio Designer to ActiveReports Server. For further information, see Server Reports.



Report Commands

You can perform actions on reports using the commands to the right of each report in the list.



Command Descriptions

Command	Description
Upload	Uploads a report from your local files. First, drop a file or click the special area to locate a valid report file in one of the supported formats: .rdl, .rdlx, .rpx, .dll or .exe. (Reports in other formats are not uploaded.) Select a report, click Open, and then click the Upload command. The report is added to the Reports list. **Notes**: An uploaded report is automatically bound to a compatible model. If there is no compatible version of the model that the report is bound to on the server, the report is not uploaded. The user who uploads the report is the Owner of the report, with full, irrevocable permission.
Click to show report properties	Allows you to show the report properties in the pane that opens to the right of the Reports list. In the **Report properties** pane, you can edit the report's name and description in the corresponding fields, and modify the report's permissions. Under the **Categories** tab, you can assign categories to the selected report.
Preview report	Runs the report and displays it in the Designer Preview window.
Design report	Displays the selected report in the Designer. **Note**: ActiveReports Server Designer Add-On license is required to design a report.
Download report	Downloads and saves the report to the local file system in .rdlx format.
Delete report	Deletes the report and removes it from the report list.
Permissions	Opens the **Change Permissions** dialog where you can select which roles have access to the model. See Managing Permissions to Reports and Models for more information.
Categories	Allows you to assign categories to the exisitng reports. Click the **Categories** button to open the **Assign categories** dialog. See Managing System Categories for more information.

Special Reports

Some types of reports require special consideration. These include drill-through reports, and reports created with the developer version of ActiveReports or Data Dynamics Reports. Find out how to handle these reports in the following topics.

- Using Code Based Section Reports from ActiveReports Developer
- Using RPX Reports from ActiveReports Developer

- Using Data Dynamics Reports
- Uploading Drill-Through Reports

Using Code Based Section Reports

If your developers use ActiveReports, you can upload your existing ActiveReports code-based files that are compiled into a .NET assembly to take advantage of the ActiveReports Server scheduling, security, and scalability features. Note that if you also use the Designer Add-On, these developer reports cannot be edited by end users.

.NET Assembly Requirements

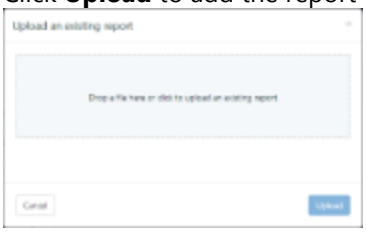
In order to upload your code-based section reports to ActiveReports Server, first compile them into a .NET assembly. Your compiled report assembly must meet all of the following requirements in order to avoid errors.

- It must be a valid .NET assembly with a strong name signature.
- It must have the target platform set to **Any CPU**.
- It must be compiled for .NET Framework 3.5.
- All its dependencies must be resolvable on the Server side.
- The version of ActiveReports 10 assemblies that you reference in the compiled report assembly cannot be newer than the version of ActiveReports 10 assemblies installed on the Server.
- It must contain at least one code-based section report.
- All its reports must have a parameterless constructor.

To upload developer ActiveReports code-based files

Developer ActiveReports appear in the list of available reports in the Report Portal as well as in the Reports list in the Administrator Dashboard. In the Report Portal, they have only Preview and Delete buttons, with the Design button omitted.

1. In the Administration section of the Administrator dashboard, click **Reports**.
2. On the **Reports** page, click the **Upload report** button.
3. In the **Upload an existing report** dialog that appears, click the special area to navigate to the .NET assembly file (dll or exe) with the code-based section report that you want to distribute via ActiveReports Server, select it and click **Upload**.
4. Select reports in the list by selecting the corresponding check boxes.
5. Click **Upload** to add the report to the Reports list in both the Administrator Dashboard and the Report Portal.



Note:

- Developer ActiveReports cannot be edited in the ActiveReports 10 Server designer. To edit such reports, please use **ActiveReports** in Visual Studio or in an End User Designer application created with the Pro Edition of ActiveReports.
- The code-based reports built with an older version of ActiveReports cannot be previewed in ActiveReports 10

> Server. Users should always rebuild all code-based reports using ActiveReports 10 and republish them on the server.

To schedule developer ActiveReports

As with any report, you can schedule code-based section reports to run at certain times, and have them automatically sent to stakeholders.

Using RPX Reports from ActiveReports

You can upload your existing ActiveReports RPX files to take advantage of the ActiveReports Server scheduling, security, and scalability features. Of necessity, there are some limits to what is supported. Here are some tips to make your RPX files compliant with ActiveReports Server.

Only self-contained RPX files are supported

- Convert any Visual Basic or C# code used in the original ActiveReport into script in the RPX file.
- Convert any references to external classes, Web services, or application code to a self-contained script inside the RPX file.
- Remove any function that requires access to the local file system, or the .NET security policies used to ensure server stability will cause errors.
- Connect reports to databases such as SQL Server or Oracle normally, using only a connection string.
- Connect to XML data sources on a remote host using the FileURL property to enable unbound data source scenarios.

To upload developer ActiveReports RPX files

Developer ActiveReports appear in the list of available reports in the Report Portal as well as in the Reports list in the Administrator Dashboard. In the Report Portal, they have only Preview and Delete buttons, with the Design button omitted.

1. In the Administration section of the Administrator dashboard, click **Reports**.
2. On the **Reports** page, click the **Upload report** button.
3. In the **Upload an existing report** dialog, you can either drop a RPX file or click inside the dotted box to browse a RPX file.
4. In the **Open** dialog that appears, navigate to the RPX file that you want to upload and click Open. The dialog closes and the file name appears in the dotted box.
5. Click **Upload** to add the report to the Reports list.

> **Note**: Developer ActiveReports cannot be edited in the ActiveReports Server designer. To edit such reports, please use **ActiveReports** in Visual Studio or in an End User Designer application created with the Pro Edition of ActiveReports.

To schedule developer ActiveReports

As with any report, you can schedule RPX reports to run at certain times, and have them automatically sent to stakeholders.

Using Data Dynamics Reports

You can upload your existing Data Dynamics Reports RDLX files to take advantage of the ActiveReports Server scheduling, security, and scalability features.

> **Caution**: In order to run RDLX reports in ActiveReports Server, the server must have access to the data source. Shared data sources are not supported for Data Dynamics Reports in ActiveReports Server.

To upload Data Dynamics Reports RDLX files

Data Dynamics Reports appear in the list of available reports in the Report Portal as well as in the Reports list in the Administrator Dashboard. In the Report Portal, they have only Preview and Delete buttons, with the Design button omitted.

1. From the Administrator Dashboard, select the **Reports** page.
2. Above the list of reports, click the **Upload report** button.
3. In the **Upload an existing report** dialog that appears, click the special area and navigate to the RDLX file that you want to distribute via ActiveReports Server, select it, and click **Open**.
4. In the **Upload an existing report** dialog, click **Upload** to add the report to the Reports list in both the Administrator Dashboard and the Report Portal.

> **Note**: Data Dynamics Reports cannot be edited in the ActiveReports Server designer. To edit such reports, please use **Data Dynamics Reports** in Visual Studio or in an End User Designer application created with Data Dynamics Reports.

To schedule Data Dynamics Reports

As with any report, you can schedule RDLX reports to run at certain times, and have them automatically sent to stakeholders.

Uploading Drill-Through Reports

You can download and upload drill-through reports like any other report created with ActiveReports Server, but to keep the drill-through functionality intact, you must keep two things in mind.

- The linked report must always be kept with the main report.
- If you change the linked report's name, you must also change it in the main report.

When you upload a report with drill-through links, if the linked reports do not exist on the server, an error message appears.

Managing Schedules

Shared schedules allow you to specify a recurring interval at which to run a report. Each schedule has a name, an optional description, and a list of roles that defines which users have permission to assign reports to it.

Once you add a shared schedule, all users in each role that is granted permission to the schedule can assign reports to run on it.

Recurrence

You can set up recurrence options on shared schedules.

Recurrence options

Option	Description
Starting On	Set the date and time for the task to start.
Repeat Task Every	Enter a number and select **Minutes**, **Hours**, **Days**, **Weeks**, or **Months** to set the frequency of the report execution.
On days	Select check boxes to specify which day or days of the week the task runs.
Result Expires After	Enter a number and select **Minutes**, **Hours**, or **Days** to specify how long to keep the report in the History. When the specified time period expires, you can no longer open the report from the **History** tab by clicking **Open**.

Note: The **Repeat Task Every** and **On days** settings work in conjunction to define the dates for a scheduled task to run.

For example, if you set the **Repeat Task Every** setting to *2 Days* and the **On days** setting to *Fri*, the scheduled task runs every second day that falls on a Friday.

Another example: if you set the **Starting On** setting to *Thursday*, the **Repeat Task Every** setting to *2 Weeks*, and the **On days** setting to *Thu*, the scheduled task runs every second Thursday. However, if a day other than Thursday is selected in the **On days** setting, the scheduled task will NOT run, as the days for the schedule to run include the starting date.

Delivery

You can optionally set delivery options for your users, or you can leave this section blank and allow your users to choose these settings.

Option	Description
Format	Select the format in which to deliver the report. **Formats**: Allow User to Choose, Excel file (xls), Mht document (archived Web page), Image file, PDF document, Word document, or XML file (xml).
Delivery Type	Select **Allow User to Choose**, **Email**, or **Windows File Share**. Below are more options that appear with these delivery types.

Delivery Type options

Option	Description
Email report to	Enter one or more e-mail addresses where the report is to be sent. Separate multiple e-mail addresses with commas.

Subject	Enter the text to use for the e-mail's subject line. You can use the following syntax in the subject: • ?*ParameterName:* To specify parameter values for the report in the subject. *R*eplace *ParameterName* with the actual name of the parameter. • &ReportName: To specify the report name. • &ExecutionTime: To specify the execution date and time of the report.
Body	Enter text to use in the body of the e-mail. You can use the following syntax in the e-mail body: • ?*ParameterName:* To specify parameter values for the report in the e-mail body. *R*eplace *ParameterName* with the actual name of the parameter. • &ReportName: To specify the report name. • &ExecutionTime: To specify the execution date and time of the report. **Caution:** When using a syntax such as, &ReportName, add a whitespace character after the syntax to replace the placeholder correctly. For example: ○ Correct: This is &ReportName template. ○ Incorrect: This is &ReportNametemplate. You can also enclose the syntax with curly braces to replace the placeholder correctly. For example: ○ &{ReportName}
Delivery method	Select **Allow User to Choose**, or **As Link** to insert a hyperlink to the report in the body of the message, or **As Attachment** to attach the report to the message in the format specified above. If you select As Link, please see Site Settings Notification URL.
Attachment template	Enter the resource name template. You can use the following syntax in the attachment template: • ?*ParameterName:* To specify parameter values for the report in the attachment template. *R*eplace *ParameterName* with the actual name of the parameter. • &ReportName: To specify the report name. This is default template. • &ExecutionTime: To specify the execution date and time of the report.

Note: ExecutionTime token is specified as **&{ExecutionTime:<TokenFormat>}**. Different **DateTime** format strings for .NET framework such as "d", "F", "T" etc. can be specified for the TokenFormat. For example, **&{ExecutionTime:d}** will be displayed as **06/10/2014**.

Caution: When **"g"** DateTime format is specified using special characters in the "&ExecutionTime" token, the generated attachment that is received through scheduled e-mail is downloaded in an unsupported format.

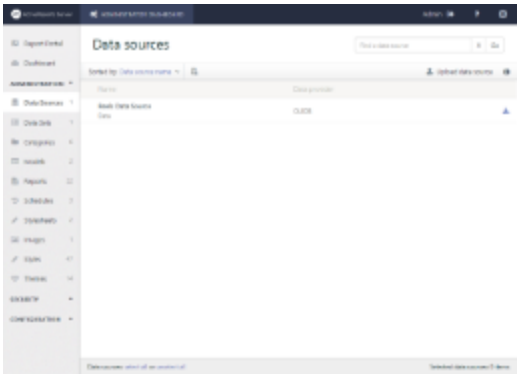
Windows File Share Delivery Type Options

Option	Description

File name	Enter the name of the report file. You can can use the following styntax in the file name: • ?*ParameterName:* To specify parameter values for the report in the file name. *R*eplace *ParameterName* with the actual name of the parameter. • &ReportName: To specify the report name. This is default template. • &ExecutionTime: To specify the execution date and time time of the report.
Add file extension	Select to add the file extension for the report's format to the file name.
Path	Enter the fully qualified UNC path of the location at which to share the report.
User name	Enter the user name to connect to the UNC path.
Password	Enter the password of the user for the UNC path.
Overwrite	Select **Allow User to Choose**, **Overwrite the existing file if it exists** , or **Increment filename as newer versions are added** to control how stored versions of the report are handled.

Managing Data Sources

ActiveReports Server provides you with the ability to upload, download or delete data sources from the Administrator dashboard. Data sources are files in RDSX format that contain data connection information. You can access data sources uploaded on ActiveReports Server to design reports in ActiveReports. See Server Shared Data Sources, for more information.



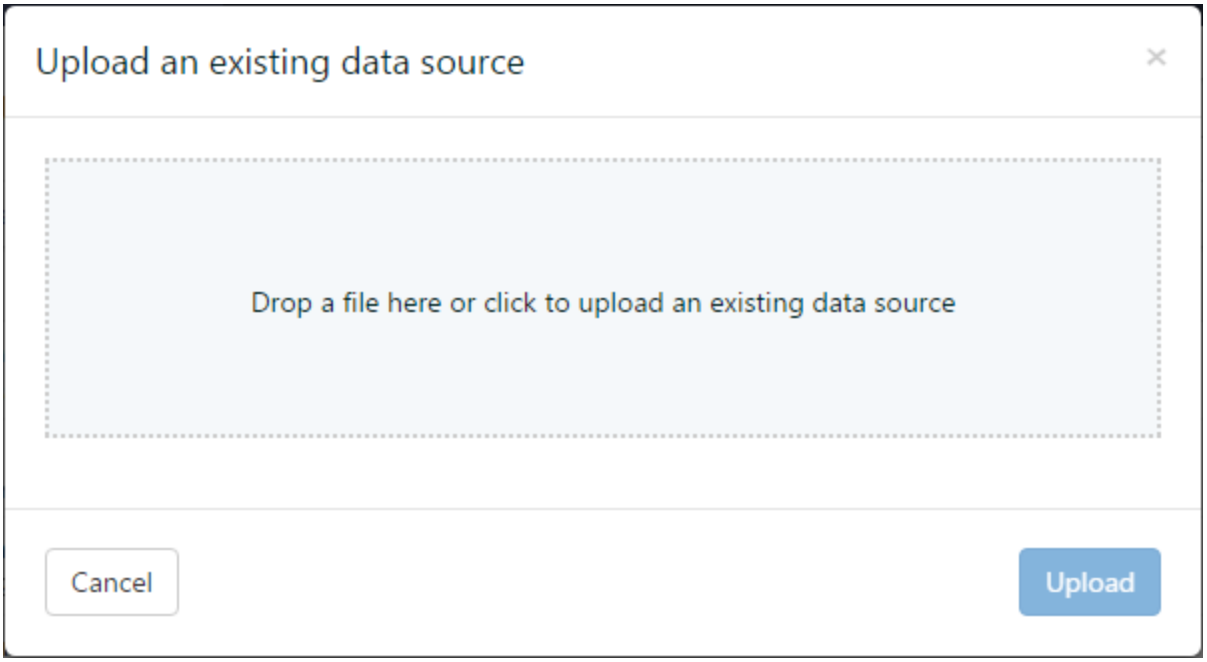
You can perform the following operations from the **Data Sources** page on the Administrator dashboard.

Download

1. In the Administration section of the Administrator Dashboard, click **Data Sources**.
2. On the **Data Sources** page, click the **Download data source** button next to the data source.
3. The data source file is downloaded on your system in .rdsx format.

Upload

1. In the Administration section of the Administrator Dashboard, click **Data Sources**.
2. On the **Data Sources** page, click the **Upload data source** button.
3. In the **Upload an existing data source** dialog that appears, you can either drop a data source file or click inside the dotted box to browse to a data source file.



4. In the **Open** dialog that appears, navigate to the ***.rdsx** file that you want to upload and click Open. The dialog closes and the file name appears in the dotted box.
5. Click **Upload** to add the data source to the server.

Delete

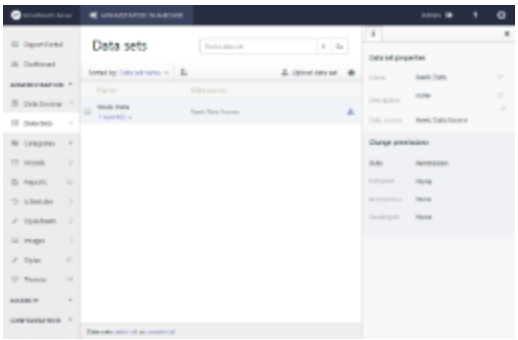
1. In the Administration section of the Administrator Dashboard, click **Data Sources**.
2. On the **Data Sources** page, click the **Delete data source** button next to the data source. The data source is removed from the list.

Rename

1. In the Administration section of the Administrator Dashboard, click **Data Sources**.
2. On the **Data Sources** page, select a data source from the list, and then click the **Data source properties** button to display the data source properties.
3. In the **Name** box, enter the name to update.

Managing Data Sets

ActiveReports Server provides you with the ability to upload, download or delete data sets from the Administrator dashboard. The server administrator sets up permissions to control who can access data sets uploaded on ActiveReports Server. You can use these server shared data sets to design reports in ActiveReports. See Server Shared Data Sets, for more information.



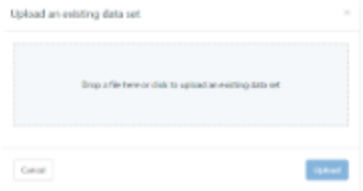
You can perform the following operations from the **Data Sets** page on the Administrator dashboard.

Download

1. On the **Data Sets** page of the Administrator Dashboard, click the **Download data set** button next to the data set.
2. The data set file is downloaded on your system in .rsdx format.

Upload

1. On the **Data Sets** page of the Administrator Dashboard, click the **Upload data set** button.
2. In the **Upload an existing data set** dialog that appears, you can either drop a data set file or click inside the dotted box to browse to a data set file.



3. In the **Open** dialog that appears, navigate to the ***.rsdx** file that you want to upload and click Open. The dialog closes and the file name appears in the dotted box.
4. Click **Upload** to add the data set to the server.

Delete

1. On the **Data Sets** page of the Administrator Dashboard, click the **Delete data set** button next to the data set. The data set is removed from the list.

Rename

1. On the **Data Sets** page of the Administrator Dashboard, select a data set from the list, and then click the **data set properties** button to display the data set properties.
2. In the **Name** box, enter the name to update.

Manage Permission

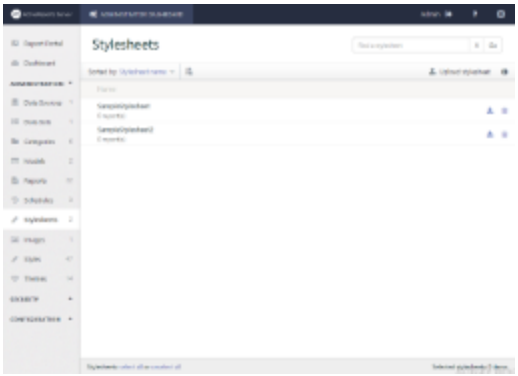
1. On the **Data Sets** page of the Administrator Dashboard, select a data set from the list, and then click the **data set properties** button to display the data set properties.

2. In the **Change permissions** section, next to each role, drop down the box to select whether to allow its users **Read** or **None** to hide the data set from users in that role.

   Note: Users should at least have **Read** permission to access the Server Shared Data Sets in ActiveReports.

Managing Style Sheets

ActiveReports Server provides you with the ability to upload, download or delete style sheets from the Administrator dashboard. Style sheets are a collection of style objects. You can provide access to style sheets uploaded on ActiveReports Server to all of your developers who design reports in ActiveReports. See Working with Styles in ActiveReports help, for more information.



You can perform the following operations from the **Stylesheets** page on the Administrator dashboard.

Download

1. In the Administration section of the Administrator Dashboard, click **Stylesheets**.
2. On the **Stylesheets** page, click the **Download stylesheet** button next to the stylesheet.
3. The style sheet file is downloaded on your system in ***.rdlx-styles** format.

Upload

1. In the Administration section of the Administrator Dashboard, click **Stylesheets**.
2. On the **Stylesheets** page, click the **Upload stylesheet** button.
3. In the **Upload an existing stylesheet** dialog that appears, you can either drop a style sheet file or click inside the dotted box to browse to a style sheet file.
4. In the **Open** dialog that appears, navigate to the ***.rdlx-styles** file that you want to upload and click Open. The dialog closes and the file name appears in the dotted box.
5. Click **Upload** to add the style sheet to the server.

Delete

1. On the **Stylesheets** page, click the **Delete stylesheet** button next to the style sheet. The style sheet is removed from the list.

Rename

1. On the **Stylesheets** page, select a style sheet from the list, and then click the **stylesheet properties** button to display the style sheet properties.
2. In the **Name** box, enter the name to update.

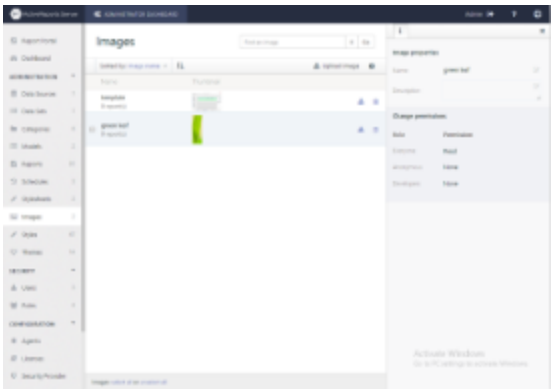
Manage Permission

1. On the **Stylesheets** page of the Administrator Dashboard, select a style sheet from the list, and then click the **stylesheet properties** button to display the style sheet properties.
2. In the **Change permissions** section, next to each role, drop down the box to select whether to allow its users **Read, Read & Modify** or **None** to hide the style sheet from users in that role.

> **Note:** Users require **Read** permission to access the shared style sheets in ActiveReports.

Managing Images

ActiveReports Server provides you with the ability to upload, download or delete images from the Administrator dashboard. You can provide access to images uploaded on ActiveReports Server to all of your developers who design reports in ActiveReports. For more information, see Working with Images in the ActiveReports help.



You can perform the following operations from the **Images** page on the Administrator dashboard.

Download

1. In the Administration section of the Administrator Dashboard, click **Images**.
2. On the **Images** page, click the **Download image** button.
3. The image file is downloaded on your system.

Upload

1. In the Administration section of the Administrator Dashboard, click **Images**.
2. On the **Images** page, click the **Upload image** button.
3. In the **Upload an existing image** dialog that appears, you can either drop an image file or click inside the dotted box to browse to an image file.
4. In the **Open** dialog that appears, navigate to the image file that you want to upload and click Open. The dialog closes and the file name appears in the dotted box.

5. Click **Upload** to add the image to the server.

Delete

1. On the **Images** page, click the **Delete image** button next to the image. The image is removed from the list.

Rename

1. On the **Images** page, select an image from the list, and then click the **image properties** button to display the image properties.
2. In the **Name** box, enter the name to update.

Manage Permissions

1. On the **Images** page of the Administrator Dashboard, select an image from the list, and then click the **image properties** button to display the properties.
2. In the **Change permissions** section, next to each role, drop down the box to select whether to allow its users **Read, Read & Modify** or **None** to hide the image from users in that role.

> **Note:** Users require **Read** permission to access the images in ActiveReports.

Managing Themes and Styles

Themes and styles allow you to control the appearance of reports. Themes determine the overall look of reports and all of their tables and charts, whereas styles are applied to individual report items.

Working with Themes
: Learn to delete themes and styles from the server, rename them, and set your default theme.

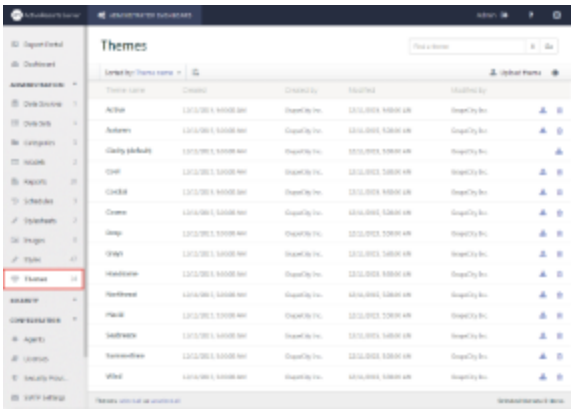
Working with Styles
: Learn to download themes and styles to your local machine and upload them to the server.

Working with Themes

The upload and download operations offer an additional means of backing up or transferring themes between servers without requiring access to the server console. Themes are specially formatted xml files with a file extension of .rdlx-theme.

> **Tip**: If you also use Data Dynamics Reports, .rdlx-theme files automatically open in that product's Theme Editor dialog, giving you a way to visually edit your theme files.

You can delete and rename themes, and change the default theme to control the look and feel of all of the reports your users create.



To download a theme

Save a working copy of a theme to your local machine

1. From the Administrator Dashboard, select **Themes**.
2. On the **Themes** page, next to the theme that you want to save, click **Download theme** button.
3. The xml theme file is copied onto your machine in .rdlx-theme format.

> **Tip**: If your browser does not automatically offer you a choice of where to save the file, you can right-click **Download** and select **Save As**.

To upload a theme

Save a file from your local machine onto the server

1. From the Administrator Dashboard, select **Themes**.
2. On the **Themes** page, click the **Upload theme** button.
3. In the **Upload an existing theme** dialog that appears, you can either drop a **.rdlx-theme** file or click inside the dotted box to browse to a data source file.
4. In the Open dialog that appears, navigate to the file with the extension **.rdlx-theme** to upload and click **Open**. The dialog closes and the file name appears in the box.
5. Click **Upload** to add the theme from your computer to the server.

To delete a theme

You can delete any themes that you do not want your users to access when they design reports.

Remove a theme from the list

1. From the Administrator Dashboard, select **Themes**.
2. On the **Themes** page, next to the theme that you want to delete, click the **Delete theme** button. The theme is removed from the list.

To rename a theme

You can name themes after departments to let them know which one to use, or use any name that suits your business needs.

Make a theme name relevant for your users

1. From the Administrator Dashboard, select **Themes**.
2. On the **Themes** page, select a theme from the list, and then click the **theme properties** button to display the properties.
3. In the **Name** box, enter the name to update.

To change your default theme

You can select which theme is used by default when your users create new reports. Users can still select any other theme that you have in the themes list.

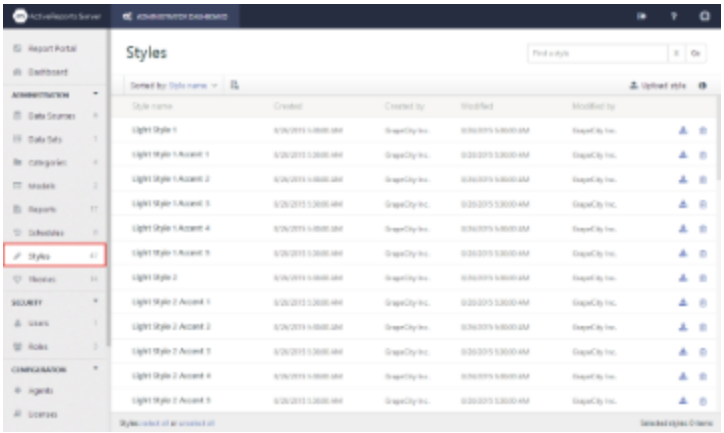
▸ Designate the most commonly used theme for your company

1. From the Administrator Dashboard, select **Themes**.
2. On the **Themes** page, select a theme from the list, and then click the **theme properties** button to display the properties.
3. Select the **Set as default** checkbox to set the theme as your default. The Set as default command is removed from the selected theme, and added back to the theme that was previously the default theme.

Working with Styles

The upload and download operations offer an additional means of backing up or transferring styles between servers without requiring access to the server console. Styles are specially formatted xml files with a file extension of .style.

You can delete and rename styles to control the look of text in the reports your users create, as well as to suggest where they are used for a more consistent company-wide look.



To download a style

▸ Save a working copy of a style to your local machine

1. From the Administrator Dashboard, select **Styles**.
2. On the **Styles** page, next to the stylee that you want to save, click **Download style** button.
3. The xml style file is copied onto your machine in .style format.

> **Tip**: If your browser does not automatically offer you a choice of where to save the file, you can right-click **Download** and select **Save As**.

To upload a style

Save a file from your local machine onto the server

1. From the Administrator Dashboard, select **Styles**.
2. On the **Styles** page, click the **Upload style** button.
3. In the **Upload an existing report style** dialog that appears, you can either drop a **.style** file or click inside the dotted box to browse to a data source file.
4. In the Open dialog that appears, navigate to the file with the extension **.style** to upload and click **Open**. The dialog closes and the file name appears in the box.
5. Click **Upload** to add the style from your computer to the server.

To delete a style

You can delete any styles that you do not want your users to access when they design reports.

Remove a style from the list

1. From the Administrator Dashboard, select **Styles**.
2. On the **Styles** page, next to the style that you want to delete, click the **Delete style** button. The style is removed from the list.

To rename a style

You can name styles to reflect where and how they are used in your company to keep your users consistent.

Make a style name relevant for your users

1. From the Administrator Dashboard, select **Styles**.
2. On the **Styles** page, select a style from the list, and then click the **style properties** button to display the properties.
3. In the **Name** box, enter the name to update.

Managing Security

This section contains information that helps administrators manage users, roles, and permissions.

Managing Users
: Learn to add and delete users, and to modify user credentials, password, and roles.

Managing Roles
: Learn to add and delete roles, and to modify permissions for roles.

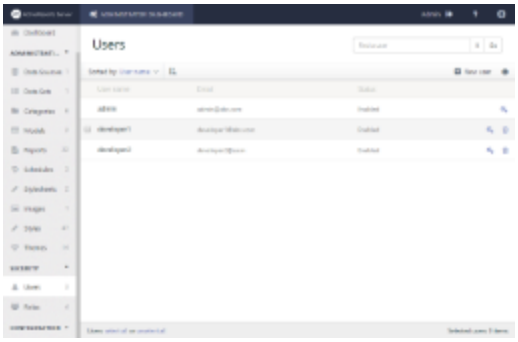
Managing Permissions
: Learn what permissions you can set on reports and models.

Using LDAP and Active Directory for Single Sign-On
: Learn about how you can integrate your LDAP and Active Directory users into ActiveReports Server security.

> **Note**: You can also create row-level security by creating custom security providers. the Managing Single Sign-On Security Providers topic for more information.

Managing Users

An administrator can add and delete users, and modify existing users' credentials, passwords, and roles. All of these functions are in the Users list, which you can access from the Administrator Dashboard by clicking Users in the Security section.



To add a new user

Each user needs an e-mail address, password, and user name.

Create a new user

1. In the Security section of the Administrator Dashboard, click **Users**. The Users list appears.
2. Click the **New user** button. The Create new user dialog appears.
3. In the dialog, fill in the User Name, Password, Confirm password, E-mail, and Description fields.

4. Click **Next** and then select the check box under any roles to which to assign the user.
5. Click **Finish**. The new user is added to the Users list and any selected roles, and the user can log in to the Report Portal with the user name and password specified.

> **Note**: By default each new user is assigned to the **Everyone** role. For more information, see Managing Roles.

To delete a user

You can remove users from ActiveReports Server so that they no longer have access.

Delete a user

1. In the Security section of the Administrator Dashboard, click **Users**. The Users list appears.
2. Select the user that you want to remove, and then click **Delete user**. The user is removed from the list.

> **Note:** The currently signed-in user Admin cannot be deleted, so there is no Delete command for this user.

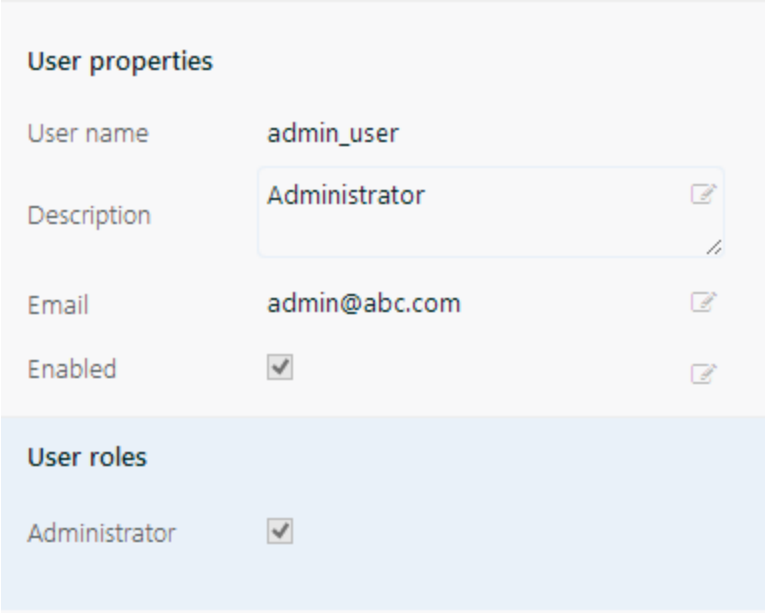
To modify the credentials of an existing user

In addition to changing a user's e-mail address or description, you can lock or unlock user accounts.

▸ Change user credentials

1. In the Security section of the Administrator Dashboard, click **Users**. The Users list appears.
2. On the **Users** page, select a user from the list, and then click the **user properties** button to display the properties.
3. Modify the information in the activated fields of the **Users** table to update.

   The **Description** allows you to enter a description of the user.
   The **E-mail** address of the user.
   The **Enabled** check box is cleared automatically if the user has made several unsuccessful log-in attempts.

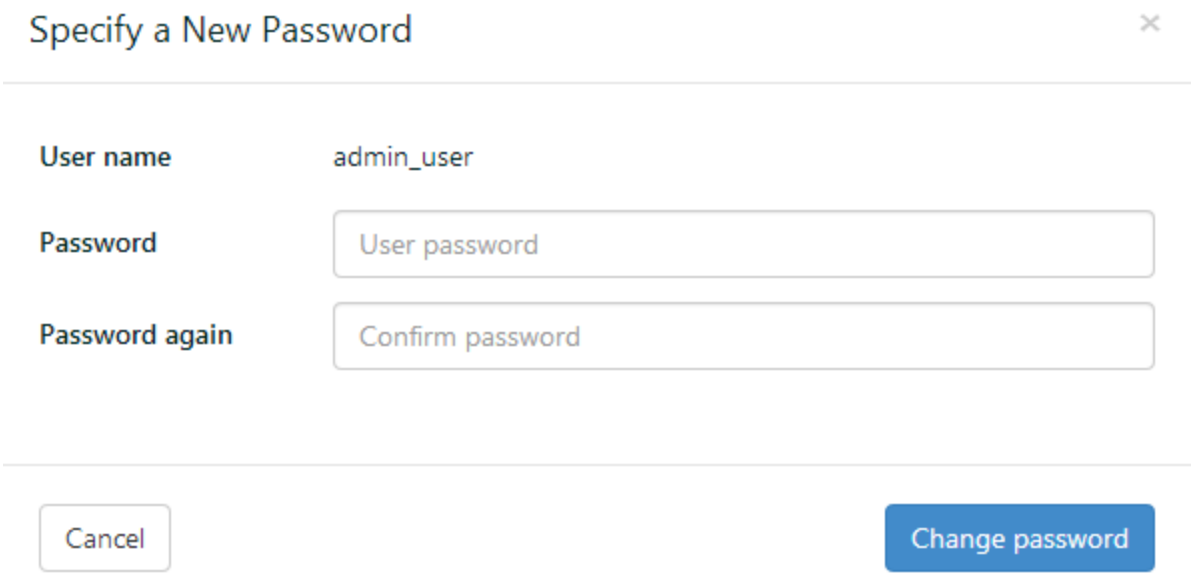


To change a user's password

You can change a user's password when they forget it, or if their password security is in doubt.

▸ Reset a password

1. In the Security section of the Administrator Dashboard, click **Users**. The Users list appears.
2. Select the user whose password you want to change, and then click **Reset password**. The Specify a New Password dialog appears.
3. In the **Specify a New Password** dialog, enter the new password and then once again to confirm it.



4. Click the **Change password** button to confirm the password change.

To manage a user's roles

In addition to editing the user's credentials, you can add or remove the user from roles.

▸ Edit roles

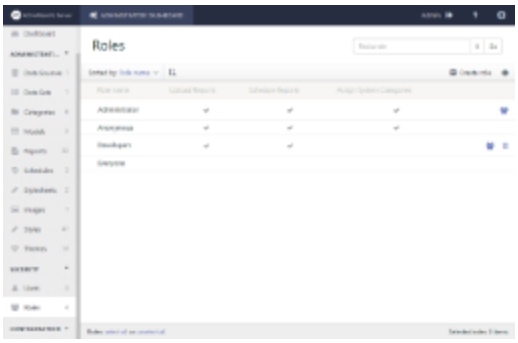
1. In the Security section of the Administrator Dashboard, click **Users**. The Users list appears.
2. On the **Users** page, select a user from the list, and then click the **user properties** button to display the properties.
3. In the **User roles** section, select or clear the roles to assign to the user.

> **Note:** ActiveReports Server has role-based security, which means that permissions are assigned to roles, not users.

Managing Roles

All permissions are granted to a Role. To grant permission to users, add the user to one or more roles. By default, four roles are configured. These roles cannot be deleted, but you can add new roles to the list.

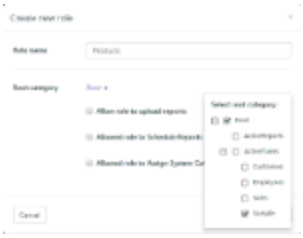
- **Administrators** contains users with the permission to enter the Administration area.
- **Anonymous** contains no users, but it allows users to work with the Report Portal without credentials.
- **Everyone** By default each new user is assigned to the **Everyone** role. The Administrator can also enable or disable the **Everyone** role from the Site Settings page of the Administrator dashboard. Once you disable the **Everyone** role, a user who created a report while belonging only to that role loses the ability to assign or revoke permissions of that report even if their new role has Full permissions for the report.
- **Owner** contains the user who originally created or uploaded the report. By default, this is the only role with full permissions on a newly created or uploaded report.



To create a new role

You can add as many roles as you need to the list. Then if you decide to grant permission for a report or model to an entire department, you can make the change once instead of for each individual.

1. In the Security section of the Administrator Dashboard, click **Roles**.The Roles list appears.
2. From the **Roles** page, click **Create role** button. The Create new role dialog appears.



3. In the **Role Name** box, enter a name for the new role.
4. In the **Create new role** dialog that appears, select the Root category drop-down list box to assign a role root category.
5. If you want to allow the role to upload reports, select the **Allow role to upload reports** check box.
6. If you want to allow the role to schedule reports, select the **Allow role to schedule reports** check box.
7. If you want to allow the role to assign system categories, select the **Allow role to Assign System Categories** check box.
8. Click **Create role**. The new role is added to the list, with an extra command that allows you to delete it. To begin using the role, add users to it. To add more permissions, see Managing Permissions.

To add a user to a role

When you create a new role, it has no users.

1. In the Security section of the Administrator Dashboard, click **Roles**.
2. In the Roles list that appears, to the right of the role to which you want to add a user, click the **Manage users** command. The Manage users dialog appears.
3. In the **Manage users** dialog, select check boxes next to users to assign them to the role.

> **Note:** To remove a user from the role, clear the corresponding check box. You cannot add or remove users from the Owner role, which has a user for each report based on the Created By value for that report.

4. Click the **Update** button to save the changes.

To grant permission to upload reports

By default, only the Administrator role has permission to upload reports.

1. In the Security section of the Administrator Dashboard, click **Roles**.
2. On the **Roles** page, select a role from the list, and then click the **role properties** button to display the Role properties.
3. In the **Role permission** section, select the **Upload Reports** checkbox. All users assigned to the role are able to upload reports.

To grant permission to schedule reports

By default, only the Administrator role has permission to schedule reports.

1. In the Security section of the Administrator Dashboard, click **Roles**.
2. On the **Roles** page, select a role from the list, and then click the **role properties** button to display the Role properties.
3. In the **Role permission** section, select the **Schedule Reports** checkbox. All users assigned to the role are able to schedule reports.

To grant permission to assign system categories

By default, only the Administrator role has permission to assign system categories.

1. In the Security section of the Administrator Dashboard, click **Roles**.
2. On the **Roles** page, select a role from the list, and then click the **role properties** button to display the Role properties.
3. In the **Role permission** section, select the **Assign System Categories** checkbox. All users assigned to the role are able to assign system categories.

To delete a role

1. In the Security section of the Administrator Dashboard, click **Roles**.
2. In the Roles list that appears, to the right of the role that you want to delete, click the **Delete role** button.
3. If the role does not contain any users, it is removed from the list immediately. If the role does contain users, you are asked to confirm.
4. Click **OK** to delete the role.

You can grant other permissions to reports and models by selecting roles to allow for each individual report and each model. See Managing Permissions for more information.

Managing Permissions

Administrators can modify permissions to data models, reports, and shared schedules by assigning rights to a role. Once a role has permission to use a model, report, or shared schedule, all users added to that role have the same permission.

▸ Table defining what permissions are granted for each combination of settings for reports and models

Permission Setting	Model: Read	Model: Create Report	Model: None

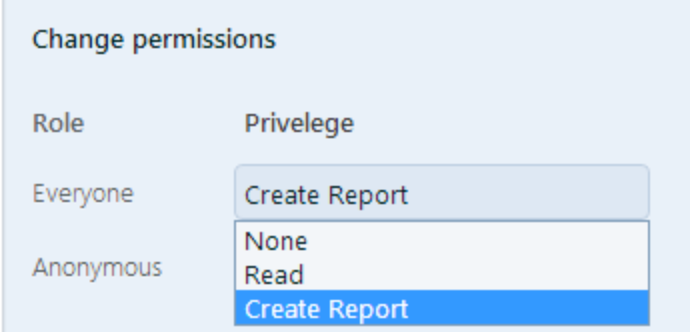
Report: None	No operations	No operations	No operations
Report: Read	Create a copy of report and modify the copy	Create a copy of report and modify the copy	Preview only
Report: Read & Modify	Preview, delete, and modify report	Preview, delete, and modify report	Preview and delete report
Report: Full	Preview, delete, and modify report Change access	Preview, delete, and modify report Change access	Preview and delete report Change access

To manage permissions to a model

You can grant read or read and create permission to different roles for each model.

Allow or deny read permission to a model

1. From the Administrator Dashboard, select **Models**.
2. On the **Models** page, select a model from the list, and then click the **model properties** button to display the properties.
3. In the **Change Permissions** section, next to each role, drop down the box to select whether to grant **Read** permission, **Create Report**, or **None** to hide the model from users in that role.



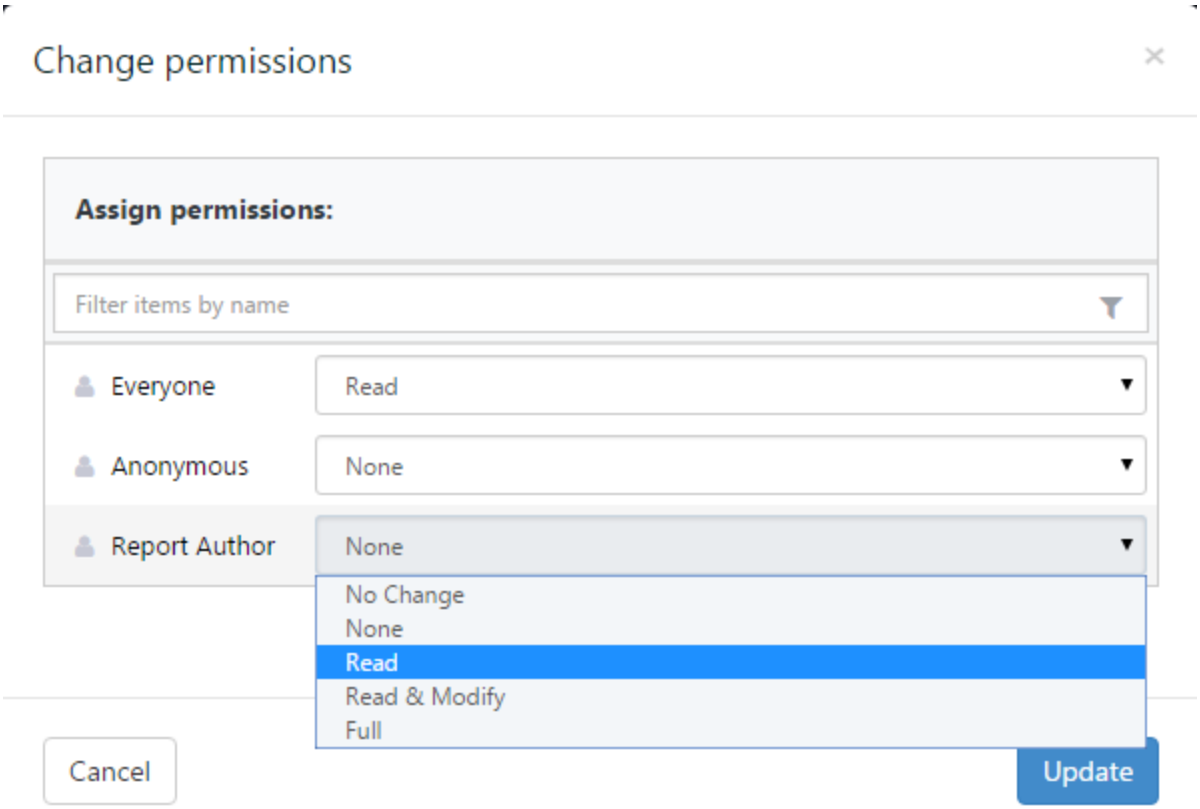
4. The selected roles have permission to read or read and create reports on the model.

To manage permissions to a report

You can grant different roles permission to read, modify, or change permissions on each report. The Owner of the report (the user who created or uploaded it) has full permission which cannot be modified.

Allow or deny read, modify, or change permissions

1. From the Administrator Dashboard, select **Reports**.
2. On the **Reports** page, select a report from the list, and then click **Permissions** button.
3. In the **Change permissions** dialog that appears, next to each role, drop down the box to select whether to allow its users **Read**, **Read & Modify**, or **Full** permissions on the report, or **None** to hide the report from users in that role.



- **Read** allows users in the role to see the report in the Report Portal **Open** page and preview the rendered report, but does not allow them to modify it in the Report Portal.
- **Read & Modify** allows users in the role to modify the report in the Report Portal in addition to Read permissions.
- **Full** allows users in the role to modify permissions on the report in addition to Read & Modify permissions.

4. Click the **Update** button to save the changes. All users in the selected roles are granted the specified access.

To change ownership of a report

By default, the user who created or uploaded the report has full permission to it. The Owner role always has full permission to the report, so the only way to change that is to change ownership by deleting the report and having a new owner upload it to the server.

Change ownership

1. From the Administrator Dashboard, select **Reports**.
2. On the **Reports** page, next to the report for which you want to change ownership, click **Download report** button. The RDLX file of the report is copied to your local machine.
3. Next to the same report, click **Delete**. In the message box that appears, click **Yes** to confirm that you want to delete the report. The report is removed from the server.
4. Log in with a new user.
5. On the Reports page, click the **Upload report** button.
6. In the **Upload an existing report** dialog, you can either drop a report file or click inside the dotted box to browse to a report file.
7. In the Open dialog that appears, navigate to the report and click **Open**. The dialog closes and the file name appears in the dotted box.

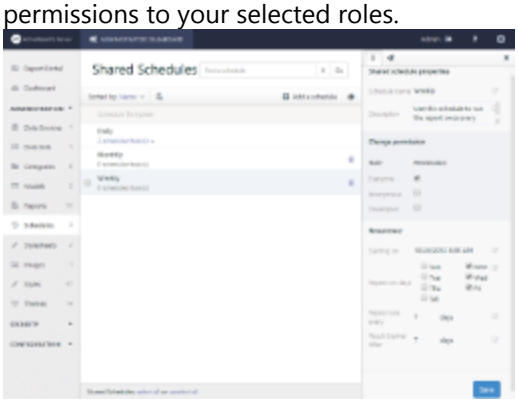
8. Click **Upload**. The report appears in the list with the new owner listed under the **Created By** column.

To manage permissions to a shared schedule

You can grant different roles permission to use shared schedules to schedule reports to run. You can also set options in the Delivery section to allow or prevent users from changing the settings. For more information, see Managing Schedules.

▸ Allow or deny read, modify, or change permissions

1. From the Administrator Dashboard, select **Schedules**.
2. On the **Schedules** page, select the schedule on which you want to set permissions, and then click the **schedule properties** button to display the properties.
3. In the **Change permission** section, next to each role, select the checkbox for each role whose users you want to allow to schedule reports. By default, **Everyone** is allowed to schedule reports, so clear that checkbox to limit permissions to your selected roles.



Using LDAP and Active Directory for Single Sign-On

You can use your existing LDAP or Active Directory® with ActiveReports Server so that your users can log in with their existing user names and passwords, and so that you can use existing groups to provide access to specific data.

> **Note**: ActiveReports Server supports OpenLDAP v2.3 or later. Earlier versions are not supported because they do not have a MemberOf attribute.

To map your user directory to ActiveReports Server, you specify LDAP (Active Directory) as your custom security provider, and set properties to give ActiveReports Server access to it. These properties are described in the table below.

▸ Security provider properties table

Property	Description
LDAP Server URL (required)	The URL to the directory where you can find users. The syntax is: `ldap://host:port/domain` • **ldap://** is an optional prefix • **host** is the LDAP server name or IP address • **port** is the optional LDAP server port number (default is 389) • **domain** is the optional distinguished name of the domain in which to search for users **Example:** `ldap://example.org:888/DC=grapecity,DC=net` In this example, *example.org* is the name of the server answering LDAP queries, *888* is a non-standard port

	used on the LDAP server, and *grapecity.net* is the ActiveDirectory domain.
LDAP Admin User (required)	The name or DN (distinguished name), or UPN (user principal name) of the user under which the LDAP connection is established. This user is also used to implement the ISecurityProvider.GetAdminContext method when you add a custom entity. **Note**: Most non-Active Directory LDAP servers require a DN (distinguished name) for this, so a DN in the administrator account would be the directory manager. **Example:** display name: Directory Manager distinguishedName (DN): `cn=Directory Manager,cn=Root DNs,cn=config` userPrincipalName (UPN): DirectoryManager@grapecity.net **Note:** For user authentication, we recommend to use UPN of the user as it is based on **samaccountname** that is unique in the domain.
LDAP Admin Password (required)	The password associated with the administrative user on the LDAP server. **Example:** `*****`
User Name Attribute (optional)	By default, this value is **samaccountname** and specifies the LDAP attribute to use in resolving the user name passed in the ISecurityProvider.GetUserToken method. **Example:** `uid`
User Display Name Attribute (optional)	By default, this value is **displayName** and specifies the LDAP attribute to use in resolving the user-friendly name with the ISecurityProvider.GetUserDescription method. **Example:** `sn`
User Email Attribute (optional)	By default, this value is **mail** and specifies the LDAP attribute to use in resolving the user e-mail with the ISecurityProvider.GetUserDescription method. **Example:** `mail`
Define Mappings	Description
User Context Attribute Name	LDAP User Attribute Name

To configure ActiveReports 10 Server to use your LDAP or Active Directory

1. In the Configuration section of the Administrator Dashboard, click **Security Provider**. The Custom security provider drop-down list appears.
2. From the drop down list box, select **Active Directory**.
3. In the **Security Provider Settings** that appear, enter a value for each of the properties.

4. Below the Security Provider Settings, in the **Define mappings** section, click on **Add row** button..
5. In the boxes that appear, enter a valid **User Context Attribute Name**, and a **LDAP User Attribute Name** that you will be able to select when you create security filters in the model editor. (See the **To create a new security filter** section below.)

To test your custom security provider

You can place percent signs around a UserContext attribute to use it in a connection string when you create a model. This is useful when each tenant in a multi-tenant application has a separate database, and you need to supply a dynamic value for the database.

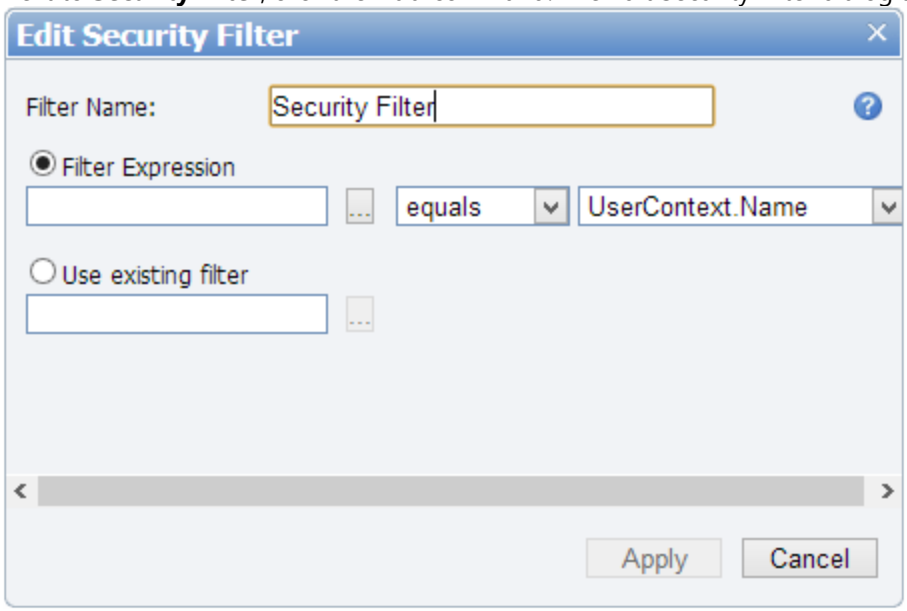
```
Provider=SQLOLEDB.1;Data Source=HQ;Initial Catalog=%TenantDatabase%;
User Id=myUsername;Password=myPassword;
```

When you create a security filter, you can re-use it for other entities that contain the attribute on which the filter expression is based.

To create a new security filter for row-level security

From the Administrator Dashboard, in the Administration section on the left, click **Models**. The Models list appears.

1. Select the model to which you want to add a filter, and then click **Edit model** button. The model editor appears.
2. Select the entity that you want to filter. The editable properties appear in the center workspace.
3. Next to **Security Filter**, click the **Add** command. The Edit Security Filter dialog appears.



4. In the box next to **Filter Name**, enter a name for your filter.
5. With **Filter Expression** selected, click the ellipsis button to open the Select Attribute dialog where you can select an attribute of the entity on which to filter data, and click **OK**.
6. Drop down the list next to the attribute to select whether to display data that **equals** or does **not equal** the following value.
7. Drop down the list to the right, and select a User Context Attribute Name mapping against which to filter the selected attribute.
8. Click the **Apply** button to create the filter, or to discard the changes, click **Cancel**.

Managing Configuration

This section contains information that helps administrators manage SMTP settings and diagnose problems.

Managing Agents
: This section describes how ActiveReports Server uses agents to handle scale-out deployment so that you can serve up more reports to more users as needed.

Managing Licenses
: Learn about different licenses for ActiveReports Server and how to manage them.

Managing Single Sign-On Security Providers
: Learn about using custom security providers as well as LDAP or Active Directory® security providers.

Configuring Email
: Learn to configure SMTP settings for your site to enable the error reporting and password reminder functions.

Site Settings
: Learn how to set a base URL for links sent in notifications for lost passwords and scheduled reports.

Managing Audit Settings
: This section describes how to set up auditing on your site, and reports and data are automatically set up.

Server Diagnostics
: Learn about monitoring your server with the ActiveReports Server Diagnostics page.

Managing Agents

You can use agents to manage your report execution load and concurrent users in ActiveReports Server. The agents list displays information about each agent to help you manage them. For more information, see Agents and Scalability.

Agent Information and Command

You can obtain information on agents and activate or deactivate them using the command to the right of each agent in the list.

Column Descriptions

Column	Description

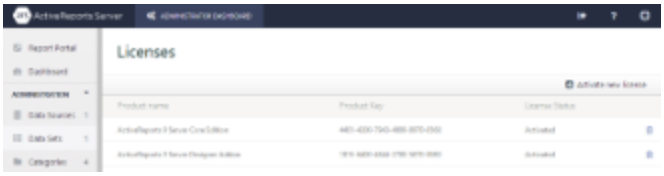
Agent ID	Displays a unique internal identification code for the agent.
Host	Displays the name of the hosting machine.
Version	Displays the ActiveReports Server version number.
License Status	Displays licenses and serial numbers for licensed agents, or lets you know when no license is available. If the agent is disabled or cannot connect, the license goes back into circulation, and the license status is N/A.
Connection Status	Displays the state of the connection: **Connected** or **Cannot Connect**.
Deactivate (or Activate)	Takes the agent out of action, but allows you to enable it later. It remains connected, but no tasks are assigned to it, and its license is freed up for use by another agent. You can deactivate an agent and re-activate it to force it to check for an available license. When a new agent is installed, it is deactivated until an Administrator activates it.

Managing Licenses

When you purchase an initial license, you receive a product key to unlock them. You can manage these licenses on the Administrator Dashboard Licenses page. For more information, see Agents and Scalability.

License Information and Commands

You can obtain information on licenses and remove them using the command to the right of each license in the list. You can add a new license using the command below the list.



Column Descriptions

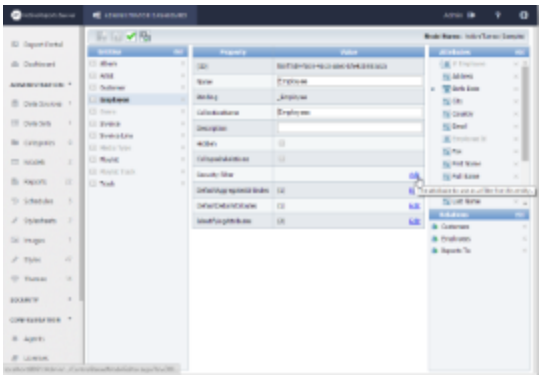
Column	Description
Product name	Displays the type of license you purchased, which may be a Trial, Core, and Designer Add-On Edition.

Product Key	Displays the product key that you received when purchasing the license.
License Status	Displays the status of the license, which may be Activated, or may show the number of days left on a trial license.
Delete license	Click **Delete license** button to remove a license from the list. This allows you to clean up any expired evaluation licenses.
Activate New License	Click this button to add a new license. It opens a dialog where you can enter your product key and activate the license online. See Activate a License for more information.

Managing Single Sign-On Security Providers

You can provide row-level security with security filters. While editing a model, you can add and delete security filters from entities, and modify existing ones, as well as configure and use an LDAP security provider. All of these functions are in the Model Editor, which you can access from the Administrator Dashboard.

> **Note**: In order to use single sign-on security filters, you must first create a custom security provider. For information on creating custom security providers, please refer Create a Custom Security Provider.



To use an Active Directory® or LDAP security provider

When using a Lightweight Directory Access Protocol (LDAP) security provider, you can create User Context attributes that map to LDAP user attributes, and use them in security filter expressions to enable single sign-on for your users.

▸ Use an LDAP security provider

1. In the Configuration section of the Administrator Dashboard, click **Security Provider**. The Custom security provider drop-down list appears.
2. From the drop down list box, select **Active Directory**.
3. In the **Security Provider Settings** that appear, enter a value for each of the properties.
4. Below the Security Provider Settings, in the **Define mappings** section, click on **Add row** button..
5. In the boxes that appear, enter a valid **User Context Attribute Name**, and a **LDAP User Attribute Name** that you will be able to select when you create security filters in the model editor. (See the **To create a new security filter** section below.)

To add your new security provider to the server

Once you have compiled your custom security provider into a DLL, you can begin using it to filter entities in your models.

Add a new security provider

1. In Windows Explorer, create a SecurityProviders folder on the same level as your site, for example, if you have C:\ActiveReports 10 Server, create the folder C:\ActiveReports 10 Server\SecurityProviders.
2. Copy your custom security provider DLL file into the new folder.
3. In the Configuration section of the Administrator Dashboard, click **Security Provider**. The Custom security provider drop-down list appears.
4. Drop down the list and select your new security provider.
5. Depending on your security provider, there may be settings that you can select.
6. When you have the settings the way that you want them, click the **Apply changes** button. The security filter becomes available to your model entities.

To delete a security provider

You can stop the server site and remove security providers from ActiveReports Server.

Delete a security provider

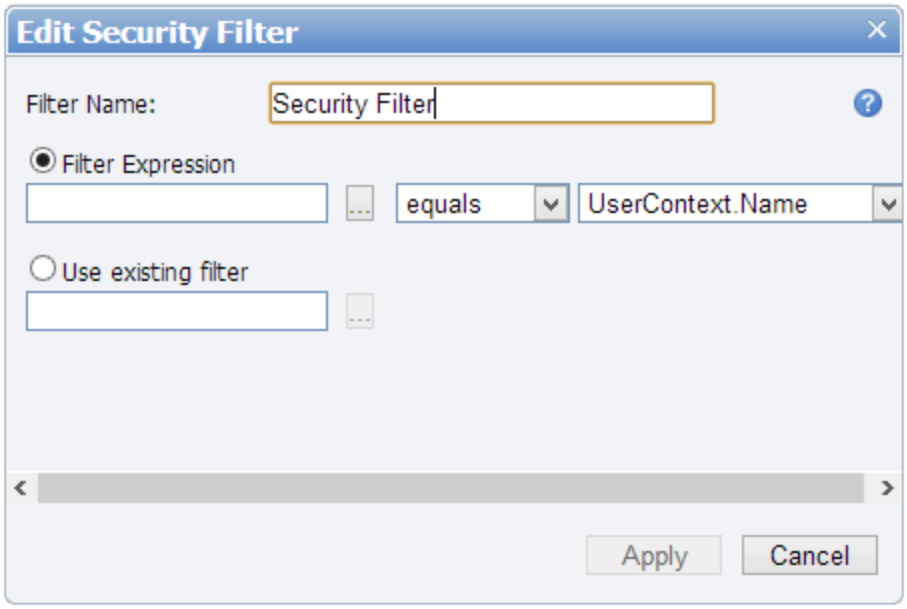
1. Open Internet Information Services (IIS) Manager, and expand the **Sites** node.
2. Select the **ActiveReports 10 Server** site, and on the right, in the Manage Web Site section, click **Stop**.
3. In Windows Explorer, locate your SecurityProviders folder.
4. Delete the DLL containing the security provider that you long longer wish to use. The provider is removed from the list.
5. Back in IIS, with the ActiveReports 10 Server site selected, in the Manage Web Site section, click **Start**.

To create a new security filter

When you create a security filter, you can re-use it for other entities that contain the attribute on which the filter expression is based, providing row-level security for your data.

Create a new filter

1. Select the model to which you want to add a filter, and then click **Edit model** button. The model editor appears.
2. Select the entity that you want to filter. The editable properties appear in the center workspace.
3. Next to **Security Filter**, click the **Add** command. The Edit Security Filter dialog appears.



4. In the box next to **Filter Name**, enter a name for your filter.
5. With **Filter Expression** selected, click the ellipsis button to open the Select Attribute dialog where you can select an attribute of the entity on which to filter data, and click **OK**.
6. Drop down the list next to the attribute to select whether to display data that **equals** or does **not equal** the following value.
7. Drop down the list to the right, and select a User Context Attribute Name mapping against which to filter the selected attribute.
8. Click the **Apply** button to create the filter, or to discard the changes, click **Cancel**.

To use an existing filter

You can re-use security filters with other entities that use the same attributes.

Use an existing filter

1. In the Administration section of the Administrator Dashboard, click **Models**. The Models list appears.
2. Select the model to which you want to add a filter, and then click **Edit model** button. The model editor appears.
3. Select the entity that you want to filter. The editable properties appear in the center workspace.
4. Next to **Security Filter**, click the **Add** command. The Edit Security Filter dialog appears.
5. In the box next to **Filter Name**, enter a name for your filter.
6. With **Use existing filter** selected, click the ellipsis button to open the Select Existing Filter dialog where you can select a previously created filter (if it is based on an attribute shared by this entity), and click **OK**.
7. Click the **Apply** button.

Configuring Email

To ensure that error reporting and password reminders function properly for your site, you can configure the SMTP settings. The report server also uses these settings to connect with the e-mail server to deliver scheduled reports. You can configure e-mail SMTP settings from the Administrator Dashboard.

To configure e-mail from the Administrator Dashboard

1. From the Administrator Dashboard, at the bottom left under Configuration, click **SMTP Settings**.
2. In the SMTP Settings dialog that appears, enter values for your SMTP server.

   - **SMTP host** Enter the host address for your server, usually with syntax like "smtp.yourdomainname.com."
   - **SMTP port** Enter the port number for your server, for example, 587.
   - **Use SMTP authentication** Select this check box to add a user name and password.
   - **Sender e-mail** Enter the e-mail address to use as the return address for error reporting, password reminders, and scheduled report delivery.
   - **Sender name** Enter the name to use for the sender.
   - **Use SSL** Select this check box to enable the Secure Sockets Layer protocol.
3. When you have finished entering the values for your server, click **Apply changes**.

Troubleshooting

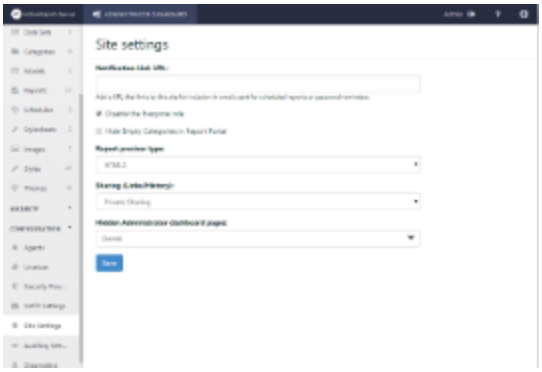
The table below explains the possible reasons for e-mail failure messages that you may get in the diagnostic report.

Cause	Error description
Incorrect host	System.Net.Mail.SmtpException: Failure sending mail. ----> System.Net.WebException: Unable to connect to the remote server ----> System.Net.Sockets.SocketException: A connection attempt failed because the connected party did not properly respond after a period of time, or established connection failed because connected host has failed to respond.
Incorrect port	System.Net.Mail.SmtpException: Failure sending mail. ----> System.Net.WebException: Unable to connect to the remote server ----> System.Net.Sockets.SocketException: A connection attempt failed because the connected party did not properly respond after a period of time, or established connection failed because connected host has failed to respond .
Incorrect login	System.Net.Mail.SmtpException: The SMTP server requires a secure connection or the client was not authenticated. The server response was: 5.7.1 Authentication required.
Incorrect password	System.Net.Mail.SmtpException: The SMTP server requires a secure connection or the client was not authenticated. The server response was: 5.7.1 Authentication required.
Incorrect sender e-mail	System.Net.Mail.SmtpFailedRecipientException: Mailbox name not allowed. The server response was: 5.7.1 <INCORRECT_EMAIL>: Sender address rejected: not owned by user CORRECT_EMAIL.
SSL is enabled but not required	System.Net.Mail.SmtpException: Error in processing. The server response was: 4.3.0 Error: queue file write error. OR

	System.Net.Mail.SmtpException: Failure sending mail. ----> System.IO.IOException: Unable to read data from the transport connection: net_io_connectionclosed.
SSL is disabled but required	System.Net.Mail.SmtpException: Failure sending mail. ----> System.IO.IOException: Unable to read data from the transport connection: net_io_connectionclosed.
SMTP authentication is enabled but not required	System.Net.Mail.SmtpException: The SMTP server requires a secure connection or the client was not authenticated. The server response was: 5.7.1 Authentication required.
SMTP authentication is disabled but required	System.Net.Mail.SmtpException: The SMTP server requires a secure connection or the client was not authenticated. The server response was: 5.7.1 Authentication required.

Site Settings

ActiveReports Server lets you manage a number of site settings. The following settings are available to the administrator.



Notification Link URL

The Notification Link URL field allows you to enter the URL address of the server that hosts ActiveReports Server. The URL specified in the **Notification Link URL** field is included in the notification link that goes out to the users via email. When you schedule reports and choose **Email** as the delivery type, and **As Link** as your delivery method, the email that goes out to your users contains a link to the report location.

For example, if you enter 'http://localhost:3000'; in the **Notification Link URL** field, the following link is in the email notification.

http://localhost:3000/?view=changePassword&key=VAtHJJrldNySQdknuC15KcfQL6M

Disable the Everyone role

Check this option to disable the **Everyone** role that contains all users by default.

Hide Empty Categories in the Report Portal

Check this option to hide any system categories that do not have any reports.

Report preview type

ActiveReports Server provides you with an option to view your reports in the **HTML5 Viewer** or **Flash Viewer**. Select **HTML5** or **Flash** from the **Report preview type** dropdown and click the **Save** button to apply the selected changes.

Sharing (Links/History)

ActiveReports Server allows you to provide different settings for scheduling links and history. Scheduling links and history are visible to the user depending upon the settings the administrator selects here. You can select from the following settings.

Public Access - Everyone can share links and history when scheduling.
Role Access - Select which roles can share links and history when scheduling.
Private Sharing - Select a particular user who can share links and history when scheduling.

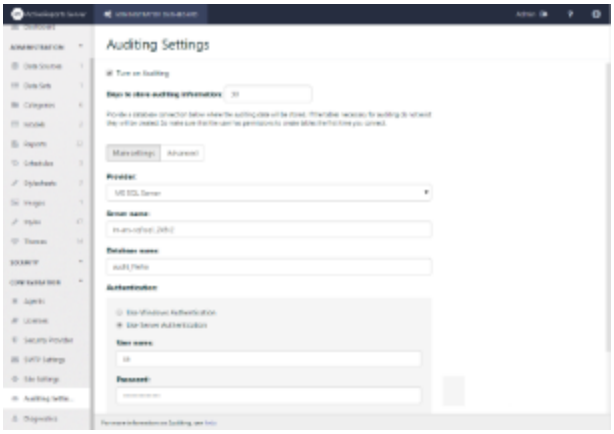
Hidden Administrator dashboard pages

ActiveReports Server provides you with an option to hide pages from the Administrator dashboard. To hide a page in the dashboard, click beside the page name you want to hide and click the **Save** button. You can select multiple pages to hide in the Administrator dashboard.

Managing Audit Settings

Whether you need to be HIPAA compliant and able to track who accessed personally identifiable data, or you just want to know which reports your clients actually use and which ones they do not use, auditing allows you to track which reports are accessed and what is done with them.

When you turn on auditing, in addition to setting the number of days to store audit data, you also specify a database in which to store the data.



Depending on the Provider you select, a different set of fields appears, allowing you to provide the information needed to connect to the database. Once you provide all of the information, click **Connect** to save it and begin auditing report usage statistics. ActiveReports Server asks you whether to create its own tables, and when you OK the action, it takes care of everything else, including adding an Audit Model and two Audit Reports.

Connection String Editor

If you just want to enter a connection string, you can click the **Advanced** button to open the Connection String Editor, where you can also specify a connection timeout period.

> **Tip**: You can use any UserContext attribute in the connection string by putting the attribute name between percent signs. For example:
> `Provider=SQLOLEDB.1;Data Source=HQ;Initial Catalog=%TenantDatabase%;`
> `User Id=myUsername;Password=myPassword;`

Provider

Here are descriptions of the fields that you need to fill in for each type of provider.

▸ MS SQL Server

Field	Description
Server Name	Enter the name of the server on which to store auditing data, for example, HQ.
Database Name	Enter the name of the database in which to store auditing data, for example, REPORTAUDITS.
Authentication	Select whether to use Windows authentication or server authentication to provide access to the database.
User Name	Enter the user name under which to log in to the database.
Password	Enter the password to use for logging in to the database.

▸ MySQL Server

Field	Description
Server Name	Enter the name of the server on which to store auditing data, for example, localhost.
Database Name	Enter the name of the schema object in your SQL Editor Object Browser in which to store auditing data, for example, ARServerAudit.
User Name	Enter the user name under which to log in to the database.
Password	Enter the password to use for logging in to the database.

▸ Oracle

Field	Description
Service Name	Enter the name of the service on which to store auditing data.
User Name	Enter the user name under which to log in to the database.
Password	Enter the password to use for logging in to the database.

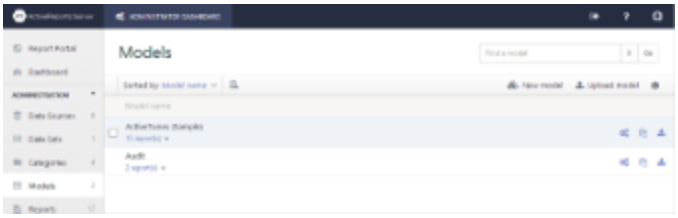
▸ Postgre SQL

Field	Description

Host	Enter the name of the host machine on which to store auditing data, for example, localhost.
Port	Enter the port number to use on the host machine, for example, 8080.
Database Name	Enter the name of the database in which to store auditing data, for example, REPORTAUDITS.
User Name	Enter the user name under which to log in to the database.
Password	Enter the password to use for logging in to the database.

Audit Model

Once you turn on auditing, provide the information needed to connect to a database, and click **Save** to apply changes, ActiveReports Server asks you whether to create its own tables. When you OK that action, it immediately begins auditing report usage statistics, saving the data in the specified database, and automatically creates a data model called Audit.



In the Models list, the Audit model has two reports by default, and differs from other models in that you cannot edit or clone it. This is to satisfy requirements for secure audit data.

Like other models, you can create new reports using its data, rename it, set permissions on it to allow other users to access it, view its history, and download it.

Audit Reports

Once you turn on auditing, provide the information needed to connect to a database, and click **Connect** to save it, ActiveReports Server asks you whether to create its own tables. When you OK that action, it immediately begins auditing report usage statistics, saving the data in the specified database, and automatically creates two auditing reports.

Reports contain data for the number of days that you specify when you set up auditing.

Auditing Access Summary

The Auditing Access Summary report logs all of the following types of access for the most recent seven day period:

- Recently accessed reports, including date, report name, user, and client IP address.
- Recently accessed models, including date, model name, action type, user, and client IP address.
- User logon and logoff attempts, including date, user, client IP address, and success or fail.

Auditing Activity Summary

The Auditing Activity Summary report logs all of the following action types for every report for the most recent 30 day period:

- Download

- Logon
- ModelGeneration
- ParametersResolving
- Rendering
- Upload

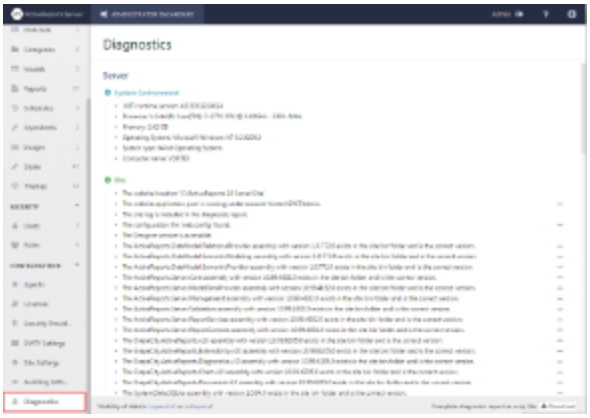
Creating Custom Report

You can also create custom reports on your auditing data, as you would on any other data model. ActiveReports Server provides you with the following tables and columns that you can use to create your own auditing report.

- **OperationInfo**
  RequestID
  ActionType
  Date
  ResourceType
  ResourceName
  ResourceID
  ModelID
  ModelName
  ModelVersion
  Parameters
  RenderingFormat
  Details
- **RequestInfo**
  RequestID
  ActionType
  Date
  RequestSource
  Link
  ClientIP
  ServerIP
  User
  Email
  UserContext
  Details
- **SemanticAttribute**
  RequestID
  Name
  EntityType
- **ServiceRecord**
  RequestID
  ActionType
  Date
  RequestSource
  User
  Details

Server Diagnostics

The **Diagnostics** page allows you to perform run-time monitoring of the server that hosts ActiveReports Server. This page provides information on the application system environment, its installed files, assemblies and services.



To open the Diagnostics page

1. In the **Configuration** section of the Administrator Dashboard, click **Diagnostics**.
2. In the item description list that appears, click the Toggle visibility icon next to an item to display details.
3. If you want to send a diagnostic report in for support, you can click the command at the bottom of the page to **Download the complete diagnostic report as a zip file.**

If you encounter an error, you are presented with the Error Reporting dialog. The information we collect from this dialog is protected by our privacy policies. For more information, see Privacy.

Privacy

Privacy Statement for the ActiveReports 10 Server Error Reporting Service

Last updated: 19/01/2016

GrapeCity is committed to helping protect your privacy. This statement explains how the ActiveReports 9 Server Error Reporting Service collects information and how collected information can be used. This statement does not apply to other online or offline GrapeCity Web sites, software, or services.

Why does GrapeCity collect information about errors and problems?

The information helps GrapeCity and ActiveReports 10 Server diagnose problems in the software you use and provide solutions. Not all problems have solutions but when solutions are available, they are offered as steps for solving a problem you've reported or as updates to install. To help prevent problems and make software more reliable, some solutions are also included in service packs and future versions of the software.

How is information collected?

GrapeCity ActiveReports 10 Server includes its own error reporting service. If a problem occurs in this software, you are asked if you want to report it. You can view the details of the report before sending it, although some files might not be in a readable format.

Some software also allows you to report problems automatically instead of requesting your consent each time a problem occurs. If you use automatic reporting, you are not prompted to review the information in a report before it is sent. However, no information is collected unless you (or your system or network administrator) choose to report problems. You can choose to stop reporting problems at any time.

What types of information can be collected?

The reporting service can collect information about problems that interrupt you while you work and about errors that occur behind the scenes. It is important to diagnose errors that occur behind the scenes because these problems, if left unsolved, may cause additional problems such as performance or program failures.

Reports contain information that is most useful for diagnosing and solving the problem that has occurred, such as:

- Where the problem happened in the software or hardware. Occasionally, empty files might be included as an initial indication of a problem.
- Type or severity of the problem, if known.
- Files that help describe the problem (typically system or report-generated files about software behavior before or after the problem occurred).
- Basic software and hardware information (such as operating system version and language, device models and manufacturers, or memory and hard disk size).

Your Internet Protocol (IP) address is also collected because you are connecting to an online service (web service) to send error reports. However, your IP address is used only to generate aggregate statistics. It is not used to identify you or contact you.

Reports might unintentionally contain personal information, but this information is not used to identify you or contact you. For example, a report that contains a snapshot of memory might include your name, part of a document you were working on, or data that you recently submitted to a Web site. If you are concerned that a report might contain personal or confidential information, you should not send the report.

Who can use the information and how can it be used?

GrapeCity uses information about errors and problems to improve ActiveReports 10 Server. GrapeCity employees and contractors may be provided access to information collected by the reporting service. However, they may use the information only to repair or improve the products that they publish or manufacture.
For example, if an error report indicates that a third-party product is involved, GrapeCity may send that information to the vendor of the product. The vendor may provide the information to sub-vendors and partners. If a third-party product is involved, and information is transferred to the vendor of that product, that vendor will be subject to its own policies. GrapeCity cannot monitor such vendor's compliance with its policies.

About surveys and report tracking

After you report a problem, you might be asked to complete a survey about the error experience. If you choose to provide a phone number or e-mail address in response to the survey, your error report will no longer be anonymous. GrapeCity may contact you to request additional information to help solve the problem you reported.

You might also be provided with the opportunity to track some error reports so you can check to see if a solution has been found. If you choose to track an error report, the report is associated with your e-mail address and is no longer anonymous.

Information storage, processing, and release

Information that is collected by or sent to GrapeCity may be stored and processed in the United States or any other country in which GrapeCity or its affiliates, subsidiaries, or agents maintain facilities. GrapeCity may disclose this information if required to do so by law or in the good faith belief that such action is necessary to: (a) conform to the edicts of the law or comply with legal process served on GrapeCity or the site; (b) protect or defend the rights or property of GrapeCity and its affiliates, or (c) act in urgent circumstances to protect the personal safety of GrapeCity employees, users of GrapeCity software or services, or members of the public.

GrapeCity occasionally hires other companies to provide limited services on its behalf, such as providing customer support, processing transactions, or performing statistical analysis of reports. GrapeCity will provide these companies only the information they need to deliver the service. They are required to maintain the confidentiality of the information and are prohibited from using it for any other purpose.

Security practices

GrapeCity is committed to helping protect the security of the information we collect. The Error Reporting Service uses multiple security technologies and procedures to help protect information from unauthorized access, use, or disclosure. For example, reports are sent to GrapeCity from your computer using encryption technology. The information is then stored on computer servers with controlled access.

Changes to this statement

GrapeCity may update this privacy statement. If we do, we will revise the "Last updated" information at the top of the statement. To stay informed about how we are helping protect the information collected by the Error Reporting Service, we encourage you to periodically review the privacy statement.

For more information

GrapeCity welcomes your comments regarding this privacy statement. If you believe that GrapeCity has not adhered to this statement, please contact us. We will use commercially reasonable efforts to promptly determine and remedy the problem.

GrapeCity Contact Information

8025 Creedmoor Rd., Raleigh NC 27613
TEL: +1 (866) 459-5567 | +1 (984) 242-0700

To find contact details for the GrapeCity subsidiary or affiliate in your country or region, see the GrapeCity corporate Web site at http://activereports.grapecity.com/.

How To

This section provides quick answers to your questions about how to perform specific tasks with ActiveReports Server.

Create Users and Roles
: This topic explains how to set up roles and groups so that users only have access to specified functions.

Activate a License
: This topic explains how to license ActiveReports Server with your product key.

Add Row-Level Security
: This topic explains how to license ActiveReports Server with your product key.

Use the Designer Web Control

This topic explains how to use the Designer control on a Web Form.

Hide Log In and Log Out Buttons

This topic explains how to hide authentication UI elements when you use the Designer in your own web applications.

Open the Designer without Selecting a Model

This topic explains how to open the Designer without first visiting the list of models.

Create a Custom Security Provider

This topic explains how to create a custom security provider and use it with a model.

Debug a Security Provider

This topic explains how to debug a custom security provider.

Work with the HTML5 Viewer Using Javascript

This topic explains how to work with the HTML5 Viewer using the Javascript.

Get a Security Token

This topic explains how to get a security token.

Create Users and Roles

To control which users have access to data models and reports, you can set up users and group them by roles. You can assign roles access to each report and data model, and you can designate which roles are able to upload reports.

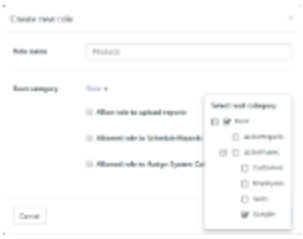
To set up a user

1. In the Security section of the Administrator Dashboard, click **Users**. The Users list appears.
2. Click the **New user** button. The Create new user dialog appears.
3. In the dialog, fill in the User Name, Password, Confirm password, E-mail, and Description fields.

4. Click **Next** and then select the check box under any roles to which to assign the user.
5. Click **Finish**. The new user is added to the Users list and any selected roles, and the user can log in to the Report Portal with the user name and password specified.

To set up a role

1. In the Security section of the Administrator Dashboard, click **Roles**. The Roles list appears.
2. Click the **Create role** button. The Create Role dialog appears.



3. In the **Role Name** box, enter a name for the new role.
4. If you want to allow the role to upload reports, select the **Allow Upload Report** check box.
5. Click **Create role**. The new role is added to the list, with an extra command that allows you to delete it. To begin using the role, add users to it.

To add a user to a role

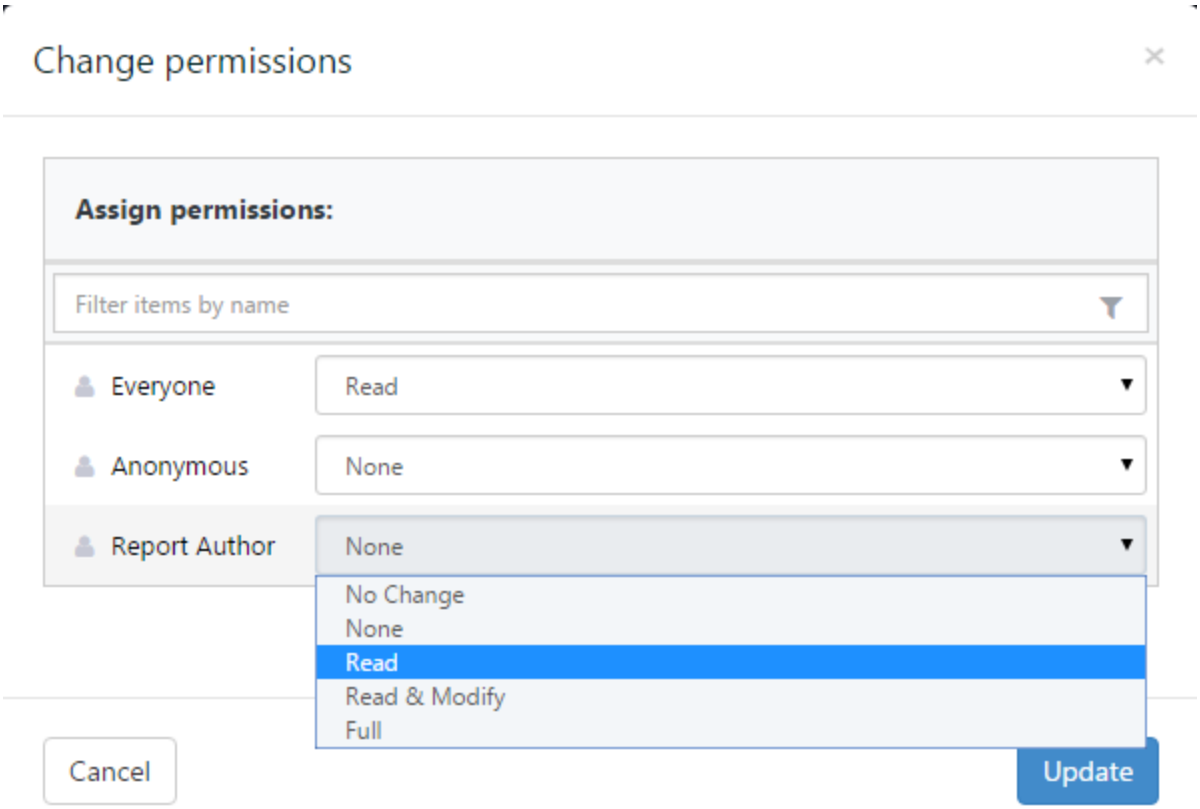
1. In the Security section of the Administrator Dashboard, click **Roles**.
2. In the Roles list that appears, to the right of the role to which you want to add a user, click the **Manage users** button. The Manage users dialog appears.
3. In the Manage users dialog, select check boxes next to users to assign them to the role.

   > **Note:** To remove a user from the role, clear the corresponding check box.

4. Click the **Update** button to save the changes.

To give a role permissions on a report

1. From the Administrator Dashboard, select **Reports**.
2. On the **Reports** page, select a report from the list, and then click **Permissions** button.
3. In the **Change permissions** dialog that appears, next to each role, drop down the box to select whether to allow its users **Read**, **Read & Modify**, or **Full** permissions on the report, or **None** to hide the report from users in that role.

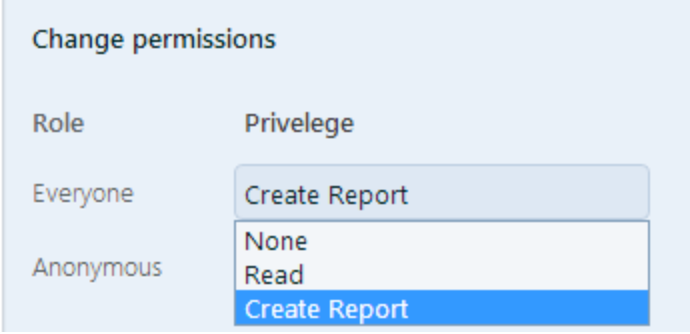


- **Read** allows users in the role to see the report in the Report Portal **Open** page and preview the rendered report, but does not allow them to modify it in the Report Portal.
- **Modify** allows users in the role to modify the report in the Report Portal.
- **Change Permissions** allows users in the role to modify permissions on the report.

4. Click the **Apply** button to save the changes. All users in the selected roles are granted the specified access.

To give a role permissions on a model

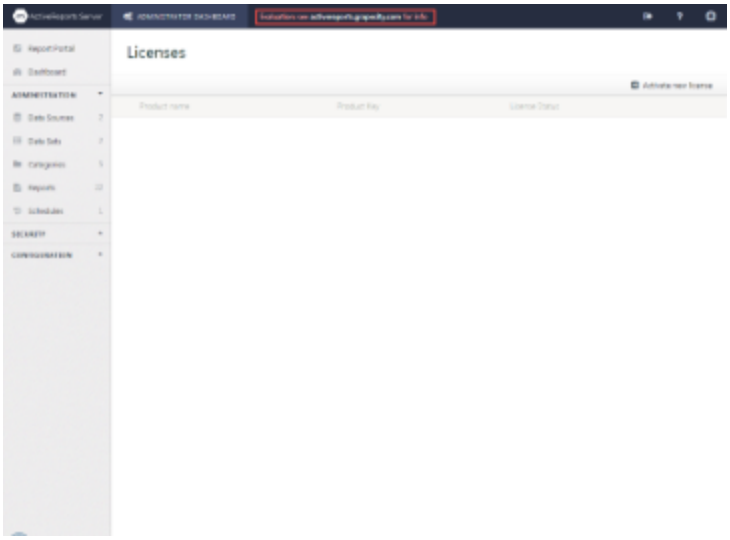
1. From the Administrator Dashboard, select **Models**
2. On the **Models** page, select a model from the list, and then click the **model properties** button to display the properties.
3. In the **Change permissions** section, next to each role, drop down the box to select whether to grant **Read** permission, **Create Report**, or **None** to hide the model from users in that role



4. The selected roles have permission to create reports on the model.

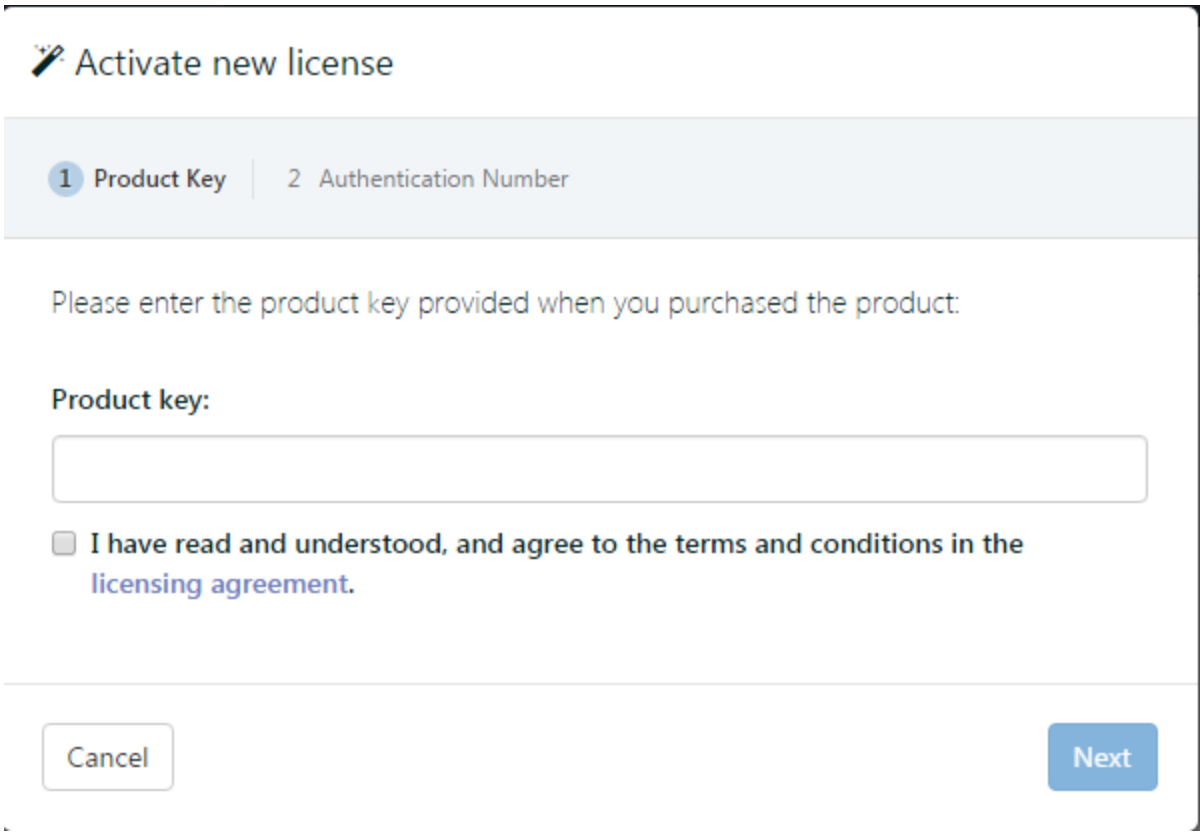
Activate a License

If you have not yet licensed ActiveReports Server, you will see an evaluation message in the top bar of both the Administrator Dashboard and the Report Portal. To remove the evaluation message, license the server using the product key you received via e-mail when you purchased the product.



To license your server

1. In the configuration section of the Administrator dashboard, click **Licenses**. The Licenses list appears.
2. Click **Activate new license.** The Enter Product Key dialog appears.
3. In the Activate new license dialog, fill in the Product Key that you received when you purchased ActiveReports Server.



4. Select the check box to indicate your acceptance of the licensing agreement terms, and click **Next**. The Authentication Number fills in automatically.
5. Click the link that appears in Step 2 to open the activation site in a new browser tab. Your Product Key and Authentication Number fill in automatically.
6. Click **Send Request**. Your License Key appears, along with your activation date. Select the License Key number and copy it.
7. Return to the browser tab with the Administrator Dashboard, paste the License Key into Step 3 of the dialog and click **Finish**. The evaluation message is removed, and the product key is listed in the Licenses list with a License Status of Activated.

Add Row-Level Security

To control access to row-level data based on user permissions, you must create security filters and apply them to the model entities that you want to secure. Security filters can be based on LDAP or Active Directory, or you can create a custom security provider.

For information on creating custom security providers, please see Create a Custom Security Provider topic, or the sample in C:\ActiveReports 10 Server\SDK\ActiveTunes.SecurityProvider.

If you use LDAP or Active Directory, please see Using LDAP and Active Directory for Single Sign-On. If you have created a custom security provider, follow the steps below.

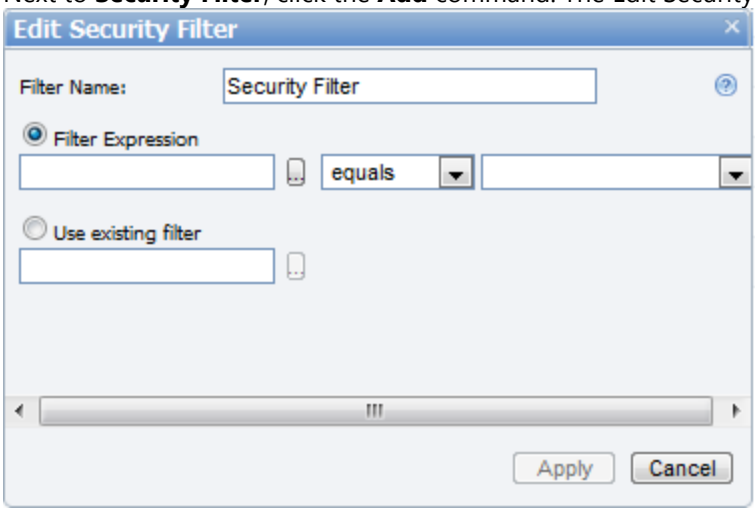
▸ To configure ActiveReports Server to use your custom security provider

1. In the configuration section of the Administrator dashboard, click **Security Provider**. The Custom security provider drop-down list appears.
2. Drop down the list and select your custom security provider.

3. In the Security Provider Settings that appear, enter a value for any properties that appear.
4. When you have entered all of the information, click the **Apply changes** button. Your custom security provider becomes available for use in security filters.

To create a new security filter for row-level security

1. In the Administration section of the Administrator dashboard, click **Models**. The Models list appears.
2. Select the model to which you want to add a filter, and then click **Edit model** button. The model editor appears.
3. Select the entity that you want to filter. The editable properties appear in the center workspace.
4. Next to **Security Filter**, click the **Add** command. The Edit Security Filter dialog appears.



5. In the box next to **Filter Name**, enter a name for your filter.
6. With **Filter Expression** selected, click the ellipsis button to open the Select Attribute dialog where you can select an attribute of the entity on which to filter data, and click **OK**.
7. Drop down the list next to the attribute to select whether to display data that **equals** or does **not equal** the following value.
8. Drop down the list to the right, and select a User Context Attribute Name mapping against which to filter the selected attribute.
9. Click the **Apply** button to create the filter, or to discard the changes, click **Cancel**.

Use the Designer Web Control

The Web controls need to be directed to the server used for ActiveReports 10 Server in order to function. In your production application, specify the user name and password based on the current user and store information in the current session to persist these values.

To add the Designer control to a Web Form

1. In Visual Studio, create a new C# ASP.NET Web Site.

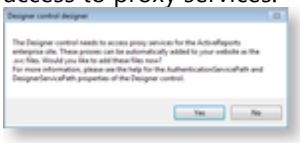
   If you need to add the control to your toolbox, drop down these steps.

   1. Right-click in the General tab and select **Choose items**.
   2. In the Choose Toolbox Items dialog that appears, the .NET Framework Components tab is selected by default. Click the **Namespace** column header to sort by namespace.
   3. The ActiveReports.Server.ReportControls namespace is at or near the top of the list. Select the checkbox next

to **Designer** and click **OK**.

4. The Designer control appears in your Visual Studio toolbox.

2. Open the Design view of your Web Form and from the toolbox, drag and drop the Designer control onto the body section of the Web Form.
3. On the message box that appears, click **Yes** to automatically add .svc files to your Web site to give the designer access to proxy services.



4. You can resize the designer using the Properties grid by changing the **Width** property.
5. So that the link at the top of the Designer will work for your Admin users, change the **AdminPath** property to the URL of your Admin site, for example, http://localhost:8080/Admin/.

To direct the Designer to the server used for ActiveReports Server

You can specify the security token and ActiveReports 10 Server host using code in a Global Application Class.

1. From the Visual Studio **Website** menu, select **Add New Item**.
2. In the Add New Item dialog that appears, select Global Application Class and click **Add**.
3. In the Global.asax file that appears, provide a handler for the ResolveRemoteEndPoint event using code like the following in the Application Start event:

To provide a handler for the event

C# code. Paste INSIDE the Application_Start event.

```
ReportServiceProxy.ResolveRemoteEndpoint += ResolveRemoteEndpoint;
```

4. Below the Application Language tag at the top of the file, import the Servicing namespace so that you can use the ReportServiceProxy using a directive like the following:

To import the Servicing namespace

ASP.NET code. Paste in the Global.asax file on the line BELOW the Application Language line.

```
<%@ Import Namespace="ActiveReports.Server.ReportControls.Servicing" %>
```

5. Below the Application Start event, create the ResolveRemoteEndpoint event using code like the following, but with your address, user name, and password:

To create the ResolveRemoteEndpoint event

C# code. Paste in the Global.asax file AFTER the Application_Start event.

```
    static void ResolveRemoteEndpoint(RemoteEndpoint remoteEndpoint)
    {
        remoteEndpoint.Address = "http://localhost:8080";
        remoteEndpoint.SecurityToken = "MySecurityToken";
    }
```

You can also specify the user name, password, and ActiveReports 10 Server host in your web application's web.config file.

1. In the web.config file, provide a handler for the ResolveRemoteEndPoint event using code like the following.

▶ To provide a handler for the event

C# code. Paste BETWEEN the <configSections> and </configSections> tags.

```
    <sectionGroup name="activereports.server">
      <section name="reportServiceProxy"
type="ActiveReports.Server.ReportControls.Configuration.ActiveReportsServerSection,
      ActiveReports.Server.ReportControls, Version=x.x.xxxx.x, Culture=neutral,
PublicKeyToken=d557f2f30a260da2" allowDefinition="Everywhere" />
    </sectionGroup>
```

2. Below the Application Start event, create the ResolveRemoteEndpoint event using code like the following, but with your address, user name, and password

▶ To import the Servicing namespace

ASP.NET code. Paste in the web.config file BELOW the </configSections> tag.

```
  <activereports.server>
    <reportServiceProxy remoteReportServicePath="http://localhost:8080/"
username="Admin" password="1" />
  </activereports.server>
```

Hide Log In and Log Out Buttons

When you add the Designer control to your custom web application and use a custom security provider, you probably do not want to show our built-in Log In and Log Out buttons on the designer.

To hide the authentication buttons

1. On the Design view of your web form, select the Designer control.
2. In the Properties window, locate the **SecurityToken property (on-line documentation)**.
3. Enter a string value representing a valid security token.

You can obtain a security token from the IReportService Login Method.

Open the Designer without Selecting a Model

If you want to make report designing even faster for your business users, you can allow them to skip the steps involved in selecting a data model. Use the new DataModel property so they can just click a button to open the designer on a blank report with a pre-selected data model, or copy the Create report link from the models list.

This topic assumes that you are already familiar with how to Use the Designer Web Control.

To code a button click event to open a blank report with data

1. From the Visual Studio Standard toolbox, drag a Button control and drop it onto your Web form.
2. Double-click the button to create a Click event handling method in the code view.

3. Paste code like the following into the event to open a blank report in the designer with a pre-selected data model.

C# code. Paste INSIDE the Button Click event.

```
this.Designer1.ModelName = "ActiveTunes (Sample)";
this.Designer1.ReportAction =
ActiveReports.Server.ReportControls.ReportAction.Create;
```

To copy the Create report link

1. In the Administration section of the Administrator dashboard, click **Models.**
2. In the list of models, find the model that you want to use and, to the right of the model, click the **Create report** command.
3. In the browser's address bar, select the URL and copy it.
4. You can use this link as a hyperlink or in a response.redirect command.

Create a Custom Security Provider

You must create custom security providers in order to provide security filters for model entities. This allows you to add single sign-on functionality using LDAP and Active Directory, and to control access to data with row-level security.

Once you implement interfaces from the Extensibility assembly and configure the resulting assembly in the Administrator Dashboard by selecting it in the Security Provider list, you and your administrators can add security filters to entities when you edit a model. For more information, see the Managing Security Providers and Modifying an Entity topics.

You can use your UserContext attributes in the connection string of your data models by encasing them in percent signs. For more information, see the Changing the Connection String topic.

▸ To implement Extensibility interfaces in a .NET 3.5 Class Library

1. In Visual Studio, create a new C# .NET Framework 3.5 Class Library project.
2. From your ActiveReports 10 Server installation folder, copy the **ActiveReports.Server.Extensibility.dll** file into your project's **bin** folder.
3. In the Visual Studio Solution Explorer, right-click **References** and select **Add Reference**.
4. In the Add Reference dialog that appears, on the **Browse** tab, look in the bin folder, select the **ActiveReports.Server.Extensibility.dll** and click **OK**.
5. Add a class to your project that implements the ISecurityProvider interface, and another that implements the ISecurityProviderFactory interface.

   > **Note**: For details on how to do this, please see the **ActiveTunes.SecurityProvider** sample in the C:\ActiveReports 10 Server\SDK directory.

   Important methods in the ISecurityProvider interface include:
   - **CreateToken** creates the security token that the server holds to identify the current logged in session with **username** and **password** parameters, as well as a **custom** parameter for any other credentials data used by the provider.
   - **FilterRoles** gets the collection of **roles** associated with the specified security **token**.
   - **GetUserContext** returns the UserContext object for the specified security **token**.
   - **GetUserDescription** returns two useful objects:
     - **UserDescription.Email** automatically fills the email field in the error log submission dialog.
     - **UserDescription.FriendlyUserName** automatically fills the CreatedBy and ModifiedBy fields for reports.
6. Once all of the necessary classes are in place with your custom data, from the Build menu, select **Build**

ClassLibrary1. A DLL is created in your project's **bin** > **Release** folder.

To configure the provider

1. In Windows Explorer, create a SecurityProviders folder on the same level as your site, for example, if you have C:\ActiveReports 10 Server, create the folder C:\ActiveReports 10 Server\SecurityProviders.
2. Copy your custom security provider DLL file into the new folder.
3. On the Administrator Dashboard, in the Configuration section, click **Security Provider**.
4. From the **Custom security provider** drop-down that appears, select your new provider.
5. Select values for any properties that may be available, depending on your provider.
6. Click the **Apply changes** button.

To add a row-level security filter

1. On the Administrator Dashboard, in the Administration section, click **Models**.
2. In the list of models that appears, next to the model containing the entity that you want to filter, click the **Edit** command.
3. In the model editor that appears, from the **Entities** list to the left, click the entity that you want to filter.
4. In the workspace at the center, next to the **Security Filter** property, click the **Add** command.
5. In the Edit Security Filter dialog that appears, select **Filter Expression**, and click the ellipsis button next to the box under that label.
6. In the Select Attribute dialog that appears, expand nodes as necessary to select the attribute to which you want to apply a filter condition, and click **OK**.
7. In the center, you can choose whether the attribute **equals** or does **not equal** the value to the right.
8. In the drop-down box to the right, select the value from your security provider to compare to the attribute on the left.
9. Click **Apply** to add the filter to the entity.

Debug a Security Provider

Once you have created a custom security provider, you may need to debug it. The following steps walk you through how to do this using the sample custom security provider project installed with ActiveReports Server. By default, this sample is in the following folder.

C:\ActiveReports 10 Server\SDK\ActiveTunes.SecurityProvider

To deploy the custom security provider for debugging

1. If you have not done so already, open the project in Visual Studio and from the Build menu, select **Build Solution**.
2. If you have not done so already, in the C:\ActiveReports 10 Server directory, create a new folder named **SecurityProviders**.
3. From your project bin\Debug\ folder, copy the following files, and paste them into C:\ActiveReports 10 Server\SecurityProviders\ to make the security provider and debugging file available to ActiveReports 10 Server.
   - ActiveTunes.SecurityProvider.dll
   - ActiveTunes.SecurityProvider.pdb
   - System.Data.SQLite.dll
4. Open the ActiveReports Administrator Dashboard in your browser, and under Configuration, select **Security Provider**.
5. Next to **Custom security provider**, drop down the list and select ActiveTunesSecurityProviderFactory. The page refreshes to allow you specify security provider settings and to test the configured provider.

> **Important**: If you need to update the DLL at any point, you must first change this value back to **Default (builtin)**. You can change it back after you finish updating the DLL.

6. Under Security Provider Settings, for the ConnectionString property, enter the following connection string and click **Apply changes**.

Connection string
`Data Source=C:\ActiveReports 10 Server\SDK\ActiveTunes.SecurityProvider\ActiveTunes.sqlite`

To debug the custom security provider

In order to perform the following steps, you must run Visual Studio as Administrator.

1. From the Visual Studio **Tools** menu, select **Attach to Process**.
2. In the Attach to Process dialog that appears, select the **Show processes in all sessions checkbox**, or navigate to the remote machine where the server web site resides.
3. Scroll down and select the **w3wp.exe** process for server's web site and click **Attach**. The project goes into Debugging mode.

> **Caution**: There may be multiple w3wp.exe processes, so be sure to attach the one on the server machine running the ActiveReports services.

4. In the ActiveTunesSecurityProvider.cs file, locate the **ActiveTunesSecurityProvider.CreateToken** method and click in the margin to the left to set a breakpoint on it.
5. In your browser, on the Administrator Dashboard Security Provider page in the **Login to test** and **Password to test** boxes, enter test values from the table below.

Login	Password
andrew@chinookcorp.com	1
nancy@chinookcorp.com	2
jane@chinookcorp.com	3
margaret@chinookcorp.com	4
steve@chinookcorp.com	5
michael@chinookcorp.com	6
robert@chinookcorp.com	7
laura@chinookcorp.com	8

6. Click the **Test provider** link. The Visual Studio window takes focus and shows the breakpoint, allowing you to debug the code.

To debug your custom security provider with Visual Studio on the server

Here is another way to debug your security provider if you have Visual Studio installed on the server and have direct access to the server machine.

> **Caution**: Ensure that you remove this code before you go live with your deployment.

1. Open your SecurityProvider implementation, and in one of its methods, for

example, ISecurityProviderFactory.Create, add the following code.

C# code. Paste inside the method to debug.

```
System.Diagnostics.Debugger.Break();
```

2. From the Build menu, select **Rebuild Solution**.
3. Copy the resulting DLL and PDB files and paste them into C:\ActiveReports 10 Server\SecurityProviders\.
4. Open the ActiveReports Administrator Dashboard in your browser, and under Configuration, select **Security Provider**.
5. Next to **Custom security provider**, drop down the list and select your security provider. The page refreshes to allow you specify any security provider settings specified in your provider.

   > **Important**: If you need to update the DLL at any point, you must first change this value back to **Default (builtin)**. You can change it back after you finish updating the DLL.

6. When you run a test or try to log in as a user, The Visual Studio Just-In-Time Debugger prompt appears asking whether you want to debug w3wp.exe.
7. Click **Yes** to open Visual Studio and begin debugging.

View Reports Using the HTML5 Viewer

This topic walks you through how to initialize the HTML5 Viewer in JavaScript. It also provides details on all public initialization options, plus all of the public API methods and properties that you can use with the viewer after you initialize it.

Initializing the HTML5 Viewer

1. Copy the following files from the **...\ActiveReports 10 Server\SDK\HTML5 Viewer.Source** folder to a folder accessible from the target HTML page:
   - GrapeCity.ActiveReports.Viewer.Html.js
   - GrapeCity.ActiveReports.Viewer.Html.min.js
   - GrapeCity.ActiveReports.Viewer.Html.cs
2. In the target HTML page, add the references to the GrapeCity.ActiveReports.Viewer.Html.css, GrapeCity.ActiveReports.Viewer.Html.js and its dependencies:
   - jQuery 1.9.0 or higher
   - Bootstrap 3.0
   - Knockout.js 2.3.0 or higher

   > **Note:** You can access the dependencies like jQuery in a Content Delivery Network (CDN) or copy them locally.

   HTML

   ```
   <link rel="stylesheet" href="GrapeCity.ActiveReports.Viewer.Html.css" >
   <script src="//ajax.googleapis.com/ajax/libs/jquery/1.9.0/jquery.min.js"></script>
   <script src="//netdna.bootstrapcdn.com/bootstrap/3.0.3/js/bootstrap.min.js"></script>
   <script src="http://cdnjs.cloudflare.com/ajax/libs/knockout/2.3.0/knockout-min.js"
   type="text/javascript"></script>
   <script src="Scripts/GrapeCity.ActiveReports.Viewer.Html.js" type="text/javascript"></script>
   <script type="text/javascript">
   </script>
   ```

3. In the target HTML page, add the DIV element that will contain the HTML5 Viewer.

   HTML

   ```
   <div id="viewer" style="width:600px;height:480px;"></div>
   ```

4. Add the following code to initialize the HTML5 Viewer. The code might vary depending on the technology that you use to develop the HTML5 Viewer component. To learn how to get a security token for the reportService, see Get a Security Token.

   Javascript

   ```
   $(function ()
   {
   ```

```
   var viewer = GrapeCity.ActiveReports.Viewer(
   {
      element: '#viewer',
      report: {
         id: "myreport"

      },

      reportService: {
          url: 'http://myAR10server.com/ReportService.svc/json/',
          securityToken: securityToken,
          resourceHandler : 'http://myAR10server.com/TemporaryResource.axd/'
      },
      uiType: 'desktop',
      documentLoaded: function reportLoaded()
      {
         console.log(viewer.pageCount);
      },
      reportLoaded: function (reportInfo)
      {
         console.log(reportInfo.parameters);
      },

      error: function (error)
      {
         console.log("error");
      }

   });
});
```

Alternatively, you can use the REST API endpoint URL for the HTML5 Viewer initialization.

Javascript

```
$(function ()
{
   var viewer = GrapeCity.ActiveReports.Viewer(
   {
      element: '#viewer',
      report: {
         id: "myreport"

      },

      reportService: {
          url: 'http://myAR10server.com/api/',
          securityToken: securityToken,
          resourceHandler : 'http://myAR10server.com/TemporaryResource.axd/'
      },
      uiType: 'desktop',
      documentLoaded: function reportLoaded()
      {
         console.log(viewer.pageCount);
      },
      reportLoaded: function (reportInfo)
      {
         console.log(reportInfo.parameters);
      },
```

```
      error: function (error)
      {
         console.log("error");
      }

   });
});
```

5. Add the following code to the system.webServer section of the web.config file of your ActiveReports 10 Server web site.

XML

```
<system.webServer>

        <httpProtocol>

                <customHeaders>

                        <add name="Access-Control-Allow-Origin" value="*" />

                        <add name="Access-Control-Allow-Headers" value="Authorization, Origin, Content-
Type, Accept, X-Requested-With" />

                </customHeaders>

        </httpProtocol>

...

</system.webServer>
```

Initialization Options

The following options can be set during initialization or at runtime while working with the HTML5 Viewer.

uiType

Description: Sets the UI mode of the HTML5 Viewer.

Type: String

Accepted Value: 'Custom', 'Mobile' or 'Desktop'

Example:
```
viewer.option('uiType', 'Mobile');
```

element

Description: JQuery selector that specifies the element that hosts the HTML5 Viewer control.

> **Note**: This option is used during initialization only.

Type: String

Example:
```
var viewer = GrapeCity.ActiveReports.Viewer(
{
   element: '#viewerContainer2',
   reportService: {
      url: '/ActiveReports.ReportService.asmx'
   },
});
```

reportService

Description: The report service that can use ActiveReports Server or ActiveReports Web Report Service.

Type: Object that has the URL and optional securityToken properties

Example:
```
reportService: {
   url: 'http://remote-ar-server.com',
   securityToken: '42A9CD80A4F3445A9BB60A221D042FCC',
   resourceHandler: 'http://remote-ar-server.com/resourceHandler.aspx'
};
```

reportService.url

Description: The URL of ActiveReports 10 Server instance of the ActiveReports Web service that provides the reportInfo and output.

Type: String

Example:
```
reportService: {
   url: 'http://remote-ar-server.com'
};
```

reportService.securityToken

Description: The security key needed to log in to ActiveReports Server.

Type: String

Example:
```
reportService: {
   securityToken: '42A9CD80A4F3445A9BB60A221D042FCC'
};
```

reportService.resourceHandler

Description: The URL of the ActiveReports Server resource handler.

Type: String

Example:
```
reportService: {
   resourceHandler: 'http://remote-ar-server.com/resourceHandler.aspx'
};
```

report

Description: The report that is displayed in ActiveReports Server or ActiveReports Web Report Service.

Type: An object that has id and parameters properties.

Example:
```
report: {
   id: 'CustomersList',
   parameters: [
   {
      name: 'CustomerID',
      value: 'ALFKI'
   }]
};
```

reportID

Description: The id of the report to be shown by the HTML5 Viewer.

Type: String

Example:

Example:
```
report: {
   id: 'CustomersList',
   parameters: [
   {
      name: 'CustomerID',
      value: 'ALFKI'
```

```
   }]
};
```

reportParameters

Description: The array of the {name, value} pairs that describe the parameters values used to run the report.

Type: Array

Example:
```
report: {
   id: 'CustomersList',
   parameters: [
   {
      name: 'CustomerID',
      value: 'ALFKI'
   }]
};
```

reportLoaded

Description: The callback that is invoked when the HTML5 Viewer obtains the information about the requested report. The reportInfo object is passed in the callback including the TOC info, Parameters info and the link to the rendered report result.

Type: function(reportInfo)

Example:
```
var reportLoaded = function reportLoaded(reportInfo)
   {
      console.log(reportInfo.parameters);
   }
viewer.option('reportLoaded', reportLoaded);
```

action

Description: The callback that is invoked before the HTML5 Viewer opens the hyperlink, bookmark link, drill down report or toggles the report control visibility.

Type: function(actionType, actionParams)

Example:
```
function onAction(actionType, actionParams)
{
   if (actionType === 0)
   {
      window.open(params.url, "Linked from report", "height=200,width=200");
   }
}
viewer.option('action', onAction);
```

availableExports

Description: The array of export types available via Export functionality of HTML5 Viewer. By default, PDF, Word, Image, Mht, and Excel exports are available.

Type: Array

Example:
```
viewer.option("availableExports", ['Pdf']);
```

maxSearchResults

Description: The number of search results received for a single search invoke.

Type: Number

Example:
```
maxSearchResults: 10
```

error

Description: The callback that is invoked when an error occurs in the process of displaying the report. The error message can be customized.

Type: function(error)

Example:
```
var options = {
  error: function(error){
    error.message = "My error message";
  },
  // other properties
};
var viewer = GrapeCity.ActiveReports.Viewer(options);
```

documentLoaded

Description: The callback that is invoked when a document is loaded entirely on the server.

Type: function()

Example:
```
var documentLoaded = function documentLoaded()
   {
      setPaginator();
   }
viewer.option('documentLoaded', documentLoaded);
```

localeUri

Description: The URL of the file containing the localization strings.

> **Note**: This option is used during initialization only.

Type: String

Example:
```
var viewer = GrapeCity.ActiveReports.Viewer(
{
   localeUri: 'Scripts/i18n/ru.txt'
});
```

Public API Methods and Properties

After initializing the HTML5 Viewer, the following API methods and properties can be used.

Methods

option

Description: Gets or sets the option value by name if the value parameter is specified.

Syntax: `option(name,[value])Object`

Parameters:

- **name**: The option name to get or set.
- **value**: (optional) The option value to set. If this argument is omitted than the method returns the current option value.

Example:
```
viewer.option('uiType', 'mobile');
viewer.option('report', {
   id: 'my report'
});
```

Return Value: The current option value.

refresh

Description: Refreshes the report preview.

Syntax: `option(name,[value])Object`

Example:

```
viewer.refresh()
```

Return Value: Void

print

Description: Prints the currently displayed report if any.

Syntax: `print()Void`

Example:
```
viewer.print()
```

Return Value: Void

goToPage

Description: Makes the viewer to display the specific page, scroll to the specific offset (optional) and invokes the callback once it's done.

Syntax: `goToPage(number, offset, callback)Void`

Parameters:

- **number**: The number of pages to go to.
- **offset object**: The object such as {left:12.2, top:15}.
- **callback**: The function to call after perform action.

Example:
```
viewer.goToPage(1, {
   2, 3
}, function onPageOpened()
{});
```

Return Value: Void

backToParent

Description: Makes the viewer to display the parent report of the drill-down report.

Syntax: `backToParent()Void`

Example:
```
viewer.backToParent()
```

Return Value: Void

destroy

Description: Removes the viewer content from the element.

Syntax: `destroy()Void`

Example:
```
viewer.destroy()
```

Return Value: Void

export

Description: Exports the currently displayed report.

Syntax: `export(exportType,callback,saveAsDialog,settings)Void`

Parameters:

- **exportType**: Specifies export format.
- **callback**: Function that is invoked once the export result is available (its URL is passed in the callback).
- **saveAsDialog**: Indicates whether the save as dialog should be shown immediately once the export result is ready.
- **settings**: The export settings, vary for each export type.

Example:
```
viewer.export('Word', function ()
{
```

```
   console.log('export callback');
}, true, {
   FileName: 'Document.doc'
})
```

Return Value: Void

search

Description: Performs the search of a specific term with specific search options (match case, whole word) and invokes the specific callback with the search result passed.

Syntax: `search(searchTerm, searchOptions, callback)Void`

Parameters:

- **searchTerm**: String to find.
- **searchOptions**: The object optionally defines the search options:
  - **matchCase**: Whether the search should respect the case.
  - **wholePhrase**: Whether the search should look for a whole phrase.
- **callback**: The function to call after performing search.

Example:

```
viewer.search('a', {
   matchCase: true,
   wholePhrase: false
}, function (results)
{
   console.log(results);
});
```

Return Value: Void

getToc

Description: Obtains the part of the report TOC specified by parent(node), startChild(index), count parameters and invokes callback function passing the result as parameter.

Syntax: `getToc(callback) Void`

Parameters:

- **callback**: The callback to handle TOC tree.

Example:

```
viewer.getToc(function (toc)
{
   console.log(toc);
})
```

Return Value: Void

Properties

pageCount

Description: Gets the page count of the currently displayed report.

Syntax: `viewer.pageCount`

Example:

```
console.log(viewer.pageCount)
```

Return Value: An integer representing page count.

currentPage

Description: Gets the currently displayed page number.

Syntax: `viewer.currentPage`

Example:

```
console.log(viewer.currentPage)
```

Return Value: An integer representing currently displayed page number.

Toolbar

Description: Returns the HTML element that displays the toolbar in desktop UI mode. The developer may use it to add the custom elements or remove the existing ones using jQuery/Html/Css capabilities.

Syntax: `viewer.Toolbar`

Example:
```
// Toolbar, MobileToolbarTop, MobileToolbarBottom
$(viewer.toolbar).hide();
$(viewer.toolbarTop).hide();
$(viewer.toolbarBottom).hide();
```

ToolbarTop

Description: Returns the HTML element that displays the top toolbar in mobile UI mode. The developer may use it to add the custom elements or remove the existing ones using jQuery/Html/Css capabilities.

Syntax: `viewer.ToolbarTop`

Example:
```
// Toolbar, MobileToolbarTop, MobileToolbarBottom
$(viewer.toolbar).hide();
$(viewer.toolbarTop).hide();
$(viewer.toolbarBottom).hide();
```

ToolbarBottom

Description: Returns the HTML element that displays the bottom toolbar in mobile UI mode. The developer may use it to add the custom elements or remove the existing ones using jQuery/Html/Css capabilities.

Syntax: `viewer.ToolbarBottom`

Example:
```
// Toolbar, MobileToolbarTop, MobileToolbarBottom
$(viewer.toolbar).hide();
$(viewer.toolbarTop).hide();
$(viewer.toolbarBottom).hide();
```

Get a Security Token

When you use the HTML5 Viewer to access the ActiveReports Server web site, you need to provide the Report Service with a security token to use.

Below are the steps to retrieve a security token.

To get a security token by the Javascript function

1. In the target HTML page, add the following code to get a security token. The code might vary depending on the technology that you use to develop the HTML5 Viewer component. For more information, see the **HTML5 Viewer Sample**. By default, this sample is in the following folder.

   C:\ActiveReports 10 Server\SDK\Samples\HTML5 Viewer.

   > **Note:** You need to change these settings in the code below.
   >
   > - **url** parameter represents the location (reportservice.svc/json/login) which stays constant where as the arsEndpoint represents the URL used to access the Report Portal Website. **Example:**

http://<ActiveReports10ServerName>:<portnumber>.
- **username** is your username to log into the ActiveReports Server web site.
- **password** is your password to log into the ActiveReports Server web site.

Javascript

```
function getSecurityToken() {
    if (_securityToken) return _securityToken;
    _securityToken = "error";

    try {
        $.ajax({
             async: false,
             type: "POST",
             url: arsEndpoint + "ReportService.svc/json/Login",
             data: JSON.stringify({
                   username: "username",
                   password: "password"
             }),
             contentType: "application/json",
             dataType: "json"
        }).done(function(res) {
              _securityToken = res.d;
        }).fail(function() {
              _securityToken = "error";
        });
    } catch(e) {
          return "error";
    }

    return _securityToken;
}
```

If you are using the REST API, add the following code to get a security token.

Javascript

```
function getSecurityToken() {
    if (_securityToken) return _securityToken;
    _securityToken = "error";

    try {
        $.ajax({
             async: false,
             type: "POST",
             url: arsEndpoint + "api/accounts/login",
             headers: JSON.stringify({
                   user: "username",
                   password: "password"
```

```
            }),
            contentType: "application/json",
            dataType: "json"
        }).done(function(res) {
              _securityToken = res.Token;
        }).fail(function() {
              _securityToken = "error";
        });
    } catch(e) {
          return "error";
    }

    return _securityToken;
}
```

Samples and Walkthroughs

These topics describe the Server Core samples and provide step-by-step walkthroughs detailing ActiveReports Server project creation.

Samples
: These topics explain how to access the sample files installed with ActiveReports Server.

Walkthroughs
: Learn to work with ActiveReports 10 Server controls in Visual Studio.

Samples

The ActiveReports Server installation includes several sample projects in a Visual Studio 2012 solution. You can find the solution in:

C:\ActiveReports 10 Server\SDK\Samples

To open the solution, double-click the **ActiveReports 10 Server Samples.sln** file.

Sample Project	Description
ASP.NET Integration	Shows how you can embed the Report Designer control in your own applications, and how you can customize the Report List control. It also demonstrates how to use the Web service features using javascript.
Console Web Service Client	Shows how you can use the Web service features to serve reports to Windows applications.

Also included is a separate Security Provider Sample. See Security Provider Sample for details.

Security Provider Sample

The ActiveReports Server installation includes a security provider sample that you can find in:

C:\ActiveReports 10 Server\SDK\Samples\ActiveTunes.SecurityProvider

To open the solution, double-click the **ActiveTunes.SecurityProvider.sln** file.

Sample Project	Description
ActiveTunes.SecurityProvider	Shows how you can provide proprietary or single-sign-on authentication instead of the built-in ActiveReports Server security and use row-level security.
ActiveTunes.SecurityProvider.UnitTests	Shows how you can test the row-level security using sample data.

To compile and use the Security Provider sample

1. With the sample open in Visual Studio, from the **Build** menu, select **Build Solution**.
2. In Windows Explorer, open the following folder, and copy the **ActiveTunes.SecurityProvider.dll** assembly.
   C:\ActiveReports 10 Server\SDK\ActiveTunes.SecurityProvider\ActiveTunes.SecurityProvider\bin\Debug
3. Paste the assembly into the following folder in your ActiveReports 10 Server installation folder.
   C:\ActiveReports 10 Server\SecurityProviders
4. On the Administrator Dashboard, in the Configuration section, click **Security Provider**.
5. From the **Custom security provider** drop-down that appears, select **ActiveTunesSecurityProviderFactory**.
6. In the **ConnectionString** property that appears, enter the connection string found in the ActiveTunes.SecurityProvider.UnitTests project file App.config:

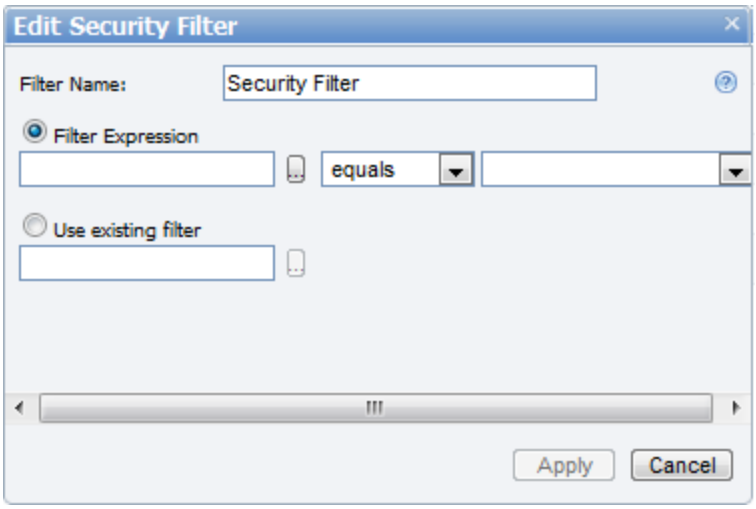
ConnectionString
`Data Source=C:\ActiveReports 10 Server\SDK\ActiveTunes.SecurityProvider\ActiveTunes.sqlite`

7. Click the **Apply changes** button.

Now, when you edit the ActiveTunes sample model, you can apply either the EmployeeID or CustomerID filter to any entity.

To add row-level security to the ActiveTunes sample model

1. On the Administrator Dashboard, in the Administration section, click **Models**.
2. In the list of models that appears, next to the **ActiveTunes (Sample)** model, click the **Edit** command.
3. In the model editor that appears, from the **Entities** list to the left, click the entity that you want to filter.
4. In the workspace at the center, next to the **Security Filter** property, click the **Add** command.
5. In the Edit Security Filter dialog that appears, in the box next to **Filter Name**, enter a name for your filter.
6. Select **Filter Expression**, and click the ellipsis button next to the box under that label.



7. In the Select Attribute dialog that appears, select the attribute to which you want to apply the filter condition, and click **OK**.
8. To the right of the **equals** option, click the drop-down arrow and select **UserContext.CustomerID** or **UserContext.EmployeeID**.
9. Click **Apply** to add the filter to the entity.

Walkthroughs

Learn to use ActiveReports Server controls and API in Visual Studio.

Using the ReportList Web Control
: This topic explains how to use the ReportList control on a Web Form.

Uploading a Code-Based Section Report
: This topic explains how to upload a code-based section report by the UploadResource method.

Rendering a Code-Based Section Report
: This topic explains how to render a code-based section report by the RenderReport method.

Downloading a Code-Based Section Report
: This topic explains how to download a code-based section report by the Download method.

UserContext for Multi-Tenancy
: The topic explains how to provide specific report data to the user logged onto the Server by setting the UserContext attribute in ActiveReports report.

Using the ReportList Web Control

ActiveReportsServer allows you to retrieve the reports list from ActiveReportsServer site using the ReportList control. It also allows you to incorporate the feature of exporting each report into PDF, Word or Excel format in the application developed by the user. This topic explains how to retrieve reports from a site by entering credentials in the log-in screen or through the Web.config file.

To create a Log In page

Note: The following steps are not required if you are adding credentials in the Web.config file.

1. In Visual Studio, create a new C# ASP.NET Web Site.
2. Open the Design view of your default Web Form and from the Standard section of the Toolbox, drag the following controls and drop them onto the body section of Web Form, setting their properties as in the table below.

Controls for the Log In page

Control	Text Property
Label	User Name:
TextBox	?
Label	Password:
TextBox	?
Button	Log In

3. Select **Add New Item** from the Website menu of Visual Studio.
4. In the **Add New Item** dialog that appears, select **Class**, enter SessionStorage.vb/SessionStorage.cs in the name field and click **Add**.
5. Replace the SessionStorage.vb/SessionStorage.cs code with the following code:

Visual Basic

Visual Basic (Paste INSIDE SessionStorage.vb)

```
Imports System
Imports System.Web

Namespace App_Code
    Public Class SessionStorage

        Private Const SecurityTokenKey As String = "{DEBDB016-040A-48f4-B568-
897E6D410919}"

        Public Shared Property SecurityToken() As String
            Get
                Return TryCast(HttpContext.Current.Session(SecurityTokenKey),
String)
            End Get
            Set(value As String)
                HttpContext.Current.Session(SecurityTokenKey) = value
            End Set
        End Property
    End Class
End Namespace
```

C#

C# (Paste INSIDE SessionStorage.cs)

```
using System;
using System.Web;

namespace App_Code
{
    public static class SessionStorage
    {
        private const string SecurityTokenKey = "{DEBDB016-040A-48f4-B568-
897E6D410919}";

        public static string SecurityToken
        {
            get { return HttpContext.Current.Session[SecurityTokenKey] as string; }
            set { HttpContext.Current.Session[SecurityTokenKey] = value; }
        }
    }
}
```

6. At the top of Default.aspx.vb/Default.aspx.cs, with the other Imports/using directives, paste the following code to use Servicing namespace and App_Code namespace.

▶ Visual Basic

Visual Basic (Paste at the top of file)

```
Imports ActiveReports.Server.ReportControls.Servicing
Imports App_Code
```

▶ C#

C# (Paste at the top of file)

```
using ActiveReports.Server.ReportControls.Servicing;
using App_Code;
```

> ⚠ **Note:** An error will be thrown in the following line until you add **ActiveReports.Server.ReportControls.dll** reference using the steps mentioned in **To add the ReportList control to a Web Form**.
>
> using ActiveReports.Server.ReportControls.Servicing;

7. Double-click the button to create button click event. The *.cs file for the page appears with the cursor inside the button click event.
8. In the button click event, paste the following code to get the security token from the user's credentials entered in the log-in screen.

▶ Visual Basic

Visual Basic (Paste INSIDE the button click event)

```
Dim username = TextBox1.Text
Dim password = TextBox2.Text

Dim reportService = New ReportServiceProxy()
```

```
SessionStorage.SecurityToken = Nothing

Dim securityToken = reportService.Login(username, password, Nothing, False)

SessionStorage.SecurityToken = securityToken
Server.Transfer("Default2.aspx")
```

▶ C#

C# (Paste INSIDE the button click event)

```
var username = TextBox1.Text;
var password = TextBox2.Text;

var reportService = new ReportServiceProxy();
SessionStorage.SecurityToken = null;

var securityToken = reportService.Login(username, password, null, false);

SessionStorage.SecurityToken = securityToken;
Server.Transfer("Default2.aspx");
```

Note: An error will be thrown in the following line(s) until you add **ActiveReports.Server.ReportControls.dll** reference using the steps mentioned in **To add the ReportList control to a Web Form**.

Dim reportService = New ReportServiceProxy()
OR
var reportService = new ReportServiceProxy();

9. In the Web.config file, add the following code to enable the compatibility mode.

▶ To enable the ASP.NET compatibility mode

ASP.NET code. Paste BETWEEN the <configSections> tab and </configSections>

```
<system.serviceModel>
    <serviceHostingEnvironment aspNetCompatibilityEnabled="true" />
</system.serviceModel>
```

To create .svc file for WCF service

Note: These steps are required when using ASP.NET website. You do not need to create .svc file manually in case of ASP.NET Web application, as service file is added automatically when a ReportList control is dropped upon the web form.

1. Select **Add New Item** from the **Website** menu of Visual Studio.
2. In the **Add New Item** dialog that appears, select **WCF**, enter **ReportService.svc** in the name field and click **Add**.
3. Replace the ReportService.svc code with the following code:

▶ To create .svc file for WCF service

ASP.NET code

```
<%@ ServiceHost
Service="ActiveReports.Server.ReportControls.Servicing.ReportServiceProxy"
Factory="ActiveReports.Server.ReportControls.Servicing.ReportServiceHostFactory" %>
```

4. In the App_Code file, the following files are added by default. You need to delete these files as they are not required.
   - IReprotService.cs
   - ReprotService.cs

To add the ReportList control to a Web Form

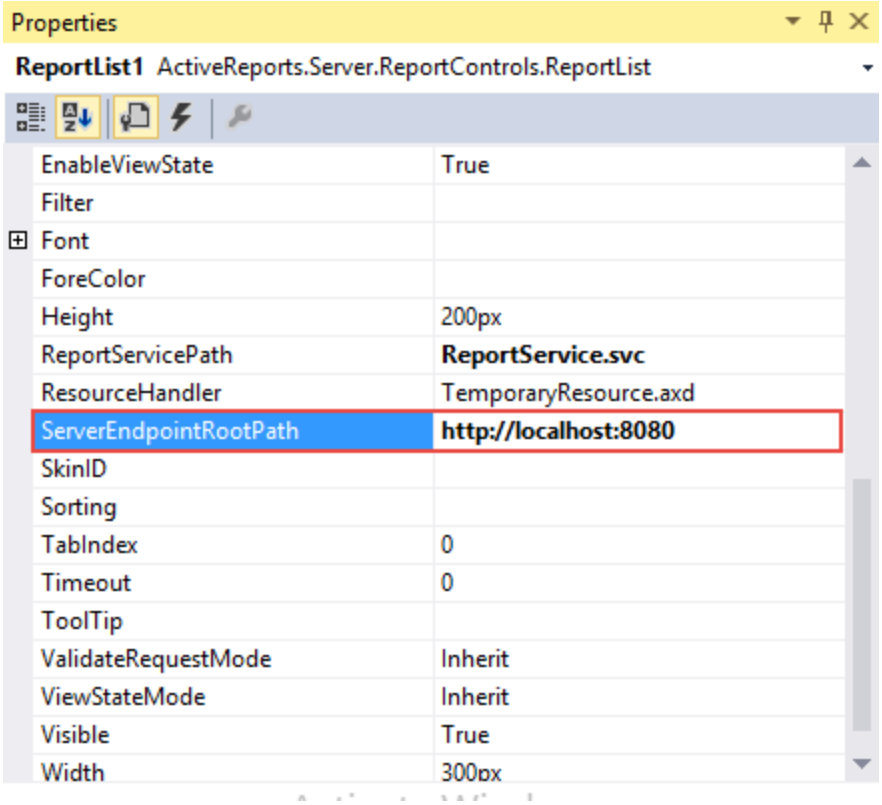
1. Add a second Web Form to your project, **Default2.aspx**. (Add to the default Web form if you have not created the Log-in page)
2. Open the Design view of your Web Form and from the toolbox, drag and drop the ReportList control onto the body section of the Web Form.

   If you need to add the control to your toolbox, drop down these steps.

   1. Right-click in the General tab and select **Choose items**.
   2. In the Choose Toolbox Items dialog that appears, the .NET Framework Components tab is selected by default. Click the **Namespace** column header to sort by namespace.
   3. The ActiveReports.Server.ReportControls namespace is at or near the top of the list. Select the checkbox next to **ReportList** and click **OK**.
   4. The ReportList control appears in your Visual Studio toolbox.
3. On the message box that appears, click **Yes** to automatically add .svc file to your Web site to give the report list access to proxy services.

   > **Note:** The.svc file is not added automatically when using ASP.NET Website. You need to manually add **ReportService.svc** file by using the above steps mentioned in **To create .svc file for WCF service**.

4. In the Properties window, set the **ServerEndpointRootPath** property to the URL that you want to use as the basis for all of the other URLs in your site, which can then be set as relative URLs. (Replace localhost:8080 with the site port where you installed ActiveReports Server)



> **Note:** The **ReportServicePath** property is not automatically set when using ASP.NET website. You need to manually set the file path of ReportService.svc.

5. You can resize the control by changing the **Width** property in the Properties grid.

To display the report list on the ReportList control

You can specify the security token and ActiveReports 10 Server host using code in a Global Application Class.

1. From the Visual Studio **Website** menu, select **Add New Item**.
2. In the Add New Item dialog that appears, select **Global Application** Class and click **Add**.
3. Below the Application Language tag at the top of Global.asax, import the Servicing namespace so that you can use the ReportServiceProxy with a directive like the following:

To import Servicing namespace

ASP.NET code. (Paste on the line BELOW the Application Language line)

```
<%@ Import Namespace="ActiveReports.Server.ReportControls.Servicing" %>
<%@ Import Namespace="App_Code" %>
```

> **Note:** You need to add directives in Global.asax.vb/Global.asax.cs file when using ASP.NET Web application.
>
> Visual Basic (Paste at the top of Global.asax.vb file)
>
> ```
> Imports ActiveReports.Server.ReportControls.Servicing
> Imports App_Code
> ```
>
> C# (Paste at the top of Global.asax.cs file)

```
    using ActiveReports.Server.ReportControls.Servicing;
    using App_Code;
```

4. Below the Application Start event, create the ResolveRemoteEndpoint event using code like the following, but with your address and security token:

▶ Visual Basic

Visual Basic (Paste in the Global.asax file AFTER the Application_Start event)

```
Private Shared Sub ResolveRemoteEndpoint(remoteEndpoint As RemoteEndpoint)
    remoteEndpoint.Address = "http://localhost:8080"
    remoteEndpoint.SecurityToken = SessionStorage.SecurityToken
End Sub
```

▶ C#

C# (Paste in the Global.asax file AFTER the Application_Start event)

```
static void ResolveRemoteEndpoint(RemoteEndpoint remoteEndpoint)
{
    remoteEndpoint.Address = "http://localhost:8080";
    remoteEndpoint.SecurityToken = SessionStorage.SecurityToken;
}
```

5. Provide a handler for the ResolveRemoteEndPoint event using code like the following in the Application Start event:

▶ To provide a handler for the event

C# code. Paste INSIDE the Application_Start event.

```
ReportServiceProxy.ResolveRemoteEndpoint += ResolveRemoteEndpoint;
```

▶ You can also specify the user name, password, and ActiveReports 10 Server host in your web application's web.config file.

Note: Use the following steps to add credentials in the Web.config file without creating the log-in page.

In the web.config file, add credentials using code like the following.

ASP.NET code. Paste in the web.config file AFTER the </configSections> tag.

```
<configSections>
    <sectionGroup name="activereports.server">
      <section name="reportServiceProxy"
type="ActiveReports.Server.ReportControls.Configuration.ActiveReportsServerSection,
      ActiveReports.Server.ReportControls, Version=9.x.xxxx.1, Culture=neutral,
PublicKeyToken=d557f2f30a260da2" allowDefinition="Everywhere" />
    </sectionGroup>
</configSections>
<activereports.server>
```

```
    <reportServiceProxy remoteReportServicePath="http://localhost:8080/"
username="MyUserName" password="MyPassword" />
</activereports.server>
```

Note: Replace "Version=9.*x.xxxx*.1" with the version number of the installed ActiveReports 10 Server.

At run time, the ReportList control retrieves all of the reports from ActiveReports 10 Server, and provides buttons to export each to PDF, Word, and ExcelNotw.

Uploading a Code-Based Section Report

You can upload all types of developer-created ActiveReports to the server using the UploadResource method. This topic explains how to upload a code-based section report.

1. From the Visual Studio **File** menu, select **New Project**.
2. In the **New Project** dialog that appears, select Visual C# or Visual Basic from the template list, click Windows and select **Console Application**.
3. Rename the project to **UploadReport** and click **OK**.

   Note: The target framework of the project must be set to .NET Framework 4.5 or above.

4. From the Visual Studio **Project** menu, select **Add Service Reference**.
5. In the **Add Service Reference** dialog that appears, enter the following address in the **Address** field.

   Paste in the Address box, replacing 8080 with the site port where you installed ActiveReports 10 Server.

   http://localhost:8080/ReportService.svc

6. Click **Go**, and when the ReportService appears in the Services pane, you can expand it and select the service interface to reveal its available operations in the pane to the right.
7. Rename the Namespace that you want to use to **ReportService** and click **OK**. The reference appears in the Solution Explorer.
8. Open the app.config file, in the address attribute of endpoint element, check if the service address set in step 5 is correct. Replace the address if it is not correct.

   confirm the value of address attribute in the endpoint element of app.config file

   ```
   <client>
       <endpoint address="http://localhost:8080/ReportService.svc"
           binding="wsHttpBinding" bindingConfiguration="WSHttpBinding_IReportService"
           contract="ReportService.IReportService" name="WSHttpBinding_IReportService" />
   </client>
   ```

   Note: The service address may not get update automatically by the server configuration of the installed ActiveReports Server.

9. In Program.cs or Module1.vb, add the following statements to the using/Imports statements at the top of the code.

   Visual Basic

   Visual Basic code (Add to the list of Imports statements at the top of the code.)

   ```
   Imports System.IO
   Imports UploadReport.ReportService
   ```

   C#

   C# code(Add to the list of using statements at the top of the code.)

   ```
   using System.IO;
   ```

```
using UploadReport.ReportService;
```

10. In Program.cs or Module1.vb, add the following code into the Main method.

Visual Basic

Visual Basic code (Paste the following code into the Main method of the Module1 module declaration)

```
Dim filepath As String = "C:\section_report.rpx"
Dim serverUserName = "USER"
Dim serverUserPwd = "PASS"
Dim uploadOptions = New UploadOptions() With { _
    .Overwrite = True _
}
Dim reportService = New ReportServiceClient("WSHttpBinding_IReportService")
Dim securityToken = reportService.Login(serverUserName, serverUserPwd, Nothing, True)

' Name: Report name to display after the upload.
' ReportType: Type of the report to upload.
' ClassName: Class name. Specify while uploading code-based section report.
' AssemblyResourceId: Assembly Resource ID. Specify while uploading code-based section
report.
Dim reportDescription As New ReportDescription()
With reportDescription
    .Name = "Test Report"
    .ReportType = ReportType.SectionReport
    .ClassName = Nothing
    .AssemblyResourceId = Nothing
End With
reportService.UploadResource(securityToken, ReportDescription,
Convert.ToBase64String(File.ReadAllBytes(filepath)), UploadOptions)
```

C#

C# code (Paste the following code into the Main method of the Program class declaration)

```
string filepath = @"C:\section_report.rpx";
var serverUserName = "USER";
var serverUserPwd = "PASS";
var uploadOptions = new UploadOptions { Overwrite = true };
var reportService = new ReportServiceClient("WSHttpBinding_IReportService");
var securityToken = reportService.Login(serverUserName, serverUserPwd, null, true);


// Name: Report name to display after the upload.
// ReportType: Type of the report to upload.
// ClassName: Class name. Specify while uploading code-based section report.
// AssemblyResourceId: Assembly Resource ID. Specify while uploading code-based section
report.
var reportDescription = new ReportDescription()
{
    Name = "Test Report",
    ReportType = ReportType.SectionReport,
    ClassName = null,
    AssemblyResourceId = null
```

```
};
reportService.UploadResource(securityToken, reportDescription,
Convert.ToBase64String(File.ReadAllBytes(filepath)), uploadOptions);
```

11. Run the project.

Rendering a Code-Based Section Report

You can render the reports saved in ActiveReports Server to PDF or Excel format using the RenderReport method. This topic explains how to render a code-based section report (XML) to PDF format.

> **Note:** You can also render a code-based section report or a page report to PDF format using the same steps.

1. From the Visual Studio **File** menu, select **New Project**.
2. In the **New Project** dialog that appears, select Visual C# or Visual Basic from the template list, click Windows and select **Console Application**.
3. Rename the project to **RenderReport** and click **OK**.

   > **Note**: The target framework of the project must be set to .NET Framework 4.5 or above.

4. From the Visual Studio **Project** menu, select **Add Service Reference**.
5. In the **Add Service Reference** dialog that appears, enter the following address in the **Address** field.

Paste in the Address box, replacing 8080 with the site port where you installed ActiveReports 10 Server.
http://localhost:8080/ReportService.svc

6. Click **Go**, and when the ReportService appears in the Services pane, you can expand it and select the service interface to reveal its available operations in the pane to the right.
7. Rename the Namespace that you want to use to **ReportService** and click **OK**. The reference appears in the Solution Explorer.
8. Open the app.config file, in the address attribute of endpoint element, check if the service address set in step 5 is correct. Replace the address if it is not correct.

   confirm the value of address attribute in the endpoint element of app.config file

   ```
   <client>
       <endpoint address="http://localhost:8080/ReportService.svc"
           binding="wsHttpBinding" bindingConfiguration="WSHttpBinding_IReportService"
           contract="ReportService.IReportService" name="WSHttpBinding_IReportService"
   />
   </client>
   ```

   > **Note:** The service address may not get update automatically by the server configuration of the installed ActiveReports Server.

9. In Program.cs or Module1.vb, add the following statements to the using/Imports statements at the top of the code.

   Visual Basic

   Visual Basic code (Add to the list of Imports statements at the top of the code.)

   ```
   Imports System.IO
   Imports System.Net
   Imports RenderReport.ReportService
   ```

▶ C#

C# code(Add to the list of using statements at the top of the code.)

```
using System.IO;
using System.Net;
using RenderReport.ReportService;
```

10. In Program.cs or Module1.vb, add the following code into the Main method.

▶ Visual Basic

Visual Basic code (Paste the following code into the Main method of the Module1 module declaration)

```
Dim serverUserName = "USER"
Dim serverUserPwd = "PASS"
Dim reportService = New ReportServiceClient("WSHttpBinding_IReportService")
Dim securityToken = reportService.Login(serverUserName, serverUserPwd, Nothing,
True)
' Get the report list from internal storage.
Dim description = reportService.[Select](securityToken, New Query())
' Get the report description of output target.
Dim TargetRepDescription As New ReportDescription()
For Each des As ReportDescription In description
    If des.Name = "Payment Slip" Then
        TargetRepDescription = des
        Exit For
    End If
Next
' Output into PDF
Dim result = reportService.RenderReport(securityToken, TargetRepDescription, New
RenderOptions() With {.Extension = "PDF"})
Dim info As RequestInfo
Do
    info = reportService.GetRequestStatus(securityToken,
Result.Info.RequestId).Info
Loop While info.State = RequestState.Pending OrElse info.State =
RequestState.Running
If info.State = RequestState.Accomplished Then
    Using client = New WebClient()
        client.DownloadFile("http://localhost:8080/TemporaryResource.axd/" +
info.PrimaryUrl, "C:\work\test.pdf")
    End Using
End If
```

▶ C#

C# code (Paste the following code into the Main method of the Program class declaration)

```
var serverUserName = "USER";
```

```
var serverUserPwd = "PASS";
var reportService = new ReportServiceClient("WSHttpBinding_IReportService");
var securityToken = reportService.Login(serverUserName, serverUserPwd, null, true);
// Get the report list from internal storage.
var description = reportService.Select(securityToken, new Query());
// Get the report description of output target.
ReportDescription TargetRepDescription = new ReportDescription();
foreach (var des in description)
{
    if (des.Name == "Payment Slip")
    {
        TargetRepDescription = des;
        break;
    }
}
// Output into PDF
var result = reportService.RenderReport(securityToken, TargetRepDescription, new
RenderOptions { Extension = "PDF" });
RequestInfo info;
do
{
    info = reportService.GetRequestStatus(securityToken,
result.Info.RequestId).Info;
} while (info.State == RequestState.Pending || info.State == RequestState.Running);
if (info.State == RequestState.Accomplished)
{
    using (var client = new WebClient())
    {
        client.DownloadFile("http://localhost:8080/TemporaryResource.axd/" +
info.PrimaryUrl, @"C:\work\test.pdf");
    }
}
```

11. Run the project.

Downloading a Code-Based Section Report

You can download the reports uploaded in the ActiveReports Server using the Download method and can save it into report definition format. This topic explains how to download a code-based section report (XML) to PDF format.

> **Note:** You can also download a page report or an RDL report using the same steps.

1. From the Visual Studio **File** menu, select **New Project**.
2. In the **New Project** dialog that appears, select Visual C# or Visual Basic from the template list, click Windows and select **Console Application**.
3. Rename the project to **RenderReport** and click **OK**.

   > **Note**: The target framework of the project must be set to .NET Framework 4.5 or above.

4. From the Visual Studio **Project** menu, select **Add Service Reference**.
5. In the **Add Service Reference** dialog that appears, enter the following address in the **Address** field.

Paste in the Address box, replacing 8080 with the site port where you installed ActiveReports Server.
http://localhost:8080/ReportService.svc

6. Click **Go**, and when the ReportService appears in the Services pane, you can expand it and select the service interface to reveal its available operations in the pane to the right.
7. Rename the Namespace that you want to use to **ReportService** and click **OK**. The reference appears in the Solution Explorer.
8. Open the app.config file, in the address attribute of endpoint element, check if the service address set in step 5 is correct. Replace the address if it is not correct.

confirm the value of address attribute in the endpoint element of app.config file

```
<client>
    <endpoint address="http://localhost:8080/ReportService.svc"
        binding="wsHttpBinding" bindingConfiguration="WSHttpBinding_IReportService"
        contract="ReportService.IReportService" name="WSHttpBinding_IReportService"
/>
</client>
```

> **Note:** The service address may not get update automatically by the server configuration of the installed ActiveReports Server.

9. In the app.config file, add maxReceivedMessageSize attribute inside the binding element and set appropriate value. If you want to send or receive large amount of data in WCF service, you need to customize this value. In the following example, the value is set to 10 MB.

Add INSIDE the binding element of app.config file

```
<bindings>
    <wsHttpBinding>
        <binding name="WSHttpBinding_IReportService"
maxReceivedMessageSize="10000000">
            <security mode="None" />
                </binding>
        </wsHttpBinding>
</bindings>
```

10. In Program.cs or Module1.vb, add the following statements to the using/Imports statements at the top of the code.

▶ Visual Basic

Visual Basic code (Add to the list of Imports statements at the top of the code.)

```
Imports System.IO
Imports DownloadReport.ReportService
```

▶ C#

C# code(Add to the list of using statements at the top of the code.)

```
using System.IO;
using DownloadReport.ReportService;
```

11. In Program.cs or Module1.vb, add the following code into the Main method.

Visual Basic

Visual Basic code (Paste the following code into the Main method of the Module1 module declaration)

```
Dim serverUserName As String = "USER"
Dim serverUserPwd As String = "PASS"
Dim reportService As New ReportServiceClient("WSHttpBinding_IReportService")
' Get the security token using Login method
Dim securityToken As String = reportService.Login(serverUserName, serverUserPwd,
Nothing, True)
' Get the list information of the uploaded form
Dim query As New ReportService.Query()
Dim description As ReportDescription() = reportService.[Select](securityToken,
query)

' Get the report description of the output target
Dim TargetRepDescription As New ReportDescription
For Each des As ReportDescription In description
    If des.Name = "Payment Slip" Then
        TargetRepDescription = des
        Exit For
    End If
Next
' Get the encoded form data
Dim result As DataResult = reportService.Download(securityToken,
TargetRepDescription.Id)
' Return to byte array
Dim s As String = result.Data
Dim bs As Byte() = System.Convert.FromBase64String(s)

' Specify the target folder
Dim outFileName As String = "C:\work\PaymentSlip.rpx"
Dim outFile As New System.IO.FileStream(outFileName, System.IO.FileMode.Create,
System.IO.FileAccess.Write)
outFile.Write(bs, 0, bs.Length)
outFile.Close()
```

C#

C# code (Paste the following code into the Main method of the Program class declaration)

```
var serverUserName = "USER";
var serverUserPwd = "PASS";
var reportService = new ReportServiceClient("WSHttpBinding_IReportService");
var securityToken = reportService.Login(serverUserName, serverUserPwd, null, true);
// Get the report list from internal storage
var description = reportService.Select(securityToken, new Query());
```

```
// Get the report description of the output target
ReportDescription TargetRepDescription = new ReportDescription();
foreach (var des in description)
{
    if (des.Name == "Payment Slip")
    {
        TargetRepDescription = des;
        break;
    }
}
// Get the encoded form data
var result = reportService.Download(securityToken, TargetRepDescription.Id);
// Return to byte array
string s = result.Data;
byte[] bs = System.Convert.FromBase64String(s);
// Specify the target folder
string outFileName = @"C:\work\PaymentSlip.rpx";
System.IO.FileStream outFile = new System.IO.FileStream(outFileName,
System.IO.FileMode.Create, System.IO.FileAccess.Write);
outFile.Write(bs, 0, bs.Length);
outFile.Close();
```

12. Run the project.

UserContext in Multi-Tenant Reports

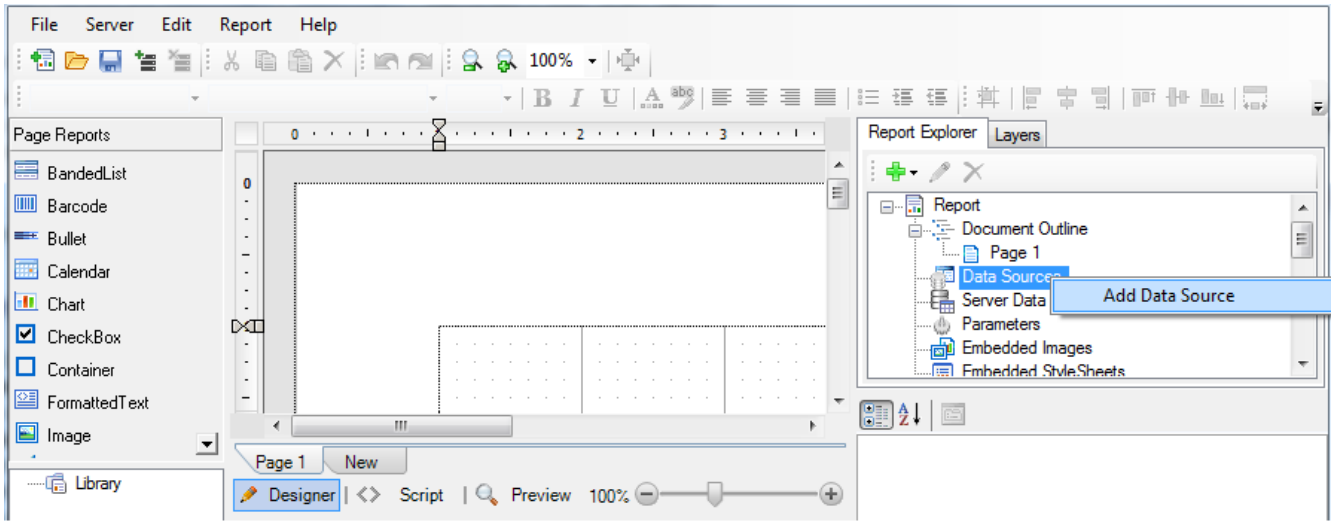
The topic demonstrates how to provide specific report data to a current user by setting the UserContext attribute in a report.

In the developer version of ActiveReports, you can set the UserContext attribute in Page, RDL, and Section report, then upload the report to ActiveReports Server. On the Server, the UserContext attribute filters report data to display only data specific to the current user. The UserContext attribute is particularly helpful when you have multiple tenants, logging into the same portal and you want to restrict the data available to each tenant.

> **Note:** The topic uses the Chinook database as a custom security provider. For information about the Chinook database, see https://chinookdatabase.codeplex.com.

To create a report for multiple tenants using the Chinook database

1. In the ActiveReports Designer, under **File**, click **New**.
2. In the **Create New Report** wizard, select Page Report and click **OK**.
3. To connect to a data source, in the Report Explorer, right-click the Data Sources node and select the **Add Data Source** option.

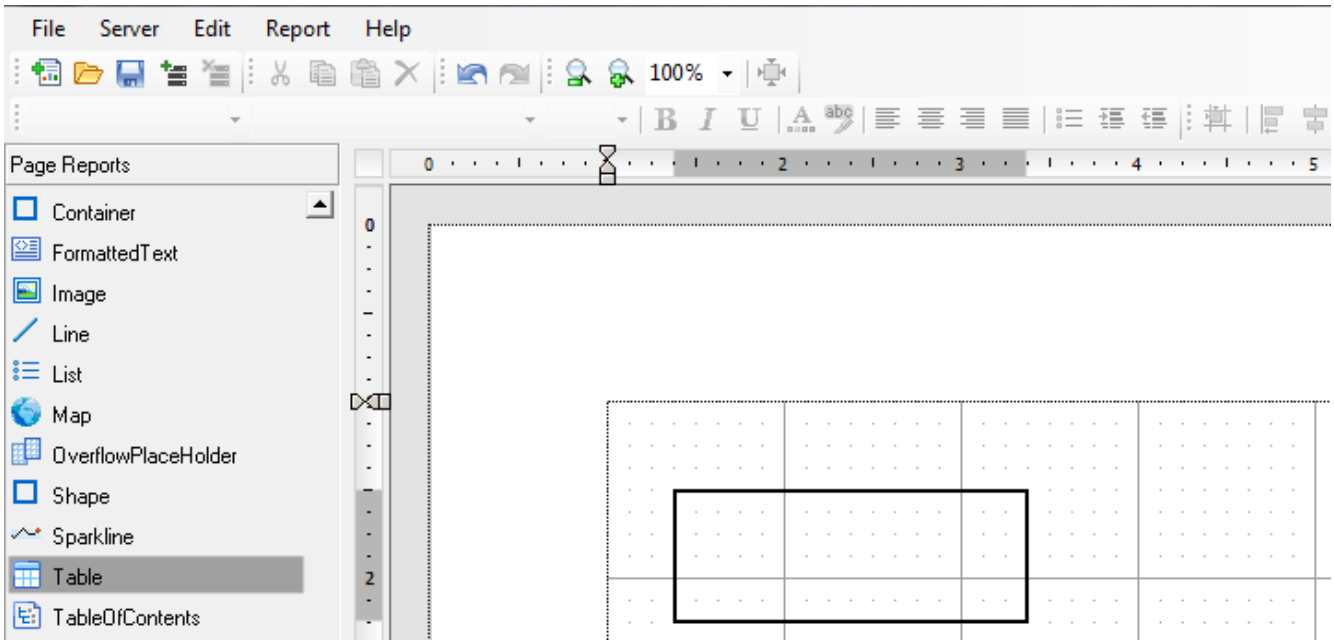


4. In the **Report Data Source** dialog, create a connection to the Chinook database. For information, see Connect to a Data Source.
5. To add a dataset, in the Report Explorer, right-click the Data Sources node and select the **Add Data Set** option.
   - In the **DataSet** dialog, on the **General** page, enter **Invoice** as the name of the dataset.
   - On the **Query** page of this dialog, in the **Query** field enter the following SQL query.

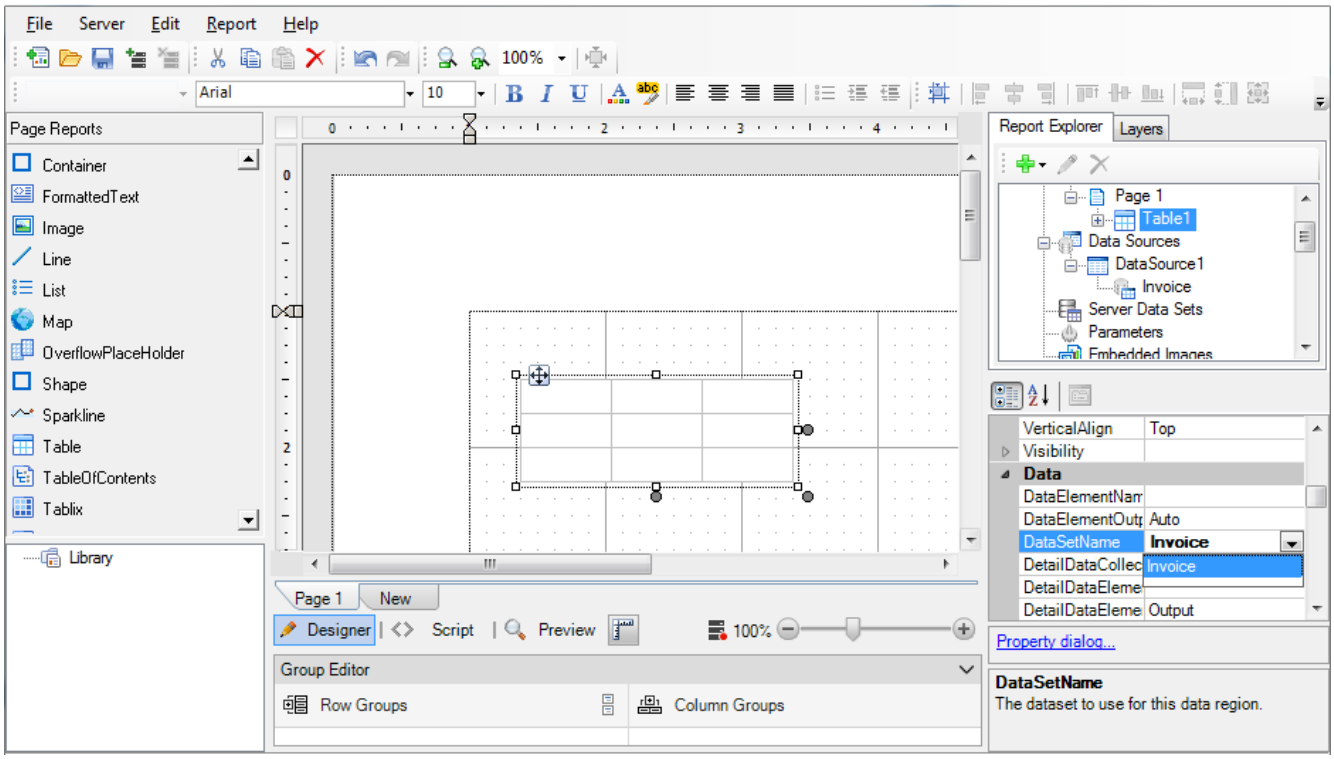
     SQL Query

     ```
     Select * from Invoice where CustomerID=@Invoice
     ```

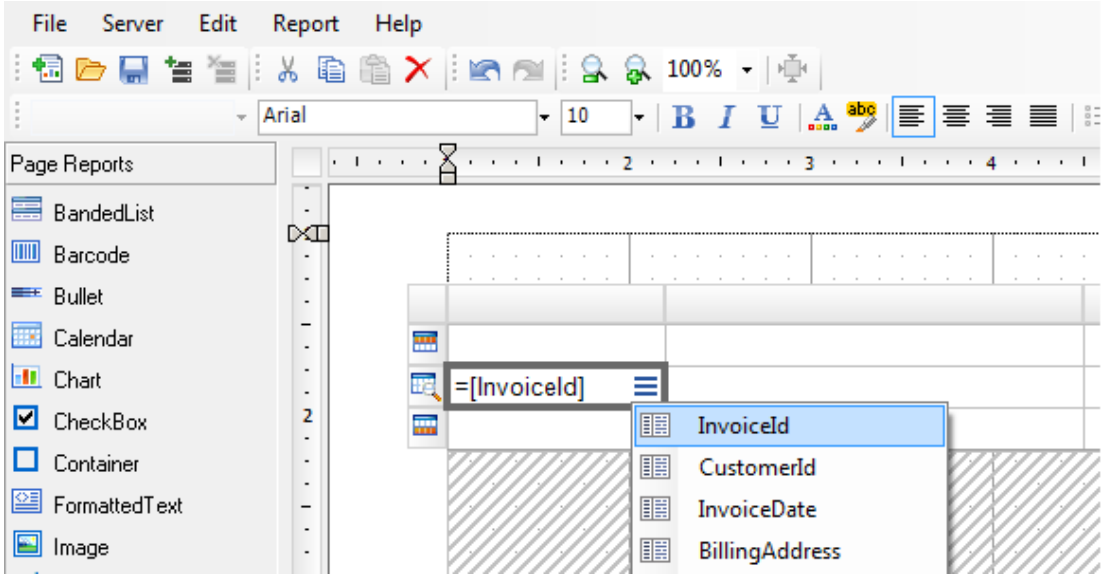
   - On the **Parameters** page of this dialog, click the **Add** button.
   - In the **Name** field, enter **Invoice** and set its value to **=[@Customer]**.
   - Click **OK** to close the dialog.
6. From the Toolbox, drag the **Table** control onto the Design surface.



7. Select the **Table** control and in the **Properties** window, set the DataSetName property to **Invoice**.



8. In the Table's detail row, click the adorner in each cell to show a list of available fields from the data set and add the following fields to the cells.



Cell	Field
Left Cell	InvoiceID
Middle Cell	InvoiceDate
Right Cell	Total

To set the UserContext attribute

The UserContext attribute is set in the report parameter that will retrieve specific information for the user that is logged into the server. See Add Parameters for detailed information about parameters in a page report.

1. In the Report Explorer, right-click the Parameters node and select **Add Parameter**. The **Report Parameters** dialog appears.
2. On the **General** tab, set the Name to **Customer**, set the Data type to **Integer** and select the **Hidden** box.
3. On the **Available Values** tab, select **Non-queried** and in the **Value** field, enter **= Code.UserContext.GetValue("CustomerID")**.
4. Click **OK**.

To format the table (optional)

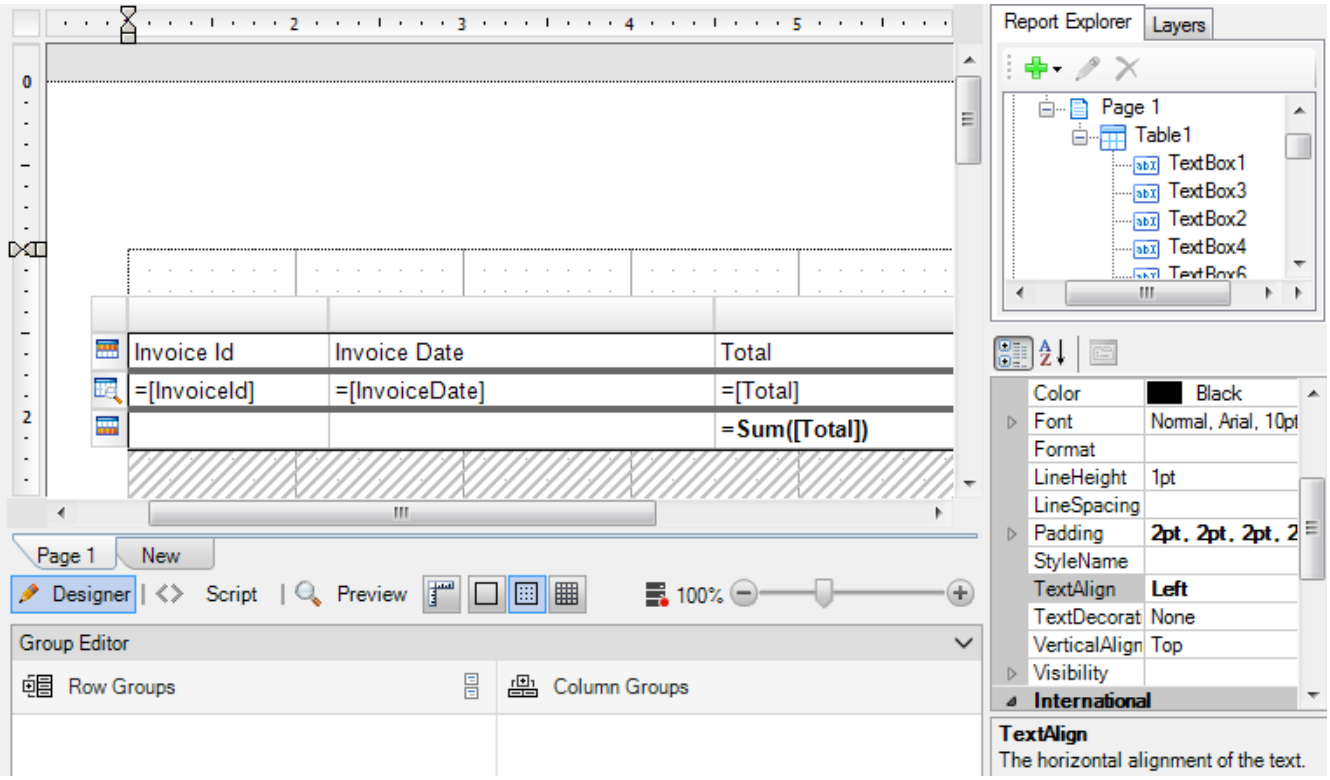
There are many ways you can format the table that you have added to the report in ActiveReports Designer. See Table in the ActiveReports User Guide for more information on the formatting possibilities.

Invoice Id	Invoice Date	Total
98	Thursday, March 11, 2010	$3.98
121	Sunday, June 13, 2010	$3.96
143	Wednesday, September 15, 2010	$5.94
195	Friday, May 06, 2011	$0.99
316	Saturday, October 27, 2012	$1.98
327	Friday, December 07, 2012	$13.86
382	Wednesday, August 07, 2013	$8.91
		$39.62

1. Select the **Table** control and go to the Properties window to set the following properties.

FixedSize	6.5in, 7in
Location	0in, 0.5in
RepeatHeaderOnNewPage	True
Size	6.4in, 0.7in

2. Select the Table's detail row and in the Properties window set the **TextAlign** property to **Left** and the **BorderStyle** property to **Solid**.

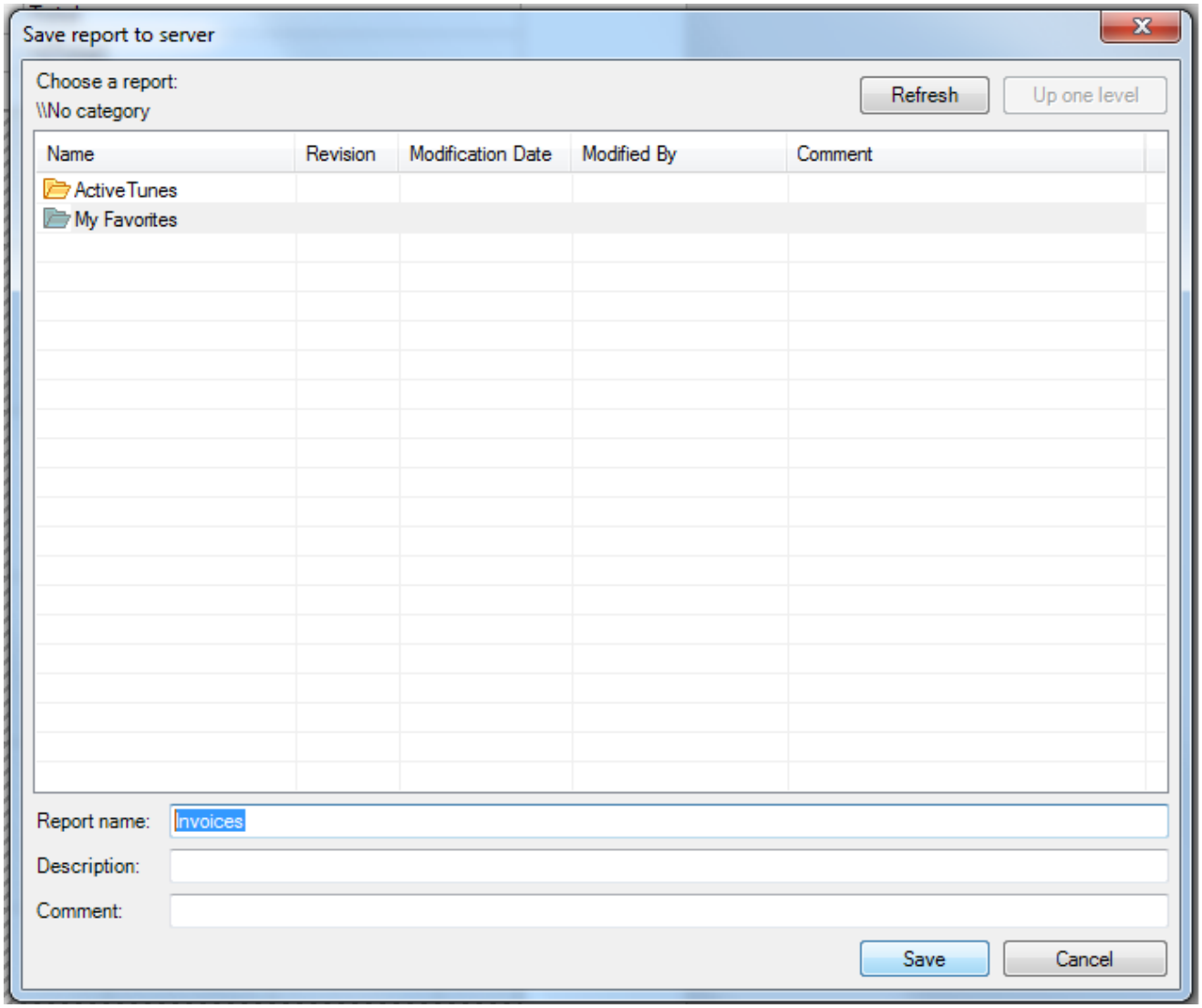


3. Select the middle cell (InvoiceDate) of the details row and in the Properties window set the **Format** property to **d**.
4. Select the right cell of the Table Footer and go to the Properties window to set the following properties.

Format	c
Value	=Sum(Fields!Total.Value)

To upload the Page report to ActiveReports Server

1. Connect to ActiveReports Server as described in Connecting to ActiveReports Server.
2. Select the directory in the folders tree to choose the location in which to save the report on the server.
3. Enter the report name in the Report name field as **Invoices** and click **Save**.



To set up the custom security provider

This topic uses the sample custom security provider project installed with ActiveReports Server. By default, this sample is in the following folder.

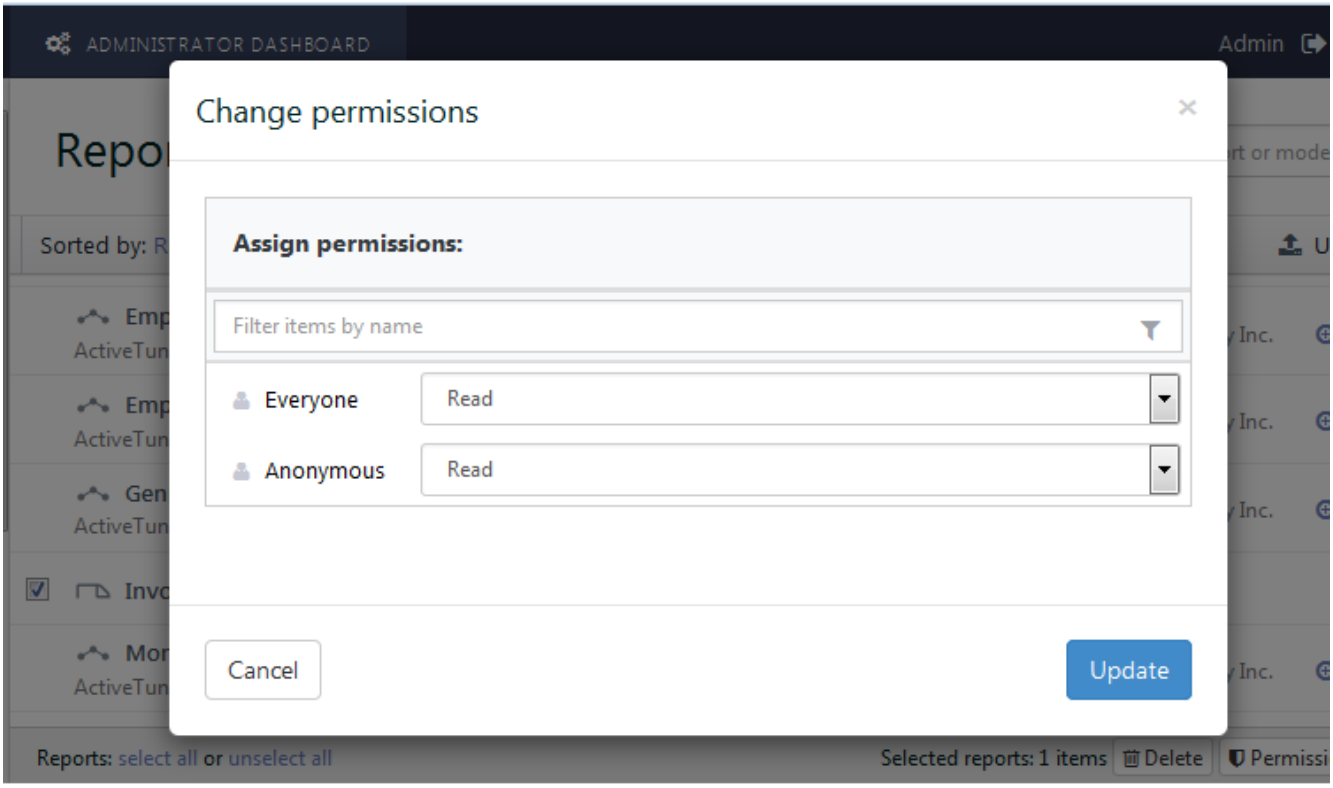
C:\ActiveReports 10 Server\SDK\ActiveTunes.SecurityProvider

For detailed steps on how to set up the custom security provider, see Deploy the Custom Security Provider for Debugging.

To set permissions for the report on ActiveReports Server

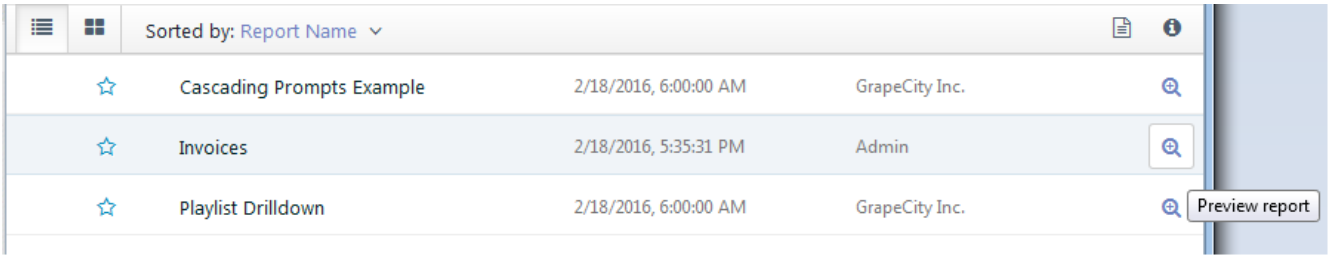
On ActiveReports Server, the report permissions have to be modified so that its multiple users can read it.

See To Manage Permissions to a Report on how to grant the Read permissions on the report.



To preview the report on ActiveReports Server

1. Open the ActiveReports 10 Server **Report Portal**.
2. On the Login screen that appears, in the **User Name** field, enter a customer ID, for example, luisg@embraer.com.br.
3. In the **Password** field, enter 1 (for the luisg@embraer.com.br customer ID).
4. Click **Login**.
5. In the ActiveReports Server **Report Portal (on-line documentation)** that appears, click the **Preview report** button for the **Invoices** report to see all invoices for the logged-in user.



Troubleshooting

Here you can find solutions to issues that you may encounter while working with ActiveReports Server. Click a description below to view the symptoms, cause, and solution.

▸ When trying to open the Report Portal page, you get the Server Error "Could not connect to endpoint DesignerService.svc on the server."

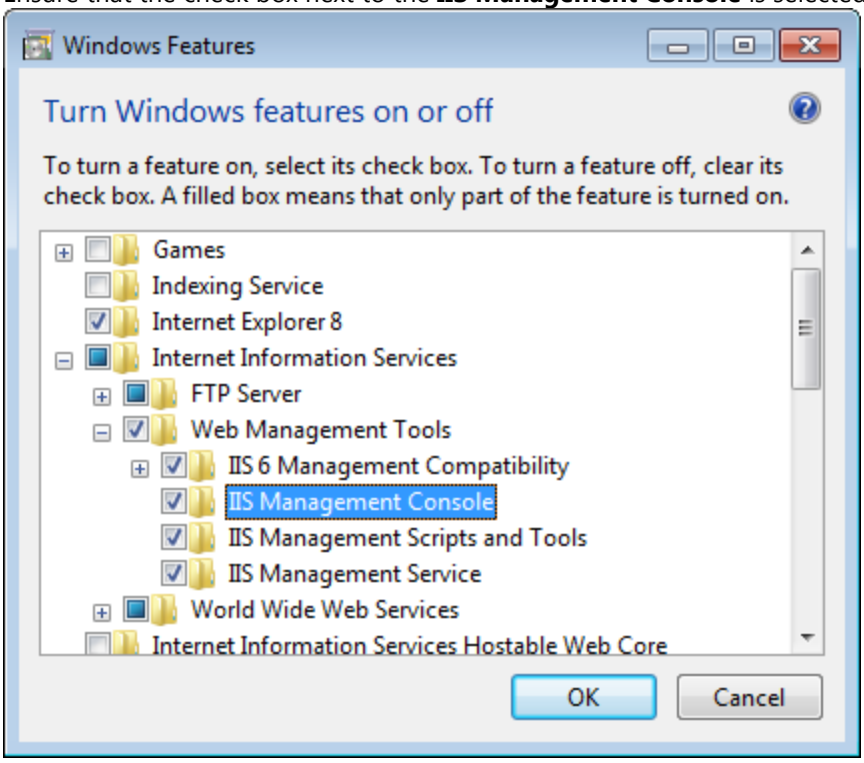
Symptoms: The Report Portal page will not open. Instead, you get a Server Error that reads: "Could not connect to

endpoint DesignerService.svc on the server. Please verify connectivity to the server and check the server configuration. HTTP request error"

Causes: The IIS Management Console is turned off by default, or WCF is not configured on your machine.

Solution 1: Turn on the IIS Management Console.

1. Open Control Panel and select **Programs and Features**.
2. On the left, select **Turn Windows features on or off**.
3. In the Windows Features window that appears, expand the **Internet Information Services** node, then the **Web Management Tools** node.
4. Ensure that the check box next to the **IIS Management Console** is selected and click **OK**.



Solution 2: Configure WCF on your machine.

1. From the Start menu, open a command prompt (cmd.exe).
2. In the command window, enter **cd %SystemRoot%\Microsoft.Net\Framework\v3.0\Windows Communication Foundation** to change to the directory where the ServiceModel Registration Tool is installed.
3. For Framework 4 or 4.5, enter the command: **ServiceModelReg.exe -ia** to run the tool to install all WCF and WF components. (For Framework 3 or 3.5, the command is: **ServiceModelReg.exe -ir**.)

Solution 3: Add features to WCF Services on the server.

1. From the Start menu, select **Server Manager**.
2. On the Manage menu, select **Add Roles and Features**.
3. In the Add Roles and Features window, select the **Features** tab.
4. Under Features, expand the .NET Framework 4.5 (Installed) node, then the **WCF Services (Installed)** node.
5. Under WCF Services (Installed), select the following three features:
    - HTTP Activation
    - Named Pipe Activation
    - TCP Activation

6. Click the **Install** button.

When trying to open the Administrator Dashboard or Report Portal page, you get HTTP Error 503 Service Unavailable.

Symptoms: The Administrator Dashboard and Report Portal pages will not open. Instead, you get HTTP Error 503.

Cause: The application pool for the service is stopped.

Solution: Restart the application pool.

1. Open IIS and expand your localhost node.
2. Select **Application Pools**.
3. In the Application Pools page that appears, find **ASP.NET v4.0 AR_AppPool** and check its Status.
4. If the Status is Stopped, right-click **ASP.NET v4.0 AR_AppPool** and select **Start**.

When trying to reinstall, you get the error: "Service 'ActiveReports Server' failed to start. Verify that you have sufficient privileges to start system services."

Symptoms: The service will not start when you try to reinstall. Instead, you get "Service 'ActiveReports 10 Server' failed to start."

Cause: The previous version did not uninstall completely.

Solution: Uninstall the remnants of the previous version before reinstalling.

1. In Windows Explorer, delete the directory C:\ActiveReports 10 Server.
2. Open IIS, expand the localhost, then the **Sites** node, and delete the **ActiveReports 10 Server** site.
3. Right-click the installer and select **Run as administrator** and follow the installation instructions.

The reports list is not displayed on the sample web page

Symptoms: The report list is not displayed on the sample web page.

Cause: When deploying samples, two access options in the Internet Information Services (IIS) settings are selected: "Enable anonymous access" and "Integrated Windows authentication."

Solution: Change the IIS settings to use only a single authentication scheme, "Enable anonymous access."

1. In the Control Panel, open **Administrative Tools**, then **Internet Information Services Manager**.
2. In the **Internet Information Services** window that appears, expand the tree view in the left pane until you see the **Web sites/sample** node.
3. Right-click **Web Sites/sample** to open the **Web Sites Properties** dialog.
4. On the Directory Security tab, in the **Authentication and access control** section, click **Edit**.

5. In the Authentication Methods dialog that appears, ensure that only one authentication scheme is selected in the corresponding checkbox.

6. Click **OK** to apply the changes, and **OK** again to close the dialog.
7. Back in Visual Studio, run the sample again.

Configuring Email

To ensure that error reporting and password reminders function properly for your site, you can configure the SMTP settings. The report server also uses these settings to connect with the e-mail server to deliver scheduled reports. You can configure e-mail SMTP settings from the Administrator Dashboard.

To configure e-mail from the Administrator Dashboard

1. From the Administrator Dashboard, at the bottom left under Configuration, click **SMTP Settings**.
2. In the SMTP Settings dialog that appears, enter values for your SMTP server.

   - **SMTP host** Enter the host address for your server, usually with syntax like "smtp.yourdomainname.com."
   - **SMTP port** Enter the port number for your server, for example, 587.
   - **Use SMTP authentication** Select this check box to add a user name and password.
   - **Sender e-mail** Enter the e-mail address to use as the return address for error reporting, password reminders, and scheduled report delivery.
   - **Sender name** Enter the name to use for the sender.
   - **Use SSL** Select this check box to enable the Secure Sockets Layer protocol.
3. When you have finished entering the values for your server, click **Apply changes**.

Troubleshooting

The table below explains the possible reasons for e-mail failure messages that you may get in the diagnostic report.

Cause	Error description
Incorrect host	System.Net.Mail.SmtpException: Failure sending mail. ----> System.Net.WebException: Unable to connect to the remote server ----> System.Net.Sockets.SocketException: A connection attempt failed

	because the connected party did not properly respond after a period of time, or established connection failed because connected host has failed to respond.
Incorrect port	System.Net.Mail.SmtpException: Failure sending mail. ----> System.Net.WebException: Unable to connect to the remote server ----> System.Net.Sockets.SocketException: A connection attempt failed because the connected party did not properly respond after a period of time, or established connection failed because connected host has failed to respond .
Incorrect login	System.Net.Mail.SmtpException: The SMTP server requires a secure connection or the client was not authenticated. The server response was: 5.7.1 Authentication required.
Incorrect password	System.Net.Mail.SmtpException: The SMTP server requires a secure connection or the client was not authenticated. The server response was: 5.7.1 Authentication required.
Incorrect sender e-mail	System.Net.Mail.SmtpFailedRecipientException: Mailbox name not allowed. The server response was: 5.7.1 <INCORRECT_EMAIL>: Sender address rejected: not owned by user CORRECT_EMAIL.
SSL is enabled but not required	System.Net.Mail.SmtpException: Error in processing. The server response was: 4.3.0 Error: queue file write error. OR System.Net.Mail.SmtpException: Failure sending mail. ----> System.IO.IOException: Unable to read data from the transport connection: net_io_connectionclosed.
SSL is disabled but required	System.Net.Mail.SmtpException: Failure sending mail. ----> System.IO.IOException: Unable to read data from the transport connection: net_io_connectionclosed.
SMTP authentication is enabled but not required	System.Net.Mail.SmtpException: The SMTP server requires a secure connection or the client was not authenticated. The server response was: 5.7.1 Authentication required.
SMTP authentication is disabled but required	System.Net.Mail.SmtpException: The SMTP server requires a secure connection or the client was not authenticated. The server response was: 5.7.1 Authentication required.

Support Options

We provide a number of technical support options to help you succeed with our products.

Free Community Support

Our community web site is available free of charge. Community members and ActiveReports Server team members provide answers to your most common questions.

In addition, we offer the following support options:

Annual Maintenance Package

Customers who purchase annual maintenance can benefit from the full suite of technical support services for one year.

These benefits include:

1. **All major releases.**
2. **All service releases and minor upgrades.**
3. **Email support**

   For e-mail support, please submit your request on this page of the ActiveReports Server web site.
   Please include the build number, the environment you are using, and a detailed description of the problem.
   This will help speed up the support process.

4. **Phone support**

   Support telephone number: (425) 952-6362
   Support hours: Monday through Friday, 9:00 am-5:00 pm US Pacific Standard Time (PST)
   When contacting support, please be prepared to provide your serial number, a complete description of the problem, as well as hardware and operating environment specifications.

Web Service Library

The Web Service Library contains documentation and syntax for the ActiveReports Server ReportService.

Service

ReportService
: Checks user credentials and creates a security token if the credentials are valid.

Objects

DataResult Object
: Represents an object containing the status of an IReportService operation.

EMailDistribution Object
: Defines e-mail options to use in distributing the report.

ExceptionDetail Object
: Defines detailed exception information for an IReportService request operation error.

ItemDescription Object
: Contains descriptive information about stored resources.

ItemDescriptionsResult Object
: Represents an object that is returned as a result of an Upload operation and contains an array of item descriptons.

Query Object
: Defines a query used to find reports in the Select operation.

RenderOptions Object
: Defines options for how to render a report.

ReportDescription Object
: Contains descriptive information about stored reports.

ReportParameter Object
: Defines a parameter for the report.

RequestInfo Object
: Defines status information for an IReportService request operation.

RequestResult Object
: Represents an object containing the status, or result contract, of an IReportService request operation.

Result Object
Represents an object that is returned as a result of an IReportService operation.
UploadOptions Object
Defines options for how to upload a resource.
UserCapabilities Object
Represents a user capabilities object that is contained in a result from an IReportService get user capabilities operation.
UserCapabilitiesResult Object
Represents a user capabilities result object that is returned as a result of an IReportService get user capabilities operation.

Enumerations

ModelPermission Enumeration
Specifies the byte value of the enumerated level of permissions granted for the model associated with the report.
ReportParameterDomain Enumeration
Specifies the byte value of the enumerated report parameter domain.
ReportType Enumeration
Specifies the byte value of the enumerated report type for all supported reports.
RequestState Enumeration
Specifies the state of an IReportService request operation for a RequestInfo object.

ReportService

Checks user credentials and creates a security token if the credentials are valid.

Syntax

C#
public interface IReportService

Parameters

username
A string representing the user name to validate.

password
A string representing the password for the specified user.

custom
A string representing the value or values to validate in addition to user name and password, if any.

isPersistent
A Boolean value that indicates whether the security token remains valid across sessions.

Return Value

A security token if user credentials are valid; otherwise, **null**.

IReportService Methods

	Name	Description
	CancelRequest Method	Cancels the specified request.
	Delete Method	Deletes the report specified in the ID parameter.
	Download Method	Downloads the specified report from storage.
	GetRequestStatus Method	Retrieves the status of the specified request.
	GetUserCapabilities Method	Retrieves the capabilities of the user with the specified security token.
	IsLoggedIn Method	Determines whether the security token is valid.
	Login Method	Checks user credentials and creates a security token if the credentials are valid.
	Logout Method	Recalls the security token.
	RenderReport Method	Renders the report specified in the description using the specified options.
	ResolveParameters Method	Populates the parameters for the specified report.
	Select Method	Retrieves an array of report descriptions matching the criterion defined by the specified query.
	SendReportEmail Method	Renders the specified report and sends it via email according to the specified options.
	Upload Method	Uploads the specified report to storage.
	UploadResource Method	Uploads the specified resource to storage.

CancelRequest Method

Cancels the specified request.

Syntax

C#
Result CancelRequest(string token, string id);

Parameters

token
The security token to use.

id
The identifier of the request to cancel.

Return Value

A Result object.

Delete Method

Deletes the report specified in the ID parameter.

Syntax

C#
DataResult Delete(string token, string id);

Parameters

token
The security token to use.

id
The identifier of the report to delete.

Return Value

A DataResult object containing the status of the delete operation.

Download Method

Downloads the specified report from storage.

Syntax

C#
DataResult Download(string token, string id);

Parameters

token
The security token to use.

id
The identifier of the report to download.

Return Value

A DataResult object containing the report downloaded by the operation.

GetRequestStatus Method

Retrieves the status of the specified request.

Syntax

C#
RequestResult GetRequestStatus(string token, string requestId);

Parameters

token
The security token to use.

id
The identifier of the request for which to return a status.

Return Value

A Result object.

GetUserCapabilities Method

Retrieves the capabilities of the user with the specified security token.

Syntax

C#
UserCapabilitiesResult GetUserCapabilities(string token);

Parameters

token
The security token to use.

Return Value

A UserCapabilitiesResult Object.

IsLoggedIn Method

Determines whether the security token is valid.

Syntax

C#

```
bool IsLoggedIn(string token);
```

Parameters

token
The security token to use.

Return Value

A Boolean value indicating whether the security token is valid.

Login Method

Checks user credentials and creates a security token if the credentials are valid.

Syntax

C#

```
string Login(string username, string password, string custom, bool isPersistent);
```

Parameters

username
A string value indicating the user name to validate.

password
A string value indicating the password for the specified user.

custom
Other string values to validate in addition to the user name and password, if any.

isPersistent
A Boolean value indicating whether the security token remains valid across sessions.

Return Value

A string value containing the security token if the user credentials are valid; otherwise, **null**.

Logout Method

Recalls the security token.

Syntax

C#

```
void Logout(string token);
```

Parameters

token
The security token to recall.

RenderReport Method

Renders the report specified in the description using the specified options.

Syntax

C#
RequestResult RenderReport(string token, ReportDescription description, RenderOptions options);

Parameters

token
The security token to use.

description
The ReportDescription object containing information about the report.

options
The RenderOptions object containing information about how to render the report.

Return Value

A RequestResult object.

ResolveParameters Method

Populates the parameters for the specified report.

Syntax

C#
RequestResult ResolveParameters(string token, ReportDescription description, RenderOptions options);

Parameters

token
The security token to use.

description
The ReportDescription object containing information about the report.

options

The RenderOptions object containing information about how to render the report.

Return Value

A RequestResult Object.

Select Method

Retrieves an array of report descriptions matching the criterion defined by the specified query.

Syntax

C#
ReportDescription[] Select(string token, Query query);

Parameters

token
The security token to use.

query
The Query object to use in finding reports to select.

Return Value

A array of ReportDescription objects.

SendReportEmail Method

Renders the specified report and sends it via email according to the specified options.

Syntax

C#
Result SendReportEmail(string token, ReportDescription description, RenderOptions options, EMailDistribution distribution);

Parameters

token
The security token to use.

description
The ReportDescription object containing information about the report.

options
The RenderOptions object containing information about how to render the report.

distribution
The EMailDistribution object containing information about how to distribute the report.

Return Value

A Result object.

Upload Method

Uploads the specified report to storage.

Syntax

C#
ItemDescriptionsResult Upload(string token, ReportDescription description, string data);

Parameters

token
The security token to use.

description
The ReportDescription object containing information about the report to upload.

data
The string value representing the report data encoded in base 64 digits to upload.

Return Value

An ItemDescriptionsResult object.

UploadResource Method

Uploads the specified resource, a valid .NET assembly containing compiled ActiveReports, to storage.

Syntax

C#
ItemDescriptionsResult UploadResource(string token, ItemDescription description, string data, UploadOptions options);

Parameters

token
The security token to use.

description
The ItemDescription object containing information about the resource to upload.

data
The string value representing the resource data encoded in base 64 digits to upload.

options
The UploadOptions object containing optional settings such as Overwrite and SkipResourceValidation.

Return Value

An ItemDescriptionsResult object.

DataResult Object

Represents an object containing the data returned by an IReportService operation.

Object Properties

	Name	Description
	Data Property	Gets the base 64 encoded data returned by the operation.

EMailDistribution Object

Defines e-mail options to use in distributing the report.

Object Properties

	Name	Description
	To Property	Gets or sets the string containing the address collection that contains the e-mail recipients.
	Subject Property	Gets or sets the string containing the subject for the e-mail.
	MessageBody Property	Gets or sets the string containing the message body of the e-mail.
	AsLink Property	Gets or sets a Boolean value indicating whether to add a link to the report within the e-mail.
	AsAttachment Property	Gets or sets a Boolean value indicating whether to attach the report to the e-mail.
	BaseUri Property	Gets or sets the base URI to use for the report link in the e-mail.

ExceptionDetail Object

Defines detailed exception information for an IReportService request operation error.

Object Properties

	Name	Description

	HelpLink Property	Gets the string for the help link from the exception.
	InnerException Property	Gets the ExceptionDetail object that represents the inner exception.
	Message Property	Gets the string message from the exception.
	StackTrace Property	Gets the string stack trace information from the exception.
	Type Property	Gets the string representing the type of the exception.

ItemDescription Object

Contains descriptive information about uploaded resources.

Object Properties

	Name	Description
	CreatedBy Property	Gets the string containing the name of the user who uploaded the resource.
	CreationDate Property	Gets the DateTime object containing the date on which the resource was uploaded.
	Id Property	Gets the string containing the ID for the resource.
	ModifiedBy Property	Gets the string containing the name of the user who last modified the resource.
	ModifiedDate Property	Gets the DateTime object containing the date on which the resource was last modified.
	Name Property	Gets the string value representing the name of the resource.
	Permissions Property	Gets the byte value representing the permissions for the resource.

ItemDescriptionsResult Object

Represents an object that is returned as a result of an Upload operation and contains an array of item descriptons.

Object Properties

	Name	Description
	isAuthenticated Property	Gets the Boolean value indicating whether the user has been authenticated.

Query Object

Defines a query used to find reports in the Select operation.

Object Properties

	Name	Description

	FilterBy Property	Gets the string containing the comma-separated role values by which to filter reports.
	OrderBy Property	Gets the string representing the property name to use in ordering the report descriptions returned by the Select operation.

RenderOptions Object

Defines options for how to render a report.

Object Properties

	Name	Description
	Extension Property	Gets the string value representing the rendering extension to use.
	ReportParameters Property	Gets any ReportParameter objects associated with the report.
	ReportId Property	Gets the string containing the identity of the report to render.
	Name Property	Gets the string containing the name of the preview file to render.
	ReportType Property	Gets the enumerated ReportType value for the report to render.

ReportDescription Object

Contains descriptive information about stored reports.

Object Properties

	Name	Description
	CreatedBy Property	Gets the string containing the name of the user who created the report.
	CreationDate Property	Gets the DateTime object containing the date on which the report was created.
	Description Property	Gets the string containing a description of the report.
	IsBroken Property	Gets the Boolean value indicating whether the report is broken by recent model version changes.
	IsParametrized Property	Gets the Boolean value indicating whether the report contains parameters.
	MasterReportIds Property	Gets the string value representing any master report used by the report. For internal use only.
	ModelId Property	Gets the string containing the identity of the model associated with the report.
	ModelName Property	Gets the string containing the name of the model associated with the report.
	ModelPermissions Property	Gets the enumerated ModelPermission value for the model associated with the report
	ModelVersion Property	Gets the int value for the history version of the model associated with the report.
	ModifiedBy Property	Gets the string containing the name of the user who last modified the report.

	ModifiedDate Property	Gets the DateTime object containing the date on which the report was last modified.
	Name Property	Gets the string value representing the name of the report.
	Permissions Property	Gets the byte value representing the permissions for the report.
	Id Property	Gets the string value representing the ID of the report.
	ReportType Property	Gets the enumerated ReportType value for the report.
	SubReportIds Property	Gets the string value representing any subreport used by the report. For internal use only.
	ThemeId Property	Gets the int value representing the theme used by the report.

ReportParameter Object

Defines a parameter for the report.

Object Properties

	Name	Description
	Name Property	Gets the string containing the name of the report parameter.
	Domain Property	Gets the enumerated ReportParameterDomain value for the report parameter.
	Values Property	Gets the string representing values to supply for the report parameter.

RequestInfo Object

Defines status information for an IReportService request operation.

Object Properties

	Name	Description
	Exception Property	Gets the ExceptionDetail object containing error information for the request.
	PrimaryUrl Property	Gets the URL of a primary resource for execution results.
	RequestId Property	Gets the string that identifies the request.
	State Property	Gets the enumerated RequestState value signifying the current request state.

RequestResult Object

Represents an object containing the status, or result contract, of an IReportService request operation.

Object Properties

	Name	Description
	Info Property	Gets the RequestInfo object containing status information for a request.

Result Object

Represents an object that is returned as a result of an IReportService operation.

Object Properties

	Name	Description
	Error Property	An Error object that represents an error that occurs during operation execution.
	IsAuthenticated Property	A Boolean value that indicates whether the request has been authenticated.

UploadOptions Object

Defines options for how to upload a resource.

Object Properties

	Name	Description
	ConnectionString Property	Gets the string value representing the connection string to use.
	Overwrite Property	Gets the Boolean value indicating whether to overwrite existing resources.
	SkipResourceValidation	Gets the Boolean value indicating whether to validate that all requirements are met for using the uploaded resource to serve compiled reports.

UserCapabilities Object

Represents a user capabilities object that is contained in a result from an IReportService get user capabilities operation.

Object Properties

	Name	Description
	Email Property	A string value that represents the e-mail address of the user.
	IsAdministrator Property	A Boolean value indicating whether the user is the administrator.
	IsEvaluationVersion Property	A Boolean value indicating whether the system is in evaluation mode.
	IsSBEVersion Property	A Boolean value indicating whether the system is the small business edition.
	IsSchedulePermitted Property	A Boolean value indicating whether the user has permission to scheduling.

	IsUploadPermitted Property	A Boolean value indicating whether the user has permission to upload reports.

UserCapabilitiesResult Object

Represents a user capabilities result object that is returned as a result of an IReportService get user capabilities operation.

Object Properties

	Name	Description
	Capabilities Property	A UserCapabilities object that contains the capabilities for the user.

ModelPermission Enumeration

Specifies the byte value of the enumerated level of permissions granted for the model associated with the report.

Enumeration Members

Member	Description
None	The indicated security token has no permission to access the data model.
Read	The indicated security token has only read access to the data model.
CreateReport	The indicated security token has both read and reporting access to the data model.

ReportParameterDomain Enumeration

Specifies the byte value of the enumerated report parameter domain.

Enumeration Members

Member	Description
SpecifiedValues = 0	Indicates that the parameter values are specified in the value collection.
SelectAll = 1	Indicates that all valid values are set.
AcceptingDynamicValues = 2	Indicates that the specified values include values to be evaluated dynamically at run time.

ReportType Enumeration

Specifies the byte value of the enumerated report type for all supported reports.

Enumeration Members

Member	Description
Unknown = 0	Indicates that the report type is unknown.
SW = 1	Indicates a semantic report, created with ActiveReports Server.
AR = 2	Indicates an ActiveReports developer report.
DDR = 3	Indicates a Data Dynamics Reports developer report.

RequestState Enumeration

Specifies the state of an IReportService request operation for a RequestInfo object.

Enumeration Members

Member	Description
Accomplished	Indicates that the request operation completed successfully.
Cancelled	Indicates that the request operation was cancelled.
Pending	Indicates that the request operation is in a request queue.
Rejected	Indicates that the request operation was rejected.
Running	Indicates that the request operation execution has started, but has not yet completed.
Unavailable	Indicates that no request status information is available due to connection problems or other issues.

Section 508 Compliance

Section 508 requires that when Federal agencies develop, procure, maintain, or use electronic and information technology, Federal employees with disabilities have access to and use of information and data that is comparable to the access and use by Federal employees without disabilities, unless an undue burden would be imposed on the agency. Section 508 also requires that individuals with disabilities seeking information or services from a Federal agency have access to and use of information and data that is comparable to that provided to the general public, unless an undue burden would be imposed on the agency.

Accessibility Summary:

All major features of ActiveReports software are accessible via keyboard navigation.

DISCLAIMER:

GRAPECITY MAKES NO WARRANTIES, EXPRESS OR IMPLIED, IN THIS DOCUMENT. The following information reflects the general accessibility features of GrapeCity software components as related to the Section 508 standards. If you find that the information is not accurate, or if you have specific accessibility needs that our products do not meet, please contact us and we will attempt to rectify the problem, although we cannot guarantee that we will be able to do so in every case.

ActiveReports Server Administrator and Designer controls

Section 1194.21 Software applications and operating systems

Criteria	Status	Remarks
(a) When software is designed to run on a system that has a keyboard, product functions shall be executable from a keyboard where the function itself or the result of performing a function can be discerned textually.	Supported with exceptions	ActiveReports Server partially supports navigation using the keyboard. However, a user cannot execute report functions by using only the keyboard on Administrator and Designer sites.
(b) Applications shall not disrupt or disable activated features of other products that are identified as accessibility features, where those features are developed and documented according to industry standards. Applications also shall not disrupt or disable activated features of any operating system that are identified as accessibility features where the application programming interface for those accessibility features has been documented by the manufacturer of the operating system and is available to the product developer.	Supported	The controls do not disrupt or disable industry standard accessibility features of other products.
(c) A well-defined on-screen indication of the current focus shall be provided that moves among interactive interface elements as the input focus changes. The focus shall be programmatically exposed so that Assistive Technology can track focus and focus changes.	Supported with exceptions	Some elements in the Report Designer, Viewer, Admin site, and Report Portal do not provide focus information to Assistive Technology.
(d) Sufficient information about a user interface element including the identity, operation and state of the element shall be available to Assistive Technology. When an image represents a program element, the information conveyed by the image must also be available in text.	Supported with exceptions	ActiveReports Server supports images for representing some user interface elements, and also provide a textual conveyance of the information.
(e) When bitmap images are used to identify controls, status indicators, or other programmatic elements, the meaning assigned to those images shall be consistent throughout an application's performance.	Supported	Any images used to identify a programmatic element have a consistent meaning throughout the

		controls.
(f) Textual information shall be provided through operating system functions for displaying text. The minimum information that shall be made available is text content, text input caret location, and text attributes.	Not Applicable	Operating system functions for displaying text are not supported in ActiveReports Server.
(g) Applications shall not override user selected contrast and color selections and other individual display attributes.	Supported with exceptions	ActiveReports Server does not override contrast and color in Internet Explorer 11 and Firefox. Google Chrome requires an extension to be installed.
(h) When animation is displayed, the information shall be displayable in at least one non-animated presentation mode at the option of the user.	Supported with exceptions	ActiveReports Server provides textual information for some animated elements.
(i) Color coding shall not be used as the only means of conveying information, indicating an action, prompting a response, or distinguishing a visual element.	Supported	ActiveReports Server does not have any feature or text that conveys any information or message using color coding.
(j) When a product permits a user to adjust color and contrast settings, a variety of color selections capable of producing a range of contrast levels shall be provided.	Supported	ActiveReports Server supports a wide variety of colors, range of contrast levels, themes and styles that can be applied to controls in a report.
(k) Software shall not use flashing or blinking text, objects, or other elements having a flash or blink frequency greater than 2 Hz and lower than 55 Hz.	Supported	The controls do not use flashing or blinking text or objects.
(l) When electronic forms are used, the form shall allow people using Assistive Technology to access the information, field elements, and functionality required for completion and submission of the form, including all directions and cues.	Supported	Any form-type dialogs or windows associated with the controls provide Assistive Technology with access to

		information on all directions, cues, field elements, and functionality required for completion.

Section 1194.22 Web-based Internet information and applications

Criteria	Status	Remarks
(a) A text equivalent for every non-text element shall be provided (e.g., via "alt", "longdesc", or in element content).	Supported with exceptions	Some of the non-text element has a text equivalent.
(b) Equivalent alternatives for any multimedia presentation shall be synchronized with the presentation.	Not Applicable	There is no multimedia presentation associated with the software.
(c) Web pages shall be designed so that all information conveyed with color is also available without color, for example from context or markup.	Supported	By default, no information is conveyed with color.
(d) Documents shall be organized so they are readable without requiring an associated style sheet.	Supported with exceptions	Some elements are not visible on the Admin site and the HTML5 Report Portal when styles are disabled.
(e) Redundant text links shall be provided for each active region of a server-side image map.	Not Applicable	There are no server-side image maps associated with the software.
(f) Client-side image maps shall be provided instead of server-side image maps except where the regions cannot be defined with an available geometric shape.	Not Applicable	There are no client-side image maps associated with the software.
(g) Row and column headers shall be identified for data tables.	Not Supported	By default, there are no data tables associated with the software.
(h) Markup shall be used to associate data cells and header cells for data tables that have two or more logical levels of row or column headers.	Not Supported	By default, there are no data tables associated with the software.
(i) Frames shall be titled with text that facilitates frame identification and navigation.	Not Applicable	By default, there are no frames associated with the software.
(j) Pages shall be designed to avoid causing the screen to flicker with a frequency greater than 2 Hz and lower than 55 Hz.	Supported	The software does not cause the screen to flicker outside the recommended range.
(k) A text-only page, with equivalent information or functionality, shall be provided to make a web site comply with the provisions of this part, when compliance cannot be accomplished in any other way. The content of the text-only	Not Applicable	Text-only page is not available in ActiveReports Web Viewer or Web controls.

page shall be updated whenever the primary page changes.		
(l) When pages utilize scripting languages to display content, or to create interface elements, the information provided by the script shall be identified with functional text that can be read by Assistive Technology.	Supported	The ActiveReports HTML5 Viewer control uses scripting that is supported by Assistive Technology.
(m) When a web page requires that an applet, plug-in or other application be present on the client system to interpret page content, the page must provide a link to a plug-in or applet that complies with §1194.21(a) through (l).	Supported	ActiveReports provides a link to install the Silverlight plug-in when an application using ActiveReports Silverlight viewer control is executed.
(n) When electronic forms are designed to be completed on-line, the form shall allow people using Assistive Technology to access the information, field elements, and functionality required for completion and submission of the form, including all directions and cues.	Supported with exceptions	The components used in ActiveReports have been configured for maximum compliance. However, some form controls do not support assistive technology.
(o) A method shall be provided that permits users to skip repetitive navigation links.	Not Applicable	By default, the software contains no repetitive navigation links.
(p) When a timed response is required, the user shall be alerted and given sufficient time to indicate more time is required.	Not Applicable	No timed responses are required with ActiveReports.

ActiveReports

Section 1194.31 Functional Performance Criteria

Criteria	Status	Remarks
(a) At least one mode of operation and information retrieval that does not require user vision shall be provided, or support for assistive technology used by people who are blind or visually impaired shall be provided.	Supported with exceptions	ActiveReports provide partial support to assistive technologies like screen readers.
(b) At least one mode of operation and information retrieval that does not require visual acuity greater than 20/70 shall be provided in audio and enlarged print output working together or independently, or support for assistive technology used by people who are visually impaired shall be provided.	Supported	ActiveReports does not require visual acuity greater than 20/70. Also, it is exposed to magnification tools like “Screen Magnifier” thus magnifying the screen for ease of use.
(c) At least one mode of operation and information retrieval that does not require user hearing shall be provided, or support for assistive technology used by people who are deaf or hard of hearing shall be provided.	Not Applicable	ActiveReports does not use any sound output and hearing is not necessary to use the product.
(d) Where audio information is important for the use of a product, at least one mode of operation and information retrieval shall be provided in an enhanced auditory fashion, or	Not Applicable	ActiveReports does not provide functionalities that require audio or video aids.

support for assistive hearing devices shall be provided.		
(e) At least one mode of operation and information retrieval that does not require user speech shall be provided, or support for assistive technology used by people with disabilities shall be provided.	Not Applicable	ActiveReports does not provide any functionality that use user speech.
(f) At least one mode of operation and information retrieval that does not require fine motor control or simultaneous actions and that is operable with limited reach and strength shall be provided.	Supported	ActiveReports does not provide any functionality that requires fine motor control or simultaneous action.

Documentation

Section 1194.41 Information, Documentation, and Support

Criteria	Status	Remarks
(a) Product support documentation provided to end-users shall be made available in alternate formats upon request, at no additional charge.	Supported	Documentation is available in four formats: HTML(Web), *.chm, *.PDF, and Help3(Visual Studio help content)
(b) End-users shall have access to a description of the accessibility and compatibility features of products in alternate formats or alternate methods upon request, at no additional charge.	Supported	ActiveReports provides information about accessibility and compatibility features in the documentation. Also, documentation is exposed to screen readers.
(c) Support services for products shall accommodate the communication needs of end-users with disabilities.	Supported	Support services such as telephone, email and forum support are provided to the customers.

Glossary

A

attribute tree
Displays the attributes that are associated with the entity selected under the **entity tree**. For example, the Customer entity has attributes like Address, Phone, Country, etc. If no entity is selected, the attribute tree is empty.

D

drill down
Allows users to expand collapsed table rows to reveal more data. See also drill through.

drill through
Allows users to click links to reports with more detailed data. Located in the Design tab toolbar, the Drilldown button is enabled when you select a table cell or chart data point. Specify parameters in the target report to supply relevant detail data.

R

report design surface
Visible on the Design tab and the Report tab, it is a visual page designer where you can drag and drop entities and attributes to create tables and charts and design your reports.

Report Info
An insert that you can add to a textbox to show page numbering, report name, or run time. Located in the Report tab toolbox, it is enabled when you click inside a textbox.

role root category
Role Root Category is a system category that is set as the top level category for a User Role. All the sub-categories under this level are available to the users in the user role.