TIBCO JASPERREPORTS® SERVER

ADMINISTRATOR GUIDE

RELEASE 6.0

This is version 1214-JSP60-22 of the *JasperReports Server Administrator Guide*.

TABLE OF CONTENTS

CHAPTER 1 OVERVIEW OF JASPERREPORTS SERVER ADMINISTRATION

TIBCO™ JasperReports® Server builds on TIBCO™ JasperReports® Library as a comprehensive family of Business Intelligence (BI) products, providing robust static and interactive reporting, report server, and data analysis capabilities. These capabilities are available as either stand-alone products, or as part of an integrated end-to-end BI suite utilizing common metadata and provide shared services, such as security, a repository, and scheduling. The server exposes comprehensive public interfaces enabling seamless integration with other applications and the capability to easily add custom functionality.

The heart of the TIBCO™ Jaspersoft® BI Suite is the server, which provides the ability to:

- Easily create new reports based on views designed in an intuitive, web-based, drag and drop Ad Hoc Editor.
- Efficiently and securely manage many reports.
- Interact with reports, including sorting, changing formatting, entering parameters, and drilling on data.
- Schedule reports for distribution through email and storage in the repository.
- Arrange reports and web content to create appealing, data-rich Jaspersoft Dashboards that quickly convey business trends.

For business intelligence users, Jaspersoft offers TIBCO™ Jaspersoft® OLAP, an optional component of JasperReports Server, controlled by license and described in its own user guide.

While the Ad Hoc Editor lets users create simple reports, more complex reports can be created outside of the server. You can either use TIBCO™ Jaspersoft® Studio or manually write JRXML code to create a report that can be run in the server. We recommend that you use Jaspersoft Studio unless you have a thorough understanding of the JasperReports file structure.

You can use the following sources of information to extend your knowledge of JasperReports Server:

- Our core documentation describes how to install, administer, and use JasperReports Server. Core documentation is available as PDFs in the docs folder of your JasperReports Server installation. You can also access PDF and HTML versions of these guides online from the Documentation section of the Jaspersoft Community website.
- Our Ultimate Guides document advanced features and configuration. They also include best practice recommendations and numerous examples. These guides are available online in PDF and HTML from the Documentation section of the Jaspersoft Community website.
- Our Online Learning Portal lets you learn at your own pace, and covers topics for developers, system administrators, business users, and data integration users. The Portal is available online from Professional Services section of our website.
- Our free samples, which are installed with JasperReports, Jaspersoft Studio, and JasperReports Server, are documented online. The samples documentation can be found on our community website.

This administrator guide describes features that are only available to users who have administrator roles. Many of the configuration procedures also assume you have unlimited access to the JasperReports Server host computer.

This chapter contains the following sections:
- **Overview of Organizations**
- **Overview of the Repository**
- **Overview of Users and Roles**
- **Overview of Security**
- **Administrator Login**
- **Administrator Pages**

This section describes functionality that can be restricted by the software license for JasperReports Server. If you don't see some of the options described in this section, your license may prohibit you from using them. To find out what you're licensed to use, or to upgrade your license, contact Jaspersoft.

JasperReports Server is a component of both a community project and commercial offerings. Each integrates the standard features such as security, scheduling, a web services interface, and much more for running and sharing reports. Commercial editions provide additional features, including Ad Hoc charts, flash charts, dashboards, Domains, auditing, and a multi-organization architecture for hosting large BI deployments.

1.1 Overview of Organizations

The commercial version of JasperReports Server is built on organizations, which are logical entities within JasperReports Server that have their own users, roles, and branches of the repository. As with any business structure or hierarchy, organizations may have suborganizations, which may in turn have suborganizations, and so on.

In the default JasperReports Server installation, a single organization mimics the simple structure of older versions of JasperReports Server. If you want to deploy multiple organizations, you must be aware of certain design considerations.

Organizations are not available in the community version of JasperReports Server.

1.1.1 Single Default Organization

After a default installation, JasperReports Server contains a single organization in which you can deploy your reports. For example, if you install the sample data, you'll see a single organization that holds all sample resources, users, and roles.

Single organizations are designed to handle most business cases and are straightforward to administer. Even in a single organization, a system admin and an organization admin share administrative duties. If your business needs more organizations, you'll need to manage several levels of administrators and possibly create shared resources in the repository. The following sections provide use cases and explain the functioning of multiple levels of administrators.

1.1.2   Multiple Organizations

There are many use cases for defining multiple organizations in JasperReports Server. For instance:

- An application provider, such as a software-as-a-service (SaaS) company or a computer department, has a hosted application offered to many customers It integrates JasperReports Server in its application to offer dashboards, reports, and analysis. A number of common reports and data sources are useful across customers, but there are customer specific reports, as well. Machines and databases are shared by customers, according to the provider's own architecture, but within the functionality provided by JasperReports Server, each customer is a separate organization. Customers can manage their own users in the hosted application, and JasperReports Server maps the application's authentication scheme to the correct organization. The organization structure provides the full power of JasperReports Server to each of the provider's customers, while ensuring that their data and reports are secure.

- A company has many departments but wants to consolidate the BI environment so that all departments are sharing a common BI infrastructure. Corporate IT needs to deploy and maintain only a single instance of JasperReports Server, while each department is represented by an organization that manages its own users. For security and simplicity, the departments do not share databases, except in the case of sub-departments, such as Accounts Payable being a sub-department of Finance. Users access JasperReports Server directly, logging in with their department name and user name. Organization administrators have defined the data sources and Domains specific to the needs of their department's users.

The organization feature is flexible enough to accommodate any combination of these scenarios and many others like it. In all cases, administrators can configure secure environments for any number of organizations, and end-users experience a powerful BI platform tailored to their needs.

All organizations or hierarchies of organizations exist independently within a single instance of JasperReports Server. Neighboring organizations are isolated from one another, while parent organizations have full control of their suborganizations. Users can access only the data and resources in their own organizations or suborganizations, and administrators can define roles and set permissions to further control access.

1.1.3   Levels of Administrators

Each organization has an administrator who can manage users, roles, and repository permissions in that organization and any of its suborganizations.

There are essentially three levels of administration:

- The system administrator – Also called *system admin*. The default login name of the system admin is `superuser`. He exists at the root level, outside all organizations. The system admin manages the JasperReports Server installation, creates top-level organizations, and configures server-wide settings. The system admin can create, modify, and delete users, roles, and repository objects of any organization.
- The administrator of a top-level organization – Also called *organization admin*. The organization admin manages all users, roles, and repository objects in an entire organization, including any suborganizations. The default login name of the organization admin is `jasperadmin`.
- The administrator of a suborganization – Functionally equivalent to an organization admin, but limited to the suborganizations's users, roles, and repository objects. This administrator can be overridden by a top-level organization admin.

1.2 Overview of the Repository

The repository is a hierarchical structure of folders where JasperReports Server, administrators, and users store resources for creating, running, and viewing reports. In its appearance and function, the repository resembles a file system with a structure of folders containing files. However, the repository is actually implemented as a database that is private to the server instance. As a result, it lacks a few of the functions of a file system.

1.2.1 Folder Structure

The root of the repository tree structure is accessible only to the system admin logged in as `superuser` in the commercial edition, or `jasperadmin` in the community edition. The tree contains the folders for each organization and folders for certain configuration settings.
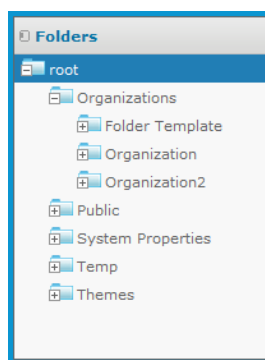
Figure 1-1 **Root of the Repository Showing Default Folders**

In the repository, each organization has its own branch, contained in a folder named for the organization. Each organization's top-level folder also contains a folder called Organizations where suborganizations can be created.

The previous figure shows the superuser's view, starting at root. The Organizations folder contains a folder for the default organization, named Organization. This figure also shows a second top-level organization. Commercial edition users logged in as `jasperadmin` will see a similar structure.

JasperReports Server automatically restricts every user's access to their own organization's branch of the repository. System admins (`superuser`) can view and create folders in all organizations, while organization admins (`jasperadmin`) can view and create folders only in the organizations they administer. In general, We recommend that you avoid placing resources directly in the root or organization folder. Instead, use folders for various resource types, as in the sample data.

1.2.2 Resources

Resources are stored in the repository and used as input for creating reports and performing analysis. Some resources, such as images, fonts, or JRXML files created in Jaspersoft Studio, are uploaded from files. Others, such as data sources and Domains[1], are created in JasperReports Server itself. Of course, dashboards, data views,

---

[1]A domain is a configured data source with selected joins, tables, and fields from your data base.

and reports can also be saved in the repository to be run as often as needed. Output such as PDF or HTML can be saved in the repository as well.

All resources, including folders, have unique IDs, names, and optional descriptions. The ID of a resource, along with the ID of its enclosing folder comprise the path used to reference that resource. The name and description appear in the user interface when you browse or search the repository.

Resources are stored in an internal format not accessible to users or administrators. Any resource can be exported with the `js-export` utility, but the resulting files are for backup or transfer to another JasperReports Server instance and cannot be modified.

JasperReports Server restricts access to folders and resources based on organizations, user names, and roles. The system admin and organization admin can define permissions as explained in **"Overview of Users and Roles" on page 17**.

1.2.3   Sample Data

When you install the sample data in JasperReports Server, the default organization (Organization) has sample content. Below you'll see the System Admin and Organization Admin views of sample the data.
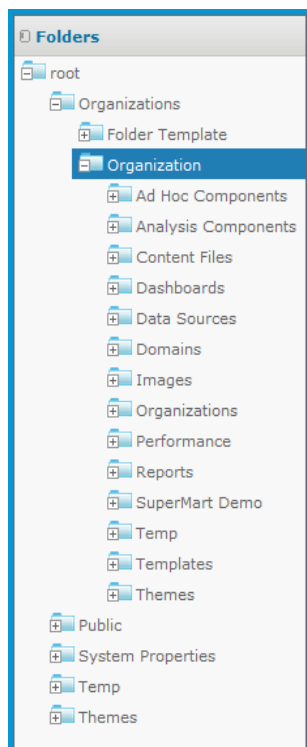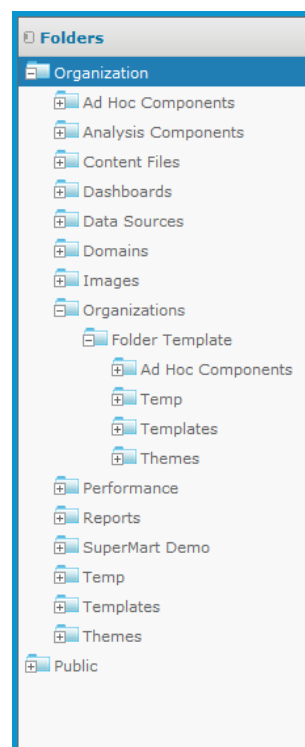
 Figure 1-2 **System Admin and Organization Admin Views of Sample Data**

The sample data includes dashboards, reports, Domains, data sources, and many of their components, such as input types and image files. Each type of content is stored in a separate folder, making it easy to locate. The Supermart Demo folder contains a complete example of dashboards, reports, and resources for various business scenarios in a fictional grocery store company.

The Public folder is a special system folder that appears at the root and in every organization folder. Its contents are shared with all organizations. The system admin should manage the Public folder and set permissions so that users can access shared resources (such as data sources, logos, and report templates) but not modify them.

1.2.4   Browsing and Searching

Users and administrators can browse or search the repository, depending on what action they want to perform and how resources are organized. Administrators have more actions available to them, such as creating resources. Searching the repository finds specific resources faster.

- Browse - On the Home page, click **View > Repository**.

Figure 1-3 **Browsing the Repository**

In Browse mode, the Folders panel on the left lists the folders in the repository and the Repository panel lists the contents of the selected folder. The tool bar in the Repository panel allows you to perform actions such as **Copy**, **Cut**, **Paste**, and **Delete**; select several resources in a same folder to perform actions in bulk.

- Search - Enter a search term in the search field at the top of any page, or select **View > Search Results**.

Figure 1-4 **Searching the Repository**

The filters in the left-panel enable users to search the repository. The **View > Reports** and **View > OLAP Views** also use the search feature with preset filters to find all reports and all OLAP views to which the user has access.

For more information on browsing and searching the repository, see the *JasperReports Server User Guide*.

1.3 Overview of Users and Roles

User accounts and role membership provide authentication and authorization settings for access control in JasperReports Server. Users enter an organization name if required, a login name, and a password to access JasperReports Server. Administrators assign named roles to users and then create role-based permissions to further control access to the repository.

In the commercial version of JasperReports Server, users and roles are associated with the organizations in which they are defined. Users and roles defined in an organization can be granted or denied access to any repository folder or object in the organization or its suborganizations. However, a suborganization administrator has no access to the roles and users in the parent organization, even if they are used in access permission within the suborganization.

User names and role names are unique within an organization, but not necessarily among suborganizations or across all organizations in the server. For example, the default organization administrator is called `jasperadmin` in every organization. Because login credentials include the organization and user name, JasperReports Server can distinguish each user. In some cases such as web services, a user is identified by the unique string `username|organization_ID`.

The community edition of JasperReports Server has only a single default organization. All user and role names belong to this organization.

Administrators define access to the repository directly on the repository resources. You can define a level of access, such as read-write, read-only or no access, and assign each permission based either a user name or a role.

1.3.1 Administering Users and Roles

Administrators perform the following actions to manage users in their organization:

- Create, modify, and delete users.
- Set user account properties such as name, email, and setting the password. However, no administrator can ever view a user's existing password in clear text.
- Login as any user in the organization to test permissions.
- Create, modify, and delete roles.
- Assign roles to users.
- Set access permissions on repository folders and resources.

1.3.2 Delegated Administration

JasperReports Server enables three levels of delegated administration:

- In a multi-organization implementation administrators in each organization are limited to actions within their organization.
- The Administer permissions allow a user to view and set permissions on a folder or resource. This can allow a power-user to manage a section of the repository, but not to create or manage users.
- Granting `ROLE_ADMINISTRATOR`, `ROLE_SUPERUSER`, or both allows a user to see the management interface and create users and roles. This is true delegated administration, whereby a user other than `superuser` or `jasperadmin` has administration abilities.

In the case of true delegated administration, three factors determine the scope of a user's administrative privileges:

- `ROLE_ADMINISTRATOR` – JasperReports Server confers the organization-level privileges to any user with this role. This includes managing users, roles, and permissions, as well as creating resources in the repository. When a user with this role logs in, the server displays the additional menus to access the admin pages and manage repository resources. Any administrator can assign this role to any other user.
- `ROLE_SUPERUSER` – When a user already has `ROLE_ADMINISTRATOR`, this additional role grants access to the system configuration functions. Only a system admin can assign this role to another user.
  In a multi-organization environment, `ROLE_SUPERUSER` should not be given to organization admins or organization users, because this allows access to the Ad Hoc cache shared by all organizations. In the case of a single organization such as in the default installation, you may assign this role to the organization admins to grant access to system settings without granting privileges to create top-level organizations or other system administrators.
- The user's organization – Regardless of roles, an administrator is always limited in scope to the organization in which the user account is created, including any suborganizations thereof. In no case can a user, even with the `ROLE_SUPERUSER`, ever view or modify any organization, user, role, or folder outside of the organization to which that user belongs.

Any administrator can grant `ROLE_ADMINISTRATOR` to any user. That user then becomes equivalent to an organization admin of the organization in which he belongs. In order to delegate system administration, the existing system admin must first create other users at the root level, outside of any organization. The system admin can then assign both `ROLE_ADMINISTRATOR` and `ROLE_SUPERUSER` to grant them system admin privileges. For further information about these roles, see **"Permissions" on page 57**.

1.4 Overview of Security

JasperReports Server ensures that people can access only the data they are allowed to see. The mechanisms that define organizations, users, roles, and repository resources work together to provide complete access control that includes:

- Authentication – Restricts access to identified users and protects that access with passwords. Defines roles for grouping users and assigning permissions. Authentication is further explained in the next section.
- Authorization – Controls access to repository objects, pages, and menus based on users and roles. Authorization is further explained in a following section.
- Data level security (commercial version only) – Defines row and column level permissions to access your data. Row and column level permissions can be defined and enforced in Domains. For more information, refer to the *JasperReports Server User Guide*. If you implement Jaspersoft OLAP, you can use roles to secure your data at any level of an analysis schema's hierarchy. For more information, refer to the *Jaspersoft OLAP User Guide*.

Administrators must keep security in mind at all times when managing organizations, user, roles, and resources, because the security settings behind each of these rely on the others.

1.4.1   Authentication

The first part of security is to define user accounts and secure them with passwords. Users must log in with their user ID and password so that they have an identity in JasperReports Server. The server stores user definitions, including encrypted passwords, in a private database. Administrators create, modify, and delete user accounts through the administrator pages, as described in **"Managing Users" on page 32**.

JasperReports Server also implements roles that can be assigned to any number of users. Roles let administrators create groups or classes of users that are granted similar permissions. A user may belong to any number of roles and receive the privileges from each of them. The server stores role definition in its private database, and administrators create, modify, and delete roles through the administrator pages, as described in **"Managing Roles" on page 38**.

JasperReports Server relies on the open source Spring security framework; it has many configurable options for:
- External authentication services such as LDAP (used by Microsoft Active Directory and Novell eDirectory)
- Single sign-on using JA-SIG's Central Authentication Service (CAS)
- Java Authentication and Authorization Service (JAAS)
- Container security (Tomcat, Jetty)
- SiteMinder
- Anonymous user access (disabled by default)

JasperReports Server also supports these encryption and authentication standards:
- HTTPS, including requiring HTTPS
- HTTP Basic
- HTTP Digest
- X509

The Spring framework is readily extensible to integrate with custom and commercial authentication services and transports.

Authentication occurs by default through the web user interface, forcing login, and/or through HTTP Basic authentication for web services, such as Jaspersoft Studio and for XML/A traffic. The server can automatically synchronize with an external authentication service. The external users don't need to be created manually in the server first. Both users and roles are created automatically in the server from their definitions in an external authentication service. For an overview of the authentication system and details about external authentication, see the *JasperReports Server Authentication Cookbook*.

1.4.2   Authorization Overview

With a user's identity and roles established, JasperReports Server controls the user's access in these ways:

Menu options and pages	The menus that appear in JasperReports Server depend on the user's roles. For example, only users with the administrator role can see the **Manage** menu and access the administrator pages. By modifying the server's configuration, you can modify access to menus, menu items, and individual pages. Refer to the *JasperReports Server Source Build Guide* and *JasperReports Server Ultimate Guide* for more information.
Organization scope	Users belong to organizations and are restricted to seeing resources within their organization. Organizations have their own administrators, but they see only the users, roles, and resources from their organization. When JasperReports Server is configured with multiple organizations, they are effectively isolated from each other, although the system admin can share resources through the Public folder. For more information, see **"Multiple Organizations in the Repository" on page 55**.
Resource permissions	Administrators can define access permissions on every folder and resource in the repository. Permissions can be defined for every role and every user, or they can be left undefined so they are inherited from the parent folder. For example, user may have read-write access to a folder where they create reports, but the administrator can also create shared reports in the same folder that are set to read-only. The possible permissions are: no access, execute only, read-only, read-delete, read-write-delete, and administer (see **"Permissions" on page 57**). Permissions are enforced when accessing any resource either directly through the repository interface, indirectly when called from a report, or programmatically through the web services. Permissions are cumulative, meaning that a user has the most permissive access that is granted to any of the roles to which the user belongs.
Administrator privileges	JasperReports Server distinguishes between reading or writing a resource in the repository and viewing or editing the internal definition of a resource. For security purposes, granting a user read or write permission on a resource does not allow viewing or editing the resource definition. For example, users need execute or read permission on a data source to run reports that use it, but they cannot view the data source's definition which includes a database password. Also, only administrators may interact with theme folders to upload, download, and activate CSS files that control the user interface.

Data-level security	Data-level security determines the data that can be retrieved and viewed in a report, based on the username and roles of the user who runs the report. For example, a management report could allow any user to see the management hierarchy, managers would see the salary information for their direct employees, and only human resource managers would see all salary values.
	Data-level security in Domains is explained in the *JasperReports Server User Guide*. Data-level security through OLAP views is covered in the *Jaspersoft OLAP User Guide*.
	Note: This type of security is only available in the commercial edition of JasperReports Server.
attributes	Profile attributes are name-value pairs associated with a user, organization, or server.
	User attributes provide additional information about the user and can also be used to restrict a user's access to data through Domain security files and OLAP schemas. For information on defining user attributes, see **"Editing User Attributes" on page 35**.
	User, organization and server attributes can be used to customize the definition of a data source or as parameters of a report. See **"Attributes in Data Source Definitions" on page 66** and **"Attribute-Based Parameters for Queries and Reports" on page 107**

1.5 Administrator Login

Administrators log in on the standard login page, using the following default passwords:

Commercial editions:	system admin: username `superuser` and password `superuser`
	organization admin: username `jasperadmin` and password `jasperadmin`
Community project:	system admin: username `jasperadmin` and password `jasperadmin`

For security reasons, always change the default administrator passwords immediately after installing JasperReports Server. For instructions, see **"Editing a User" on page 34**.

For more information about options on the Login page and logging in with multiple organizations, see the *JasperReports Server User Guide*.

The first time you log in as an administrator, you may be prompted to opt-into the Heartbeat program. You should also set the administrator passwords and email.

1.5.1 JasperReports Server Heartbeat

When you log into JasperReports Server for the first time after installation, users of the commercial edition may be prompted to opt into the server's Heartbeat program. It reports specific information to Jaspersoft about your implementation: the operating system, JVM, application server, database (type and version), and JasperReports Server edition and version number. By tracking this information, we can build better products that function optimally in your environment. No personal information is collected.

To opt into the program, click **OK**. To opt out, clear the check box then click **OK**.

1.5.2   Administrator Email

After logging in for the first time, you should set the email on the `superuser` and `jasperadmin` accounts to your email address. In very rare cases, the server may notify you by email about issues with your license.

> This is also a good time to change the default passwords on the `superuser` and `jasperadmin`accounts.

To set the email and passwords on the administrator accounts, edit the user account information as described in **"Editing a User" on page 34**.

1.6   Administrator Pages

Administrators have access to special pages to manage the server. After logging in, click **View options** in the Admin box on the Getting Started page or select an item from the **Manage** menu on any page.

Figure 1-5 Different Manage Menus for Different Admins

In the commercial edition of JasperReports Server, the administrator controls are different for system and organization admins, as shown in the previous figure. Organization administrators can manage users, roles, and suborganizations, but only within their organization. System admins can manage top-level organizations, as well as users and roles in any organization. And only system admins have access to the server-settings that apply to logs, Jaspersoft OLAP, and Ad Hoc cache and data policies.

The next figure shows the **Admin Home** page for system admins that includes the **View settings** button not available to organization admins.
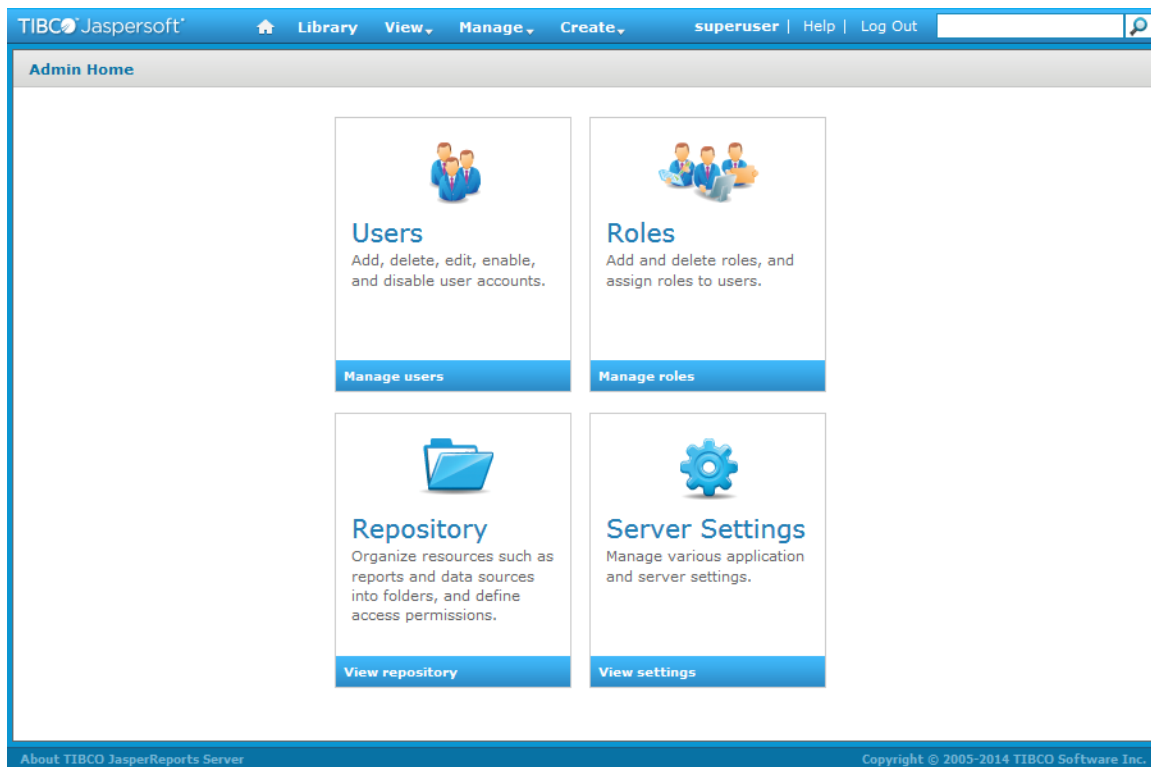
Figure 1-6 **Manage Server Page for System Admins in Commercial Editions (superuser)**

The **About JasperReports Server** link in the footer of all pages displays the product version along with the software build, and your license type and expiration. Please have this information available if you need to contact Jaspersoft for support.
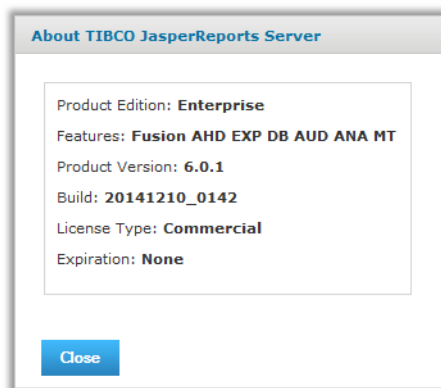
Figure 1-7 **About JasperReports Server Dialog**

CHAPTER 2   ORGANIZATION, USER, AND ROLE MANAGEMENT

In a single-organization deployment, the administrator needs to create only the users and roles. In deployments with multiple organizations, administrators need to create organizations, populate them with users, and create the roles they use afterward to set access permissions.

> This section describes functionality that can be restricted by the software license for JasperReports Server. If you don't see some of the options described in this section, your license may prohibit you from using them. To find out what you're licensed to use, or to upgrade your license, contact Jaspersoft.

In a deployment with multiple organizations, there can be administrators at every level of the hierarchy, as described in **"Delegated Administration" on page 18**. Part of any large deployment is to designate the administrators responsible for every task. For example, system administrators might set up the top-level organizations and default roles, but each organization's admin would then create and manage the users of their particular organization.

The interface in JasperReports Server for managing organizations (commercial edition users), users, and roles (both commercial and community editions) accommodates all levels of administrators and makes it easy for them to search among hundreds of users and roles, whether in a single organization or spread across many. The interface also enforces the scope of administrative privileges. For example, it ensures that an organization administrator cannot see roles and users from parent organizations.

This chapter contains the following sections:
* **Managing Organizations**
* **Managing Users**
* **Managing Roles**

2.1   Managing Organizations

System admins and organization admins use the same pages for managing organizations, the only difference is that system admins can manage top-level organizations, and organization admins are limited to suborganizations.

> Community edition users and administrators of deployments with a default single organization can generally skip this section. However, this procedure can be used to change the name of the default organization.

The following figure shows the organizations the system admin (superuser) can view, that is, all the organizations in the repository. As shown in the Organizations panel on the left, the system admin's view begins at the root of the organization hierarchy and includes all defined organizations and suborganizations, so he can manage any organization or suborganization in the repository. In this example, there are two top-level organizations, and one of them has several suborganizations.
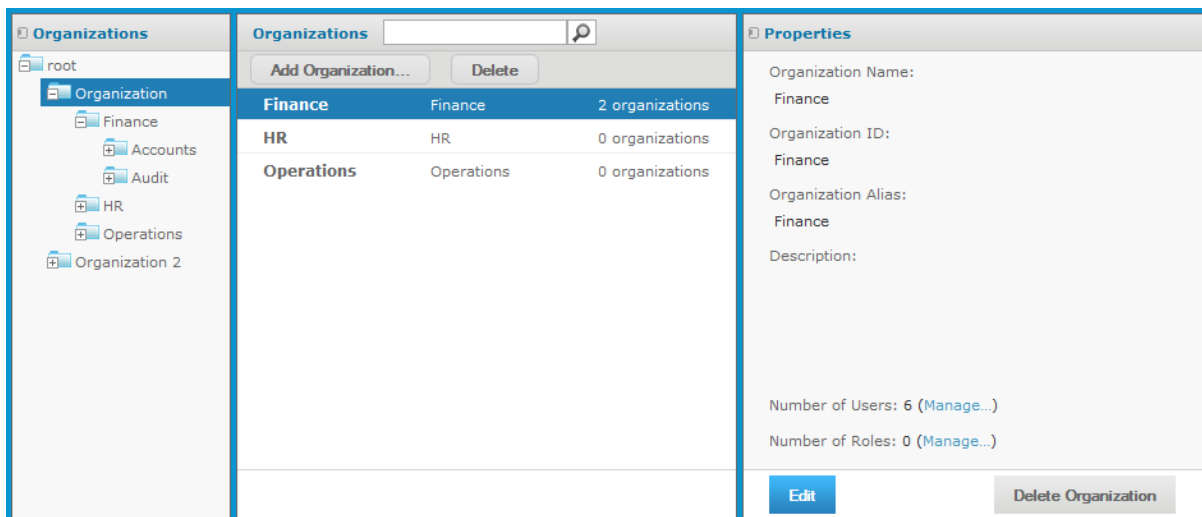
Figure 2-1 **System Admin View of Manage Organizations Page**

The following figure shows the same repository as seen by the admin of Organization (jasperadmin). It shows that this admin's view is limited to his own organization and its suborganizations, and he can access and manage only those.
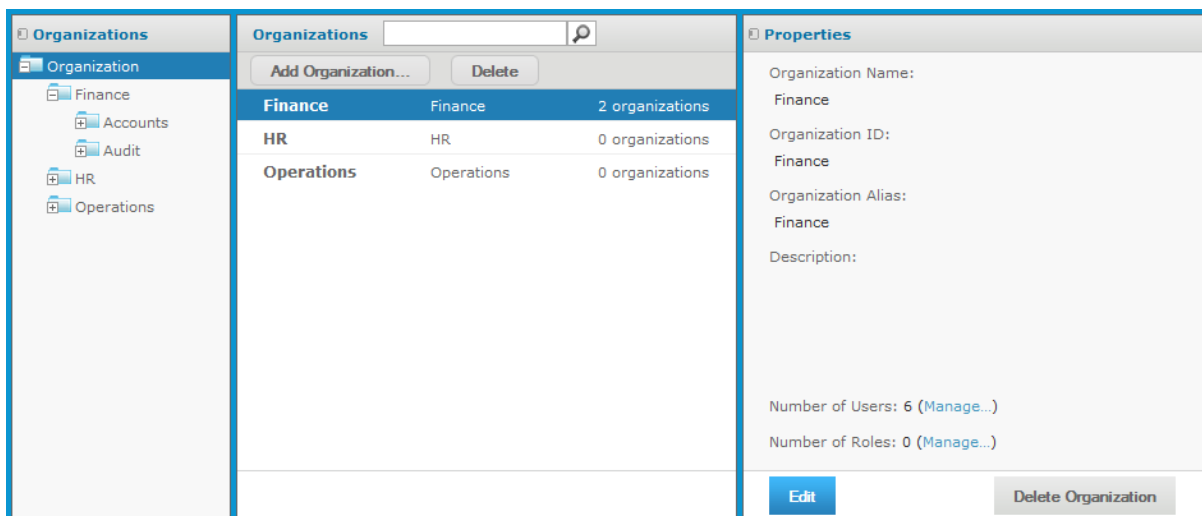
Figure 2-2 **Organization Admin View of Manage Organizations Page**

2.1.1 Viewing Organization Properties

1. Log in as a user with administrative privileges for the organization.
2. Select **Manage > Organizations**.

   The organization management page appears, as shown in **Figure 2-1** or **Figure 2-2**.
3. To select an organization, click its parent in the left Organizations panel, then select the organization in the center panel. If there are many organizations, you can search for a specific organization, but the search will find only organization in the center Organizations panel.
4. Once you select an organization, the Properties panel on the right shows information about it:
   • Name – Display name of the organization that appears on the organization's top folder.
   • ID – Unique and permanent organization ID used for logging into it.
   • Alias – Unique but editable short organization name that can also be used when logging in.
   • Description – Optional description that appears only in this Properties panel.
   • Number of Users – Count of all users, including those in any suborganizations. Click **Manage** to see the list of users on the user management page.
   • Number of Roles – Count all roles, including those in any suborganizations. The number of roles does not include the system roles (such as ROLE_USER) that appear at every organization level but are defined at the root level. Click **Manage** to see the list of roles on the role management page.

2.1.2 Creating an Organization

1. Log in as a user with administrative privileges for the parent of the new organization.
2. Click **Manage > Organizations**.
3. In the left-Organizations panel, expand the hierarchy of organizations to select the parent organization, for example Finance, then click **Add Organization** in the middle panel.
4. The Add Organization dialog appears.

Figure 2-3 Adding an Organization

5. Enter the organization name; the server automatically fills in the ID and alias based on the name. You can change the ID and alias if needed before saving the organization. Once saved, the organization ID can no longer be modified. The Description is optional. The previous figure shows this dialog with sample values.

6. To save the new organization, click **Add Organization to <organization>**.

The new organization appears in the Organizations panels. When you select it in the center panel, its properties appear in the Properties panel on the right.

The properties panel shows the number of users and roles in the organization and provides links to manage them. By default, new organizations have the following:

• Two users with default passwords: the organization admin (`jasperadmin/jasperadmin`) and a sample user (`joeuser/joeuser`).

> For security reasons, always change the default passwords immediately after creating a new organization. For instructions, see **"Managing Users" on page 32**.

• The organization has no roles of its own. The default users have the system-wide roles `ROLE_USER` and `ROLE_ADMINISTRATOR`.

• In the repository, a new folder is created in the parent's Organization folder. This new organization folder contains a copy of the parent's Organization/Folder Template folder. To manage the Organization folders, select **View > Repository**.

2.1.3 Default Folders for Organizations

Every organization contains a special folder named Organizations where you can create suborganizations. The Organizations folder always contains a folder named Folder Template. When you create a new organization, the entire content of the Folder Template is copied to create your new organization's folders. You can add folders and resources in the Folder Template, and these are copied into each later-created organization.

The default folders in the Folder Template are:

• Ad Hoc Components\Topics – The location where the Ad Hoc Editor looks for Topics to create new reports.
• Temp – A folder visible only administrators, used by the server to store temporary files.
• Templates – A folder that holds templates used when generating reports from Ad Hoc views.
• Themes – A special folder managed by the system to contain CSS files that define the appearance of the user interface.

> The Public folder visible in every organization is a special shared folder at the root level. The repository makes it accessible to every organization, but it is not within the organization folder.

There is a Folder Template at every level of the organization hierarchy, including the root. The system admin can add content to the top-level Folder Template for use in creating top-level organizations. Organization admins can add content to their Folder Templates for use in creating suborganizations.

Finally, the Folder Template itself is copied into a new organization, so each new suborganization has the same default folders and resources as its parent.

2.1.4 Editing an Organization

1. Log in as a user with administrative privileges for the organization.

2. Click **Manage > Organizations**.

3. In the left-Organizations panels, select the organization's parent. In the center Organizations panel, select the organization.

4. In the Properties panel, click **Edit** so you can change the organization's properties.
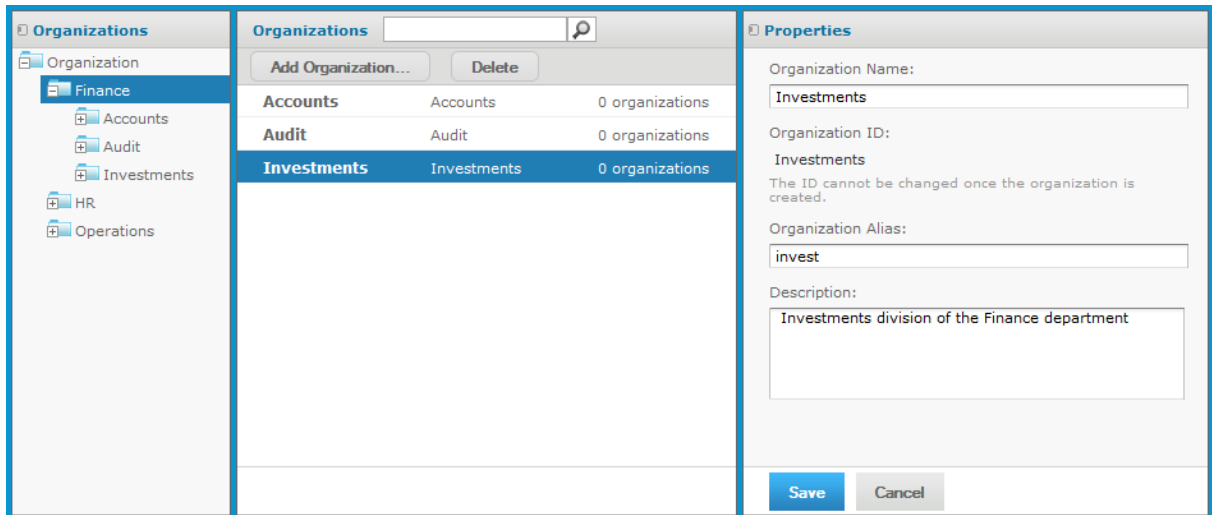
Figure 2-4 **Editing Properties of an Organization**

5. Edit the organization properties as needed. Changing the organization name changes the name of the organization's folder, as well, but no other data. You can change the alias and description. The organization ID is value defined when the organization was created.

6. Click **Save** to keep your changes, or **Cancel** to quit without saving.

2.1.5   Creating Attributes on Organizations

Organization attributes can be used to derive the parameters of data sources used for reports. For example, an attribute with the same name could have different values in each organization, so that the same data source defiition would access a different database specific to each organization. For more information, see **4.1, "Attributes in Data Source Definitions," on page 66**.

You can create, edit, and delete organization attributes using the REST v2 API. For more information about these REST API requests, see the corresponding section of the *JasperReports Server Web Services Guide*.

To view, create, modify, or delete organization attributes:

1. Download and install the REST browser plugin from one of these links:
   • REST plugin for Firefox
   • REST plugin for Chrome

2. Log into JasperReports Server as system admin (superuser) or the organization admin (jasperadmin) of the organization you want to modify.

3. To view all attributes on an organization, use the REST plugin to send this API request:

   GET http://<host>:<port>/jasperserver-pro/rest_v2/organizations/*organizationID*/attributes

   To view individual attributes on an organization, add name=... paramters to the request above:

GET http://<host>:<port>/jasperserver-pro/rest_v2/organizations/*organizationID*/attributes?
name=*Attr1*&name=*Attr2*&...

To view single attributes on an organization, add the attribute name to the request above:

GET http://<host>:<port>/jasperserver-pro/rest_v2/organizations/*organizationID*/attributes/*Attr3*

4. To create attributes on an organization, or to replace all of its exisiting attributes, send this REST API request:

PUT http://<host>:<port>/jasperserver-pro/rest_v2/organizations/*organizationID*/attributes

```
{
 "attribute":[
 {
  "name": "Attr1",
  "value":"Value1"
 },
 {
  "name": "Attr2",
  "value":"Value2a, Value2b"
 },
 {
  "name": "Attr3",
  "value":"Value3"
  "secure":"true"
 },
 ...]
}
```

5. To add one attributes on an organization, or to change the value of an existing attribute, send this REST API request. The attribute name in the URL must match the one in the content:

PUT http://<host>:<port>/jasperserver-pro/rest_v2/organizations/*organizationID*/attributes/*Attr1*

```
{
 "name": "Attr1",
 "value":"newValue1"
}
```

6. To delete a single attribute from an organization, use the following REST API request:

DELETE http://<host>:<port>/jasperserver-pro/rest_v2/organizations/*organizationID*/attributes/*Attr1*

To delete multiple attributes:

DELETE http://<host>:<port>/jasperserver-pro/rest_v2/organizations/*organizationID*/attributes?name=*Attr2*&name=*Attr3*&...

To delete all attributes:

DELETE http://<host>:<port>/jasperserver-pro/rest_v2/organizations/*organizationID*/attributes

2.1.6 Creating Attributes at the Server Level

Attributes can also be defined the server level, which is equivalent to the root organization. If a given attribute name is not defined on an organization or a user, it can have a value at the server-level to act as a default value. Server-level attributes may also be used as parameters in data source definitions. For example, the same data source may be imported to several servers, such as test and production servers, but a server-level attribute with the same name on each server makes the data source connect to a different database. For more information, see **4.1, "Attributes in Data Source Definitions," on page 66**.

To view, create, modify, or delete server-level attributes:

1. Log on as the system administrator (`superuser` on commercial editions, `jasperadmin` on community editions).

2. Select **View > Repository** and open the System Properties folder at the root.

3. Right click the GlobalPropertiesList resource and select Edit from the context menu.

   The GlobalPropertiesList is a list of name-value attribute pairs. Some of the attributes are used internally, such as JDBC driver names and configuration settings. New attributes defined here have the same behavior as server-level attributes.

Figure 2-5 **Editing Attributes at the Server Level**

4. Use the fields to enter new attribute name-value pairs, and click **Add** to create new attributes.

5. Click **Remove** to delete a server-level attribute. Do not remove attributes defined by the system.

6. To modify an attribute, remove it first, then add it with the same name and a new value.

7. When secure attributes appear in the GlobalPropertiesList, their value is shown as *** symbols. The GlobalPropertiesList does not support adding a secure attribute. To add a secure server-level attribute, use the procedure in **2.1.5, "Creating Attributes on Organizations," on page 29**, but specify the REST API request as follows:

   PUT http://<host>:<port>/jasperserver-pro/rest_v2/attributes/*Attr3*

```
{
  "name": "Attr3",
  "value":"Value3"
  "secure":"true"
}
```

You can also create, edit, and delete server-level attributes using the REST v2 API. For more information about these REST API requests, see the corresponding section of the *JasperReports Server Web Services Guide*.

2.1.7 Deleting an Organization

1. Log in as a user with administrative privileges for the organization.

2. Click **Manage > Organizations**.

3. In the left-Organizations panels, select the organization's parent. In the center Organizations panel, select the organization and click **Delete**.

   Administrators cannot delete the organization to which they belong. Confirming the delete completely removes all users, roles, and folders of the organization and all of its suborganizations from JasperReports Server.

2.2 Managing Users

As with organizations, system admins can manage all users in all organizations and create users outside of organizations, as described in **"Delegated Administration" on page 18**. Organization admins can manage only the users in the organizations they administer.

The default installation of JasperReports Server includes the following users:

Table 2-1 Default Users in JasperReports Server Installations

User Name	Default Password (case sensitive)	Organization	Description
superuser	superuser	*none*	Default system admin (commercial edition only).
anonymousUser	anonymoususer	*none*	Allows anonymous login; disabled by default. If you do not allow anonymous access, this user can be deleted.
jasperadmin	jasperadmin	Organization	Default organization admin in every organization.
joeuser	joeuser	Organization	Default end user in every organization.
demo	demo	Organization	Included for use with sample data.
CaliforniaUser	CaliforniaUser	Organization	Included for use with sample data.

> You should advise your users to change their passwords regularly. To configure periodic expiration of passwords, refer to **"Configuring User Password Options" on page 155**.

> Some editions of the server do not include all these users; for example, superuser is not included in the Community edition of the server, and demo and CaliforniaUser are included only if you install the sample data.

2.2.1 Viewing User Properties

1. Log in as a user with administrative privileges for the user's organization.

2. Select **Manage > Users** or, on the Admin Home page, click **Users**. The Manage Users page displays the users in each organization and properties of the selected user.
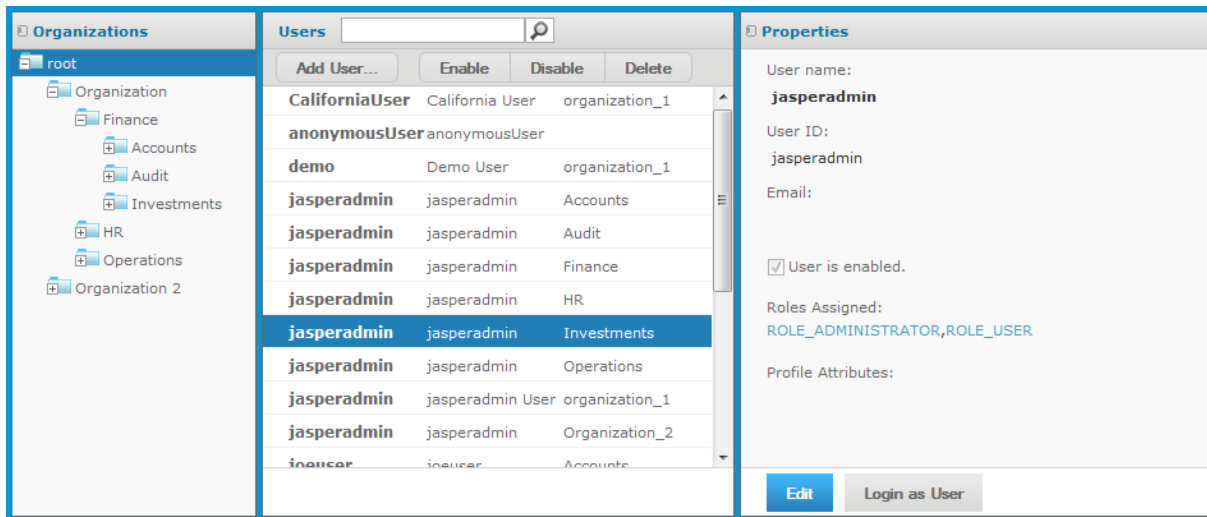
Figure 2-6 **Manage Users Page**

The list of users includes everyone in the chosen organization and its suborganizations. The same user ID may appear more than once, indicating that users with the same ID were created in different organizations. The third column gives the name of the selected user's organization.

In this example, the system admin can see all users in all organizations by selecting the root of the Organization hierarchy. There are always multiple jasperadmin users in a hierarchy of organizations, because it's the default administrator ID in each organization.

3. To locate a user:

   • Browse for users. Expand the organization hierarchy in the left-panel, and select an organization or suborganization.

   • Search for a user, Select the organization (or any parent organization) and enter a search string in the **Search** field of the Users panel. The search results show all usernames in the selected organization and suborganizations that match the search string.

4. Select a user to view attributes in the Properties panel.

   User status can be **Enabled** or **Disabled**; disabled users are displayed in gray text in the Users list. For convenience, the role names link to the role management page for each role.

> As the admin of a given organization, you can see the roles defined in your organization and its suborganizations but not the parent organization (except for certain system-wide roles). A user may have roles defined and assigned from a parent organization that are not visible to the administrator of the user's organization. For more information, see **"Managing Roles" on page 38**.

2.2.2   Creating a User

1. Log in as a user with administrative privileges for the organization to which the user will belong.

2. Select **Manage > Users** or, on the Admin Home page, click **Users**.

3. In the Organizations panels, select the organization for the new user and click **Add User**. For community edition admins, simply click **Add User**. The Add User dialog appears.

Figure 2-7 Adding a User

4. Enter the following information:
   - User name – The new user's full name. The name is optional but recommended; It will appear in the top right-corner of the UI when the user is logged in.
   - User ID – Generated automatically from the user name; you can accept the suggested value or type your own. The user ID is used to log into JasperReports Server, and for administrators to manage users and resources. The User ID must be unique within the organization, but may exist in multiple organizations.
   - Email – This is optional but must be in a valid email format.
   - Password and confirmation – Enter and confirm a password for the user.
   - User is enabled – To enable the user to log in, select this checkbox. Users who are not enabled can't log in. If you implement role-based permissions, you might want to delay enabling the user until you assign more roles. For more information on roles, see **"Managing Roles" on page 38**.
5. Click **Add User to <organization>** (**Add User** for community edition).

   The new user is available in the Users panel. To assign roles to the user, click **Edit** in the user's Properties panel.

2.2.3 Editing a User

One way to assign roles to a user is to edit the user's properties. Alternatively, when you edit a role, you can assign it to any number of users. To edit a user's properties:

1. Log in as a user with administrative privileges for the user's organization.
2. Click **Manage > Users** or, on the Admin Home page, click **Users**.
3. In the Organizations panel, select the user's organization. (Commercial users only. Community users skip to **step 4**.)
4. In the Users panel, select the user.
5. In the user's Properties panel, click **Edit**.

Figure 2-8 **Editing the Properties of a User**

6. Edit the user's properties as needed. You can't edit the user ID; it always has the value defined when the user is created originally.

7. To assign or remove roles from the user, select the roles, and use the arrow buttons between the Roles Available and Roles Assigned lists.

    The Roles Available list includes any role in the organizations of the current administrator, as well as the special system-wide roles. For more information on creating and adding roles, see **"Managing Roles" on page 38**.

8. Click **Save** to keep your changes.

9. In the Properties panel, click **Login as User** to test the user's permissions, as explained in **"Testing User Permissions" on page 62**.

    Logging in as another user is also necessary when you are maintaining resources that use absolute references in the repository. For more information, see **"Referencing Resources in the Repository" on page 56**.

2.2.4   Editing User Attributes

User attributes are name-value pairs that provide additional information about a user and to user restrict access to data through Domain security files and OLAP schemas. User attributes can also enable you to derive data source parameters based on the user running a report. See **4.1, "Attributes in Data Source Definitions," on page 66**.

As of version 5.0, we recommend using the user interface to add, edit, and delete user profile attributes instead of accessing the private repository database.

To add, edit, or remove user attributes:

1. Log in as a user with administrative privileges for the user's organization.
2. Click **Manage > Users** or, on the Admin Home page, click **Users**.
3. In the Organizations panel, select the user's organization. (Commercial users only. Community users skip to **step 4**.)
4. Select the user in the Users panel.
5. In the Properties panel, click **Edit** and select the **Attributes** tab at the top of the panel.

Figure 2-9 **Editing the Attributes of a User**

6. To create a new attribute, enter an attribute name and attribute value(s) and click **Add**. Use commas to separate multiple attribute values.

   You cannot define two attributes with the same name.
7. To remove an attribute, click **Remove** in that row.
8. To edit an attribute, remove it and create it again with a new value.
9. Click **Save**.

Figure 2-10 **Viewing the Attributes of a User**

2.2.5   Enabling or Disabling Multiple Users

You may sometimes need to disable user accounts. For example, when making configuration changes, you may want to lock out all users until the changes are finished. Administrators can select any number of users, and `superusers` can select all users in the server, except themselves.

1. Log in as a user with administrative privileges for the users' organization.
2. Click **Manage > Users** or, on the Admin Home page, click **Users**.
3. In the Organizations panel, select the users' organization; to enable or disable users in different organizations, select the common parent organization.
4. In the Users list use Control-click and Shift-click to make multiple selections. If the User list is too long, enter a search term to find users and enable or disable them individually.
5. Click **Enable** or **Disable** in the menu bar.

2.2.6   Deleting One or More Users

1. Log in as a user with administrative privileges for the user's organization.
2. Click **Manage > Users** or, on the Admin Home page, click **Users**.
3. In the Organizations panel, select the user's organization; to delete users in different organizations, select the common parent organization.
4. In the Users list use Control-click and Shift-click to make multiple selections. If the list of users is too long, enter a search term to find and select the user.
5. In the tool bar of the Users panel, click **Delete** and confirm the action.

2.3 Managing Roles

Roles define sets of users who are granted similar permissions. Administrators create roles, assigned them to users, and set permissions in the repository (see **"Permissions" on page 57**). By default, JasperReports Server includes the following roles; some are needed for system operation, some are included as part of the sample data:

Table 2-2 Default Roles in JasperReports Server Installations

Role	Description
ROLE_SUPERUSER	Commercial editions only. This role determines system admin privileges, as explained in **"Delegated Administration" on page 18**. It's a system-level role, but in a single-organization deployment, the system admin can assign it to an organization admin. Never delete this role, it's required for proper administration of the server.
ROLE_ADMINISTRATOR	This role determines organization admin privileges, as explained in **"Delegated Administration" on page 18**. This role is automatically assigned to the default jasperadmin user in every new organization. It's a special system-level role visible in every organization, and organization admins can assign to other users. Never delete this role, it's required for proper administration of the server.
ROLE_USER	Required to log in. This role is automatically assigned to every JasperReports Server user. It's a special system-level role visible in every organization. Never delete this role, it's required to create users and allow them to log in.
ROLE_ANONYMOUS	When anonymous access is enabled,this role is automatically assigned to any agent accessing the server without logging in. It's a special system-level role visible in every organization. This role is also assigned to the default anonymous user. By default, anonymous access is disabled and this role isn't used. It's a system role that even the system admin can't delete.
ROLE_PORTLET	JasperReports Server assigns this role to users that are created automatically when a portal such as Liferay requests authentication for a connection. If the specified user name does not exist in the server, it is created, assigned the password of the user in the portal, and assigned the ROLE_PORTLET and ROLE_USER roles. If you do not use a portal server, you can delete this role.
ROLE_DEMO	This role grants access to the SuperMart demo Home page, reports, and if you implement Jaspersoft OLAP, OLAP views. This role is assigned to the demo user in the default organization. These objects are available only if you installed the sample data when you installed JasperReports Server. It is a special system-level role that is visible in every organization When you no longer need the sample data, this role can be deleted.

Role	Description
ROLE_SUPERMART_MANAGER	This role is used to assign permissions relative to the sample data. It is a special system-level role that is visible in every organization. It demonstrates data security features available in Jaspersoft OLAP. See the *Jaspersoft OLAP Ultimate Guide* for more information. When you no longer need the sample data, this role can be deleted.
ROLE_ETL_ADMIN	This role no longer governs any JasperReports Server permissions or functionality, unless your server is integrated with Talend Integration Suite Enterprise Edition (TIS EE). Otherwise, it does not appear in the server.

Except for the five special system-level roles visible in every organization, roles are defined within organizations. A single role ID can be defined in multiple organizations, as long as it is unique within each organization. Admins can manage all roles in their organizations and any suborganization, but they can never see roles in a parent or sibling organization. JasperReports Server enforces this scheme to ensure that organizations are secure and only valid roles are assigned to users.

It is possible for an administrator to assign a role to a user in a suborganization, where the role is defined in a parent organization of the user. The admin of the user's organization cannot see the role when managing the user, but the admin of the role's organization can, and permissions associated with the role are properly enforced.

2.3.1   Viewing Role Properties

1. Log in as a user with administrative privileges for the role's organization.
2. Select **Manage > Roles** or, on the Admin Home page, click **Roles**. The Manage Roles page displays the roles in each organization and properties for each role.
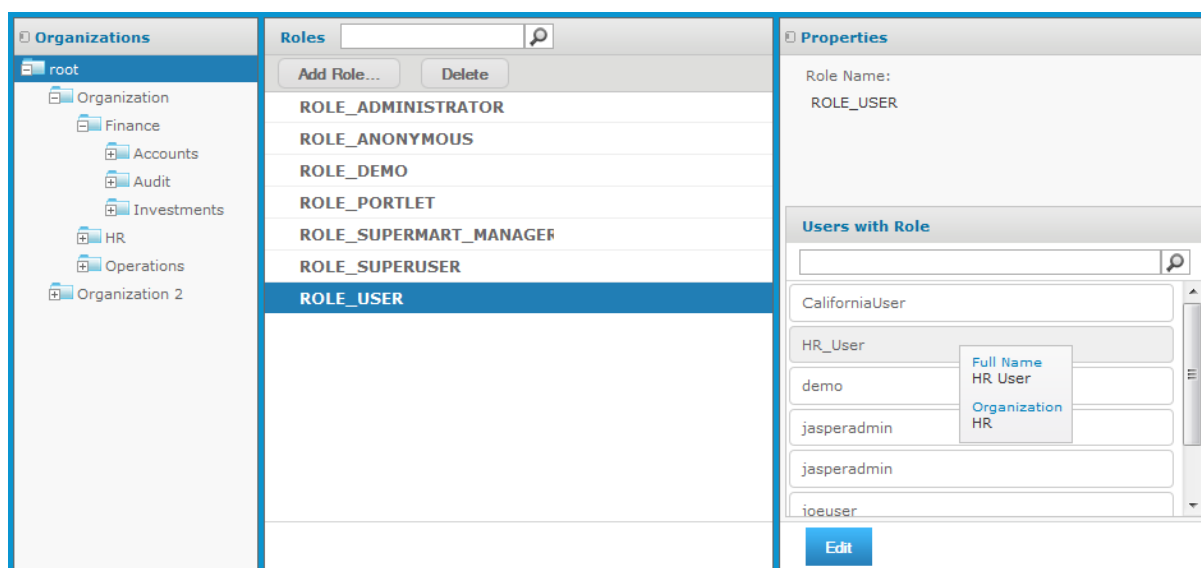
**Figure 2-11** **Manage Roles Page**

The Roles list includes all roles in the chosen organization and its suborganizations along with the five default system-level roles. The same role name may appear more than once if roles with the same name were created in different organizations. The second column (blank in this figure) gives the organization name of a particular role.

In this example, the system admin can see all roles in all organizations by selecting the root of the Organization hierarchy.

3. To select a role, click its organization in the Organizations panel. (Commercial users only. Community users skip to step 4.) The Roles panel is displayed.

4. Click the role in the Roles panel.

   To filter the list of roles, enter a search string in the **Search** field of the Roles panel. The search results show all of the roles in the selected organization and suborganizations whose names contain the search string. If necessary, scroll through the new list or refine your search.

5. Select the role in the Roles panel. The role's properties appear in the Properties panel.

   The Properties panel shows the role name, the organization where it's defined, and the users assigned the role. Hover over a user ID to see a user's full name and organization, as shown in **Figure 2-11**.

> When you view the properties of a special system-level role, you only see the users with that role in your organization or any suborganization. An organization admin can never see users outside of his organization or its suborganizations.

2.3.2   Creating a Role

1. Log in as a user with administrative privileges for the organization in which the role will be used.

2. Select **Manage > Roles** or, on the Admin Home page, click **Roles**.

3. In the Organizations panel, select the organization to which the role will belong. (Commercial users only. Community users skip to step 4.)

4. Click **Add Role**. The Add Role dialog appears.

Add Role

Role Name (required):

ROLE_DIRECTOR

Add Role to organization_1   Cancel

Figure 2-12 Adding a Role

5. Enter the name of the role.

6. Click **Add Role to <organization>** (**Add Role** for community edition users) to create the role.

   The new role is included in the Roles panel. If you want to assign users to the role, click **Edit** in the Properties panel of the new role.

2.3.3   Assigning Users to a Role

You can assign multiple users to one role. To assign multiple roles to one user, edit the user's properties as described in **"Editing a User" on page 34**.

1. Log in as a user with administrative privileges for the organization in which the role is defined.

2. Select **Manage > Roles** or, on the Admin Home page, click **Roles**.

3. In the Organizations panel, select the role's organization. (Commercial users only. Community users skip to step 4.) The Roles panel is displayed.

4. Select the role in the Roles panel.

> Unless you're logged in as the system admin, you can't edit or delete the five special system-level roles.

5. In the Properties panel, click **Edit**. The role's properties become editable. .

Figure 2-13 **Editing the Members of a Role**

6. Enter a different name to change the role name throughout the server.

> Permissions in the repository that use the role name are automatically updated. However, role names in security files for Domains and OLAP are *not* updated with the new role name and may cause a security risk. If you use security files for Domains or OLAP, do not change role names without verifying the files as well. For more information, see the *JasperReports Server User Guide*.

7. To assign or remove role users, select the users, and click the arrow buttons between the Users Available and Users Assigned lists.

8. Click **Save** to keep your changes, or **Cancel** to quit without saving.

2.3.4   Deleting One or More Roles

1. Log in as a user with administrative privileges for the organization in which the role is defined.

2. Select **Manage > Roles** or, on the Admin Home page, click **Roles**.

3. In the Organizations panels, select the role's organization. (Commercial users only. Community users skip to step 4.) The Roles panel is displayed.

4. Select the role in the Roles panel. Use Control-click and Shift-click to make multiple selections.

> Unless you're logged in as the system admin, you can't edit or delete the five special system-level roles.

5. In the tool bar of the Roles panel, click **Delete** and confirm the action.

CHAPTER 3   REPOSITORY ADMINISTRATION

JasperReports Server provides a powerful and flexible environment for deploying and running JasperReports. The repository stores all the resources used to run and create reports, including data source definitions, JRXML files, datatypes, and helper files such as images. Administrators create the folders and resources so users can create, run, and save the reports they need. For administrators who want to customize the user interface, the repository also holds the CSS and image files that define themes.

This section describes functionality that can be restricted by the software license for JasperReports Server. If you don't see some of the options described in this section, your license may prohibit you from using them. To find out what you're licensed to use, or to upgrade your license, contact Jaspersoft.

The repository is structured as a hierarchy of folders based on the hierarchy of organizations. The JasperReports Server web interface enables you to browse the repository's resources, manage its folder structure, and secure its contents. This chapter covers the basic tasks of administering the repository, including:

- Creating folders and organizing repository objects.
- Managing references to data sources, images, fonts, and other resources upon which reports rely.
- Controlling access to resources in the repository through roles and object-level permissions.

Further information about the repository is available in the following sections:

- **"Overview of the Repository" on page 14**
- **"Other Resources in the Repository" on page 95**
- **"Themes" on page 119**

You can also access the repository using the web services (see the *JasperReports Server Web Services Guide*) and APIs (see the *JasperReports Server Ultimate Guide*).

This chapter contains the following sections:

- **Resource Types**
- **JasperReport Structure**
- **Managing Folders and Resources**
- **Multiple Organizations in the Repository**
- **Permissions**

3.1   Resource Types

Resource types are based on various reports and report elements that users and administrators store in the repository.

Two fundamental resource types are those end users can create, and those administrators must create. Procedures for end users to create resources are described in the *JasperReports Server User Guide*. End users create the following resources:

Table 3-1 Resources Created by End Users

Resource Type	Description
Ad Hoc view	Professional edition only. Users Create Ad Hoc views interactively in the Ad Hoc Editor by dragging and dropping columns of data onto a table, chart, or crosstab. Users can then explore their data by applying filters and performing pivot operations. An Ad Hoc view can also be saved as an interactive report and shared with other users.
Dashboard	Professional edition only. A collection of reports, input controls, graphics, labels, and web content displayed together. Users create dashboards interactively in the Dashboard Designer and save them in the repository.
Content resource	Report output of any format, either from running a report in the background or from scheduling a report. A content resource is a simple file that the repository allows users to view or download.
JasperReport or simply report	A report combines a JRXML file, a data source, and optional components such as input controls. Users can run reports in the server. Depending on the use case, both users and administrators create JasperReports in the server. For more information, see **"JasperReport Structure" on page 46**. Optionally, reports may also store a snapshot of the report data to improve performance when many users access the same reports.
Report version	Professional edition only. Reports with input controls allow you to save combinations of input data so that you can run a custom version of the report directly. In the repository, report versions are always listed under the original report.

The other types of resources are all created and managed in the repository by administrators. The following resources generally support the report creation:

Table 3-2 Resources Created by Administrators

Resource Type	Description
Data source	A connection that points to a database or other data store. Data sources define where data is stored for reporting. There are several types of data sources, based on the type of connection or location of the data: JDBC, JNDI, and bean data sources. For more information, see **"Data Sources" on page 65**.

Resource Type	Description
Datatype	A basic resource that defines the format for input values, for example text, number, or date. A datatype may also specify a valid range for the input value.
Domain	A metadata layer that selects, joins, and filters tables and fields from your data. A Domain can be the basis for an Ad Hoc report. Domains also support row-level security and localization of labels. Domains are further documented in the *JasperReports Server User Guide*.
File	A resource that stores data in the repository. **"File Resource Types" on page 115** gives the list of supported file formats and their purpose.
Input Control	A complex type that specifies the values users can input for a report and how the input field appears when running the report, for example radio buttons or check boxes. Input controls depend on datatypes or lists of values to specify the format of the input.
List of Values	A basic resource that defines a list of arbitrary labels for input. Each label is associated with a value that can correspond to your data. For example, the Month Names List in the sample data associates the name of each month with a value 1 to 12.
Query	A database query string, for example in SQL. The JRXML doesn't necessarily include the query, in which case, you must define a query resource for use in the JasperReport.

Administrators can also manage OLAP resources in the repository, if their license supports Jaspersoft OLAP. For more information about OLAP and Mondrian resources, see the *Jaspersoft OLAP User Guide*.

Table 3-3 OLAP Resources Created by Administrators

Resource Type	Description
Mondrian XML/A Source	A server-side XMLA source definition of a remote client-side XML/A connection.
OLAP Client Connection	Specifies how to retrieve data for an OLAP view. An OLAP client connection is either a direct Java connection (Mondrian connection) or an XML-based API connection (XML/A connection).
OLAP View	If you implement Jaspersoft OLAP, a view of multidimensional data based on an OLAP client connection and an MDX query. Like JasperReports, OLAP Views are collections of individual resources that define how to access and present the data.

3.2   JasperReport Structure

The repository resource that aggregates all information needed to run a report is called a JasperReport. A JasperReport is based on a JRXML file that conforms to the JasperReports open source library the server uses to render reports.

A JasperReport is a complex resource composed of other resources:

•   The main JRXML file that defines the report
•   A data source that supplies data for the report.
•   A query if none is specified in the main JRXML.
    •   The query may specify its own data source, which overrides the data source defined in the report.
•   Input controls for parameters users can enter before running the report. Input controls are composed of either a datatype definition or a list of values.
•   Any additional file resources, like images, fonts, and resource bundles referenced by the report template.
•   If the report includes sub-reports, the JRXML files for the subreports.

The collection of all the resources referenced in a JasperReport is sometimes called a report unit. End users usually see and interact with a JasperReport as a single resource in the repository, but report creators must define all of the component resources.

3.2.1   Referencing Resources in the Repository

There are several ways to define and reference all the resources in a JasperReport.

In environments without JasperReports Server, reports are stored in the file system, and shared resources are usually stored on a network drive accessible to all developers and users. This solution is sometimes impractical, as you can't always add such resources to the classpath, and the use of absolute paths has its own limitations. Also, storing the resources in the file system discourages their reuse. Developers may invest time recreating resources because they don't know they exist.

By storing resources in the repository, JasperReports Server makes it easy and reliable to share resources like images, style templates, and subreports among reports. The repository mimics a folder and file structure, so references to external files can be handled as references to external resources in the repository.

In versions of JasperReports Server before 5.5, the JasperReports used the `repo:` syntax. As of 5.5, this is no longer necessary, and regular file paths are recognized and managed within the repository.

When you upload your JRXML to the repository, your file references become valid repository references, and you can store all your resources in well-known locations in the repository. This simplifies the process of uploading your reports, because you don't have to upload the resources each time. Also, you can manage these resources either through Jaspersoft Studio, the JasperReports Server user interface, or the server's APIs. For example, when you update a logo image, all reports that reference that resource display the new logo.

3.2.2   Absolute References

Absolute references are the URIs of resources in the JRXML of a report. A path may refer to the file system where the JRXML was created, but when uploaded to the server, it refers to folders in the repository.

The path must start with one of the following:

    / to represent the root of the repository in the community edition, or the current user's organization in commercial editions. For example, `/images/logo` is the path to a resource in the /images folder.

`../` to represent the folder where the JasperReport is uploaded. For example, `../myLogo` is the path to a resource in the same folder as the JasperReport.

As with a file system path, the repository path is composed of the resource ID of every parent folder, ending with the ID of the resource. Jaspersoft Studio supports absolute references by allowing you to drag resources from the repository tree view into the design area.

> If you implement organizations, the absolute path is relative the user's organization, as described in **"Multiple Organizations in the Repository" on page 55**.

When uploading the JRXML with absolute resource references as part of a JasperReport in the server, you need to ensure only that the resource with the given path exists in the repository before running the report. When the report runs, the server locates the resource in the repository and uses it to render the report.

Because file resources such as images, fonts, and JARs are the only resources for which you can create references directly in JRXML, they are the only resources for which you can create absolute references.

One disadvantage of absolute references is that JasperReports Server does not maintain the dependency between the JRXML and the absolute reference. When uploading the JRXML, there is no warning if the resource does not exist, and the server allows you to delete the resource from the repository even if it is still being referenced. If the resource is not available, running the report fails with an error.

3.2.3   Local Resources and External References

JasperReports Server provides more flexibility and power when you use indirect references instead of absolute references. Indirect references are placeholder names that must be manually linked to the resource when uploading the JasperReport. The syntax for an indirect reference contains only a resource placeholder name, for example:

```
logoImage
```

When you upload a JRXML with this reference, the server prompts you to provide the resource. You have two options:
- Create a new resource, in this case by uploading an image that becomes part of the JasperReport. This is called a local resource. You can't access this resource from elsewhere in the repository, it exists only within the JasperReport.
- Select a resource from the repository. This is called an external reference because it is external to the JasperReport. Any number of reports can link to the same external resource, and the resource can be managed independent of them.

While indirect references require slightly more work than absolute references in the JRXML, the server manages the dependency. Local resources exist as part of a JasperReport, and external references cannot be deleted until they are no longer referenced.

In cases where you don't want to reference existing resources, local resources allow reports to be highly customized and self-contained. A local resource defined inside the JasperReport has all the same properties as a repository resource, but it's not accessible in the repository. Users must edit the JasperReport to access any resources it defines locally.

> Users who are not administrators can create JasperReports but not other resources in the repository. Therefore, if an administrator does not provide them resources for external references, their only option is to create local resources for all dependencies within the JasperReport.

Indirect references are used implicitly in several other cases when you define a JasperReport:

- The main JRXML itself is either a local resource created by uploading a file or an external reference to an existing JRXML file resource in the repository.
- Every report must have a data source, and JasperReports Server gives you the option to either create a new local resource or use an external reference to an existing data source.
- Every report must also have a query that matches its data source. You may choose to create a local query resource or use an external reference to an existing query.
- Parameters in a report are implicitly handled as an indirect reference to an input control. For every parameter named in your main JRXML, you must define an input control either as a local resource or external reference.

Every level of indirect referencing is independent of the other. For example, when creating a JasperReport, you may choose to create an input control as a local resource, but that input control may have an external reference to its datatype. The server still manages the dependency between the local input control and the datatype resource in the repository.

Local resources and external references are used throughout the repository, for example when creating input controls, query resources, Domains, and OLAP resources.

3.2.4   References in Subreports

A subreport is a subordinate JRXML file within a JasperReport. As with all other resources referenced by the main JRXML, the subreport JRXML file may be specified by an absolute reference, a local resource or an external reference.

As a JRXML file, a subreport can reference other resources of its own. However, the subreport is run as part of the main JRXML, and any references in the subreport are interpreted relative to the JasperReport resource (represented by the main JRXML) and the context in which the JasperReport is being run.

3.2.5   Data Snapshots

Since JasperReports Server 4.7, report resources can also store a snapshot of the report data. A snapshot is a copy of the data that the query returns when the data is refreshed. This data snapshot is an internal structure not visible or accessible from the repository. However, when data snapshots are enabled, a data snapshot is stored in the repository with each report. When users open a report, the report viewer retrieves and displays from the snapshot. Users then have the option of refreshing the data in the report viewer, and if they have permissions, saving the data snapshot back into the report resource.

For more information about interacting with data snapshots, see the *JasperReports Server User Guide*. To enable snapshots, see **"Enabling Data Snapshots" on page 194**.

3.3   Managing Folders and Resources

Administrators and users with the proper permissions can create, modify, move, and delete folders and resources in the repository. The specific roles and permissions of the user determine the actions available. For the definition of the permissions on folders and resources, see **"Permissions" on page 57**.

One responsibility of a JasperReports Server administrator is to set up an environment for users to create and save dashboards, Ad Hoc views, and reports. That usually means creating a folder structure where users have

write permission. Users with write permission can also create their own sub-folders to store their reports and dashboards.

Only administrators can create shared data sources and other resources in the repository. So, although users with write permission can upload JRXML files and define resources inside JasperReports, they can't add these resources to the repository and share them. For this reason, we recommend that administrators create shared data sources and resources in the repository. For example, updating one shared logo file is better than many users editing their reports.

3.3.1 Resource IDs

Each resource, including each folder, has an ID, a name, and an optional description:
- The ID is used internally to reference a resource. The ID must be unique within its folder, but may exist in multiple folders.
- The resource name is displayed in the repository.
- The description, if defined, appears in the repository and in tooltips.

As in a file system, the IDs of nested folders containing a resource define the path to the resource. For example, the path to a report might be: /reports/samples/Freight. The path of every resource is shown under its name in the repository listing or search results.

To view the name and resource ID of a resource, Right click the folder's name or the resource in the repository or search results and select **Properties...** from the context menu.

Figure 3-1 **Resource Properties Dialog for a Writable Resource**

If you have write or administer permission as shown in the figure, you can also edit the name and description of the resource. For some operations such as export, you need the path, also called repository URI, which you can copy from this dialog.

3.3.2   Creating Folders

Any user with write permission on a folder can create new sub-folders.

To create a folder:

1. Log on as a user who has write permission to the parent folder.
2. Select **View > Repository** and locate the parent folder in the Folders panel.
3. Right-click the parent folder and select **Add Folder** from the context menu. The Add Folder dialog appears.
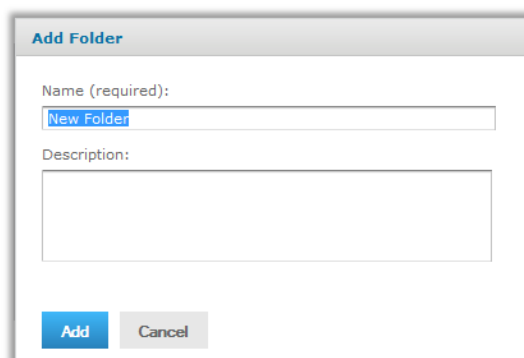
Figure 3-2 Add Folder Dialog

4. Enter the folder name and, optionally, a description, then click **Add**.

    The folder is created in the repository. The name appears in the hierarchy of folders. The description is visible only when viewing the properties of the folder, as shown in **Figure 3-1**.

    New folders and their future contents inherit the permissions of their parent folders. Users with Administrator permissions can change permissions folders, as described in **"Setting Permissions" on page 60**.

3.3.3   Adding Resources

Each resource has different requirements, for example some are created from uploaded files, others are created by defining values in a wizard. The procedures for adding each type are available in the resources below:

*   Interactive resources such as Ad Hoc views and dashboards are described in the *JasperReports Server User Guide*.
*   JasperReports are covered in the *JasperReports Server User Guide*.
*   Domains are covered in the *JasperReports Server User Guide*. Domains are only available in the Commercial Edition of JasperReports Server.
*   Mondrian and OLAP resources are covered in the *Jaspersoft OLAP User Guide*.
*   Data sources, queries, input controls, and file resources are explained in the chapter **"Other Resources in the Repository" on page 95**

Most resources are created through the Add Resource menu item on the context menu for folders in the repository. In the following figure, you can see the full menu and submenu with all the resources administrators can create:
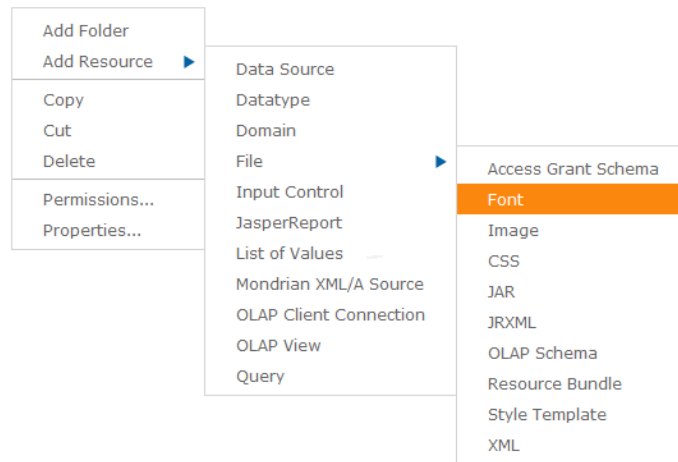
Figure 3-3 **Add Resource Context Menu Expanded**

For every resource you create, you must specify a name and resource ID for referencing the resource in the repository. each wizard also has one or more pages for specifying the values and controls specific to the resource.

> New resources inherit the permissions of the folder in which they are created. Administrators can change the permissions on the new resource, as described in section **"Permissions" on page 57**.

3.3.4   Renaming Folders and Resources
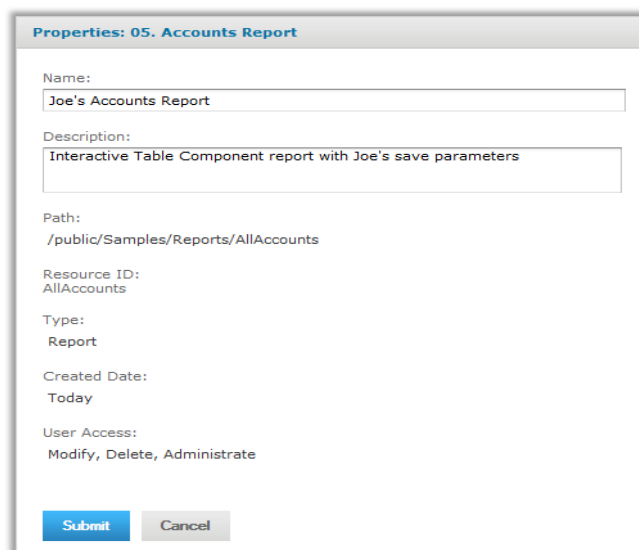
Any user with write permission on a folder or resource can change its name and description.

> You can't change the name of an organization's top-level folder in the way described here. The name of the top-level folder defaults to the name of the organization. So to change the name of the folder, you have to change the name of the organization, as described in section **"Editing an Organization" on page 28**.

To rename a folder or resource:

1. Log on as a user who has write permission for the folder or resource.
2. In the repository, browse or search for the resource. For renaming folders, select **View > Repository** and locate the folder.
3. Right-click the object and select **Properties...** from the context menu. The Properties dialog appears.

Figure 3-4 Properties Dialog for a Report Resource

You can change the folder or resource's name and description, but not the ID or the resource type; the ID is permanent once the resource is created, and the type, displayed here, is information only.

4.   Click **Submit** to save your changes.

3.3.5   Copying and Moving

The repository interface lets any user with the proper permission copy or move both resources and folders. Copying requires read permission on the source, moving requires delete permission on the source, and both require write permission on the destination folder.

You can drag-and-drop the objects, or you can copy-paste or cut-and-paste them from their context menus. Folders must be moved one at a time, but multiple resources from the same folder can be copied or moved together.

Copying and moving actions are not possible on the search interface, only on the repository interface showing the list of folders. Currently, it's not possible to create a copy of a resource in the same folder.

The moved objects inherit their permissions from the destination folder; they do *not* keep the permissions they had before the move. If you want the objects to have other permissions, you can set new permissions after the move (see **"Permissions" on page 57**).

To copy or moving folders and resources:
1.   Log on as a user who has the required permissions for the folder or resource.
2.   Click **View > Repository**, and expand the folders to display the object to be copied or moved.
3.   Right-click the resource and select **Copy** or **Cut**. (Delete permission is required to move a resource.)
     You can select multiple resources with Control-click or Shift-click, but only a single folder.
4.   Right-click the destination folder and select **Paste** in the context menu. If the **Paste**. (Write permission on the destination folder is required.)

Alternatively, you can drag to move the selected resource or folder to the destination folder. To copy, press and hold the Ctrl key then click and drag. When dragging resources, the destination folder is highlighted in blue if you have write permission to it.

3.3.6   Editing Resources

Resource editing methods differ by the resource type. For end users who work with interactive resources, editing mostly involves the **Open in Designer** action on the context menu for dashboards and Ad Hoc views. The procedures in the following table are further described in the *JasperReports Server User Guide*.

Table 3-4 Resources End Users Can Edit

Resource Type	How to Edit
Ad Hoc view	Professional edition only. Users select **Open in Designer** and modify the view interactively. After changing the content, users can overwrite the existing view or save as a new view.
	Reports created from Ad Hoc views are saved in the same format as JasperReports, but the resources referenced in the report unit are generated by the Ad Hoc editor and should not be modified. There is one exception: administrators may create a JSP file and set it as a custom report view.
Dashboard	Professional edition only. Users select **Open in Designer** and modify the dashboard interactively. After changing the content, users can overwrite the existing dashboard or save it as a new dashboard.
Content Resource	Report output is a file stored in the repository. These files cannot be edited, only downloaded or deleted.
JasperReport	Users or administrators select **Edit** and change the data source, input controls, or file resources referenced in the JasperReport. For more information, see **"JasperReport Structure" on page 46**.
	When users **Run** a report, it's displayed in the interactive report viewer. If data snapshots are enabled, the report is displayed with data that was previously returned by the data source. When data snapshots are not enabled, the server queries the data source and runs the report's query. After interacting with the report's columns and values, users may save the report, either by overwriting the original or as a new copy, depending on their permissions.
Report Version	Professional edition only. Users may select **Edit** to change the values stored as input parameters.

Table 3-5 Resources Only Administrators Can Edit

Resource Type	How to Edit
Data Source	Administrators select **Edit** from the context menu on these resources. You edit these resources in the same dialog you used to create them.
Datatype	You can view the current definition of a resource or change its values. For example, you could view the datatype of an input control, change a query, or upload a new file in a file resource.
Domain	For a Domain, you have access to the Domain Designer used when creating the Domain. You can add tables and fields, change filters, or change the display characteristics of items in the Domain. For more information about Domains, see the *JasperReports Server User Guide*.
File	
Input Control	
List of Values	For all other resource types, see the procedure for creating them in **"Other Resources in the Repository" on page 95**.
Query	

When editing a resource, you have several limitations:

- You can't change a resource's ID. If you need to change an ID, you have to create a new resource and delete the old one.
- You can't change the location of a resource. To change the location of the resource, see **"Copying and Moving" on page 52**.
- For file resources, you can't see the name of the file that was uploaded, nor in most cases download and view the contents of the file. Your only option is to upload a new file to replace the old one.

3.3.7   Deleting Folders and Resources

Users with delete permission on folders or resources can delete those objects from the repository. To delete a folder, you also need delete permissions on all the resources and folders in that folder. Folders must be deleted one at a time, but multiple resources can be deleted together.

> You can't undo a delete.

The repository won't allow you to delete resources currently referenced by other resources. For example, an input type used by a report or a properties file used by a Domain cannot be deleted as long as the report or Domain still references them.

To find the resources that reference the one you want to delete, you need to look at each report, view, Ad Hoc Topic, or Domain that you suspect of referencing it. When you edit the definition of a JasperReport or a Domain, you can see the resources it references. Then you can either remove the reference from the resource or delete the entire resource containing the reference.

To delete a folder or resource:

1. Log on as a user who has delete permission for the folder or resource.
2. In the repository, browse or search for the object to be deleted.
3. Right-click the object and click **Delete** in the context menu.

In the repository view, you can select multiple resources and click Delete in the tool bar or in the context menu. In the list of folders, you can delete only one folder at a time, but all contents of that folder will be deleted. In the search results, you can select multiple resources and right-click to select Delete in the context menu.

3.4 Multiple Organizations in the Repository

If you implement multiple organizations, there are certain considerations when designing the repository structure for your deployment.

Multiple organizations are available only in the professional edition of JasperReports Server.

3.4.1 Organization Folders

In the repository, each organization has its own branch, contained in a folder named after the organization. JasperReports Server automatically restricts a user's view and access to their organization's folder. Organization admins can create any folder structure needed within the organization.

Top-level organizations are contained in a folder called Organizations at the root of the folder hierarchy. The top folder of an organization is contained in a folder called Organizations in the parent organization. Administrators can view and browse the Organizations folder, and the folders of any suborganizations. As administrators of the parent organization, they can also create folders and resources in suborganizations.

> By default, users in an organization can also view and create folders and resources in any suborganization. An administrator can prevent this, by changing the folder permissions.

The Organizations folder in every organization is a special folder that is managed by the server. Administrators can't create folders or resources directly in the Organizations folder. The server creates the folder for each suborganization when the administrator creates a new organization through the **Manage > Organizations** page. Administrators can create folders and resources in the Folder Template folder in the Organizations folder; these resources are copied into new organizations. For more information, see **"Default Folders for Organizations" on page 28**

3.4.2 Design Considerations

Careful design of the JasperReports Server repository leads to a clear and robust BI environment and easy, secure access for users. One of the main decisions is how you want your organizations and users to access resources: which resources are shared across organizations as opposed to those specific to an organization. Consider these use cases:

- Organizations have private resources - Separate data sources, reports, OLAP views etc. These private resources would be stored in each organization's own folders, and perhaps only a few resources such as company logos would be shared among them.
- Organizations share resources - Resources are kept in the public folders where they can be used by all organizations and users. You may have common data sources and reports across customers, but the underlying data is partitioned by organization. Data level security restricts what users see when running public reports and OLAP views.
- Organization share resources, but have some customizations - For example, users in the organization create reports that are private and stored locally, but they access resources in the public folders.

- Organizations are hierarchical - One organization contains others. By default, the parent organization can access all the resources of its child organizations. If you don't want this, you must avoid creating suborganizations or customize the server's multi-organization architecture.

3.4.3   Referencing Resources in the Repository

All resources in the repository can be referenced by Universal Resource Identifiers (URIs) that specify resource names and folder paths. Because of the hierarchy of organizations, references are relative to the user accessing them. JasperReports Server transforms relative references into actual resource locations in the repository based on the user's organization and the organization's main folder. By default, folder locations are transformed in the following ways:

- For organization members, locations in /public are not transformed, but those in the organization's main folder are transformed to the actual location, for example, /organizations/organization_1.
  So if a user in org_1 runs a report that references /images/myLogo image resource, the actual path in the repository that is fetched is /organizations/org_1/images/myLogo. If the report also references /Public/sharedLogo, the server fetches the resource in /Public/sharedLogo.

- For system admins, locations throughout the repository are not transformed. They see the actual repository path names.
  If a system admin runs the same report in the example above, the reference to /images/myLogo attempts to fetch a resource named /image/myLogo, which works only if there is a folder at the root of the repository named images with a copy of the myLogo. The report fails (or is missing a resource) when run by the system admin, unless he logs in as that user through the **Manage > Users** page.

This transformation enables URIs to reference different resources depending on the organization of the user who accesses them. For example, a report may have an organization-specific logo as an image. We can set up the report as follows:

- Logo URI specified in the JRXML: /images/orgLogo. When transformed for each user, the URI accesses a location relative to his organization's main folder.
- Every organization using this report must have a folder named /images containing an image resource with the ID orgLogo. When a user in any organization runs the report, the server fetches the organization-specific image and displays it.

There are three exceptions to references being transformed. In these cases, the references must be literal:

- In report units, references to data sources, JRXMLs, or input controls.
- In OLAP views, references to OLAP connections.
- In OLAP connections, references to data sources or schemas.

Also, because these references are not transformed, you must observe the following:

- For maintenance tasks on an organization's report units, OLAP views and OLAP connections, you must log into that organization and do the tasks there. You cannot administer the resources as superuser or another organization's admin.
- The three resources (report units, OLAP views and OLAP connections) cannot reference objects across organizations or even in their own parent organization. The reference would not be transformed; it would be taken literally and would fail. For example, if the data source for a report unit is in the /dataSources folder of This_Org, users in That_Org cannot access it because their reference cannot cross organizations.

> To test the absolute references, you should log in as an admin of the organization using the references. See **"Testing User Permissions" on page 62**.

3.4.4   Best Practices

Here are some best practices for resources in a repository shared by multiple organizations:

- The system admin must log in as an organization user to maintain or run an organization's resources.
- Resources with absolute references to resources in organization folders work only for the users within the organization or the parent organization.
- If a JRXML that accesses organization resources with URIs must run across organizations, all organizations must have identical folders, object names, and expected object types for those resources.
- The public folder should be used for resources that are shared across organizations.

3.5   Permissions

Permissions on folders and resources determine what users see in the repository and what actions they're allowed to perform. In the following table, the actions granted by each permission include all of the actions granted by permissions above it, except for the No Access permission. The actions granted by each permission strictly exclude all of the actions granted by permissions below it.

Permission	Actions Granted on Repository Folders and Resources
No Access	Users can never see or access the folder or resource either directly in the repository or indirectly when running a report, dashboard, or OLAP view.
Execute Only	Users can never see the folder or resource in the repository, but the reports, dashboard, or OLAP views that they run can access them.
Read Only	- See the folder or resource in any JasperReports Server dialog - See the properties of a folder or a resource - Copy a folder and all of its readable contents - Copy resources individually or in bulk - View (run) a report, dashboard, or OLAP view - Run a report in the background - Schedule a report to run later
Read + Delete	- Cut (move) a folder and all of its contents - Delete a folder and all of its contents - Cut (move) resources individually or in bulk - Delete resources individually or in bulk
Read + Write	- Save report options for a JasperReport - Delete report options - Copy resources to a folder with this permission - Edit resources

Permission	Actions Granted on Repository Folders and Resources
Read + Write + Delete	• Add a subfolder • Paste into a folder (copy or cut) • Save a new Ad Hoc view, report, or dashboard in a folder • Save the output of a scheduled report in a folder • Rename a folder or resource and change its description string • Open an Ad Hoc view in the Ad Hoc Editor or a dashboard in the designer • Modify and overwrite an existing Ad Hoc view, report or dashboard • Add a JasperReport resource to the repository (upload a JRXML) • Edit the definition of a JasperReport resource in the repository (replace the JRXML)
Administer	• Set the permissions (by role and by user) on a folder or resource. This effectively delegates certain repository administration tasks.
Administer and ROLE _ADMINISTRATOR	• Add (create) a resource in a folder • Edit a resource, for example the components of a report unit or a Domain

Permissions apply when browsing or searching the repository and when using any dialog that accesses the repository, like browsing folders to save a report. Note that:

• Copying does *not* preserve the permissions on an object. Users may copy a read-only object, paste it into a read-write folder, then edit the object. For more details, see **"Copying and Moving" on page 52**
• Copying and cutting (moving) actions can be completed only by a user with Read + Write + Delete access to the folder in which the object is pasted. For more details, see **"Copying and Moving" on page 52**
• Cutting, deleting, and setting permissions on folders is allowed only if the user has the same permission on all folder contents. Cutting and deleting resources in bulk is allowed only if the user has at least Read + Delete permission on all selected resources.
• Deleting a resources is allowed only if no other resources rely on them. For more details, see **"Deleting Folders and Resources" on page 54**

3.5.1   Inheriting Permissions

According to the permission architecture, there is a permission setting for every user and role on every folder and resource in the repository. To simplify the definition of permissions, JasperReports Server supports the inheritance of permissions from the parent folder of a folder or resource. If no permission is explicitly defined for a user or role on a given folder or resource, the user or role has the same access permission that is defined on the parent folder. When a permission is defined explicitly, that permission is enforced, regardless of those on the parent folder.

Using permissions, you can manage large hierarchies of content and keep them secure. When you set a permission explicitly, that permission for a given user or role is inherited recursively by all of the folder's contents and subfolders, unless they have an explicit definition of their own. Permissions assigned on an organization's top folder are inherited across the entire organization. Permissions set on the root folder or Organizations folder by the system admin are inherited across multiple organizations.

For example, the system admin can make all organizations read-only by default to ordinary users, and each organization admin can make specific folders writable so that users can store their reports and output.

3.5.2 Cumulative Permissions

Because permissions can be assigned to both users and roles, a user belonging to one or more roles may have multiple permissions defined or inherited on any given folder or resource. In fact, every permission must be defined on the root, even if it has the default value of No Access, and therefore every role- and user-based permission on every folder and resource has a setting through inheritance. So, for every folder or resource, every user has a their own user-based permission and the permission assigned to the ROLE_USER.

How does JasperReports Server determine the effective permission from the many that apply? Permissions in the server are strictly cumulative, meaning that the least restrictive among the set of all permission applies. Even if a more restrictive permission, such as No Access, is set explicitly, the less restrictive permission such as Read-Only applies, regardless of whether it is inherited or set explicitly.

3.5.3 Administrator Permissions

The JasperReports Server authorization architecture distinguishes between administrators and all other users. Administrators are defined as users with either ROLE_SUPERUSER (professional edition only), ROLE_ADMINISTRATOR, or both. By design, system administrators with the ROLE_SUPERUSER always have irrevocable Administer access to the entire repository, including the contents of every organization. The system administrator cannot modify the permissions for ROLE_SUPERUSER, to prevent being locked out or unable to administer some resource. Therefore, the system admin can set permissions for all other users, on any folder or resource, and in any organization if necessary. In particular, the system administrator can modify permissions for ROLE_ADMINISTRATOR, for example to share some resources across all organizations by making them read-only to everyone, including the organization admins.

Organization admins are organization users with the ROLE_ADMINISTRATOR, like the default jasperadmin created in every organization. By default, organization admins have the Administer permissions to everything in their organization, except what the system admin has changed to a lesser permission. However, organization admins cannot change the permissions granted to ROLE_ADMINISTRATOR, to prevent them from overriding the settings of the system admin and from locking themselves out of a folder or resource.

3.5.4 Execute-Only Permission

As in file systems, execute-only permission in JasperReports Server allows running reports, dashboards, and OLAP views to access a resource, but keeps the resource from appearing in the repository.

Execute-only permission applies to folders as well, keeping them from appearing in the folder tree when users browse the repository, yet still allowing the resources they contain to inherit the execute-only permission. This is useful for hiding folders and resources such as data sources that only administrators and data analyst roles need to access in the repository. However, if your execute-only folder contains read-only resources, those resource are hidden when browsing folders, but can be found, with a repository search.

As with all other permissions, execute-only permission is either role-based or user-based, so only certain users can access a resource from a running report.

> If you have data or sensitive content in a resource, always set No Access permission for users or roles that must not be able to access it.
>
> Hiding a resource with execute-only permission does not protect against access, because malicious users who find the resource ID may be able to create a report, dashboard, or OLAP view that extracts the sensitive content.

3.5.5   Default User Permissions

For all non-administrator users, the default permission at the root is No Access and any permissions must be explicitly defined. In practice, the default installation of the repository contains sample data with a mix of no access, execute only, read only, and read-write permissions that allow the sample users to access folders and resources. The sample permissions demonstrate a common approach to permissions, allowing users to see the resources they can access and hiding the ones they can't, while administrators have full access.

We recommend you familiarize yourself with permissions by viewing and setting permissions in the sample data, as described in the following section.

3.5.6   Setting Permissions

Administrators can assign permissions to access any folder or resource throughout the repository. Users with the Administer permission on a folder can assign permissions to that folder and any contents that inherit the permission. Users granted Administer permission to a resource can set the permissions only on that specific resource.

To set permissions on a folder or resource in the repository:

1. Log in as a user with administrative privileges.
2. Browse or search the repository for the folder or resource.
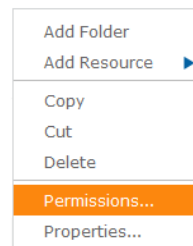
3. Right-click the object and select **Permissions...** from the context menu:

    The Permissions dialog opens showing the permissions in effect for the selected object. By default, it first shows the permissions given to roles. Permissions that are inherited from the object's parent are indicated by asterisks (*).
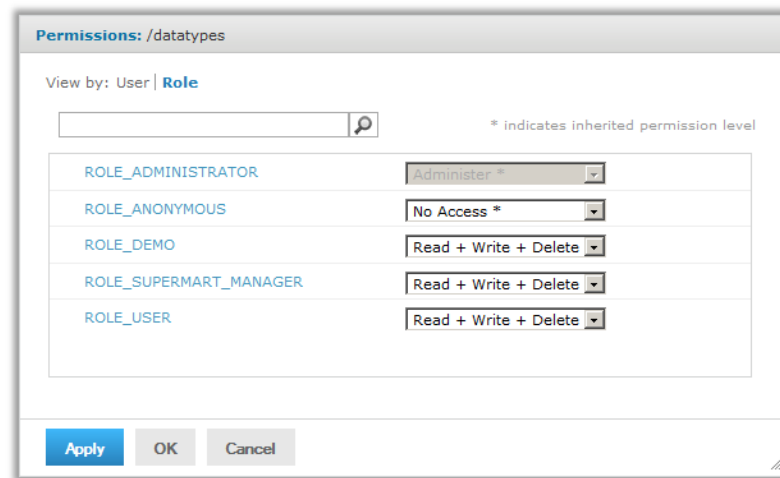
Figure 3-5 **Permissions Dialog Showing Permissions by Role**

In systems with multiple organizations, the users and roles displayed include only those within the scope of the user. For example, in the default single organization, the organization admin can't see the permission for the system admin (`superuser`) or for ROLE_SUPERUSER.

In the previous figure, you can see the default role-based permissions on the sample Input Data Types folder as seen by the organization admin (`jasperadmin`). Members of certain roles can see and modify the input data types stored in this folder; these roles likely correspond to users such as data analysts. Regular users have execute only permission so they do not see this folder, but the reports they run can access its contents. Administrators are prevented from changing the permission for their administrator role or user name, to prevent them from removing their ability to set permissions.

4. In the dialog, click **User** to view the permissions assigned to specific users. Click **Role** when viewing user permissions to toggle back.

5. For each user or role, you can select a new permission from the drop-down.

   In the next figure, you can see the default user permissions on this folder. In the default installation, all permissions are defined by role; so, all user permissions are No Access inherited from the root. The figure shows a read-only permission being granted to the sample end user. This enables `joeuser` to see but not modify the Input Data Types folder and its contents. For all other end users, but, the folder is still execute-only due to the settings in **Figure 3-5**.
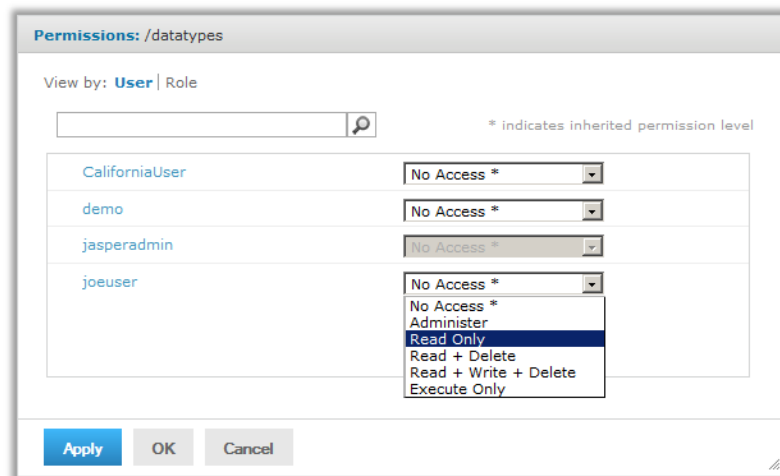
Figure 3-6 Permissions Dialog Showing Permissions by User

6. Click **Apply** to apply your changes. If you toggle between user and role permissions, first apply any changes you made.

7. Click OK to save your changes and close the permissions dialog when you're finished.

You can open several permissions dialogs for different resources or folders at the same time while navigating the repository. This helps when trying to set permissions uniformly across several folders or organizations.

> There are two special cases when setting permissions:
>
> • If a resource inherits a permission, for example Read-Only, you cannot set the permission to the same value, at least not directly. You need to temporarily change the permission level on the parent folder, then set the explicit permission, then set the parent folder's permission back to the original value. When a resource and its parent folder have been set to the same permission in this way, the permission dialog still shows the asterisk as if the permission were inherited. But if the parent is later given a different permission, for example Read-Write, the resource retains its explicit Read-Only permission instead of inheriting Read-Write.
>
> • To reset the permission level so that it once more inherits from its parent folder, select a different permissions level and click **Apply**, then select the permission with the asterisk and click **Apply** again.

3.5.7   Testing User Permissions

Once you have configured users, roles and permissions, we recommend that you test the permissions granted to a few representative users. We also recommend testing whenever you add new users, roles, and resources or make any major modifications to your access control configuration.

To test user permissions:

1. Log in as an administrator.
2. Select **Manage > Users**.
3. Select the user's organization, then browse or search for the user whose permissions you are testing.
4. In the Users panel, select the user.

5. In the Properties panel, click **Login as User**. The selected user's Home page appears. The login information in the upper-right corner shows that you are logged in as that user.

6. Verify that the expected folders and resources are available in the repository. Make a note of any objects that should be there but are not, and any that should be hidden but are displayed.

7. When you have verified the user's permissions, click **Log Out**. Your own Home page appears.

8. To change the user's permissions, edit the permissions in the repository and modify the user or role definitions.

9. Continue testing until the user's permissions are satisfactory.

10. Repeat these steps with several representative users to ensure that your access control is properly configured. An untested access control configuration can't secure your data adequately.

CHAPTER 4   DATA SOURCES

A data source is a resource in the repository that specifies how and where to obtain the data displayed by reports, Ad Hoc views, and OLAP views. Administrators must define data sources before uploading reports that rely on them. Typically, a data source specifies the URI of the database server and the details you need to access it, such as a user name and password.

This section describes functionality that can be restricted by the software license for JasperReports Server. If you don't see some of the options described in this section, your license may prohibit you from using them. To find out what you're licensed to use, or to upgrade your license, contact Jaspersoft.

JasperReports Server provides data source types for relational databases, most flavors of big data, and for specialized data such as Amazon Web Services and JavaBean data. Virtual data sources allow you to combine several data sources into one.

JasperReports Server can access any relational database that supports the SQL query language through the JDBC (Java DataBase Connectivity) API. In this case, you can configure two types of data sources in the repository:

- JDBC data source – Establishes a direct connection to the database server using its JDBC driver. After installation, JasperReports Server includes JDBC drivers to access the most popular databases: MySQL, PostgreSQL, Oracle, SQL Server, and DB2. As of release 5.6.1, the server also includes a JDBC driver that can access Hadoop-Hive. If the driver for your database is not installed, or you want to use alternate drivers, the system admin can upload and manage JDBC drivers through the UI. With JDBC data sources, JasperReports Server configures and manages the connections to the database. By default, the maximum number of simultaneous connections for each data source is 20.
- JNDI data source – Relies on the JNDI (Java Naming and Directory Interface) service of your application server to access a database connection. You must first configure your application server to install its JDBC drivers and configure its database connections. The configuration of the application server determines the number of shared connections. Note that the application server connects to the database using JDBC, meaning that JNDI data sources are available for all databases that support JDBC.

Big data stores that are not compatible with JDBC have custom data sources:

- Cassandra – This new data source introduced in JasperReports Server 5.6 is different from the community-contributed data source for Cassandra. This data source supports the Cassandra Query Language CQL 3.
- Cloudera 5 – This data source can access Apache Hadoop clusters through Cloudera 5 Hive or Impala. It uses the Hive Query Language (HiveQL), which is similar to SQL but distinct. All other Hadoop-Hive data sources should use the JDBC driver described above.
- MongoDB – This custom data source for MongoDB supports Jaspersoft's own MongoDB Query Language.

Finally, JasperReports Server also supports some specialized data sources:

- Amazon Web Services (AWS) data sources – Accesses data stored in your AWS data store using JasperReports Server, either on-premises or in the cloud.
- Virtual data source – Allows you to combine multiple data sources into a single data source and join them within a Domain. You can also wrap a data source for big data to be used in a Domain.
- Bean data source – Allows you to access data encapsulated in JavaBeans.
- Internal diagnostic data source – A custom data source for the server's own diagnostic data. The diagnostic information is available only to system admins (`superuser` by default). For more information, see **"Using the Diagnostic Data in Reports" on page 244**.

In the case of analysis data, JasperReports Server supports OLAP data sources (such as Mondrian and XML/A connections). For information about analysis data sources, refer to the *Jaspersoft OLAP Ultimate Guide*.

> You can extend JasperReports Server to support any custom data source. Custom data sources consist of Java implementation classes, a message catalog, and a Spring bean definition. For more information about custom data sources, see the *JasperReports Server Ultimate Guide*.

This chapter contains the following sections:
- **Attributes in Data Source Definitions**
- **JDBC Data Sources**
- **Managing JDBC Drivers**
- **JNDI Data Sources**
- **AWS Data Sources**
- **Cassandra Data Sources**
- **Hadoop-Hive Data Sources**
- **MongoDB Data Sources**
- **Virtual Data Sources**
- **Big Data Connectors for Virtual Data Sources**
- **Bean Data Sources**

4.1 Attributes in Data Source Definitions

You have two options for defining the parameters of a data source. You can either:
- Specifically define the parameters, like host, port, and database.
- Have the server derive these parameters at runtime based on attributes you provide.

Attributes can be used to derive all data source parameters that are not selected from drop down lists.

When defining a data source attribute, you can either specify the category in which the server should look for its value, or allow the server to locate the value hierarchically.

4.1.1 Resolving Attributes by Category

If you specify a category for the attribute value, for example user, the server will use the value specified in the logged in user's attributes for the data source parameter. If the attribute is not defined where specified, the report fails with an error. You can specify these attribute categories:
- User
- Tenant (organization)
- Server

4.1.2   Resolving Attributes by Hierarchy

If you don't specify a category for the attribute, the server searches attributes hierarchically and uses the value of the first profile attribute it finds. This search starts with the logged in user, then proceeds to the user's organization and, finally, the server level. If the specified attribute is not found, the report fails with an error.

The following figure is an example of attributes used to define data source parameters.

Figure 4-1 **Using Attributes for Data Source Parameters**

In the example above, the following data source parameters are specified by attributes:

- `{attribute('host','User')}` - Host will be derived categorically from the logged-in user's attributes, because the "User" category is specified.
- `{attribute('port','Tenant')}` - Port will be derived from the logged in user's organization attributes.
- `{attribute('port','Server')}` - Database will be derived from the server-level attributes.
- The URL defaults from the host, port, and database fields automatically.
- `{attribute('userName')}` - The user name will be derived hierarchically, because no category is specified. JasperReports Server will look for a host first in the logged in user's attributes, then in the organization attributes, and finally the server-level attributes.

- `{attribute('password')}` - The password field can also reference an attribute, here a hierarchical attribute.

For information about user attributes, see **2.2.4, "Editing User Attributes," on page 35**

For information about defining organization and server attributes, see **2.1.5, "Creating Attributes on Organizations," on page 29**

4.2 JDBC Data Sources

JDBC data sources are direct connections to your database managed by JasperReports Server. To create one, you must provide the URL and credentials to access your database, along with any database-specific configuration parameters.

JasperReports Server includes JDBC drivers for the most used databases. If your data base is not included, or if you want to use different JDBC drivers, the system administrator must upload the appropriate JDBC driver before creating a data source. For more information on JDBC drivers, see **"Managing JDBC Drivers" on page 70**.

To create a JDBC data source:

1. Log on as an administrator.

2. Click **View > Repository**, right-click a folder's name, and select **Add Resource > Data Source** from the context menu. Alternatively, you can select **Create > Data Source** from the main menu on any page and specify a folder location later. If you installed the sample data, the suggested folder is Data Sources. The New Data Source page appears.

3. In the Type field, select **JDBC Data Source**. The page refreshes to show the fields required for a JDBC data source.

4. Select the JDBC driver for your database. If your driver is listed as NOT INSTALLED, a system administrator must first upload the driver as described in **"Managing JDBC Drivers" on page 70**.

5. Enter the hostname, port, and database name for your database. The default hostname is localhost, and the default port is the typical port for the specified database vendor. The three fields are combined automatically to create the JDBC URL where the server will access the database.

   For more information about JDBC URLs, including optional parameters, see the troubleshooting section **"JDBC Database URLs" on page 258**.

   You have the option to use profile attributes to derive the values for data source parameters. See **"Attributes in Data Source Definitions" on page 66**

   As of JasperReports Server 5.6.1, TIBCO provides a set of JDBC drivers in the installed server. These drivers support a slightly different SQL syntax. If you see errors when running reports or creating domains that use scalar functions, see **A.13.6, "SQL Functions with TIBCO JDBC Drivers," on page 258**.

Figure 4-2 **Setting the JDBC Data Source Type**

6. Fill in the database user name and password. These are the credentials the server will use to access the database.

> Set the Time Zone field when the date-time values stored in your database do not indicate a time zone. When date-time values are stored in a format other than local time zone offset relative to Greenwich Mean time (GMT), you must specify a time zone so that the server can properly convert date-time values read from the target database. Set the Time Zone field to the correct time zone for the data in the database. The list of time zones is configurable, as described in **"Specifying Additional Time Zones" on page 278**.
>
> When in doubt, leave the Time Zone field blank.

7. Click **Test Connection** to validate the data source. If the validation fails, ensure that the values you entered are correct and that the database is running. To diagnose JDBC connection issues, you can turn on logging as described in the troubleshooting section **"Logging JDBC Operations" on page 256**.

Figure 4-3 Entering the User Name and Password

8.   When the test is successful, click **Save**. The Save dialog appears.

Figure 4-4 Saving the JDBC Connection

9.   Enter the data source name and, optionally, a description. the Resource ID appears based on the name you enter.
10. Expand the folder tree and select the location for your data source, then click **Save**. The data source appears in the repository.

4.3   Managing JDBC Drivers

To access a database from JasperReports Server using JDBC you need an appropriate driver that's accessible in the server's classpath. In JasperReports Server 5.6.1, the following drivers are installed by default:

- MySQL (org.mariadb.jdbc.Driver)
- PostgreSQL (org.postgresql.Driver)
- Oracle (tibcosoftware.jdbc.oracle.OracleDriver)
- Microsoft SQL Server (tibcosoftware.jdbc.sqlserver.SQLServerDriver)
- IBM DB2 (tibcosoftware.jdbc.db2.DB2Driver)
- Hive (tibcosoftware.jdbc.hive.HiveDriver)

Drivers for other databases can be downloaded from links on the Jaspersoft community website:

http://community.jaspersoft.com/wiki/downloading-and-installing-database-drivers

The system administrator (superuser) can add JDBC drivers for other databases in the following ways:

- During installation. For more information, see the *JasperReports Server Installation Guide*.
- At any time through the UI. As described in the procedure below, the system admin can add, replace, or remove JDBC drivers through the user interface, without needing to restart the server.

Only the system administrator can manage the JDBC drivers, but once uploaded, they're available to all administrators who create data sources.

> JBoss doesn't lacks the flexibility of uploading drivers on the fly. On JBoss, drivers that are not installed don't appear in the list below, and you must configure and restart JBoss to add a driver. For more information, see **"JDBC Drivers on JBoss" on page 257**.

To add a JDBC driver:

1. Log on as the system administrator (`superuser` on commercial editions, `jasperadmin` on community editions).
2. Select **Create > Data Source** from the main menu.
3. In the Type field, select **JDBC Data Source**. The page refreshes to show the fields necessary for a JDBC data source.
4. The dropdown selector for the JDBC Driver field shows the available JDBC drivers and those that are not installed.
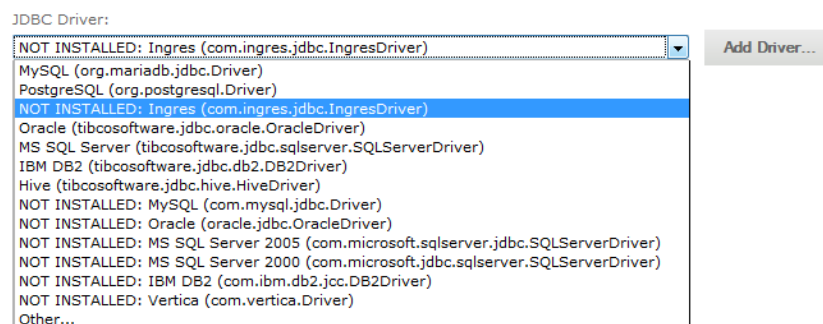
 Figure 4-5 Viewing the List of Available JDBC Drivers

5. If you want to add a driver that has not been installed, select it from the list, then click **Add Driver**. The Select Driver dialog appears.
6. If you have not yet obtained the driver, click the link for Downloading and Installing Database Drivers. That page on Jaspersoft's community website has links to the most commonly used JDBC drivers. After you download a driver to your file system, you can return to the Select Driver dialog.
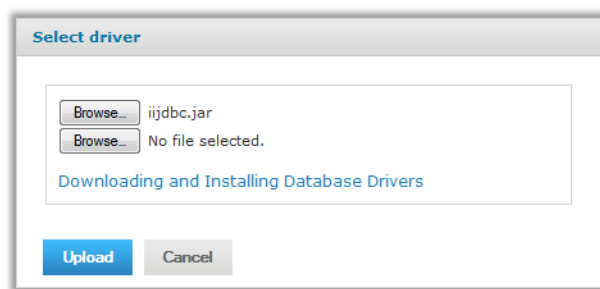
Figure 4-6 **Adding a JDBC Driver**

7. In the Select Driver dialog, click **Browse** to locate the appropriate driver JAR file. If your driver has more than one JAR file, click the **Browse** button that appears after selecting the first file.
8. Click **Upload** to install the driver and make it available immediately.

You can replace any driver that you upload with newer versions of the same driver. If you want to use the vendor's own driver instead of the TIBCO JDBC driver, you can install it as a new driver as described in the *JasperReports Server Installation Guide*.

To update a JDBC driver:

1. Log on as the system administrator (`superuser` on commercial editions, `jasperadmin` on community editions).
2. Select **Create > Data Source** from the main menu.
3. In the Type field, select **JDBC Data Source**. The page refreshes to show the fields necessary for a JDBC data source.
4. The dropdown selector for the JDBC Driver field shows the available JDBC drivers and those that are not installed.
5. To update a driver that has already been installed, select it from the list, then click **Edit Driver**. The Select Driver dialog appears and notifies you that selecting a driver will overwrite the existing one.
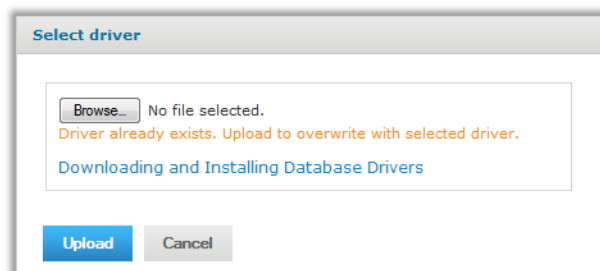
Figure 4-7 **Updating a JDBC Driver**

6. In the Select Driver dialog, click **Browse** to locate the new driver JAR file.
7. Click **Upload** to replace the existing driver and make it available immediately.
8. You can now use this driver to create a data source, and the driver will be installed when other administrators create data sources.

To remove an uploaded JDBC driver:

1. Log on as the system administrator (`superuser` on commercial editions, `jasperadmin` on community editions).
2. Select **View > Repository** and open the System Properties folder at the root.
3. Right click the GlobalPropertiesList resource and select **Edit** from the context menu.
4. Locate the driver you uploaded in the list of properties. The drivers with the value [SYSTEM] are the default drivers configured at installation time.
5. Click **Remove** beside the driver you want to remove.

Figure 4-8 Removing an Uploaded JDBC Driver

6. Click **Submit** to save your changes.

> If the JDBC driver you remove is one that updated a default driver, the default driver will reappear as a [SYSTEM] driver in the GlobalPropertiesList the next time you use the New Data Source wizard.

4.4 JNDI Data Sources

The JNDI data source accesses a database connection previously defined in the application server and published as a JNDI resource or service. Instead of specifying a driver and database as you do with JDBC data sources, you need to specify only the JNDI service name in your application server.

> Application servers use JDBC connections themselves to expose a database through JNDI. You must specify the JNDI service name of a JDBC connection. Your application server must also have the appropriate JDBC drivers and be configured to use them.

For information about setting up a JNDI connection in your application server, see the following sections:
- **"JNDI Services on Apache Tomcat" on page 259**
- **"Working With Data Sources" on page 256**
- **"JNDI Services on WebLogic" on page 260**

To create a JNDI data source:

1. Log on as an administrator.

2. Click **View > Repository**, expand the folder tree, and right-click a folder to select **Add Resource > Data Source** from the context menu. Alternatively, you can select **Create > Data Source** from the main menu on any page and specify a folder location later. If you installed the sample data, the suggested folder is Data Sources. The New Data Source page appears.

3. In the Type field, select **JNDI Data Source**. The information on the page changes to reflect what's needed to define a JNDI data source.

   You have the option to use profile attributes to derive the values for data source parameters. See **"Attributes in Data Source Definitions" on page 66**

4. Fill in the service name. This is the name the application server exposes through JNDI.

Figure 4-9 **JNDI Data Source Page**

Set the Time Zone field when the date-time values stored in the target RDBMS do not indicate a time zone. When date-time values are stored in a format other than local time zone offset relative to Greenwich Mean time (GMT), you must specify a time zone so that the server can convert date-time values read from the target database properly. Set the Time Zone field to the correct time zone for the data in the data base. The list of available time zones is configurable, as described in **"Specifying Additional Time Zones" on page 278**.
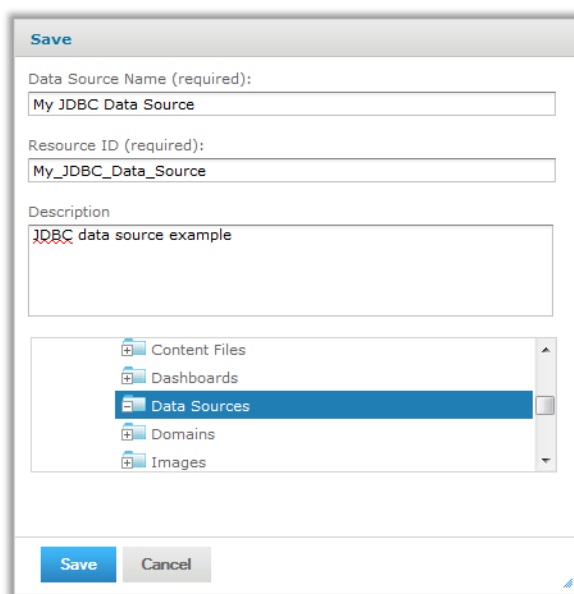
When in doubt, leave the Time Zone field blank.

5. Click **Test Connection** to validate the data source. If the validation fails, ensure that the values you entered are correct, that the database is exposed through JNDI, and that the database is running. Also see the troubleshooting section **"JNDI Services on Apache Tomcat" on page 259**.

6. When the test is successful, click **Save**. The Save dialog appears.

Figure 4-10 **Saving the JNDI Connection**

7. Enter the data source name and, optionally, a description. the Resource ID appears based on the name you enter.

8. Expand the folder tree and select the location for your data source, then click **Save**. The data source appears in the repository.

For details about configuring and exposing a JNDI database connection at the application server level, refer to the documentation provided with your application server.

4.5   AWS Data Sources

Amazon Web Services (AWS) provide computation and data storage on demand in the cloud. We partner with Amazon to deliver business intelligence solutions based on AWS.

JasperReports Server supports two of the AWS database services as data sources for reporting:

•   Amazon Relational Database Service (RDS)
•   Amazon Redshift data warehouse

JasperReports Server can access either of these services when you define a data source with the correct configuration information and credentials. The AWS data source wizard uses the AWS credentials you provide to discover RDS and Redshift data sources. Then it uses those credentials to properly configure security groups to maintain the connection between JasperReports Server and the AWS data source, even when the IP address changes. You can access AWS data sources from both stand-alone server instances that you maintain on your own computers and virtual server instances that you run on Amazon's Elastic Compute Cloud (EC2). For more information, see https://www.jaspersoft.com/cloud.

To create an AWS Data Source:

1. Log into JasperReports Server as an administrator.

2. Click **View > Repository**, expand the folder tree, and right-click a folder to select **Add Resource > Data Source** from the context menu. Alternatively, you can select **Create > Data Source** from the main menu on any page and specify a folder location later. If you installed the sample data, the suggested folder is Data Sources. The New Data Source page appears.

3. In the Type field, select **AWS Data Source**. The information on the page changes to reflect what's needed to define an AWS data source.

Figure 4-11 Selecting AWS Credentials

4. Under AWS Settings, specify your Amazon credentials in one of the following ways:
   - If your JasperReports Server is running in Amazon's EC2 service, and it has the proper instance role assigned, the server will detect this and automatically use your EC2 credentials. Using the EC2 instance credentials requires that the role was properly set up and assigned when the instance was created. If you're using the EC2 service, we strongly recommend that you use the EC2 credentials.
   - If your JasperReports Server is not running on Amazon's EC2, enter the AWS credentials associated with the RDS or Redshift service. If you don't have AWS keys, click **Generate credentials**, then look for them on the **Outputs** tab for your Stack on the Amazon console:

Figure 4-12 AWS Access and Secret Keys

5. Under Select an AWS Data Source, specify the connection details of the AWS data source that you want to connect to:

a. Select your AWS Region from the drop-down.

b. Click the **Find My AWS Data Sources** button.

The AWS data source queries your environment and displays your available data sources, as shown in the figure below.

c. Select your data source.

d. Enter your user name, password, and database name.

The AWS data source queries your environment and adds the appropriate driver and URL.

Figure 4-13 Select an AWS Data Source Section

6. When you've entered all the information, click **Test Connection**.

   If your connection is successful, a message appears at the top of the screen. Sometimes the process takes a few minutes. In that case you'll see an alert. Try the test again after one or two minutes. The test performs the following actions:

   • Validate the user name and password.
   • Create a database security group.
   • Add the IP address of your JasperReports Server instance to the security group to authorize ingress to the data service (RDS or Redshift).

   If you want to control details of the security group name or specify the IP address manually because you have a complex VPC Topology, see **"Configuring Amazon Web Services" on page 204**. You can also change the default JDBC driver through the configuration.

7. Click **Save**. The new data source appears in the repository.

4.6   Cassandra Data Sources

The Apache Cassandra database provides scalability and high availability for certain applications of big data. In JasperReports Server 5.6, we replaced the previously unsupported community connector with a supported data source for Cassandra. For more information about Cassandra, see http://cassandra.apache.org/.

The Cassandra data source relies on a new driver that has certain limitations on how your data can be structured and accessed:

• The current version of Cassandra does not support NULL values in the data. All required fields must have non-NULL default values. This also means that input controls cannot be null and must be given a value.
• The current version of the driver does not support aggregate functions (sum, min, max).
• For query parameters, the current version of the driver supports $X(IN...), but no other $X functions.

As the Cassandra driver is updated to include more functionality, we plan to update the Cassadra data source in future releases.

The Cassandra data source supports queries in the Cassandra Query Language 3 (CQL3). To improve performance, design your Cassandra data using the following guidelines:

• Specify the ALLOW FILTERING suffix to speed up queries.
• All fields referenced in WHERE clauses of a query should be indexed.

As with all big data stores, Cassandra data sources have the following limitations and usage guidelines within JasperReports Server:

• Cassandra data sources are not supported for OLAP connections
• Cassandra data sources cannot be used directly in Domains. To use Cassandra in a Domain, see **"Big Data Connectors for Virtual Data Sources" on page 90**.
• Cassandra data sources can be used in Ad Hoc Topics, but they do not support query optimization.
• You must configure your query limits to handle big data (see **"Ad Hoc Data Policies for Big Data" on page 185**).
• You must configure your JVM memory to handle the expected amount of data (see the *JasperReports Server Installation Guide*).

4.6.1   Creating a Cassandra Data Source

1. Log on as an administrator.

2. Click **View > Repository**, expand the folder tree, and right-click a folder to select **Add Resource > Data Source** from the context menu. Alternatively, you can select **Create > Data Source** from the main menu on any page and specify a folder location later. If you installed the sample data, the suggested folder is Data Sources. The New Data Source page appears.

3. In the Type field, select **Cassandra Data Source**. The information on the page changes to reflect what's needed to define a Cassandra data source.

   You have the option to use profile attributes to derive the values for data source parameters. See **"Attributes in Data Source Definitions" on page 66**

New Data Source

Set Data Source Type and Properties

First, select the type of data source you wish to add, then enter the required property values.

Type: Cassandra Data Source

Host (required):
localhost

Port (required):
9042

Keyspace (required):
test

User Name (optional):
myname

Password (optional):
••••••

Test Connection

Save   Cancel

Figure 4-14 Cassandra Data Source Page

4. Fill in the required fields, along with any optional information you choose.

   Use port 9042 with the Cassandra data source. Cassandra's default port of 9160 is for the Thrift client that is commonly used with Cassandra. To use the Cassandra Query Language (CQL) with our Cassandra data source, you may need to configure your Cassandra instance as follows:

   ```
   start_native_transport: true
   native_transport_port: 9042
   ```

5. If you have configured your Cassandra source to be password protected, specify a valid username and password. Due to compatibility issues, Cassandra authentication is supported only when you use Cassandra 1.12.18 and above.

6. Click **Test Connection** to check the values you entered. Make sure that the port is set to 9042, because the connection test will also work with the wrong port (9160).

7. When done, click **Save**. The Save dialog appears.

Figure 4-15 **Saving the Cassandra Connection**

8. Enter the data source name and, optionally, a description. the Resource ID appears based on the name you enter.

9. Expand the folder tree and select the location for your data source, then click **Save**. The data source appears in the repository.

4.6.2 Increasing File Descriptor Limits for Cassandra

Many users have reported errors when viewing multiple reports from a Cassandra data source. Cassandra generally needs more than the default limit of open file descriptors (1024).

To increase the number of file descriptors, administrators need to change the security limits on the Cassandra nodes and on the operating systems running JasperReports Server.

To test this configuration, you can increase the limits for the current session with the following Linux commands:

```
sudo ulimit -Hn 32768
```

or

```
sudo ulimit -Sn 32768
```

The effects of the commands above will be reset when the computer restarts. To make the changes permanent, edit the file /etc/security/limits.conf to add the following settings:

```
* soft nofile 32768
* hard nofile 32768
root soft nofile 32768
root hard nofile 32768
* soft memlock unlimited
* hard memlock unlimited
```

```
root soft memlock unlimited
root hard memlock unlimited
* soft  unlimited
* hard unlimited
root soft unlimited
root hard unlimited
```

4.7 Hadoop-Hive Data Sources

Unlike traditional databases, Hadoop supports huge amounts of data, often called big data. As of version 5.6.1, JasperReports Server supports two data source types that process requests to a Hadoop cluster:

- CDH 5 Hive-Impala Data Source - If you access Hadoop-Hive2 or Hadoop-Impala through a Cloudera 5 server only.
- JDBC Data Source with Hive JDBC driver - Most other Hive 1, Hive 2 and Impala servers.

Depending on whether you use Hive 1, Hive 2, or Impala, there are certain restrictions on accessing data in Hadoop.

CDH 5 Hive-Impala data sources have very low latency, and allow interactivity with Ad Hoc views, filters, and dashboards. However, Hadoop-Impala data sources still have the following limitations:

- CDH 5 Hive-Impala data sources are not supported for OLAP connections.
- CDH 5 Hive-Impala data sources cannot be used directly in Domains. To use Hadoop-Impala in a Domain, see **"Big Data Connectors for Virtual Data Sources" on page 90**.
- CDH 5 Hive-Impala data sources can be used in Ad Hoc Topics, but they do not support query optimization.
- You must configure your query limits to handle big data (see **"Ad Hoc Data Policies for Big Data" on page 185**).
- You must configure your JVM memory to handle the expected amount of data (see the *JasperReports Server Installation Guide*).

The JDBC driver for Hive works with most other Hive 1, Hive 2, and Impala servers, and it can be used with Domains. However, the original Hive 1 server has high latency with access times on the order of 30 seconds and up to 2 minutes. Hive 2 is much faster, but still not as fast as relational databases. As a result, Hadoop-Hive data sources have certain limitations and guidelines for use in JasperReports Server:

- Hadoop-Hive data sources are not suitable for creating reports interactively in the Ad Hoc Editor.
- Reports based on Hadoop-Hive are not suitable for dashboards.
- Filters and query-based input controls that rely on Hadoop-Hive data sources will be slow to populate the list of choices.
- You must configure your query limits and timeout to handle latency (see **"Ad Hoc Data Policies for Big Data" on page 185**).
- You must configure your JVM memory to handle the expected amount of data (see the *JasperReports Server Installation Guide*).

In general, reports based on JDBC-Hive data sources are best suited to be run in the background from the repository. For very large reports, consider scheduling them to run at night so the output is available when you need it during the day.

To create a Hive JDBC data source, follow the same procedure as in **"JDBC Data Sources" on page 68**.

To create a CDH 5 Hive-Impala data source:

1. Log on as an administrator.

2. Click **View > Repository**, expand the folder tree, and right-click a folder to select **Add Resource > Data Source** from the context menu. Alternatively, you can select **Create > Data Source** from the main menu on any page and specify a folder location later. If you have installed the sample data, the suggested folder is Data Sources. The New Data Source page appears.

3. In the Type field, select **Impala Data Source**. The information on the page changes to reflect what's needed to define a Hadoop-Hive data source.

   You have the option to use profile attributes to derive the values for data source parameters. See **"Attributes in Data Source Definitions" on page 66**

Figure 4-16 **Hadoop-Hive Data Source Page**

4. Fill in the required fields, along with any optional information you choose.

   The JDBC URL depends on which type of server you are using:

   Hive 2:   jdbc:hive2://<hostname>:10001/default

   Impala: jdbc:hive2://<hostname>:21050/;auth=noSasl

5. Click **Test Connection** to validate the data source.

6. When the test is successful, click **Save**. The Save dialog appears.

Figure 4-17 **Saving the Hadoop-Hive Connection**

7. Enter the data source name and, optionally, a description. the Resource ID appears based on the name you enter.

8. Expand the folder tree and select the location for your data source, then click **Save**. The data source appears in the repository.

4.8 MongoDB Data Sources

MongoDB is a big data architecture based on the NoSQL model that is neither relational nor SQL-based. We provide a connector that allows reports to use MongoDB as a data source. Reports based on a MongoDB data source can be used as Topics that allow users to create Ad Hoc views based on the fields returned by the MongoDB query.

As with all big data stores, MongoDB data sources have the following limitations and usage guidelines within JasperReports Server:

• MongoDB data sources are not supported for OLAP connections
• MongoDB data sources cannot be used directly in Domains. To use MongoDB in a Domain, see **"Big Data Connectors for Virtual Data Sources" on page 90**.
• MongoDB data sources can be used in Ad Hoc Topics, but they do not support query optimization.
• You must configure your query limits to handle big data (see **"Ad Hoc Data Policies for Big Data" on page 185**).
• You must configure your JVM memory to handle the expected amount of data (see the *JasperReports Server Installation Guide*).

To create a MongoDB data source:

1. Log on as an administrator.

2. Click **View > Repository**, expand the folder tree, and right-click a folder to select **Add Resource > Data Source** from the context menu. Alternatively, you can select **Create > Data Source** from the main menu on any page and specify a folder location later. If you have installed the sample data, the suggested folder is Data Sources. The New Data Source page appears.

3. In the Type field, select **MongoDB Data Source**. The information on the page changes to reflect what's needed to define a MongoDB data source.

   You have the option to use profile attributes to derive the values for data source parameters. See **"Attributes in Data Source Definitions" on page 66**

 Figure 4-18 **MongoDB Data Source Page**

4. Fill in the required fields, along with any optional information.

   The MongoDB URI has the form: mongodb://<hostname>:27017/<database>

5. The schema is used only if you plan to wrap the MongoDB data source in a virtual data source for use in a Domain. For instructions, see **"Relational Schema for MongoDB Connector" on page 91**.

6. Click **Test Connection** to validate the data source.

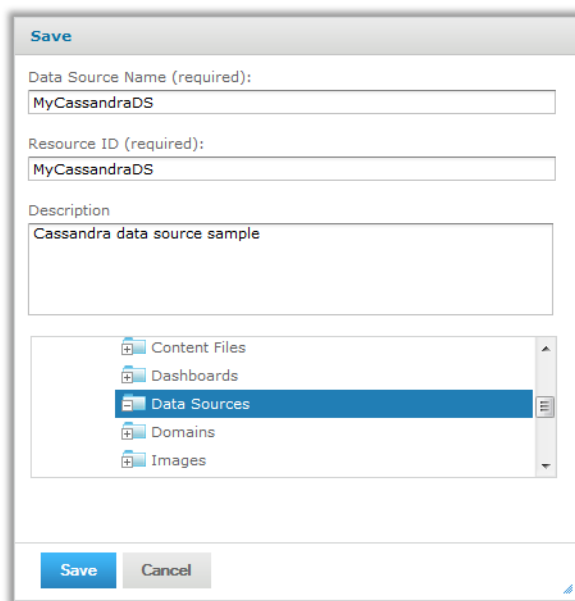7. When the test is successful, click **Save**. The Save dialog appears.
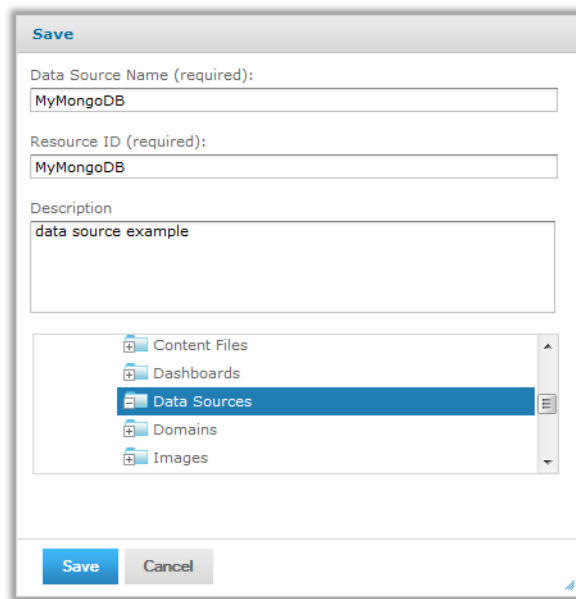
Figure 4-19 **Saving the MongoDB Connection**

8.   Enter the data source name and, optionally, a description. the Resource ID appears based on the name you enter.

9.   Expand the folder tree and select the location for your data source, then click **Save**. The data source appears in the repository.

MongoDB is designed to be accessed through API calls in an application or a command shell. As a consequence, it does not have a defined query language. In order to write queries for MongoDB data sources, we have developed a query language based on the JSON-like objects upon which MongoDB operates. JSON is the JavaScript Object Notation, a textual representation of data structures that is both human- and machine-readable.

The Jaspersoft MongoDB Query Language is a declarative language for specifying what data to retrieve from MongoDB. The connector converts this query into the appropriate API calls and uses the MongoDB Java connector to query the MongoDB instance. The following examples give an overview of the Jaspersoft MongoDB Query Language, with SQL-equivalent terms in parentheses:

•   Retrieve all documents (rows) in the given collection (table):

```
{ 'collectionName' : 'accounts' }
```

•   From all documents in the given collection, select the named fields (columns) and sort the results:

```
{
  'collectionName' : 'accounts',
  'findFields' : {'name':1,'phone_office':1,'billing_address_city':1,
                  'billing_address_street':1,'billing_address_country':1},
  'sort' : {'billing_address_country':-1,'billing_address_city':1}
}
```

- Retrieve only the documents (rows) in the given collection (table) that match the query (where clause). In this case, the date is greater-than-or-equal to the input parameter, and the name matches a string (starts with N):

```
{
  'collectionName' : 'accounts',
  'findQuery' : {
   'status_date' : { '$gte' : $P{StartDate} },
   'name' : { '$regex' : '^N', '$options' : '' }
  }
}
```

The Jaspersoft MongoDB Query Language also supports advanced features of MongoDB such as map-reduce functions and aggregation that are beyond the scope of this document. For more information, see the language reference on the Community website.

4.9 Virtual Data Sources

Virtual data sources have two usage scenarios:
- They allow you to combine multiple JDBC and JNDI data sources and make them available to be joined through a Domain. You can combine any number of data sources, including schemas from different databases, databases from different vendors, and different schemas within the same database into a single object.
- They make Cassandra, Hadoop-Hive, and MongoDB data sources available in Domains. In this case, the virtual data source acts as a wrapper for the data source for big data, with certain limitations described in **"Big Data Connectors for Virtual Data Sources" on page 90**.

A virtual data source can wrap a single data source for big data, or combine any number of JDBC, JNDI and big data data sources.

Once you have created a virtual data source, you create a Domain that joins tables across the data sources to define the relationships among the data. Ad Hoc views and reports based on the Domain can access the combined data transparently. For more information about Domains, see the chapter "Creating Domains" in the *JasperReports Server User Guide*.
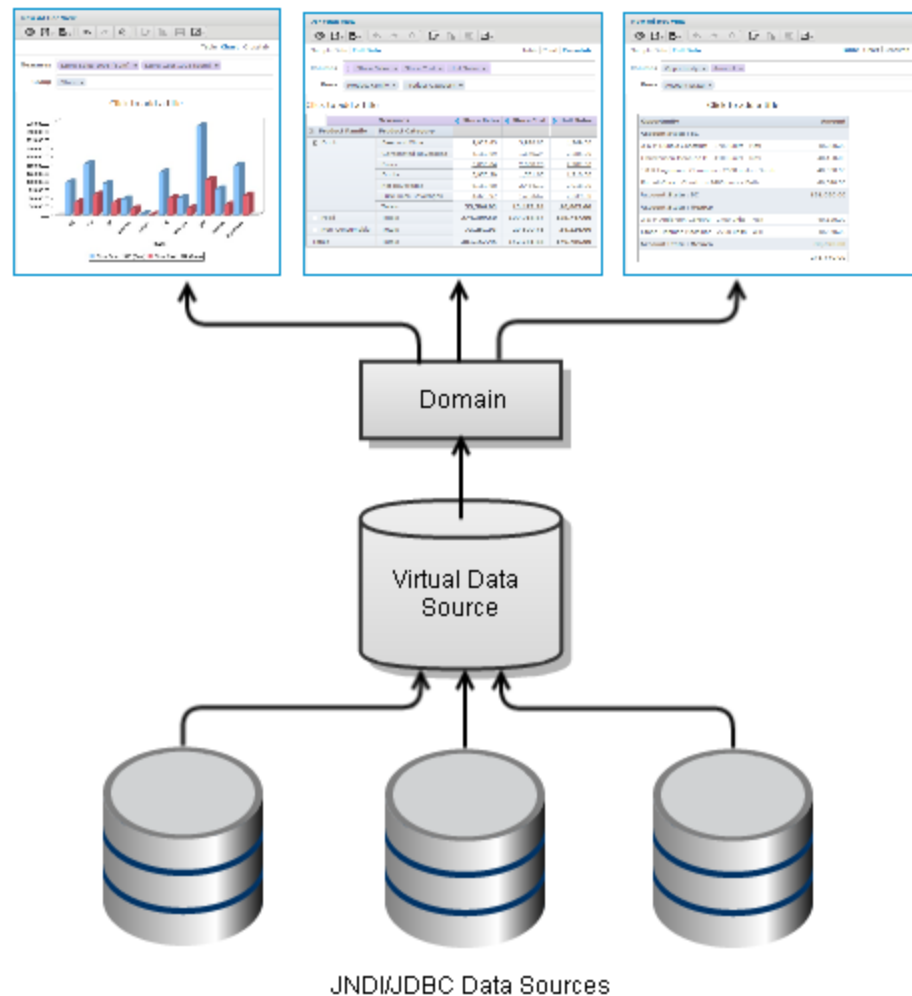
Figure 4-20 Virtual Data Source Scenario

When you combine data sources into a virtual data source, you select an alias for each data source you include; this alias is added as a prefix to the tables in the original data source to ensure that table names are unique across the virtual data source.

To create a virtual data source:

1. Log on as an administrator.

2. Click **View > Repository**, expand the folder tree, and right-click a folder to select **Add Resource > Data Source** from the context menu. Alternatively, you can select **Create > Data Source** from the main menu on any page and specify a folder location later. If you have installed the sample data, the suggested folder is Data Sources. The New Data Source page appears.

3. In the Type field, select **Virtual Data Source**.

4. Locate the data sources you want to use in the Available Data Sources pane. Double-click to select each chosen data source. The data source is shown in the Selected Data Sources pane.

5. Change the aliases by editing them directly in the Alias column (optional). The alias identifies the selected data source within the virtual data source; it is also added as a prefix to the name of each table in that data source. Spaces are *not* allowed in aliases.

The following figure shows values for creating a virtual data source by combining two of the databases included in the sample data: the Foodmart database and the SugarCRM database.
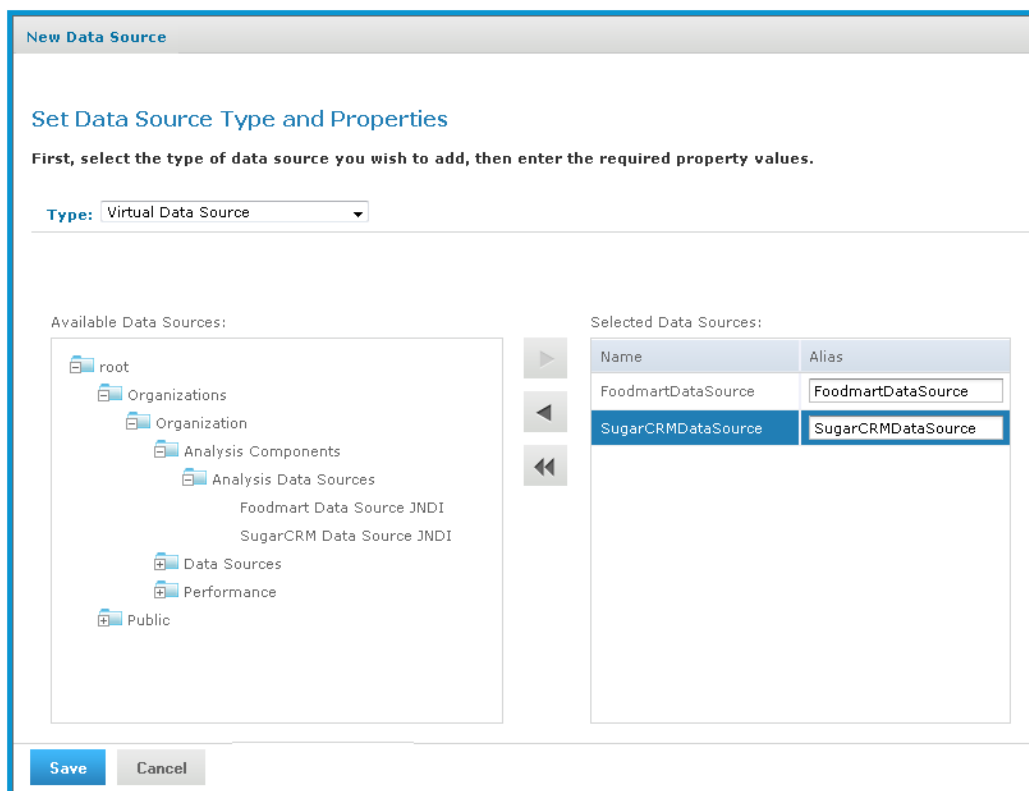
Figure 4-21 Creating a Virtual Data Source

Virtual data sources cannot use the Time Zone field that may be set on individual data sources. If used in a virtual data source, a target data source with a time zone will not return the expected date/time values. Therefore, we recommend that you do not use data sources with time zone settings in a virtual data source.

6. Click **Save**. The Save dialog appears.

Figure 4-22 **Saving a Virtual Data Connection**

7. Enter the data source name and, optionally, a description. the Resource ID appears based on the name you enter.

8. Expand the folder tree and select the location for your data source, then click **Save**. The data source appears in the repository.

You can edit a virtual data source to add or remove the data sources it uses. If the virtual data source is used by a Domain, you can add data sources, but you cannot remove them. Removing a data source from a virtual data source modifies only the virtual data source; the data source you removed remains in the repository.

To edit a virtual data source:

1. Log on as an administrator.

2. Click **View > Repository** and expand the folder tree to locate the folder containing the data source.

3. Right-click the data source and select **Edit** from the context menu.

4. To add a data source, locate the data source in the Available Data Sources pane and double-click. To remove a data source, select it in the Selected Data Sources pane and click the left arrow.

5. Click **Save**. If you try to delete a data source from a virtual data source that is used by a Domain, you'll receive a warning and the data source will remain unchanged. Otherwise, the data source is updated in the repository.

4.9.1   Performance Considerations

Virtual data sources are based on the Teiid engine to handle multiple data sources and combine the results from them. How you design the tables in each of your data sources and how you combine them in a Domain join can significantly affect performance.

Consider these issues when designing a solution that combines several databases in a virtual data source:

- When using virtual data source in a Domain, you must select at least of the tables from the Tables tab to appear in the Domain before using it to create a derived table. Virtual data sources are optimized to use the smallest set of metadata, but they do detect Domains that use only derived tables.
- The virtual data source runs in JasperReports Server and must allocate memory and use processing to handle result sets.
- The Teiid engine uses its own cache for data accessed through a virtual data source. You can clear the Teiid cache at the same time as the Ad Hoc cache, as described in **"Ad Hoc Cache Management" on page 188**.
- The virtual data source can push down certain operations (joins, filters, and aggregations) to the databases, others it must perform in memory. The more it can push down, the smaller the datasets it handles in memory
- You should design your schema to maximize the processing pushed to the database and minimize data handled in the virtual data source in a join between data sources.

For example, if you create a duplicate copy of a date dimension table in both of your data sources, they can be joined much more efficiently. If this table existed in only one database, the Teiid engine would need to retrieve all the rows from the other database and perform the join itself. With this table in both data bases, the Teiid engine can push down all of the time-dimension joins to the individual databases and perform a final join on much smaller datasets.

For more information about optimizing your data for use in Teiid, see http://www.jboss.org/teiid/.

4.9.2   Logging for Virtual Data Sources

If you have issues with your big data connections through virtual data sources, you can enable logging in the following classes:

com.jaspersoft.jasperserver.api.common.virtualdatasourcequery.VirtualDataSourceQueryService

com.jaspersoft.jasperserver.api.engine.common.virtualdatasourcequery.teiid.TeiidEmbeddedServer

For information about enabling logging, see **"Configuring System Logs" on page 199**.

4.10 Big Data Connectors for Virtual Data Sources

As of JasperReports Server 5.6, virtual data sources can also connect to several flavors of big data:

- Cassandra
- Hadoop-Hive 1, 2, and Impala
- MongoDB

Virtual data sources use the Teiid query engine internally to join the data from various sources, and to access big data stores. In the case of big data, the virtual data source extracts the connection information from the existing data source and uses an internal Teiid connector to access data. The Teiid connectors map the various structures used in each big data model to a relational model with tables and fields. This connector is distinct from what are called the native data sources for big data. For this reason, when a data source for big data is wrapped in a virtual data source, the resulting data source has the following limitations:

- The Cassandra, Hadoop, and MongoDB connectors in virtual data sources do not support query parameters ($P and $X). Therefore, if you use a big data connector wrapped in a virtual data source as the data source for a stand-alone query, report or Topic, you can't include parameters to create input controls. When used in Domains and then Ad Hoc views, you can define filters to replace this functionality.
- The Cassandra connector for virtual data sources does not support any aggregation functions.
- The MongoDB connector for virtual data sources does not support the find operations, aggregation or map reduce functions that the native MongoDB data source allows.

- The MongoDB connector for virtual data sources can't be used in stand-alone reports or Topics. It must be used in a Domain and accessed through an Ad Hoc view or report.

However, there are significant advantages to accessing big data through virtual data sources:

- When wrapped in a virtual data source, you can access Cassandra, Hadoop, and MongoDB through a Domain, Domain Topic, Ad Hoc view, and Ad Hoc report.
- A virtual data source can contain any mix of JDBC, JNDI, and big data connectors. When you define a Domain using this data source, you can access the tables from each store and define joins between compatible fields.
- Virtual data sources that use a big data connector support query optimization, unlike the native data sources for big data. In fact, the big data connectors for virtual data sources support query optimization in Ad Hoc views and reports based on stand-alone Topics, and in Ad Ho views and reports based on Domains. The only exceptions are calculated fields, which cannot be optimized when used in Ad Hoc views or reports that are based on Topics or Domains. For more information about query optimization, see **9.4.3, "Ad Hoc Data Policies for Big Data," on page 185**.

For more information about Teiid, see http://www.jboss.org/teiid/.

4.10.1   Creating Big Data Connectors

To create a virtual data source that accesses a data source for big data:

1. Create a native data source for big data, or verify that it was created as described in one of the following sections:
   - **"Cassandra Data Sources" on page 78**
   - **"Hadoop-Hive Data Sources" on page 81**
   - **"MongoDB Data Sources" on page 83**
     In the case of a MongoDB data source, you must specify the schema for the tables into which the data will be mapped. If you did not define the table schema, you can edit the data source to add one, but you must restart JasperReports Server after any modifications to the schema value. For more information, see **"Relational Schema for MongoDB Connector" on page 91**.
2. Create a virtual data source as described in **"Virtual Data Sources" on page 86**.
3. In the virtual data source creation dialog, select the big data source that you created in the first step, and save the virtual data source. You can select one or more big data sources, or any mix of big data, JDBC, and JNDI data sources.
4. Create a Domain, specify the virtual data source you just created, and then select the big data tables when you create the Domain schema. The data from the data source is mapped to tables and fields in the Domain that you can use to create joins, filters, and all other features of a Domain.

4.10.2   Relational Schema for MongoDB Connector

This schema defines a relational structure of tables and columns for the data in your MongoDB instance. The following URL gives the syntax for the schema text:

https://docs.jboss.org/author/display/TEIID/MongoDB+Translator

The following example shows a document from the collection named `customer` in MongoDB with an embedded document named `address`. The right side shows the corresponding schema for use in the virtual data source connector.

```
{                                          CREATE FOREIGN TABLE customer (
    "_id": 10,                                 _id integer PRIMARY KEY,
    "name": "John Doe",                        name varchar(255),
    "age": 27,                                 age integer,
    "gender": "male",                          gender varchar(50))
                                           OPTIONS(UPDATABLE 'TRUE');
    "address": {                           CREATE FOREIGN TABLE address (
        "_id": 10,                             _id integer PRIMARY KEY,
        "street": "123 Sesame St.",            street varchar(255),
        "city": "Anytown",                     city varchar(100),
        "state": "Rhode Island",               state varchar(25),
        "zip": 12345                           zip integer,
    }                                          FOREIGN KEY (_id) REFERENCES customer (_id), )
}                                          OPTIONS(UPDATABLE 'TRUE',
                                                   teiid_mongo:MERGE 'customer');
```

When writing your schema, keep in mind the following issues:
- For embedded relations in MongoDB (both 1-to-1 and 1-to-many), the embedded document must have the same ID as the parent document.
- The MongoDB translator supports automatic mapping of Teiid data types into MongoDB data types.
- Not all MongoDB data types are supported. Currently, the following types are not supported:
    - MongoDB Arrays
    - MongoDB::OID
    - Regular Expressions
    - MongoDB::Code
    - MongoDB::MinKey and MongoDB::MaxKey
- As a result, your documents should use integer IDs and not MongoDB::OID.
- When you change the mapping or add a new collection in the schema, you must restart JasperReports Server.

4.11 Bean Data Sources

The bean data source type is a key extension because it allows you to make use of any custom or exotic data that you might need to report on. Bean data sources serve as a bridge between a Spring-defined bean and a JasperReport. The Spring bean is responsible for providing the data or parameters that fill the report.

To use a bean data source, you must first configure the underlying Spring bean and make it available in the server's web application context. For example, you would add a bean definition to one of the WEB-INF/applicationContext*.xml files.

The bean must resolve to a ReportDataSourceService instance, either directly or by way of a factory no-argument method. You can use any Spring instantiation method (for example, a constructor or factory) and bean scope (for example, singleton or prototype) for the data source service bean.

The ReportDataSourceService instance is responsible for supplying data source parameters to the JasperReport. Custom ReportDataSourceService implementations can follow two approaches:
- If the implementation can provide the data to be used to fill a report, it needs to wrap the data into a suitable JRDataSource implementation and pass the data using the REPORT_DATA_SOURCE report parameter.

- If the data comes from the report query by way of a JasperReports query executor, the data source service must set values for the connection parameters defined by the query executor. The connection parameters are usually obtained from the properties of the data source service instance.

For example, you could implement a Hibernate data source service that would be injected in a session factory. The factory would create a Hibernate session that would be passed as a value for the `HIBERNATE_SESSION` parameter. The JasperReports Hibernate query executor then uses the parameter to run the HQL report query.

The `ReportDataSourceService` interface contains two methods: `setReportParameterValues` and `closeConnection`. The former provides data and connection parameter values; the latter is required to close and release any resources or connections created during the call to `setReportParameterValues`.
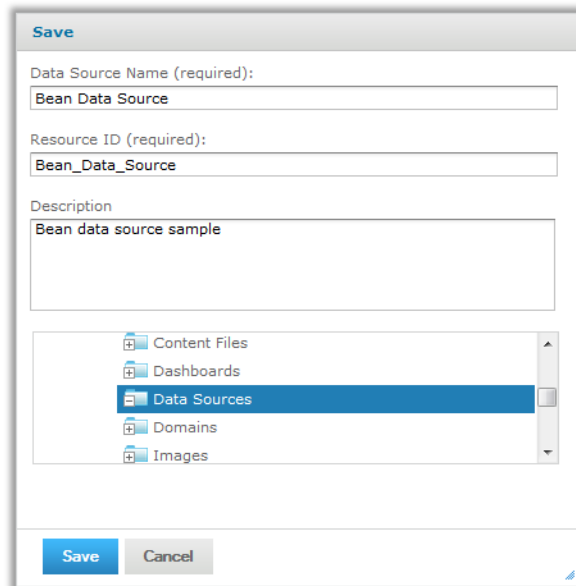
Once the data source service bean is available through Spring, you can add the bean data source to the repository.

To create a bean data source:

1. Log on as an administrator.

2. Click **View > Repository**, expand the folder tree, and right-click a folder to select **Add Resource > Data Source** from the context menu. Alternatively, you can select **Create > Data Source** from the main menu on any page and specify a folder location later. If you installed the sample data, the suggested folder is Data Sources. The New Data Source page appears, as shown in the figure below.

3. In the Type field, select **Bean Data Source**. The information on the page changes to reflect what's needed to define a bean data source.

4. Enter the bean name. If the data source service is to be instantiated through a factory method of the Spring bean, you should also enter the name of the method.

Figure 4-23 **Bean Data Source Page**

5. Click **Test Connection** to validate the data source. If the validation fails, ensure that the values you entered are correct and that the bean is in the classpath.

6. When the test is successful, click **Save**. The Save dialog appears.

Figure 4-24 **Saving a Bean Data Connection**

7.   Enter the data source name and, optionally, a description. the Resource ID appears based on the name you enter.

8.   Expand the folder tree and select the location for your data source, then click **Save**. The data source appears in the repository.

CHAPTER 5   OTHER RESOURCES IN THE REPOSITORY

The previous chapter introduced data sources necessary to create any report. But the repository also stores other resources needed by reports. This chapter goes into detail about how to create queries, input controls, and file resources. These are the resources that users reference when uploading a JRXML file to create a JasperReport.

> This section describes functionality that can be restricted by the software license for JasperReports Server. If you don't see some of the options described in this section, your license may prohibit you from using them. To find out what you're licensed to use, or to upgrade your license, contact us.

As an administrator, you need to define the resources required for your users' reports. If you have users who are proficient at creating their own reports in Jaspersoft Studio they can upload them as JasperReports to the server. In this case, you'll work with them to prepare the resources required by their reports. For less proficient users, you'll have to create and upload their JasperReports to the server.

This chapter contains the following sections:
- **Queries**
- **Datatypes**
- **Lists of Values**
- **Input Controls**
- **Query-based Input Controls**
- **Cascading Input Controls**
- **File Resources**

5.1 Queries

A JRXML report uses a query to select the data to be returned from the data source. You can define the query in the JRXML itself,, but you can also create and save queries in the repository. A query saved in the repository can be re-used by multiple JasperReports. See the sample queries in the /SuperMart Demo/Common folder in the repository.

Reusing a query enables you to adapt reports to different audiences. The query returns the same data from the same data source every time, but each report presents the data in a different way. Also, separating the query from the JRXML makes it easier to maintain large numbers of reports when the data source changes and the query needs to be updated.

See the *JasperReports Server User Guide* for complete instructions on using JRXML reports. For another means of adapting reports to different audiences, refer to the chapter on Domains in that guide.

You can also use queries to populate list input controls, as described in **"Query-based Input Controls" on page 102** and **"Cascading Input Controls" on page 109**.

To create a reusable query:

1. Login as an administrator.
2. Click **View > Repository** and locate the folder for the query. If you installed the sample data, the suggested folder is Input Data Types.
3. Right click the folder's name and select **Add Resource > Query** from the context menu. The Add Query page appears.

Figure 5-1 Add Query - Name the Query Page

4. Enter a name for the query. Resource ID is filled in automatically, and the description is optional Click **Next**. The Link a Data Source page appears.

Figure 5-2 Add Query - Link a Data Source Page

5. Select the data source and click **Next**. Your options are:

- **Do not link a data source**. If no data source is associated with the query, the server uses the data source associated with the report that references the query.
- **Create a new data source**. You can define a local data source within this query resource that's not accessible to any other resource. This new data source overrides any data source specified in reports that use the query.
- **Select data source from repository**. This creates a reference to a data source in the repository. The data source you select overrides any data source specified in reports that use the query.

Click **Next**. the Define the Query page appears.

Add Query:

Define the Query

Select a language and enter the query

Name the Query
Link a Data Source
● **Define the Query**

Query Language: SQL

Query String:

SELECT * FROM orders

Previous   Save   Cancel

Figure 5-3 **Add Query - Define the Query Page**

6. Select a Query Language. For this example choose **SQL**.

The query language Domain (sl) is selected when opening Domain-based queries created in versions of the server before 3.7. It is used only for backward compatibility and should not be selected for new Domain-based queries.

7. Enter the following test in the **Query String** field:

```
SELECT * FROM orders
```

8. Click **Save**.

By default, JasperReports Server supports SQL, HQL (Hibernate), HiveQL (Hadoop-Hive), MongoDB, and Domain queries, while JasperReports supports several more (such as EJBQL, xPath and MDX). However, JasperReports Server can support additional query languages if a query executor implementation is properly-configured for each additional language when the server is deployed.

You can use a specialized bean data source to support multiple query languages. For information about bean data sources, refer to **"Bean Data Sources" on page 92**. Another option to add new types of data sources to the server to extend the reach of the JasperReports Server platform.. Custom data sources are described in the *JasperReports Server Ultimate Guide*.

5.2 Datatypes

A datatype defines the format of a single-value input control, for example text or numerical. Datatypes determine what users can enter in the input fields that correspond to parameters in a report:

- Text
- Number
- Date
- Date/time

To create a datatype:

1. Log on as an administrator.

2. Click **View > Repository**and locate the folder for the datatype. If you installed the sample data, the suggested folder is Input Data Types.

3. Right click the folder's name's name and select **Add Resource > Datatype** from the context menu. The Add Datatype page appears.

Figure 5-4 **Add Datatype Page**

4. Enter a name and optional description for the datatype. The resource ID is filled in automatically.

5. Select the datatype and provide related information. Your options are:

- Text – You can specify a regular expression in the **Pattern** field. The expression is used to validate the text the user submits. For instance, you could enter an expression that tests for email addresses.

- Number – With numerical datatypes, you can control the range of acceptable values by specifying minimum and maximum values and whether the specified values are themselves acceptable (**Minimum is Strict/Maximum is Strict** check boxes). If you select a **Strict** check box, the specified value is *not* acceptable.

  For instance, for a percent field, you might specify a minimum of 0 and a maximum of 100. If you do not want to accept 0 percent, you would check **Minimum is Strict**. If you want to accept 100 percent, you would clear **Maximum is Strict**.

- • Date and Date/Time – Click the calendar icon to the right of the Minimum and Maximum date/date time fields to choose their values.
6. Click **Save**. The datatype resource appears in the repository.

5.3 Lists of Values

A lists of values resource is a static list of values for single-select or multi-select input controls. Depending on the type of input control, the user selects one or more of these labels using radio buttons, check boxes, or drop-down lists.

To create a list of values:

1. Log on as an administrator.
2. Click **View > Repository** and locate the folder for the new list of values. If you installed the sample data, the suggested folder is Input Data Types.
3. Right click the folder's name and select **Add Resource > List of Values** from the context menu. The Add List of Values page appears.

Figure 5-5 **Add List of Values Page**

4. Enter a name for the list of values. Resource ID is filled in automatically, and the description is optional
5. Enter the name and value for each item in the list and click **Add**.

   The name and value are both treated as strings. Users see the label only in an input control that uses the list, and the report receives only the value. To remove an item, click Remove beside the value.
6. Click **Submit**. The list of values resource appears in the repository.

5.4 Input Controls

Any JasperReport can be parameterized so that its generated output is a function of values given at runtime (query filters), or so that its layout is changed to accommodate different users (such as changing the title).

When writing JRXML, you can declare parameters and accommodate any runtime value that needs to be passed into the query executor, the rendering engine, or the calculation engine. However, the parameter information in a JRXML file does not provide everything JasperReports Server needs to build a complete user interface and prompt users for values. You must also define an input control resource that defines the following:

- The range of possible values or list of discrete values allowed.
- The type of input, for example single-select or multi-select, and the widget to display the possible values, for example drop-down list or check boxes.
- Display options such as labels and whether the value is required.
- The name of the corresponding parameter in the JRXML.

When a user runs the report, the server uses this information to prompt the user for a value and to validate that input. For example, consider a report that returns sales data for all of a company's products; the user might input a product to view by selecting a product name in a list.

JasperReports Server supports several types of input controls, each of which can map to certain types of parameters in the report's JRXML. The input control also determines the kind of widget the user interacts with:

- Boolean – Presented as a check box. These input controls return a `java.lang.Boolean` object to the report engine in response to the user's selection. Boolean input controls return .TRUE or FALSE as values, depending on whether the box is checked.
- Single value – Presented as a free-form text box. You must specify a datatype, for example text or numerical value, and the user's entry is validated against this datatype.
- Single-select – Presented either as a drop-down list or a set of radio buttons. A single-select input control returns a single value.
- Multi-select – Presented as a scrollable list of values or a set of check boxes. A multi-select input control returns a collection of values.

One advanced feature of single-select or multi-select input controls is that the values they present can be the result of a dynamic query. The query retrieves actual values from the data source before presenting them as choices to the user. These queries can contain parameters themselves, for example based on the logged-in user or the selection of a previous input control. Query parameters are described in the *JasperReports Server Ultimate Guide*.

Input controls rely on other resources in the repository, such as datatypes, static lists of values, or queries. You can manage these resources the same way you manage other resources. You can define them locally (available only to the input control) or reference them externally (reusing a resource in the repository). For more information, see **"Local Resources and External References" on page 47**.

Ad Hoc views based on Domains and Domain Topics always use locally defined input controls that are created automatically based on the chosen filters. They cannot refer to input controls stored in the repository, and you should not modify them. For more information, see the *JasperReports Server User Guide*.

Some input controls rely on queries to populate their options. These more complex controls are described in **"Query-based Input Controls" on page 102**.

As with other resources, input controls can be created locally as part of a JasperReport, in which case they cannot be seen outside that JasperReport, or they can be created separately in the repository and referenced in multiple reports.

To use an input control in a report, the control must meet two conditions:

- The parameter name in the input control must correspond to the name of the parameter in the report. No error occurs for a mismatch, but at runtime `NULL` is passed instead of actual value of the parameter.

- The input control and its corresponding parameter must be of compatible datatypes (for example, both must be text types or date types). If there is a mismatch, the report fails and an exception is returned.

This section explains how to create an input control in the repository. To reference input controls in a JasperReport, see the *JasperReports Server User Guide*.

To create an input control:

1. Log on as an administrator.
2. Click **View > Repository** and locate the folder for the input control. If you installed the sample data, the suggested folder is Input Data Types.
3. Right click the folder's name and select **Add Resource > Input Control** from the context menu. The Add Input Control page appears.

Figure 5-6 **Add Input Control Page**

4. Select the type of input control from the **Type** list. In this example, **Single Value** is selected.
5. Enter the prompt to tell users how to use the control. This example, uses the prompt `Select the text for the report title`.
6. In practice, the prompt text is often the same as the parameter, so the parameter name is automatically filled in. If you have used a different prompt, edit the Parameter Name field and enter the exact name of the parameter for your control. Remember, the parameter name must be the same here as in the reports that use this input control.

   For this example, the parameter name is `title`. Description is optional.
7. Select options for the control. Your options are:
   - **Mandatory** – Forces the end user to supply a value.
   - **Read-only** – Displays the value of the parameter without allowing the end user to modify it.
   - **Visible** – Makes the input control visible in the report options dialog.
8. Click **Next**.

   Subsequent pages depend on what type of input control you chose:

- Boolean types do not require any further information.
- Single-value types require a datatype the user can enter.
- Single-select and multi-select types based on static lists require a list of values.
- Single-select and multi-select types based on queries require a query.

9. In this single-value example, the Locate Datatype page appears. Choose the option to select a datatype from the repository and click Browse. In the repository dialog, select `/datatypes/TextGeneralDatatype`, which is similar to the datatype we created in **"Datatypes" on page 98**.

> If you choose to define a datatype, the wizard takes you through the same procedure as in section **"Datatypes" on page 98**. You can then define any datatype you need, but it's local to the input control and not reusable in other input controls.

Figure 5-7 **Locate a Datatype for an Input Control**

10. Click **Next**. The input control resource is created in the repository.
11. Locate the input control in the repository manager. Notice that the text of the prompt that you entered in **step 5** is also used as the name for the resource.

5.5   Query-based Input Controls

Query-based input controls display a dynamic set of values for the user to choose from. They are input control resources in the repository, but instead of being based on a datatype or a static list of values, they perform a query to retrieve a list of values. For example, a report could have a city parameter, and the query-based input control could display the list of cities in your data. Because the queries use standard syntax, you can include filters in a WHERE clause, for example, to restrict the list of cities to a certain country.

5.5.1   Creating a Query-based Input Control

In this first example, we create a query-based input control that returns a list of all cities the user can choose from.

1. Log in as an administrator.
2. Browse the repository and select the folder for the query-based input control.
3. Right click the folder's name and select **Add Resource > Input Control**. The Add Input Control dialog appears:

Figure 5-8 **Adding an Input Control - Naming**

4. Select the type of query-based input control from the type drop-down list. This determines how the input control appears to users, either as a drop-down list, a set of radio buttons, a multi-select list, or a set of check boxes. In this example, we choose a single-select query-based input control type.

5. Specify the prompt text, parameter name, optional description, and appearance options in the same manner as when defining a regular input control.

6. Click **Next**. Because we selected one of the query-based types, the Locate Query page appears:

Figure 5-9 **Adding an Input Control - Locating the Query**

If you have a suitable query defined in the repository, you can select it here as an external reference. In this example, we'll define a query locally inside the input control.

7. Click **Next** to define the local query resource. The query naming dialog appears:

Figure 5-10   Adding an Input Control - Naming the Query

Although the query resource is not visible in the repository, it may still have a name, ID and optional description within the query resource. However, the values for these fields are not important.

8.   Enter any name, and the ID is filled in automatically. Then click **Next**. The link data source page appears:

Figure 5-11   Adding an Input Control - Linking to a Data Source

Like all queries, the query inside the input control may optionally link to a data source, either in the repository or its own internally defined one. If no data source is linked, the query in the input control uses the same data source as the report. In this example, we use the default of not linking to a data source.

9.   Click **Next**. The query definition page appears:

Figure 5-12 **Adding an Input Control - Defining the Query**

10. Select the query language, in this example SQL, and enter a query string. The SELECT statement should contain the names of all fields used in the display, value, or filter for the input control. In this example, the query returns three fields, country, state, and city. Country limits the values to a single country. The ORDER BY clause ensures that the values from the query are sorted alphabetically when they appear in the input control.

    For an example in a different query language, see **"Domain-based Queries" on page 108**.

11. Click **Save** to complete the query definition. The parameter values page appears:

Figure 5-13 **Adding an Input Control - Setting Parameter Values**

On the parameter values page, you specify which fields in the query result are displayed, and which field contains values that become the parameter value. when chosen.

a. First, specify the value column. This is the field whose value is passed to the report. The data type of the field must match the type of the corresponding parameter in the report.

b.   Next, specify the visible columns. These are the fields whose values appear in the input control the user chooses from. In the simplest case, enter the same field as the value column. If you add multiple fields to the visible columns, the input control displays the fields together, in the order listed, separated by a vertical bar (|). In this example, the user can see and choose from:

Los Angeles | CA
San Francisco | CA
Denver | CO

Only the city value (without the state) is passed to the report. Showing additional fields in this way can help users find the value they want in long lists of results.

The value and display columns may also be entirely different, for example, displaying the full name of a sales representative, but using the employee ID as the value returned by the input control. The only restriction is that all fields used in the value or display list must be selected by the query.

5.5.2   Built-in Parameters for Query-based Input Controls

The `LoggedInUser` and `LoggedInUsername` parameters are always available for query input controls; they're always available to reports, as well, even if an input control isn't defined for them. The standard parameters are also provided for reports if they're defined as parameters in the JRXML.

Table 5-1 Built-in Parameters for Query-based Input Controls

Parameter Name	Type	Notes
LoggedInUser	User	The currently logged in user. This parameter isn't available in query input controls, but is used as a parameter to the report.
LoggedInUsername	String	The user name of the current user.
LoggedInUserFullName	String	The full name of the current user.
LoggedInUserEmail Address	String	The email address of the current user.
LoggedInUserEnabled	Boolean	Indicates whether the current user is enabled.
LoggedInUserExternally Defined	Boolean	Indicates whether the current user is authenticated externally.
LoggedInUserTenantId	String	In the commercial editions, the current user's organization.
LoggedInUserRoles	Collection <String>	The roles assigned to the current user. This is helpful for parameters that use $X.

Attributes defined on users, organizations, or at the server-level can also be used in reports and query-based input controls. For more information, see:

- **2.2.4, "Editing User Attributes," on page 35**
- **2.1.5, "Creating Attributes on Organizations," on page 29**

- **2.1.6, "Creating Attributes at the Server Level," on page 30**

Table 5-2 Attribute-Based Parameters for Queries and Reports

Parameter Name	Type	Notes
LoggedInUserAttributes	Map<String, String>	The attributes of the logged-in user. This parameter isn't usable in query input controls, but it is used as a parameter to the report. If the user has no attributes, the parameter is an empty map.
LoggedInUserAttribute Names	Collection <String>	The names of the attributes of the logged-in user. This is helpful for parameters that use $X. If the user has no attributes, the parameter is an empty map.
LoggedInUserAttribute Values	Collection <String>	The values of the attributes of the logged-in user. This is helpful for parameters that use $X. If the user has no attributes, the parameter is an empty map.
LoggedInUserAttribute_ <attribute-name>	String	For the logged-in user, the value of the attribute matching the name passed as <attribute-name> (such as att1). If there's no attribute with this name for this user, the parameter is empty. This parameter is available only if it's defined in a query or as a report parameter.
LoggedInTenantAttribute_ <attribute-name>	String	The value of the named attribute defined in the organization of the logged-in user. If there's no attribute with this name for this organization, the parameter is empty. This parameter is available only if it's defined in a query or as a report parameter.
ServerAttribute_ <attribute-name>	String	The value of the named attribute defined at the server level. If there's no attribute with this name at the server-level, the parameter is empty. This parameter is available only if it's defined in a query or as a report parameter.

Parameter Name	Type	Notes
`Attribute_` `<attribute-name>`	String	The value of the named attribute is determined hierarchically. The first value found for the named attribute in the following order: • On the logged-in user. • On the organization of the logged-in user. • At the server level. If there's no attribute with this name defined in those locations, the parameter is empty. This parameter is available only if it's defined in a query or as a report parameter.

5.5.3   Domain-based Queries

For reports that use a Domain as the data source (professional edition only), a query-based input control must contain a query against the Domain. When defining the query as shown in **Figure 5-12 on page 105**, set the query language to **Domain**.

> The query language **Domain ("sl")** is selected when opening Domain-based queries created in JasperServer 3.5 or earlier. It's used only for backward compatibility and should not be selected for new Domain-based queries.

Domain queries have their own special syntax, the same one that's used in the Domain design. A Domain-based query references fields, called items, by their item IDs, along with any set IDs that determine the path of the item within the Domain. For example, if you want your query input control to return a list of store cities, where the field with ID `ej_store_store_city` is nested in the set with ID `expense_join_store`, you would use the following Domain query:

```
<query>
  <queryFields>
    <queryField id="expense_join_store.ej_store_store_city" />
  </queryFields>
</query>
```

The list contained inside the `<queryFields>` tag in a Domain query is equivalent to the fields given in the SELECT statement of an SQL query. Given the query above, you can create an input control for a Domain-based report that lets the user select a city as a parameter to the report.

Sometimes, you may want the input control to display more information than the actual value returned. As with standard query-based input controls, you can select more fields, and then display those fields in your input control. For example, to make the list of cities unambiguous, you could include the state and country in your display. In that case, the Domain-based query must also retrieve those items:

```
<query>
  <queryFields>
    <queryField id="expense_join_store.ej_store_store_city" />
    <queryField id="expense_join_store.ej_store_store_state" />
```

```
    <queryField id="expense_join_store.ej_store_store_country" />
  </queryFields>
</query>
```

Then, when specifying your visible query columns, as shown in **"Cascading Input Controls" on page 109**, you'd add the 3 fields to the list in the order you want them to appear. When specifying fields in the list of visible query columns, use the full ID of the field, including any set IDs. For example, the following list of fields:

```
expense_join_store.ej_store_store_country
expense_join_store.ej_store_store_state
expense_join_store.ej_store_store_city
```

creates a list of values such as the following for users to choose from (the separator | is added automatically):

USA | CA | Los Angeles
USA | CA | San Francisco
USA | OR | Portland
USA | WA | Redmond

Finally, the Domain-based query also has the option to filter the query results, as shown in the following example:

```
<query>
  <queryFields>
    <queryField id="expense_join_store.ej_store_store_city" />
    <queryField id="expense_join_store.ej_store_store_country" />
    <queryField id="expense_join_store.ej_store_store_state" />
  </queryFields>
  <queryFilterString>expense_join_store.ej_store_store_country == 'USA' and
      expense_join_store.ej_store_store_state == 'CA'
  </queryFilterString>
</query>
```

The `<queryFilterString>` tag contains a DomEL (Domain Expression Language) expression that references the full ID of the fields, including any set IDs. For more information about DomEL, see the *JasperReports Server User Guide*. The `<queryFilterString>` tag in a Domain query is equivalent to the WHERE clause of an SQL query. The list of fields in the `<queryFields>` tag must include all fields being referenced in the filter string.

5.6 Cascading Input Controls

A cascading input control is one whose values depend on the selection made in a previous input control. You create a cascading input control using parameters in the query string of a related input control. In other words, the parameter you defined for an input control can be used in another query-based input control.

In a query-based example of cities and states such as:

Los Angeles | CA
San Francisco | CA
Denver | CO

the query may still generate a list of hundreds of cities to scroll through. Even though each city is easy to identify with the state, scrolling through a long list is time consuming. With cascading input controls, this

example can have two input controls, one for the state and one for the city: The city input control is empty until the user selects a state.

• When input controls are displayed, the query for the state input control returns an alphabetical list of unique state names.

• When the user selects a state, the query for the city input control is triggered and returns the list of cities in that state. When the user selects one and submits it, the city name is passed as a parameter to the report.

For an especially large number of cities, you can use more cascading input controls, such as region of state, to reduce the list.

The parameter values determined by each cascading input control may or may not be used in the report. For example, if the report only shows data about a city, the country input control exists only to speed up the choice of city. However, if the report also shows information such as city average compared to country average for a given measure, the country parameter is also used in the report.

5.6.1   Parameters in Input Control Queries

Parameter substitution in query input controls follows the same approach as for JasperReports queries. Queries of all types of data sources can use parameter substitution, and $P, $P! and $X (for SQL queries) parameters are supported. The $X notation has two principal forms explained in the following list:

• `$P{parameter_name}`

The value of the parameter is substituted into the query. In cases where the parameter contains a string, the substitution inserts the proper escape characters to create valid SQL. Use this for single-select input controls and simple comparison operators such as greater-than or less-than. For example:

```
select name from EMPLOYEES where deals_closed > $P{DEALS}
```

Do not use $P{parameter_name} with equality because the parameter value can be null, and `field = NULL` is not valid SQL. Instead use $X{EQUALS, ...} as explained below.

• `$P!{parameter_name}`

The value of the parameter is treated as raw text. The server replaces the parameter with the value of the input control without performing extra checking or value escaping. This is used in complex cases where the input control provides a piece of the query or sometimes the entire query.

• `$X{EQUALS, column, parameter_name}` or `$X{NOTEQUAL, column, parameter_name}`

This syntax performs equality verification and handles the case when the parameter value is null. Use this everywhere instead of the old `column = $P{parameter_name}` syntax. The $X{EQUALS...} syntax performs the following substitution before submitting the query:

```
column = parameter_value -- when parameter_value is non-null

column IS NULL -- when parameter_value is NULL
```

• `$X{IN, column, parameter_name}` or `$X{NOT IN, column, parameter_name}`

Use this parameter for cascading with multiple-select input controls. The $X{IN...} operator is true when the field value matches any one of the multiple values of the input control. In the country/cities example, we can allow the user to pick any number of countries, and show all the cities in the selected countries. The query-based input control would have the following query:

```
select city from ACCOUNTS where $X{IN, country, COUNTRIES}
```

If the user selects the values Canada, Mexico, and USA in the COUNTRIES multi-select input control, the `$X{}` syntax translates into the following query for the CITIES input control:

```
select city from ACCOUNTS where country IN ('USA','Canada','Mexico')
```

> When defining these parameters in a report, don't use a `defaultValueExpression` element. Due to a limitation in JasperReports Server, these parameters are null when a `defaultValueExpression` is provided.

The $X syntax also supports the following operators. They are all designed to handle null input by generating 0 = 0 when the parameter value is null:

Parameter Syntax	Meaning
`$X{GREATER, column, parameter}`	column > parameter
`$X{[GREATER, column, parameter}`	column >= parameter
`$X{LESS, column, parameter}`	column < parameter
`$X{LESS], column, parameter}`	column <= parameter
`$X{BETWEEN, column, start_param, end_param}`	start_param < column < end param
`$X{BETWEEN], column, start_param, end_param}`	start_param < column <= end param
`$X{[BETWEEN, column, start_param, end_param}`	start_param <= column < end param
`$X{[BETWEEN], column, start_param, end_param}`	start_param <= column <= end param

For more information on using $P, $P! and $X to build dynamic queries, see the *JasperReports Ultimate Guide* and the *Jaspersoft Studio User Guide*.

Any number of parameters can be used in a query, just as any number of input controls can be defined in a JasperReport. In addition to the standard input control parameters, a cascading input control query can use the built-in parameters described in **"Built-in Parameters for Query-based Input Controls" on page 106**.

5.6.2   Creating a Cascading Input Control

In this example, we'll create a cascading input control to allow users to report on orders for a specific city by selecting a country and then a city within that country. Our example uses Jaspersoft Studio to edit a JasperReport on the server. The report uses sample data from the SugarCRM database shipped with JasperReports Server.
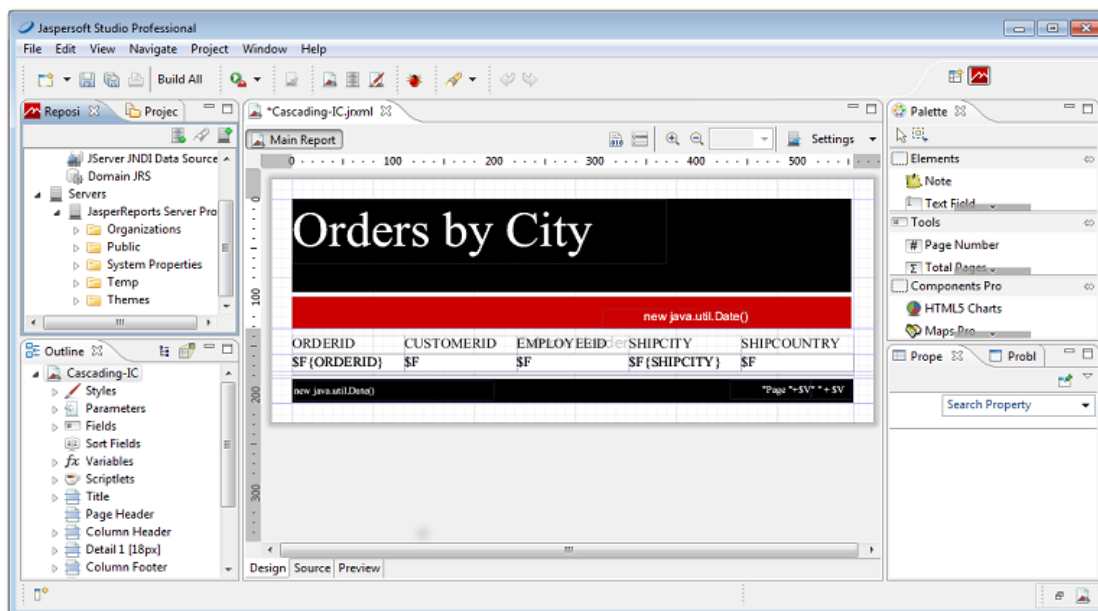
Figure 5-14 Simple Report Filtered by City

> You can connect Jaspersoft Studio to JasperReports Server. Once connected, the reports on the server will appear in the repository tree in Jaspersoft Studio. For more information see the *Jaspersoft Studio User Guide*.

We'll start with a report containing the following dataset fields: ORDERED, CUSTOMERID, EMPLOYEEID, SHIPCITY, and SHIPCOUNTRY.

Our sample report is based on the following query that requires the user to enter a city:

```
select * from orders where SHIPCITY=$P{City}
```

Creating the Country input control:

1. On the repository tab, right click your report's folder and select **New** from the context menu. The Add JasperReports Server Resource wizard appears.
2. Select **Input Control** and click **Next**.
3. Enter Country for the **Name** of the input control and click **Next**.
4. Select the query type **Single Select Query**.
5. Click the ellipsis button next to the **Local Resource** field. The Query window opens.
6. Enter the SQL query:

```
select distinct SHIPCOUNTRY from orders order by SHIPCOUNTRY
```
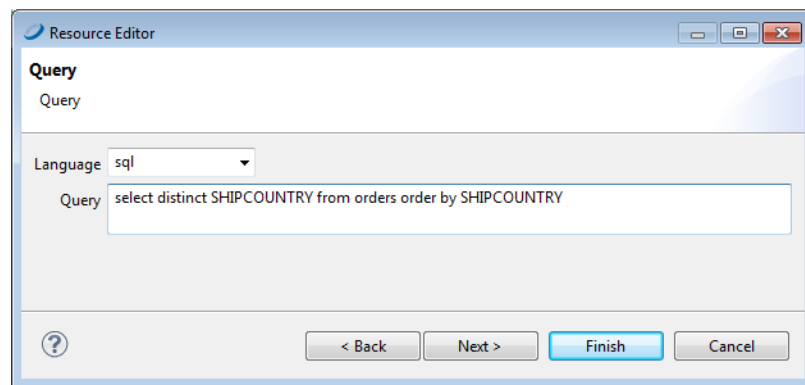
Figure 5-15 **Query for the Country Input Control**

7. Click **Next**. The Data Source window opens.

8. Click the **Data Source from Repository** option, select your data source, then click **Finish**.

9. On the **Value and Visible Columns** tab, enter SHIPCOUNTRY in the **Value Column** field and click **Add**. This is the column whose value is returned as the value of the Country input control.

10. Under **Visible Query Columns**, enter SHIPCOUNTRY. This is the column whose values populate the list of cities the user can choose from.. In our case it is the same as the value column.

11. Click **Finish**.
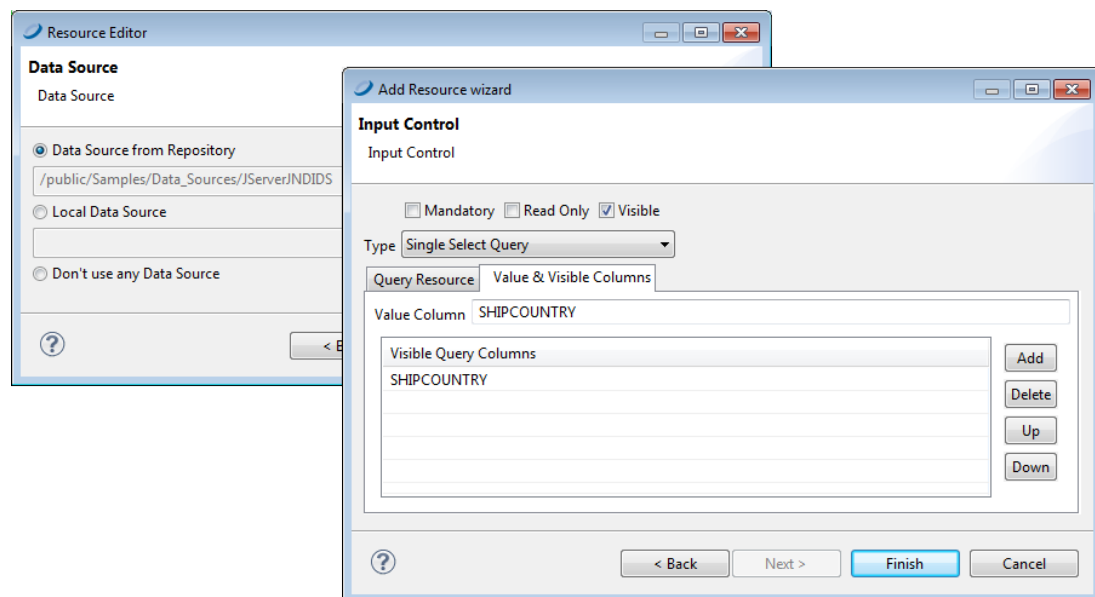
Figure 5-16 **Country Control Value and Visible Columns**

Creating the City input control:

1. Create a new City input control the same way you created the Country input control, using **Single Select Query** type.

2. Define an SQL query that uses the value of Country to derive valid values for City.

```
Select distinct SHIPCITY from orders where SHIPCOUNTRY = $P{Country} order by SHIPCITY
```
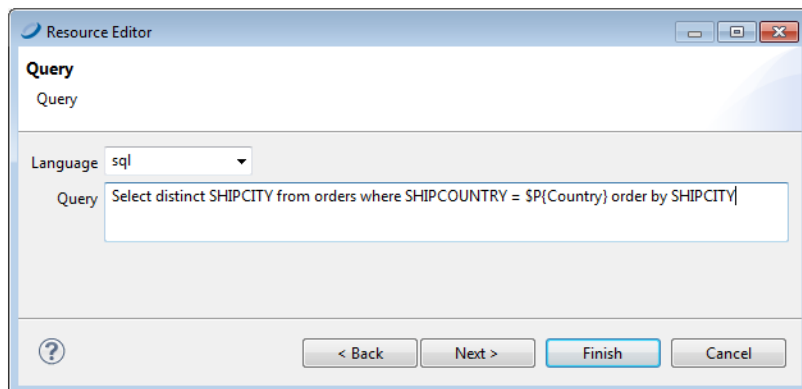

Figure 5-17 **Query Using Country to Select Cities**

3. Click **Next**. The Data Source window opens.
4. Click the **Data Source from the Repository** option, select your data source, then click **Finish**.
5. On the **Value and Visible Columns** tab, enter SHIPCITY in the **Value Column** field and click **Add**. This is the column whose value is returned as the value of the City input control.
6. Under **Visible Query Columns**, enter SHIPCITY, This is the column value used to create the list of cities the user can choose from..
7. Click **Finish**.
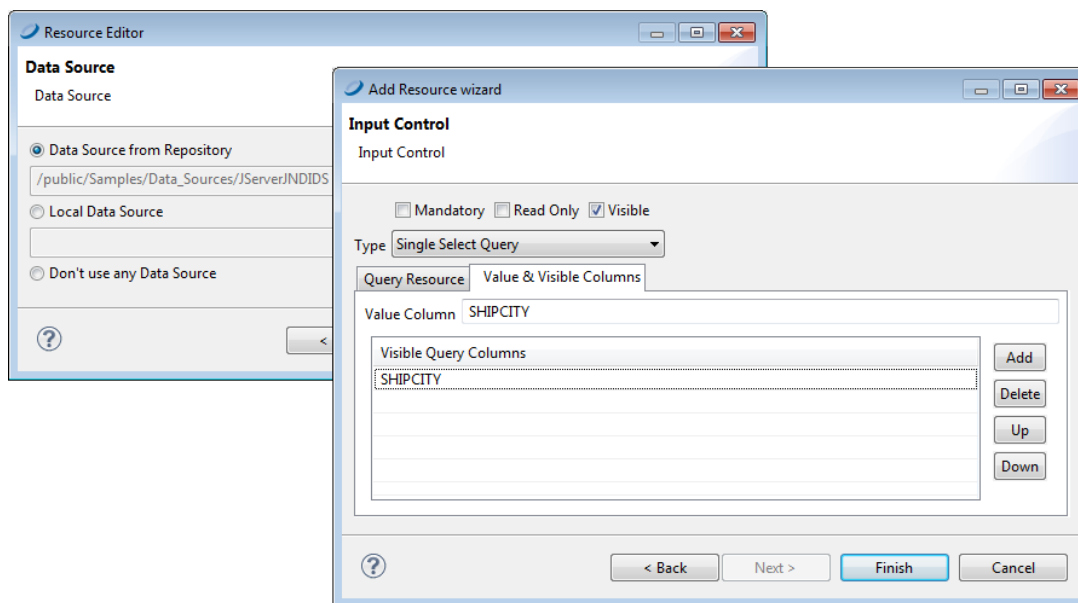
Figure 5-18 **City Control Value and Visible Columns**

8. Publish your report to JasperReports Server.

Now when you run your report in JasperReports Server, You'll see an input control for country and another for city. After you select the country, the values available for city are those within that country. Notice that the Country value is not passed to the report. It is used only to help the user select a city.
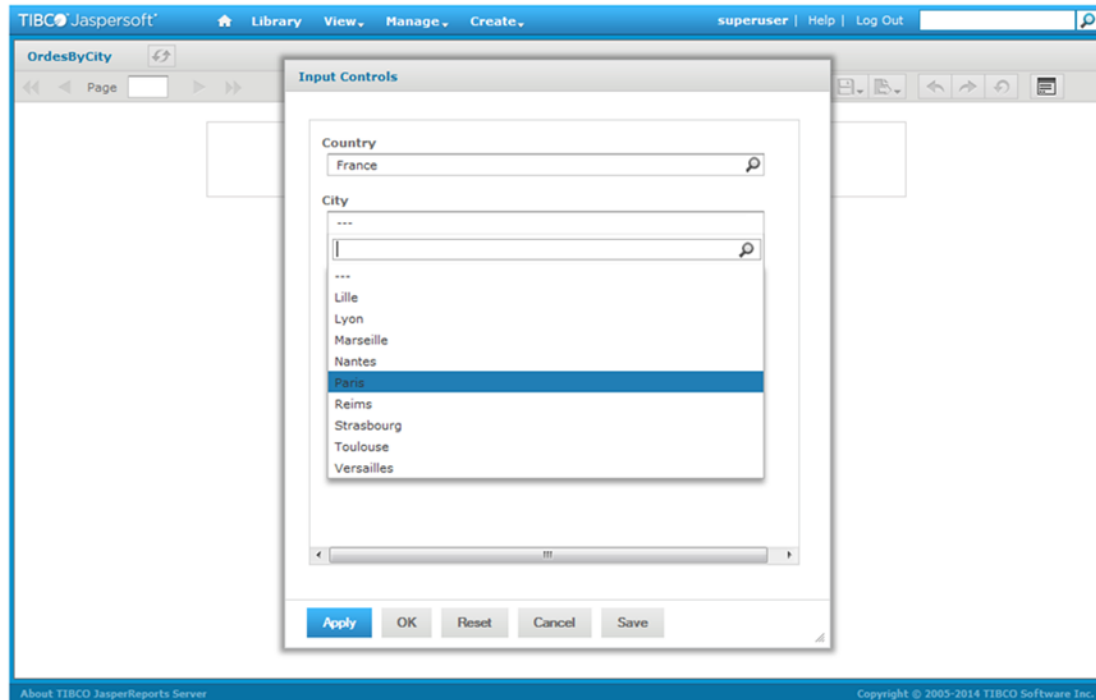
Figure 5-19 **A Report with a Cascading Input Control**

5.7 File Resources

You create file resources by uploading files so they can be referenced by Jasper Reports. JasperReports Server supports the following file types:

Table 5-3 File Resource Types

File Type	Description
CSS	Cascading Style Sheet file that helps define the user interface as part of a theme.
Font	True Type font (.ttf) file to extend the set of fonts available in a report and allow embedding of fonts in the PDF output, if needed (see **"Fonts" on page 116**).
Image	Any image format supported by the JVM (Java Virtual Machine), such as JPEG, GIF, and PNG. Image files can be referenced in JasperReports, and also in CSS files.

File Type	Description
JAR	Libraries that provide functionality for your reports (see **"JAR Files" on page 116**)
JRXML	The definition of a report in JasperReports' XML-based report definition language. A JRXML file can be uploaded separately for use in multiple JasperReports.
OLAP Schema	Defines the data in an OLAP cube, including how to aggregate the dimensions.
Resource Bundle	A Java .properties file containing key-value pairs for localization of reports (see **"Resource Bundles" on page 116**)
Style Template	A JRTX file containing a style template that can be shared among JasperReports.
XML	XML file used in Domains and analysis to define data-level security.

The way in which fonts, JAR files and resource bundles are associated with reports is further explained in the following sections.

5.7.1 Fonts

The server relies on the JasperReports Library as its content rendering engine, which enables it to produce high-quality, pixel-perfect documents. The server can use any of the fonts available to its JVM as logical or physical fonts. This solution is perfect for HTML reports that are stored in the server.

However, when exporting the report to PDF, you may need to take additional steps if the report includes fonts that the PDF viewer doesn't recognize, or if the report requires fonts that your users do not have on their computers. In this case, you must embed the font in the PDF file itself. To embed a font, you must edit the report's main JRXML file; the TTF (True Type Font) file that the report references must be available to the server at runtime. One way to ensure that the server has the correct font is to upload it to the repository by creating a file resource. Then, the report can refer to the font's URI in the repository.

For details about working with fonts and PDF export, refer to the JasperReports documentation.

5.7.2 JAR Files

JasperReports can leverage third-party APIs. When run, reports can make direct API calls to third-party code using JRXML expressions. This provides enormous flexibility for incorporating business logic or other utility code into report generation.

In some cases, you can make the third-party code available to the report generating process by adding the necessary libraries to the server's application classpath when it's deployed. In other cases, upload the third-party or additional JAR files to the repository by creating a file resource. Then the report can refer to the code by referencing them as additional file resources.

5.7.3 Resource Bundles

When a single JRXML template is used to generate documents in multiple languages, it needs a resource bundle to accommodate the locale-specific content. If you upload such resource bundles by creating a file resource, your JRXML files can refer to them.

The name of the resource bundle created as a file resource in the repository must have .properties as its file extension. For example, the default resource bundle might be named MyReport.properties, and its French translation MyReport_fr.properties. For more information about resource bundles for reports, refer to the *JasperReports Server User Guide*.

5.7.4   Creating a File Resource

Administrators should organize file resources into folders in the repository to make them easier to find when creating references.

To add a file resource:

1. Log in as an administrator and select **View > Repository**.
2. In the Folders panel, right-click the parent folder's name and select **Add Resource > File** from the context menu , and select a resource type. In this example, **Font**. The Add File dialog appears.
3. Enter the required information for the file resource. In additions to the name and ID, file resources only require you to enter the path to a file. Click **Browse** to locate a file on your file system.

   The figure below shows the dialog for adding a Font file. All file resources are created by uploading a file in this way.

Figure 5-20 Adding a File Resource

4. When done, click **Submit**. The new file resource appears in the selected folder in the Repository panel.

5.7.5   Editing a File Resource

The following example shows how to edit a file resource.

To edit a file resource:

1. Log on as an administrator.
2. In the repository, browse or search for the resource.
3. Right-click the resource and select **Edit** from the context menu. In this example, we edit the font resource created in **"Creating a File Resource" on page 117**.

Figure 5-21 **Editing a File Resource**

4. Use the Edit dialog to view or modify the resource definition and its values. In the figure above, you can see how the Description field was changed. You can also change the contents of the file resource by specifying another file to upload. The Path to File field is not required unless you want to reload the file from disk.
5. Click **Submit** to save any changes.

CHAPTER 6 THEMES

A theme is the set of all CSS (Cascading Style Sheets) and associated images that define the appearance of the user interface. Themes are stored in the Themes folder in the repository, with special menus on theme folders for activating, uploading, and downloading a theme. You can store any number of themes in the repository and switch among them, to change the UI's appearance.

> This section describes functionality that can be restricted by the software license for JasperReports Server. If you don't see some of the options described in this section, your license may prohibit you from using them. To find out what you're licensed to use, or to upgrade your license, contact us.

For multi-organization deployments, administrators can set the theme individually for each organization, or rely on theme inheritance to use the same theme everywhere without needing to set it explicitly. Theme inheritance supports a mix of explicit and inherited settings, so that you can override any setting or image in a local organization, but inherit the rest of the theme from the parent or system-wide theme.

> Themes were introduced in JasperReports Server 4.0, and the set of files in the default theme was updated in 4.7. Custom themes developed before 4.7 may require upgrading to work with the new set of files. For more information see the upgrade procedures in the *JasperReports Server Installation Guide*.

This chapter contains the following sections:
* **Introduction to Themes**
* **How Themes Work**
* **Administering Themes**
* **Creating Themes**
* **Working With CSS Files**

6.1 Introduction to Themes

The default appearance of the JasperReports Server user interface (UI) can easily be modified to suit your needs.

The UI's appearance is based on CSS (Cascading Style Sheets). A theme is a collection of CSS files and associated images that specify the appearance for all or part of the UI. A theme controls only the UI's looks-- fonts, colors, lines, spacing, and images. It has no effect on content or functionality.

Themes are defined globally at the repository root and individually for each organization. Only administrators can set the theme. Administrators can add, upload, edit, copy, and delete the files that make up the theme, just

like other resources in the repository. The repository allows special actions on theme folders for downloading and uploading themes as ZIP (compressed archive) files, and for activating the theme.

> Themes are fully integrated with the multi-organization architecture in JasperReports Server. Some features of themes discussed in this chapter apply only to deployments that are licensed to use multiple organizations. However, single-organization deployments use the same architecture, for example there are overrides and inheritance between themes in the single default organization and the system root.

Themes are hierarchical and very flexible, allowing administrators to easily change the global appearance or set organization-specific overrides. For example, all of the following scenarios are possible with themes:

Scenario	Description
Use the default theme unchanged.	If the default theme suits your needs, there's no need to customize it or develop new themes. After a standard installation, the default theme is set at the root level and is automatically inherited by all organizations so every user sees the server with this interface
Quickly modify the default theme.	You can specify overrides of individual CSS rules or replace images in the system theme. It's easy to create or upload the new files and activate customizations. Your new theme is uploaded in real time and inheritance ensures that every organization sees it at once.
Create an entirely new theme.	With CSS experience or Jaspersoft Professional Services, you can change the entire look and feel of the server. You can tailor the server to match or blend with nearly any other web design. Inheritance again ensures that every organization uses the new theme, while allowing you to manage it from a single set of files.
Override themes to customize the UI for each organization.	You can give each organization or suborganization a customization of the default theme, for example a new logo, while retaining all other aspects of the system theme. The benefit of this approach is that the system theme can still be modified and inherited by all organizations, while retaining each organization's overrides. This approach can be combined with the previous one, so overrides are applied to the custom theme.
Create a new theme in each organization.	For SaaS vendors, each organization can be a different client that needs a special interface. Themes allow each organization to fully define the UI and still retain override and inheritance for its own suborganizations. In such deployments, each organization admin can modify or create the appearance of his own user interface.

6.2   How Themes Work

Themes are stored in a special folder named Themes at the root of the repository and in every organization. Each Themes folder contains a default theme that can't be edited and any number of custom theme folders. Each theme is stored in its own folder and is known by the name of the folder.

The folder named "default" in every Themes folder is a special theme whose contents are controlled by the server. In the Themes folder at the root, the default theme contains the complete definition of every style that makes up the default theme shipped with JasperReports Server. In organization Themes, the default is a system generated theme that contains all styles inherited by the given organization. None of the default theme folders can be modified, even by administrators.

This chapter uses the following terminology to distinguish between root-level and organization-level themes.

Name	Folder	Description
Default theme	root > Themes > default	The unmodified UI as it appears at installation. The default theme is defined in the default folder in the Themes folder at the root of the repository.
System theme	root > Themes > *active-theme*	The active theme set at the root level. All users in all organizations see this theme unless an organization-specific themes activated.
Inherited theme	Organization > Themes > default	The combination of all themes applied to the parent organizations and inherited by a suborganization. The organization's inherited theme is stored in the default folder within the organization's Themes folder.
Active theme	Organization > Themes > *active-theme*	The theme that's active at the organization or system level. Users see a combination of the active and inherited theme, depending on the files in the active theme and the inheritance rules.

You cannot modify the files of the default theme through the repository. If you try to do so by circumventing the repository, you could inadvertently change rules and make the UI unusable. If this happens you'll need to re-install JasperReports Server to recover.

The following figure shows the default theme in the Themes folder at the root of the repository. The name of the folder (and its subfolders) are bold indicating that it's the active theme.

Figure 6-1 The default Theme in the Root Themes Folder

6.2.1   Theme Files

A complete theme consists of the files listed for the default theme, as shown in the previous figure, along with all referenced images. In addition, the default theme contains the file samples.css that is only used by the **View > UI Samples** page described in section **"User Interface Samples" on page 133**. The files overrides_ie7.css and overrides_ie8.css are only loaded with the style sheets when the user's browser is Internet Explorer 7 or 8, respectively.

> The set of files in the default theme was updated in 4.7. Custom themes developed before 4.7 may require upgrading in order to work with the new set of files. For more information see the upgrade procedures in the *JasperReports Server Installation Guide*.

The images associated with a theme include all the icons in the user interface and backgrounds for buttons and borders. Several icons and backgrounds can be stored in the same file called a sprite. The theme also includes the favicon.ico file that appears on browser tabs. There are approximately 60 image files in the default theme.

The default theme stores referenced image files in a folder named images. In a custom theme, there are two ways to change an image of the default theme:
• Use a folder named images and an image file with the same name as the one you want to replace.
• Modify the corresponding CSS rules to redefine the location where the new image can be found.

When you modify the CSS rules, you can use any of the following ways to reference image files, or any other helper file:

- Directly in the theme folder. In this case the file is referenced without a path, for example `"myfile.png"` in CSS.
- In any folder path located in the theme folder. For example, your custom CSS file could refer to `"MyImages/myfile.png"` if you create a folder named MyImages in the theme folder and upload your images there.
- Anywhere on the Internet. Following the CSS standard, your custom CSS can refer to images, or any helper file, with a regular URL.

6.2.2   Inheritance

In order to render the user interface, Jasper Reports Server must load each of the theme files. Because each file can be stored in multiple themes, inheritance determines which file to load.

The server loads each of the CSS files listed in **Figure 6-1**. To locate the file, the server looks in the following locations, in the orders listed below.

For professional edition users:
1. The active theme folder for the user's organization.
2. The inherited theme stored in the folder named <organization>/Themes/default.

For other users:
1. The active theme folder.
2. The inherited theme stored in the folder named /Themes/default.

When one of the CSS files references an image file or a helper file, including any path to that file, the server looks for that path and filename in the same two locations, in the same order. In this way, each file and image is resolved first in the active theme, and if not found, then in the inherited theme.

The active theme does not need to contain all the files because the inherited theme that is maintained by the server is guaranteed to contain all the files. Maintaining the inherited theme in every organization is the second task of inheritance.

The server maintains the inherited theme in each organization using the same algorithm. Whenever an administrator changes the active theme or modifies a file in the active theme, the server uses the same algorithm to find all files that define the active theme in this organization and makes a copy of them in every child organization. For nested levels of organizations, the algorithm repeats on each level after updating the copy of the inherited theme. In this way, any changes are propagated down to every organization.

> Propagating changes to the inherited themes is computationally intensive and can take several moments after making a change to a theme. However, determining inheritance when changes are made is an effective trade-off so that CSS files for rendering client request are resolved nearly instantly.

6.2.3   CSS Priority Scheme and Custom Overrides

Once inheritance determines which files to load, the standard CSS priority scheme determines which rules are visible, based on the order in which files are loaded.

This leads to two general ways of developing custom themes:

- The quickest way is to copy individual CSS rules from the default theme files, modify the rules to change the UI, and save them in the overrides_custom.css file. Because overrides_custom.css is always the last CSS file to be loaded, its rules override the same rules in other files. This allows you to easily change any number of rules, and manage them all in a single file.
  For example, if you want to increase the size of text on all the buttons in the default theme, you can do this with a few rules in the overrides_custom.css file. You may need to adjust the spacing for certain buttons, but the idea is you only need to change a limited number of rules.
- If you modify the user interface extensively, you can use the existing structure of CSS files in the default theme. In this case, copy the relevant files from the default theme, make your modifications, and save the files in your new theme. The new files are inherited when you activate the theme.
  An example of these extensive changes would be if you want to increase the size of the buttons themselves in the default theme. You would need to rewrite the majority of the rules in the buttons.css file and create images for the new buttons. In this case, it is much easier to copy the buttons.css file than to copy dozens of rules into the overrides_custom.css file. You could still use the overrides_custom.css file to adjust the spacing of elements around the buttons, because there would be fewer of those rules to modify.

We recommend using the custom overrides method for most custom themes. A custom theme that changes simple appearances such as colors, fonts, and spacing has relatively few rules and is easily manageable in a single file. And many changes can be made by copying and modifying image files in the custom theme, without writing any CSS rules. Only if you change the fundamental layout or appearance of the user interface, should you consider copying and modifying the other CSS file.

Copying and modifying CSS files is more prone to error, and is slightly less flexible due to file-based inheritance. Your copy of the file must contain all of the CSS rules as the original. If any rules are accidentally deleted or modified, even by a single character, the theme may not work properly. Also, the unmodified rules in the copy of your file now override any updates made to the same file in a parent organization.

For example, if you can copy a file that defines gray buttons with plain text, and you change the CSS rule to make the text bold, to create a theme with bold, gray buttons. However, if the theme on the parent organization or system theme is modified so that buttons are blue, your file overrides the new inherited color, and you still have bold, gray buttons. If you had defined the bold text as a single rule in the overrides_custom.css file, your theme would show bold, blue buttons now.

6.3 Administering Themes

Themes are sets of CSS and image files stored in a folder in the repository. The root of the repository and every organization has a Themes folder where active and inherited themes are stored. In the repository browser, the Themes folder and individual theme folders have special actions for administrators to manage them. You can also use the repository search to find CSS and image files.

The folders and actions for managing themes are visible only to administrators. The Themes folder has execute-only permission for ROLE_USER so that all users can load the theme files and see the user interface, but not access the folders and files in the repository.

This section gives the basic procedures for administering existing themes. To create theme folders and files, see **"Creating Themes" on page 128**. For information about how to work with CSS in themes, see **"Working With CSS Files" on page 131**.

6.3.1 Setting the System Theme

1. Log into JasperReports Server as system administrator (`superuser` in the professional edition; `jasperadmin` in other editions).

2. Click **View > Repository** and expand the Themes folder if necessary.

3. Right-click the new theme folder and select **Set as Active Theme**.

   For example, the sample data includes a two themes called jasper_dark and pods_summer that you can set as active.
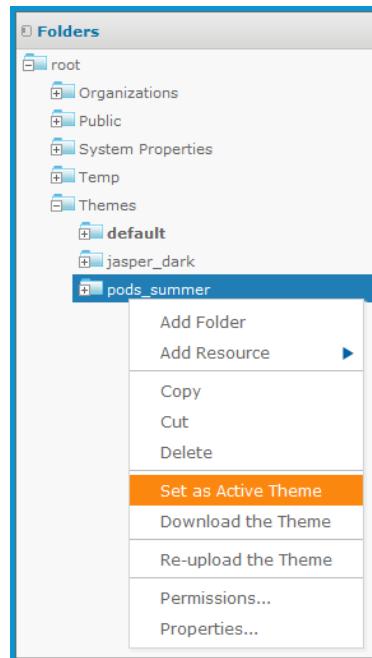
Figure 6-2 Setting a System Theme

As soon as the screen is refreshed, you see the effect of the new theme. Notice how the pods_summer theme changes the colors and the logo in the user interface with just the overrides_custom.css file and some image files.
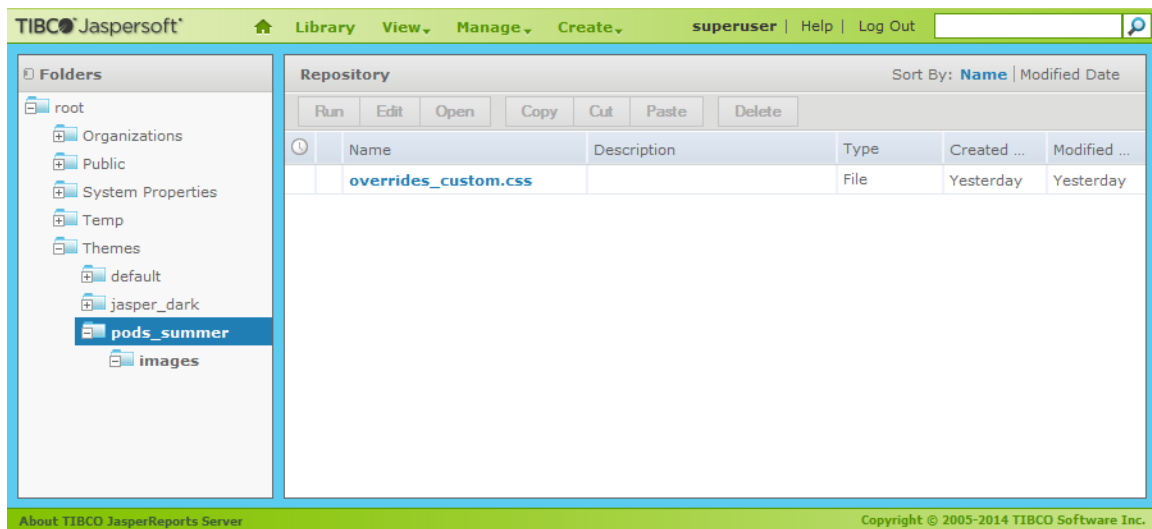
Figure 6-3 **The Sample Theme pods_summer**

Because the system theme is set at the root level, the new theme appears to all users in all organizations, unless the organization has its own theme. Also, the system theme set at the root applies to the login page, as shown in the following figure.

Figure 6-4 **The Login Page as Seen With the New System Theme**

The following procedures assume that the system theme is still set to the default theme.

6.3.2 Setting an Organization Theme

Professional edition users can give different themes to their organizations.

1. Log into JasperReports Server as the organization admin (jasperadmin).

In a server licensed to use multiple organizations, specify the organization ID or alias on the login page.

2. Click **View > Repository** and expand the Themes folder if necessary. The organization's Theme folder is shown in the following figure.

Figure 6-5 Organization Themes Seen by Organization Admin

3. Right-click the new theme folder name and select **Set as Active Theme**.

As soon as the screen is refreshed, you see the effect of the new theme. The new theme applies to all organization users and is inherited by all suborganizations, if any.

Organization admins can thus customize the user interface by creating and activating new themes within their organization.

6.3.3 Restricting Access to Themes

System admins may want to restrict access to themes, so that all themes are controlled from the system level.

This procedure only applies to system admins. Organization admins cannot modify the ROLE_ ADMINSTRATOR permission, even in suborganizations. They must request that the system admin perform the procedure for them.

1. Log into JasperReports Server as system administrator (`superuser` for commercial editions, or `jasperadmin` for community editions).

2. Click **View > Repository**. Community edition users can skip to step 5.

3. Expand the Organizations folder.

4. Locate the name of the organization where you want to restrict access to themes and expand its folder.

5. Right-click the Themes folder name and select **Permissions**.

6. Change the permission for the ROLE_ADMINISTRATOR from Administer to Execute Only.
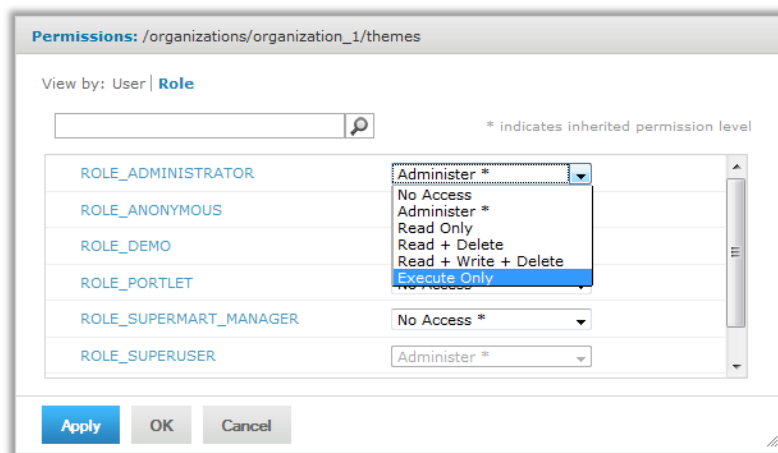
Figure 6-6 **Restricting jasperadmin Access to Organization Themes**

By setting Execute Only access, the organization admins cannot see the Themes folder in the repository, and thus cannot change themes or create a new theme.

You shouldn't change any other permissions on themes, even if the permissions dialog allows it. You could inadvertently make the user interface inaccessible.

7. To restrict access to all organizations, repeat **step 4** to **step 6** for every organization in the server, including suborganizations.

8. If you want to restrict access in the same way in all future organizations, repeat **step 5** and **step 6** in the Folder Template of every organization and suborganization in the server. Fore more information, see **"Placing Themes in the Folder Template" on page 131**.

6.4 Creating Themes

There are three ways to create the folders and files that make up a theme:

• **Creating Theme Folders and File Resources** - Create them individually as resources in the repository.

• **Downloading and Uploading Theme ZIP Files** - Copy and modify existing themes as ZIP (archive) files.

• **Placing Themes in the Folder Template** - Initialize themes automatically in multi-organization deployments.

This section explains only how to store CSS files in the repository. For information about creating CSS file contents, see **"Working With CSS Files" on page 131**.

6.4.1 Creating Theme Folders and File Resources

A theme is simply a folder in the repository that contains CSS and image files, with optional sub-folders. Administrators can use the repository menus to create theme folders. System admins can create theme folders and files at the system level or in any organization. Organization admins can create theme folders and files in their organization or any suborganization.

To create theme folders and file resources:

1. Log in as an administrator with access to the location where you want to place the theme.
2. Click **View > Repository** and expand the folder tree to view the Themes folder where you want to place the theme.
3. Right-click the Themes folder and select **Add Folder**. Give your folder a name and optional description as you would when creating any folder. The folder name is used as the name of the theme.

> Theme folders and files can be created, copied or moved anywhere in the repository, but they can only be made active, uploaded, or downloaded when properly placed in a Themes folder.

4. Right-click your new folder and select **Add Resource > File > CSS**, and use the dialog to upload an individual CSS file. In order to be used as part of a theme, it must be one of the file names shown in **Figure 6-1**.
5. To add images to your theme, create any image folders and upload image files with **Add Resource > File > Image**.
6. Repeat **step 4** and **step 5** to create all the files and images you need. If several themes use the same files or images, you can copy-paste the file resources or entire image folders from one theme to another.
7. If you need to change the contents of a CSS or image file, you can right-click it and select **Edit** to specify another file to upload and replace the current file.

> If you upload CSS and image files into the active theme, the changes are visible after reloading the page in your browser.

Interacting with theme folders and files through the repository is a convenient and flexible way to create a theme. However, this method suffers from the limitation that, like other repository resources, you cannot download the files or images to edit them. For this purpose, the repository provides special download and upload actions on theme folders.

6.4.2 Downloading and Uploading Theme ZIP Files

The process of creating a theme often starts with the files of an existing theme that you modify with CSS and image editors on your computer. To support this workflow, every Themes folder has special commands for downloading and uploading themes.

Because a theme is composed of any number of files and folders, JasperReports Server uses the ZIP archive format to store a theme in a single file.

To download a theme ZIP file:

1. Log in as an administrator with access to the theme you want to download.
2. Click **View > Repository** and expand the Themes folder if necessary.

3. Right-click the theme folder you want to download and select **Download the Theme**. This menu selection appears only on theme folders inside the Themes folder.

4. The server prompts you to save the file named <theme-name>.zip. Save it anywhere on your computer.

5. Use an archiving or compression utility to extract the files from the ZIP file and save them on your computer.

Once you have the theme files extracted on your computer, you can view the individual CSS and image files that make up the theme. For example, to create your own theme, start by downloading the default theme from the root/Themes folder (as superuser). Save the extracted file on your computer and create your custom theme in another folder by copying and editing the CSS files and images of the default theme. See **"CSS Priority Scheme and Custom Overrides" on page 123** for an explanation of how to create a theme.

When you have created all the files you need in your theme, upload it with the following procedure.

To upload a ZIP file as a theme:

1. Place the CSS files, optional folders, and images files that constitute your theme in a folder on your computer.

2. Use an archiving or compression utility to create a standard ZIP file of the contents of your theme folder.

> The ZIP file should include only the contents of your theme, not the theme folder itself.

1. Log in as an administrator with access to the location where you want to upload the theme.

2. Click **View > Repository** and expand the Themes folder if necessary.

3. Right-click the Themes folder and select **Upload a Theme**.
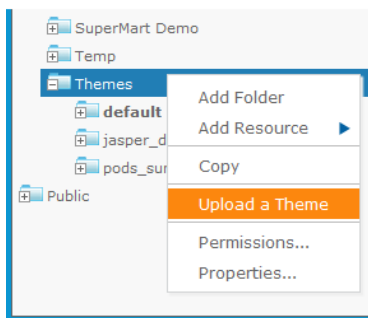
Figure 6-7 **Uploading a Theme in an Organization**

4. In the dialog that appears, enter a name for your theme, which becomes the name of its folder, and browse to find the ZIP file on your computer. Click **Upload**.

> You cannot use the ZIP upload dialog to overwrite an existing theme. You must specify a theme name that doesn't already exist in the chosen Themes folder.

The server uploads your ZIP file and extracts it contents. Then it creates a folder for the new theme and creates file resources in the folder for each of the CSS and images in your ZIP file. If you had sub-folders in your theme, they are created as well. After uploading your theme ZIP file, you can make it active to see effect of your theme on the user interface.

Creating a theme is an interactive process where you often need to make changes until you have the look and feel you want. To support this process, uploading ZIP files can be combined with the uploading of individual file resources that is described in **"Creating Theme Folders and File Resources" on page 129**. In fact, after an initial upload, it is much easier to update individual files in this way than to create the ZIP file and upload it again.

6.4.3   Placing Themes in the Folder Template

In deployments licensed to use multiple organizations, you can place a theme in the Folder Template that is used to create new organizations. The theme folder and all of its contents are copied to the Themes folder of any new organizations that are created. Upon creation, new organizations always inherit their theme from the parent organization, but having a custom theme already present can save you time when customizing and activating it for the new organization.

The Folder Template in every organization contains a Themes folder and an empty default folder. Do not modify the empty default folder, but create a new theme folder instead and place your files there. Because the Folder Template does not contain an active theme, there is no Upload Theme menu option on its Themes folder. Instead create the folder for your theme and upload files as resources, or copy an entire theme folder from the parent organization's Themes folder.

If you want to restrict access to the themes in created organizations, you can also set the permissions on the Themes folder in the Folder Template. To do this, follow the procedure in **"Restricting Access to Themes" on page 127**.

6.5 Working With CSS Files

This section is not a CSS tutorial but rather a collection of tips and tricks for working with the CSS that makes up the themes in JasperReports server. This section focuses on how to test the themes you develop and match the CSS to its behavior in the JasperReports Server UI. There are many different editors for CSS and tools for testing it, so the recommendations in this section are just one way of developing a theme.

6.5.1   Theme Development Workflow

The major choice to make when developing a theme is whether to use simple theme overrides or to duplicate and modify theme files, as described in section **"CSS Priority Scheme and Custom Overrides" on page 123**. Usually, the extent of your modifications determines which method to use.

Once you have made that determination, you are ready to create your theme. The principal steps in a theme development workflow are as follows:

Step	Reference
1. Download the default theme so you have a copy of the files and CSS rules that you want to modify.	**"Downloading and Uploading Theme ZIP Files" on page 129**
2. Create your new CSS rules, CSS files, and image files.	**"Firebug Plug-in for Firefox" on page 132**, a tool to help you create CSS rules

Step	Reference
3. Upload your new files to a test platform, and activate the theme or place them in an active theme.	**"Creating Theme Folders and File Resources" on page 129**
4. Verify your changes wherever they occur in the UI.	**"Test Platform" on page 132** and **"User Interface Samples" on page 133**
5. Repeat **step 2** through **step 4** for all your changes until the theme is finalized.	
6. Deploy your theme to your users.	**"Setting the System Theme" on page 125** or **"Setting an Organization Theme" on page 126**.

6.5.2   Firebug Plug-in for Firefox

One tool to help you find, modify, and view CSS rules in **step 2** above is the Firebug plug-in for the Mozilla Firefox browser. Firebug displays the HTML, JavaScript, and CSS rules of web pages as you browse. It has a dynamic interface that lets you select an element on the web page, and it displays the specific CSS rules that apply to the element. It also allows you to modify those rules and immediately see the effect on the web page.

The Firebug tool is ideal for modifying themes in JasperReports Server. Once you locate the pages and elements that you want to modify, you can prototype your changes directly within the tool. For example, you can see overall effect of changing a color or modifying the spacing.

If you are implementing your theme through custom overrides, you can copy the CSS rules from the Firebug frame directly into the overrides_custom.css file. Firebug displays the entire rule from its original file, so the copy overrides it exactly. If you are modifying other files from the default theme, Firebug show you the filename and line number of the rule, so that you can easily find it in your copy of the file.

And when you are testing a theme that uses overrides, Firebug displays both the active CSS rule from overrides_ custom.css and the original rule in the regular theme file of the inherited theme. The original rule is displayed in strike-through, so you can easily tell which rule is active and which rule it overrides.

For more information and downloads, see the Firebug website.

6.5.3   Test Platform

When you upload a theme and make it active, it is immediately visible to every user in the organization (if using the community edition, every user on the server), or in the case of a system theme, to every user on the server. Even editing or uploading a file into an active theme is reflected immediately in the user interface. Because developing a theme requires many iterations of uploading, activating, and testing CSS rules, you shouldn't develop themes on a production server.

In the simplest case, you can develop and test your themes before putting your JasperReports Server into production. As you test your server during the deployment, you can develop your themes without impacting real users.

For multi-organization deployments that are in production, you can test on your production server as long as you create a test organization. The test organization inherits from the system theme, creating a very realistic test environment where you can see how your theme overrides the inherited theme. Make sure your test organization

reflects your real organizations, for example having OLAP views if your real organizations perform OLAP analysis. This can help you test your theme with the elements of the user interface that your organization users see the most.

For single-organization deployments that don't have a license to create organizations, you can test your themes on a second installation of the server. For example, you could download the evaluation copy of JasperReports Server and install it on the same computer where you develop the theme. This lets you see how your theme appears either as a system theme or in the default organization. As in the case of the test organization, test your theme with all the server features, for example the Ad Hoc Editor, Dashboard Designer, input controls on reports, scheduling, and the like.

When your theme is well-tested and nearly complete, you should test it on the production server. Upload your theme to the Themes folder where you intend to deploy it, but do not activate it. Log in as a test user and add the following parameter to any URL, for example the home page URL:

```
&theme=<theme-name>
```

This activates your theme for the test user on all pages that you access until the user session times out. This allows you to navigate the entire application and see the effect of your theme in the production environment, without affecting other users.

> To set the theme back to the default append the &theme parameter to the URL with the string default (`&theme=default`). This is especially useful if a problem with the current theme has inadvertently disabled any functionality.

On all of these test platforms, you should look at the user interface generated by your theme with the same browsers and browser versions that your users have. If you see errors, you can also use Firebug to look at the CSS rules that are involved, even if the errors do not show up on Firefox.

6.5.4   Modifying the Appearance of Jaspersoft OLAP

Jaspersoft OLAP relies on a module called jPivot to display data when performing OLAP analysis. The jPivot module does not use all of the features of the new UI framework, but it supports some customizations through themes. For example, it does not use panels that can be hidden, and images for icons are not stored in a theme. However, some display characteristics of the analysis table are controlled by the theme, through the use of the analysisView ID in the theme file dataDisplays.css.

For example, you can change the lines between cells in the analysis table with the following rule in your overrides_custom.css file:

```
#analysisView td { border: thin solid black; }
```

6.5.5   User Interface Samples

When testing your theme, you should look at its effect across all pages and dialogs of JasperReports Server. Your test organization and test users should access all the features of the server to view the user interface under all conditions. An additional test is to look at the user interface samples with the theme you are developing.

The user interface sample page is a new page included with the redesign of the interface. It is only accessible to administrators:

1. Log in as administrator in your test environment at the level where you want to test your theme (superuser or jasperadmin of an organization or suborganization).

2. If you haven't already done so, upload your theme to the Themes folder at this level. See **"Creating Themes" on page 128**.

3. Select **View > UI Samples** from the main menu on any page.

4. Look at all the sample components in each of the sample galleries. For example, the buttons gallery shows all the different types of buttons in every possible action state.
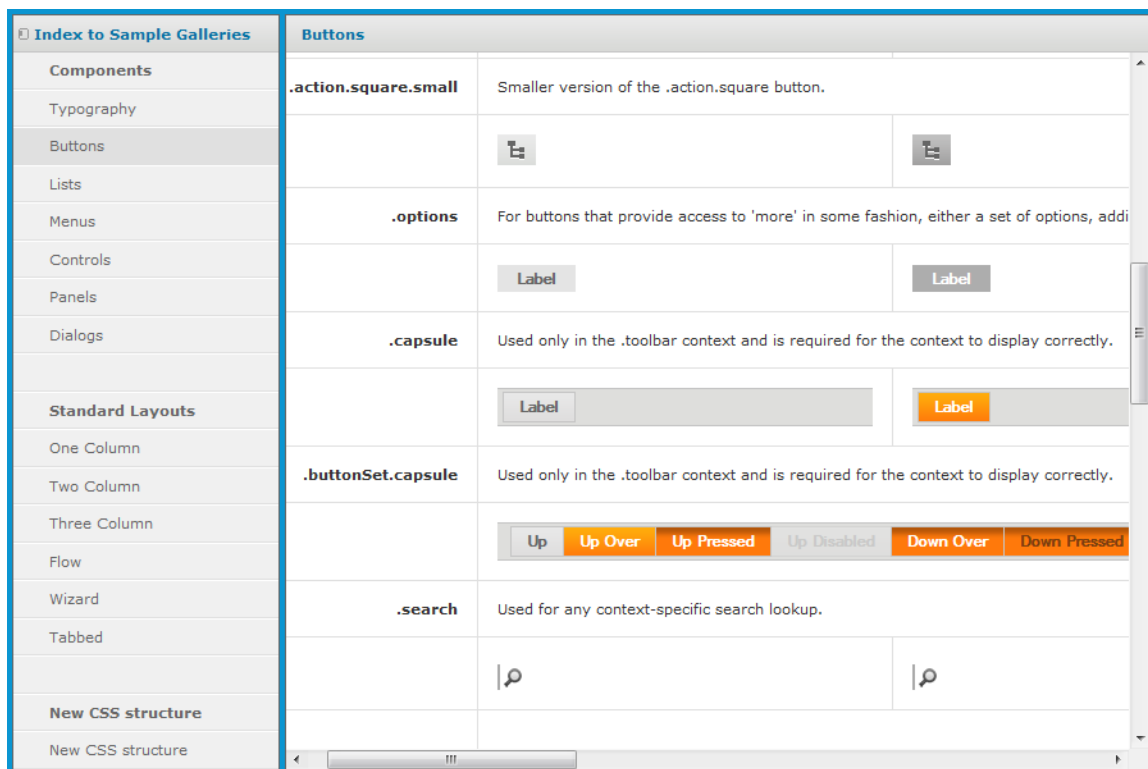
Figure 6-8 **All Possible Button Components in the Sample Galleries**

5. When you click on the standard layouts, the sample replaces the samples page. Select **View > UI Samples** from the main menu again to return to the galleries.

> The samples page relies on an extra CSS file that is not required in a theme, but that can be included. The file samples.css is located in the default theme in the system-level Themes folder. If the sample elements do not appear as you expect, add this file to your theme and customize its rules as necessary. The rules in this file are not used anywhere else in the user interface, so it should not be included in your final theme.

Viewing the sample galleries can help you quickly find errors in your theme, especially if you are changing many rules and replacing entire files in your theme. Using these samples along with the testing procedures and tools described previously, you can verify that your theme properly implements the custom user interface that you intend. Having a well-tested theme minimizes the chances of errors when you activate the theme in your production server.

CHAPTER 7   IMPORT AND EXPORT

The import and export tools enable you to move resources into and out of the JasperReports Server repository. These utilities also handle scheduled jobs, users, and roles that the server stores internally. Import and export can be helpful when migrating between versions of JasperReports Server or when moving between test and production environments.

JasperReports Server provides both a user interface and command-line utilities to perform import and export. The functionality in the UI is available only to system admins (`superuser` by default) and the command-line utilities require access to the file system where the server is installed.

This chapter contains the following sections:
*   **Import and Export Catalogs**
*   **Import and Export Through the Web UI**
*   **Import and Export Through the Command Line**
*   **Alternate Import-Export Scripts**

7.1   Import and Export Catalogs

The output of the export command and the input to the import command is called a catalog. It's a set of the folders and files that comprise the server's internal database, including organizations, users, roles, scheduled jobs, and repository resources such as reports and associated files. When you don't need the entire database, you can specify options to export only the contents you need, for example one role and its users.

The resources in the repository often have dependencies on other resources, for example a report that relies on images, input data types, and a data source. Exporting one resource includes all of its dependencies, even if they are stored in folders that were not specified in the export command. Importing a catalog that contains such dependencies will re-create the same folder structure in the target repository. Once imported, you can move and redefine the dependencies of these resources.

The catalog can be exported either as a hierarchy of folders and files, or as a single zip file (compressed archive). Regardless of the catalog format, the content of the catalog is not intended for external access. Objects in the database, such as users, roles, and folders are described in XML files, and repository resources are stored in various private formats consisting of data files and subfolders. The XML syntax of the catalog files is not publicly defined, and the data files aren't meant to be accessed.

To access and interact with the server's internal objects, use the REST v2 API. This web service has well defined data types and resource descriptors in XML or JSON formats and a complete set of methods for reading and writing objects on the server. For more information, see the *JasperReports Server Web Services Guide*.

> As of JasperReports Server 5.5, user passwords and data source passwords are encrypted in exported catalogs as well as in the server's internal database. You should still take appropriate measures to secure the catalog file from unauthorized access. Catalog files contain sensitive metadata such as user names, database URLs, and internal or external organizations. Catalog files may also contain data in the form of report output such as the PDF of an executive report.

7.1.1   Setting the Import-Export Encryption Key

As of JasperReports Server 5.5, all sensitive passwords in exported catalogs are encrypted for security. In order for two servers to share the encrypted contents of an exported catalog, they must share a private key. The default key is an AES 128-bit string stored in a configuration file.

This encryption is separate from the server's own internal encryption. All user passwords are stored encrypted in the internal database, as described in **"Encrypting User Passwords" on page 169**. The import-export encryption applies only to export catalogs.

Setting the Import-Export Encryption Key		
Configuration File		
.../WEB-INF/applicationContext-security.xml		
Property	**Bean**	**Description**

`<property name="keyBytes">`
`<value>0x2b 0x6c 0x34 0x22`
`0x44 0x42 0x6f 0xb5 0x7f`
`0x34 0xd3 0x5a 0x1f 0x92`
`0xcd 0xdc</value>`
`</property>`
	`importExport`
	`Cipher`
		Set the value of the keyBytes property to the same hexadecimal value (16 bytes = 128 bits) on all servers that will exchange export catalogs.

> When you change a private key on a server, all previous exports become unusable. Therefore, you must configure your new server soon after installing it, and you should configure it with the key from an existing server, if you have one. This way all your servers and all your export catalogs will use the same key and be mutually compatible.

7.1.2   Importing Unencrypted Catalogs

Versions before 5.5 did not encrypt the user passwords upon export. These passwords appear in plain text within the files of the export catalog. For backwards compatibility, unencrypted catalogs are still supported and can be imported into a newer server, even when an encryption key is configured.

Passwords that are encrypted by an export operation in JasperReports Server 5.5 or later have encryption markers, so older passwords without the markers can be detected and imported as plain text. Once plain-text passwords are imported, they are stored internally with encryption, and will be encrypted in any future export.

7.2   Import and Export Through the Web UI

JasperReports Server provides functionality to import and export resources and users through the user interface:

- **Exporting From the Repository**
- **Exporting From the Settings**
- **Importing From the Settings**

When importing and exporting through the web interface, remember the following:

- If you're importing to a different server, you must configure an encryption key on both servers, as described in **"Setting the Import-Export Encryption Key" on page 136**. You'll need to enter the keystore password when prompted by the import operation.
- Make sure the output location specified for an export is writable for the local user running the app server containing JasperReports Server.

7.2.1   Exporting From the Repository

To export individual resources or an entire folder of the repository:

1. Log in as system administrator (`superuser` by default).
2. Select **View > Repository**. You can also export reports, Ad Hoc views, dashboards, and OLAP views from the **Library** page.
3. Select one or more resources in the main panel, or when viewing the repository, select a folder in the left-panel.
4. Right-click the selected folder or resources and select **Export** from the context menu. The Export Resources dialog appears:

Export Resources

Export data file name:

export.zip

Export options:
☑ Include repository permissions
☑ Include report jobs

[Export] [Cancel]
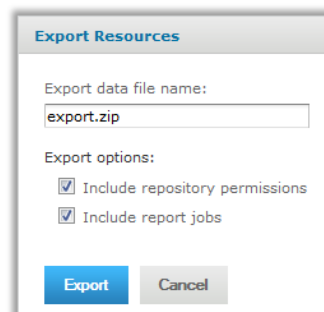
Figure 7-1 **Export Resources Dialog in the Repository**

5. Name of the zip file for the exported catalog. The dialog supports only the zip archive format.
6. Choose one or both of the include options:
   - **Include repository permissions** Enabled by default. Clear this checkbox if you want the exported items to inherit the permissions of the destination repository.
   - **Include report jobs** Select if you want to include scheduled report jobs with any reports included in the selection.
7. Click **Export**. The server generates the catalog zip file and your browser prompts you to save the file. Depending on the size of your repository and the options you've selected, it may take several minutes to generate the catalog file.

> Resources are exported along with any dependencies, even if they're not located in the selected folders. For more information, see **"Import and Export Catalogs" on page 135**.

7.2.2 Exporting From the Settings

To export users and roles in addition to, or instead of, repository contents, use the server settings page for system administrators. This page offers more options than the contextual export from the repository.

As of JasperReports Server 5.5, all exports contain encrypted passwords and if you're importing to a different server, you must configure an encryption key on both servers. See **"Setting the Import-Export Encryption Key" on page 136** for details.

To export data from the settings:

1. Log in as system administrator (`superuser` by default).
2. Select **Manage > Server Settings**, then click **Export** in the left-panel.

Figure 7-2 User Interface for Export

3. Specify the name of the zip file for the exported catalog. The web UI supports only the zip archive format.
4. Use the cascading input controls to choose the contents of your exported catalog file:
   a. Select **Export Everything** (default) to export the entire repository, including permissions report jobs, all organizations, users, and roles. Check the box for **Include access events** if you want to include resource modification times.

    b.   Clear **Export Everything** to select only users and roles to export. Then choose one of the User and Role Selection options. You can directly select users and roles to export or select them indirectly. For example, if you choose **Export users with selected roles**, you'll select users; and the server will export them with their assigned roles.

5.   To select specific users and roles:

    a.   Select **Export individually selected roles and users**.

       The **Select users** and **Select roles** lists become active.

    b.   Select the users and roles to export.

6.   To select roles:

    a.   Select **Export users with selected roles**.

    b.   Select the roles to export. The server will also export the users that are assigned those roles.

       The users assigned the roles you selected are highlighted.

7.   To select users:

    a.   Select **Export roles assigned to selected users**.

    b.   Select the users to export. Their roles will also be exported.

       The roles assigned the users you selected are highlighted.

8.   In commercial editions of JasperReports Server, you can check **Include audit events** and **Include monitoring events** as needed.

9.   Click **Export**. The server generates the catalog zip file and your browser prompts you to save the file. Depending on the size of your repository and the options you've selected, it may take several minutes to generate the catalog file.

7.2.3   Importing From the Settings

The Settings pages for system administrators include a user interface to simplify the import procedure.

This import operates on a running instance of the server, and all imported resources are visible immediately. In addition, any configuration or security settings in the imported catalog take effect immediately, with no need to restart the server.

As of JasperReports Server 5.5, all exports contain encrypted passwords and if you are importing to a different server, you must configure an encryption key on both servers. See **Setting the Import-Export Encryption Key** for details.

> You cannot use the import UI to import a catalog from a JasperReports Server 5.2 or earlier if it was created with the "export everything" option. Theme files contained in export catalogs from previous versions of the server will overwrite newer themes, but they are not compatible and cause HTML display errors. To import a catalog from a previous version of the server, use the command line utility <js-install>/buildomatic/js-import.bat or .sh with the --skip-themes option.
>
> If you have a custom theme to import, you can use the Theme UI to download it from the source server and upload it to the target server. If your theme contains the file pageSpecific.css, you must remove it from the ZIP file before uploading, and then redo your changes to the file based on pageSpecific.css in the target server from 5.5 or later. For more information, see **"Downloading and Uploading Theme ZIP Files" on page 129**.

To import data from the settings:

1.   Log in as system administrator (superuser by default).

2. Select **Manage > Server Settings** and choose **Import** in the left-hand panel.
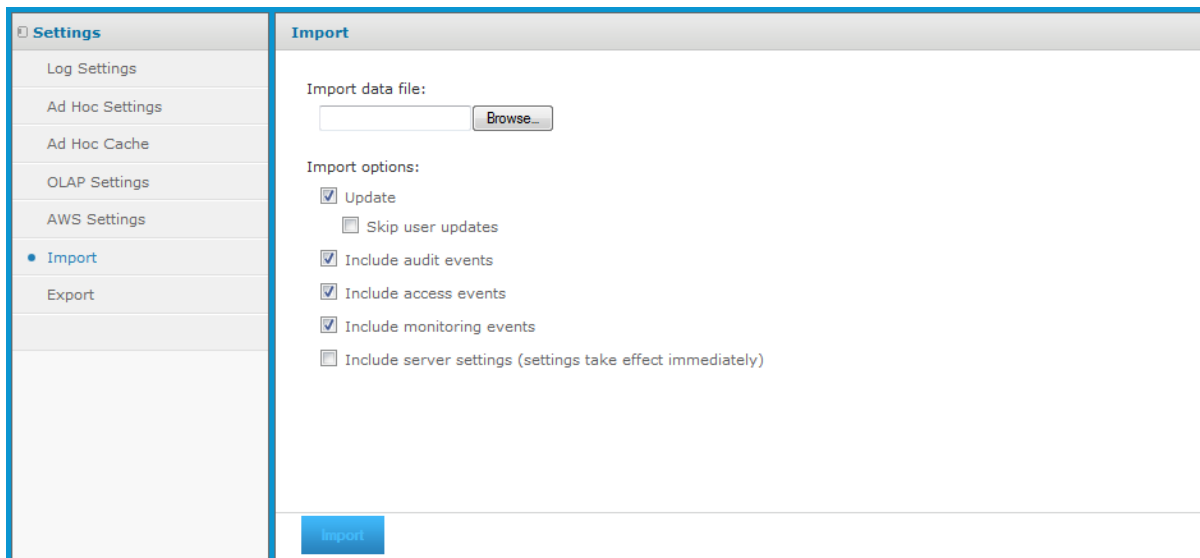
Figure 7-3 User Interface for Import

3. Click **Browse** to choose the catalog zip file to import. The web UI supports only the zip archive format. You can import any catalog zip file created by the export UI or the command-line export utility.

4. Use the check boxes to change the behavior of the import operation:

   • When checked, the **Update** option will import only resources that are newer than ones with the same URI in the current repository. The **Skip user updates** option allows you to keep the current definition of any users that also exist in the imported catalog.

   • When checked, the **Include access events** option imports the modification times of resources from the catalog. When cleared, resources keep their existing access times if they already exist.

   • In commercial editions of JasperReports Library, the **Include audit events** and **Include monitoring events** determine whether access and monitoring events from the catalog, if any, are imported.

   • The **Include server settings** option determines whether the system configuration is updated from the catalog. There are two pre-requisites in order for the catalog to contain configuration settings:

      • The originating server settings must be modified through the UI. Thus, only Log Settings, Ad Hoc Settings, AWS Settings, and OLAP Settings are affected. For more information, see **"Configuration Settings in the User Interface" on page 176**

      • The catalog must be exported with the "everything" option from the user interface or the command-line utility.

      When server settings are imported, they take effect immediately and appear in the Settings UI.

5. Click **Import**.

   The server uploads the catalog zip file and imports its contents into the repository. Depending on the size of the catalog and the options you've selected, it may take several minutes to perform the import.

7.3   Import and Export Through the Command Line

> If you installed JasperReports Server from the binary installer, the command-line utilities were configured
> by the installer. If you installed the WAR file distribution, you must follow the instructions in **"Configuring
> Import-Export Utilities" on page 145** before you can run the utilities.

The import and export utilities are shell scripts located in the <js-install>/buildomatic folder:

Windows:	<js-install>\buildomatic\js-import.bat
	<js-install>\buildomatic\js-export.bat
Linux:	<js-install>/buildomatic/js-import.sh
	<js-install>/buildomatic/js-export.sh

The examples in this chapter use the shortened Windows commands without the optional .bat extension on the
command line. If you're running JasperReports Server in Linux, be sure to add the .sh file extension.

When using the import and export utilities, keep the following in mind:

- JasperReports Server should be stopped when using the import and export utilities. This is very important
  for the import utility to avoid issues with caches, configuration, and security.
- All command line options start with two dashes (--).
- You must specify either a directory or a zip file to export to or import from.
- If you're importing to a different server, configure an encryption key on both servers, as described in **Setting
  the Import-Export Encryption Key**. Then enter the keystore password when prompted by the import
  command.
- Make sure the output location specified for an export is writable to the user running the command.
- All URIs are repository paths originating at the root. The repository paths shown in this chapter assume
  you're using a commercial edition of the server. In the community edition, paths don't include
  organizations, for example:

Commercial editions:	/organizations/organization_1/reports/interactive/CustomersReport
Community project:	/reports/interactive/CustomersReport

7.3.1   Exporting From the Command Line

Usage: `js-export [OPTIONS]`

> We recommend you stop your server instance before running the export utility. For instructions see the
> *JasperReports Server Installation Guide*.

Specifies repository resources such as reports, images, folders, and scheduled jobs to export to the file system.
You can also export the internal definitions for scheduled jobs, users, roles,and audit data. The export output is
known as a repository catalog. It's either an archive file or a set of files in a folder structure:

Table 7-1 Options in `js-export` Command

Option	Explanation
--everything	Exports everything except audit and monitoring data: all repository resources, permissions, report jobs, users, and roles. If any server settings have been modified in the UI, those are also included. This option is equivalent to: --uris --repository-permissions --report-jobs --calendars --users --roles
--help	Displays brief information about the available options.
--include-access-events	Exports events (date, time, and user name of last modification).
--output-dir	Path of the output catalog folder.
--output-zip	Path and filename of the output catalog zip file.
--report-jobs	Comma separated list of repository report unit and folder URIs for which report unit jobs should be exported. For a folder URI, this option exports the scheduled jobs of all reports in the folder and all subfolders.
--calendars	When specified, the export includes any and all calendars of all types (holiday, recurring, ...) defined in the scheduler. When calendars are present in an export catalog, they're always processed and added upon import.
--repository-permissions	This option exports repository permissions with each exported folder and resource. This option should only be used in conjunction with --uris.
--roles	Comma separated list of roles to export. If no roles are specified with this option, all roles are exported.
--role-users	Use only with --roles. This option exports all users belonging to each exported role.
--uris	Comma separated list of folder or resource URIs in the repository.
--users	Comma separated list of users to export; if no users are specified with this options, all users are exported. Exporting a user includes all user attributes and all roles assigned to each user. When specifying users, you must give their organization ID if applicable, for example: --users superuser, "jasperadmin|organization_1", ...
--include-audit-events	Includes audit data for all resources and users in the export.
--include-monitoring-events	Includes monitoring data for all resources and users in the export.

> User passwords are encrypted during the export by default, but exported catalogs may contain sensitive data. Take appropriate measures to secure the catalog file from unauthorized access.

Examples:

- Export everything in the repository:

      js-export --everything --output-dir myExport

- Export the /reports/interactive/CustomersReport report unit to a catalog folder:

      js-export --uris /organizations/organization_1/reports/interactive/
      CustomersReport --output-dir myExport

- Export the /images and /fonts folders:

      js-export --uris /organizations/organization_1/images/organizations/organization_
      1/reports --output-dir myExport

- Export all resources (except users, roles, and job schedules) and their permissions to a zip catalog:

      js-export --uris / --repository-permissions --output-zip myExport.zip

- Export all resources and report jobs:

      js-export --uris / --report-jobs / --output-dir myExport

- Export the report jobs of the /reports/interactive/CustomersReport report unit:

      js-export --report-jobs /organizations/organization_1/reports
      /interactive/CustomersReport --output-dir myExport

- Export all roles and users:

      js-export --roles --users --output-dir myExport

- Export the ROLE_USER and ROLE_ADMINISTRATOR roles along with all users belonging to either role:

      js-export --roles ROLE_USER, ROLE_ADMINISTRATOR --role-users --output-dir myExport

The `--uris` option allows you to specify one or more resource URIs. A URI can specify a resource such as a report. In this case, all associated resources (such as images, subreports, data sources, resource bundles, and class files) are exported. A URI can also specify a folder. If a folder is specified, the export operation exports all resources and folders contained in the folder. In addition, it recurses through all its subfolders.

> The folder named Temp at the root and in every organization is a special folder. None of the folders or resources in a Temp folder are exported.

7.3.2   Importing From the Command Line

See **"Import and Export Through the Command Line" on page 141** for guidelines when running the command-line utilities.

> When using the js-import command line utility, the server must be stopped to avoid issues with caches, configuration, and security. For instructions see the *JasperReports Server Installation Guide*.

Usage: `js-import [OPTIONS]`

Reads a repository catalog from your file system and creates the named resource in the JasperReports Server repository. The repository catalog must be one created by the export interface or the `js-export` command, either as a ZIP archive file or a folder structure.

As of JasperReports Server 5.5, all exports contain encrypted passwords and if you're importing to a different server, configure an encryption key on both servers. See **Setting the Import-Export Encryption Key** for details.

Table 7-2 Options in `js-import` Command

Option	Explanation
--help	Displays brief information about the available options.
--input-dir	Path for importing a catalog from a directory.
--input-zip	Path and filename for importing a catalog from a zip file.
--update	Resources in the catalog replace those in the repository if their URIs and types match.
--skip-user-update	When used with --update, users in the catalog are not imported or updated. Use this option to import catalogs without overwriting currently defined users.
--include-access-events	Restores access events (date, time, and username of last modification) on imported resources.
--include-audit-events	Professional edition only. Imports any audit data in the catalog.
--include-monitoring-events	Professional edition only. Imports any monitoring data in the catalog.
--include-server-settings	Determines whether the system configuration is updated from the catalog. There are two pre-requisites for the catalog to contain configuration settings: • The originating server settings must be modified through the UI (Log Settings, Ad Hoc Settings, Ad Hoc Cache, and OLAP Settings). For more information, see **"Configuration Settings in the User Interface" on page 176**. • The catalog must be exported with the "everything" option from the user interface or the command-line utility. Imported server settings take effect when the server is started.
--skip-themes	This flag is required when importing a catalog that includes a theme, such as when using --export-everything, from a server version 5.2 or before to version 5.5 or later. If you have a custom theme to import, you can use the Theme UI to download it from the source server and upload it to the target server. If your theme contains the file pageSpecific.css, you must remove it from the ZIP file before uploading, and then redo your changes to the file based on pageSpecific.css in the target server from 5.5 or later. For more information, see **"Downloading and Uploading Theme ZIP Files" on page 129**.

Examples:
• Import the myExport.zip catalog archive file:
```
js-import --input-zip myExport.zip
```
• Import the myDir catalog folder, replacing existing resources if their URIs and types match those found in the catalog:

```
js-import --input-dir myDir --update
```
• Import the myExport.zip catalog archive file but ignore any users found in the catalog:
```
js-import --input-zip myExport.zip --update --skip-user-update
```
• Import the myDir catalog folder with access events:
```
js-import --input-dir myDir --include-access-events
```

When a resource in the target repository has the same URI as on that you're importing, the default behavior is leave the existing resource unchanged (no overwriting occurs).

To delete the existing resource and replace it with a new one (of the same type and with the same URI), use the --update option. Note that, if the resource in the export catalog is a different type than the existing resource, the server returns an error and skips the update operation.

When you import a user whose roles exist in the repository, the user is given those roles. User properties are imported with the user.

When you import access events, the date and time of the last modification before export is restored on import for every resource. The catalog folder has to be created with access events. If you don't import access events, or if they don't exist in the imported files, the date and time of the import are used.

7.3.3   Configuring Import-Export Utilities

If you installed JasperReports Server from the binary installer, the import-export utilities were configured by the installer. If you installed the WAR file distribution, you must configure several files before you can use the import-export utilities.

Another option is to use the **"Alternate Import-Export Scripts" on page 146** because they don't require any configuration, regardless of the installation method.

1. Depending on the database you use, copy the installation configuration file:

    from: <js-install>/buildomatic/sample_conf/<database>_master.properties

    to: <js-install>/buildomatic/default_master.properties

2. Edit the default_master.properties file to set values specific to your installation. For more information about the settings in this file, see the *JasperReports Server Installation Guide*.

> Oracle users can set the sysUsername and sysPassword to the same name as dbUsername and dbPassword in the default_master.properties. The system user name and password are not required because js-import and js-export do not make changes to the database schema.

3. Run the following command:

    ```
    js-ant clean-config gen-config
    ```

    This command will generate the following files with the values you added to the default_master.properties file:

    • <js-install>/buildomatic/build_conf/default/js.jdbc.properties
    • <js-install>/buildomatic/build_conf/default/js.quartz.properties (only for DB2 and PostgreSQL)

4. Make sure the JDBC driver for your database is located in the following folder:

    <js-install>buildomatic/conf_source/iePro/lib

    If necessary, you can find links for downloading JDBC drivers from the Jaspersoft Community website:

    http://community.jaspersoft.com/wiki/downloading-and-installing-database-drivers

7.4   Alternate Import-Export Scripts

Regardless of your installation method, JasperReports Server provides a third way to run import-export commands. Buildomatic is another command-line script that is based on the Apache Ant tool to automate installations. It includes targets (sub-commands) to perform import and export operations with the same options as the scripts. The following examples compare the two commands:

Shell Scripts: `js-export.sh --everything --output-file=js-catalog-exp.zip`

Buildomatic: `js-ant export-everything -DexportFile=js-catalog-exp.zip`

Both types of scripts are located in the `<js-install>/buildomatic` folder.

7.4.1   Running Import from Buildomatic

The `import` target for ant has the following syntax:

Windows: `js-ant import -DimportFile=<filename> [-DimportArgs="<import-options>"]`

Linux and   `./js-ant import -DimportFile=<filename> [-DimportArgs=\"<import-options>\"]`
Mac OSX:

The imported file is handled as a ZIP archive if its name ends in .zip, otherwise it's handled as a directory. The `importArgs` argument is optional and can contain more than one import option. On Linux, all double quotation marks (") must be escaped with a backslash (\).

> When performing a large import using js-ant, the server should be stopped (or put into a mode with reduced load) to avoid issues with caches, configuration, and security.

The following examples are typical import commands on Windows:

```
js-ant import-help-pro
js-ant import -DimportFile=my-reports.zip
js-ant import -DimportFile=my-datasources -DimportArgs="--update"
```

The following examples are typical import commands on Linux:

```
./js-ant import-help-pro
./js-ant import -DimportFile=my-reports.zip
./js-ant import -DimportFile=my-datasources.zip -DimportArgs=\"--update\"
```

7.4.2   Running Export from Buildomatic

The `export` target for ant has the following syntax:

Windows: `js-ant export -DexportFile=<filename> -DexportArgs="<export-options>"`

Linux and   `./js-ant export -DexportFile=<filename> -DexportArgs=\"<export-options>\"`
Mac OSX:

The export file format is a ZIP file or a set of files under a new directory name. If you specify the .zip extension for your output filename, a ZIP archive is created automatically. Otherwise, an uncompressed directory with files and sub-directories is created.

The `exportArgs` argument requires double quotation marks (") and can contain more than one export option, as shown in these Windows examples:

```
js-ant export-help-pro
js-ant export -DexportFile=my-domains.zip
    -DexportArgs="--uris /organizations/organization_1/Domains"
js-ant export -DexportFile=my-reports-and-users.zip
    -DexportArgs="--uris /organizations/organization_1/reports
    --users jasperadmin|organization_1,joeuser|organization_1"
js-ant export -DexportFile=my-datasources
    -DexportArgs="--uris /organizations/organization_1/datasources --roles ROLE_USER"
js-ant export -DexportFile=js-everything.zip -DexportArgs="--everything"
```

On Linux, all double quotation marks (") and other characters, such as the vertical bar (|) separating login user and organization names must be escaped with a backslash (\). Also, when listing user names, enclose the list in single quotation marks ('), as shown in this Linux example:

```
./js-ant export-help-pro
./js-ant export -DexportFile=my-reports-and-users.zip
    -DexportArgs=\"--uris /organizations/organization_1/reports
    --users 'jasperadmin\|organization_1,joeuser\|organization_1'\"
```

CHAPTER 8   APPLICATION SECURITY

This chapter describes the configuration settings that protect JasperReports Server and its users from unauthorized access. The configuration properties appear in two locations:

- Some properties must be configured during the installation and deployment phase, before users access the server. These settings are configured through files used by the installation scripts. These settings are available only when performing a WAR file installation.
- Other properties are located in files in various folders after installation. Configuration file paths are relative to the <js-install> directory, which is the root of your JasperReports Server installation. To change the configuration, edit these files then restart the server.

Because the locations of files described in this chapter vary with your application server, the paths specified in this chapter are relative to the deployed WAR file for the application. For example, the applicationContext.xml file is shown as residing in the WEB-INF folder; if you use the Tomcat application server bundled with the installer, the default path to this location is:

C:\Program Files\jasperreports-server-6.0\apache-tomcat\webapps\jasperserver-pro\WEB-INF

Use caution when editing the properties described in this chapter. Inadvertent changes may cause unexpected errors throughout JasperReports Server that may be difficult to troubleshoot. Before changing any files, back them up to a location outside of your JasperReports Server installation.

Do not modify settings that are not described in the documentation. Even though some settings may appear straightforward, values other than the default may not work properly and may cause errors.

This chapter contains the following sections:

- **Encrypting Passwords in Configuration Files**
- **Configuring User Password Options**
- **Configuring the User Session Timeout**
- **Configuring CSRF Prevention**
- **Configuring Input Validation**
- **Restricting File Uploads**
- **Hiding Stack Trace Messages**
- **Defining a Cross-Domain Policy for Flash**
- **Encrypting User Passwords**
- **Encrypting User Session Login**

8.1   Encrypting Passwords in Configuration Files

In JasperReports Server version 5.5 or greater, administrators can obfuscate passwords that appear in the configuration files. This satisfies security audit requirements and prevents the passwords from being seen by unauthorized individuals. Typically, the following are encrypted:

- The password to JasperReports Server's internal database (`jasperserver`).
- The passwords to the sample databases (`foodmart` and `sugarcrm`).
- On Tomcat and tcServer, passwords in JNDI resource definitions.

You can change the configuration to also encrypt:

- The password for the mail server used by the scheduler (`quartz.mail.sender.password`)
- The password for LDAP external authentication.

Passwords in configuration files are encrypted during JasperReports Server installation. If the installation deploys to the Tomcat application server, the database password is also automatically encrypted in the JNDI configuration (in the file context.xml).

> Full password security cannot be guaranteed from within JasperReports Server. A user with sufficient privileges and knowledge of JasperReports Server can gain access to the encryption keys and the configuration passwords. While you could require a password on every server restart, this is impractical for most users. The only practical way to guarantee password security is through backup and restriction of access to the keystore property file.

8.1.1   Encrypting Configuration Passwords on Tomcat (or Spring tcServer)

To encrypt passwords in a Tomcat or tcServer installation, modify the installation procedure:

1. Depending on the database you use, copy the installation configuration file as usual:

    from: <js-install>/buildomatic/sample_conf/<database>_master.properties

    to: <js-install>/buildomatic/default_master.properties

2. Edit the default_master.properties file:

    - Enter values specific to your installation.
    - Enter your passwords in plain text.
    - Turn on configuration file encryption by uncommenting the `encrypt=true` property. You don't have to uncomment any of the other encryption properties because they all have the default values shown.
    - Except, if you are using Oracle, uncomment `propsToEncrypt` and set it to `dbPassword,sysPassword`.
    - Optionally, specify additional properties to encrypt as described in **"Encrypting Additional Properties in default_master.properties" on page 151**.
    - Optionally, change the settings for configuration file encryption as described in **"Encryption Options" on page 154**.

3. Run the buildomatic installation script (js-install) and all other installation steps according to the *JasperReports Server Installation Guide*. This will have the following effects:

    a. The plain text passwords in default_master.properties are overwritten with their encrypted equivalents. There is no warning when you run js-install with `encrypt=true`.

    b. The encrypted passwords are propagated to all configuration files.

    c. The installation proceeds and files are copied to their final locations.

4. After installation, passwords are encrypted in the following locations:

    - In all server configuration files in .../WEB-INF/applicationContext*.xml.

- In JNDI definitions in .../META-INF/context.xml.
- In the default_master.properties files that remains after installation.

If you get an error like the following when restarting the server:

```
javax.naming.NamingException: KeystoreManager.init was never called or there are errors
instantiating an instance
```

you may need to add the following to your Tomcat service start properties:

```
-Duser.home=c:\Users\<TomcatUser>
```

8.1.2 Encrypting Configuration Passwords on Enterprise Servers

Most enterprise servers, like JBoss, Glassfish, WebSphere, and WebLogic, have proprietary ways to set up password encryption. You should use these encryption methods. JasperReports Server does not automatically set up encrypted passwords for these servers during deployment. In this case, you can additionally encrypt the passwords in the buildomatic file after deployment:

1. Deploy JasperReports Server to your enterprise server as specified in the *JasperReports Server Installation Guide*. The resulting JasperReports Server instance will have unencrypted JNDI data source passwords. If you want to encrypt these passwords, refer to your application server's documentation.

2. After the server has been successfully configured, encrypt the JasperReports Server configuration files as follows:

   a. In default_master.properties, turn on encryption by uncommenting `encrypt=true`.

   b. Run the target `js-ant refresh-config`, which will remove and recreate all the configuration files without deploying them to the application server. Now the buildomatic files will have the database passwords encrypted. You should still be able to execute import/export or other scripts.

Do not run `js-install` or `js-ant deploy-webapp-pro`. These commands will overwrite the war file created in step 1 and render the server data sources inaccessible. If you need to redeploy the WAR file, reset the database password(s) to plain text in your default_master.properties and start again with step 1.

8.1.3 Encrypting Additional Properties in default_master.properties

You can encrypt additional properties in the default_master.properties file. To work correctly, these properties need to be decrypted when used. Currently decryption is supported for properties that are loaded into the Spring application context via the `propertyConfigurer` bean in applicationContext-webapp.xml.

If a property is defined via JNDI, we recommend pointing there instead of encrypting:

```
<property name="password">
    <jee:jndi-lookup jndi-name="java:comp/env/emailPassword" />
</property>
```

The following code sample shows the `propertyConfigurer` bean in applicationContext-webapp.xml:

```
<bean id="propertyConfigurer"
class="com.jaspersoft.jasperserver.api.common.properties.DecryptingPropertyPlaceholderConfigurer">
    <property name="locations">
      <list>
        <value>/WEB-INF/hibernate.properties</value>
        <value>/WEB-INF/js.quartz.properties</value>
        <value>/WEB-INF/js.spring.properties</value>
        <value>/WEB-INF/js.scheduling.properties</value>
        <value>/WEB-INF/mondrian.connect.string.properties</value>
        <value>/WEB-INF/js.diagnostic.properties</value>
        <value>/WEB-INF/js.aws.datasource.properties</value>
        <value>/WEB-INF/js.config.properties</value>
        <value>/WEB-INF/js.externalAuth.properties</value>
      </list>
    </property>
    ...
</bean>
</pre>
```

Because we extended Spring's `PropertyPlaceholderConfigurer` class as `DecryptingPropertyPlaceholderConfigurer`, all the loaded properties are scanned for the special marker `ENC-<value>-`. If that marker is found around the property value, that property is decrypted before being loaded into Spring context.

To determine if your property is scanned by `propertyConfigurer`, search the files in `propertyConfigurer`'s locations to see if it's defined in one of these files.

For example, suppose you want to encrypt the `password` property of the `reportSchedulerMailSender` bean in applicationContext-report-scheduling.xml:

```
<bean id="reportSchedulerMailSender" class="org.springframework.mail.javamail.JavaMailSenderImpl">
  <property name="host" value="${report.scheduler.mail.sender.host}"/>
  <property name="username" value="${report.scheduler.mail.sender.username}"/>
  <property name="password" value="${report.scheduler.mail.sender.password}"/>
  <property name="protocol" value="${report.scheduler.mail.sender.protocol}"/>
  <property name="port" value="${report.scheduler.mail.sender.port}"/>
  <property name="javaMailProperties">
    <props>
      <prop key="mail.smtp.auth">false</prop>
    </props>
  </property>
</bean>
```

The use of the `${...}` syntax tells you that `report.scheduler.mail.sender.password` is most likely defined via the `propertyConfigurer` bean. Search through the `propertyConfigurer` locations to verify. This property is defined in /WEB-INF/js.quartz.properties as follows:

`report.scheduler.mail.sender.password=${quartz.mail.sender.password}`.

Once you've verified that the `quartz.mail.sender.password` property can be encrypted using default-master.properties, you set up encryption before installation as follows:

1. Set the password for `quartz.mail.sender.password` in default-master.properties:

   `quartz.mail.sender.password=cleartextpassword`

2. Uncomment the `encrypt=true` property in the same file.

3. Uncomment `propsToEncrypt=dbPassword` in default-master.properties.

4. Add `quartz.mail.sender.password` to `propsToEncrypt`:

```
quartz.mail.sender.password=cleartextpassword
...
encrypt=true
propsToEncrypt=dbPassword,quartz.mail.sender.password
```

5. Configure and install your JasperReports Server war installation as described in the *JasperReports Server Installation Guide*.

6. Verify that `report.scheduler.mail.sender.password` was encrypted in both default-master.properties and in /WEB-INF/js.quartz.properties.

8.1.4   Password Encryption for External Authentication

As of JasperReports Server 5.6, it's possible to encrypt the passwords in the external authentication configuration files for LDAP and external database authentication. This section covers only the encryption of these passwords; for details about configuring external authentication, see the *JasperReports Server External Authentication Cookbook*.

To enable encryption during installation, property values in the external authentication sample configuration are referenced from other configuration files. For example, if you're using LDAP to authenticate, the sample configuration file contains the following reference to the LDAP password:

```
<bean id="ldapContextSource"
    class="com.jaspersoft.jasperserver.api.security.externalAuth.ldap.JSLdapContextSource">
  <constructor-arg value="${external.ldap.url}" />
  <property name="userDn" value="${external.ldap.username}" />
  <property name="password" value="${external.ldap.password}"/>
</bean>
```

The values referenced by the `${...}` format are defined in the js.externalAuth.properties file and are imported into Spring context via the `propertyConfigurer`. For example, the LDAP properties are defined in js.externalAuth.properties as follows:

```
external.ldap.url=${external.ldapUrl}
external.ldap.username=${external.ldapDn}
external.ldap.password=${external.ldapPassword}
```

The `${...}` syntax again references other configuration properties that must be set in default_master.properties before installation or upgrade. The following example shows the syntax of the properties in the default_master.properties file:

```
external.ldapUrl=ldap://hostname:389/dc=example,dc=com
external.ldapDn=cn=Administrator,dc=example,dc=com
external.ldapPassword=password
```

To encrypt the password property, set the following values in default_master.properties before installation or upgrade:

```
external.ldapPassword=cleartextpassword
...
encrypt=true
propsToEncrypt=dbPassword, external.ldapPassword
```

During the installation process, the password value in default_master.properties and its reference in js.externalAuth.properties are overwritten with the encrypted value.

If your external authentication mechanism is configured to create organizations for external users, as of JasperReports Server 6.0, there is another password to encrypt. When external authentication creates an organization, it uses the information in ExternalTenantSetupUser of the externalTenantSetupProcessor bean to create the organization administrator.

```
<bean class="com.jaspersoft.jasperserver.multipleTenancy.security.externalAuth.processors.
          MTAbstractExternalProcessor.ExternalTenantSetupUser">
  <property name="username" value="${new.tenant.user.name.1}"/>
  <property name="fullName" value="${new.tenant.user.fullname.1}"/>
  <property name="password" value="${new.tenant.user.password.1}"/>
  <property name="emailAddress" value="${new.tenant.user.email.1}"/>
  <property name="roleSet">
    <set>
      <value>ROLE_ADMINISTRATOR</value>
      <value>ROLE_USER</value>
    </set>
  </property>
</bean>
```

The values referenced by the `${...}` format are defined in the js.config.properties file as follows:

```
## New tenant creation: user config
new.tenant.user.name.1=jasperadmin
new.tenant.user.fullname.1=jasperadmin
...
new.tenant.user.password.1=jasperadmin
new.tenant.user.email.1=
```

> The default values for new tenant (organization) administrators in js.config.properties apply *only* to external authentication. They do not apply to organizations created by administrators through the UI or REST interfaces.

To encrypt this password, modify the js.config.properties file as follows:

```
new.tenant.user.password.1=${tenant.user.password}
```

Then add the following lines to default_master.properties before installation or upgrade:

```
tenant.user.password=cleartextpassword
...
encrypt=true
propsToEncrypt=dbPassword, external.ldapPassword, tenant.user.password
```

During the installation process, the password value in default_master.properties and its reference in js.config.properties are overwritten with the encrypted value.

8.1.5   Encryption Options

In buildomatic installation scripts, the passwords are symmetrically encrypted: the same secret key is used for both encryption and decryption. The key and its containing keystore file are randomly generated on each

machine during the first JasperReports Server installation. All the subsequent JasperReports Server installations on the same server rely on the same keystore; they don't regenerate the key.

The keystore is an encrypted file used to securely store secret keys. JasperReports Server uses keystore properties to access the keystore. Both the keystore and keystore properties files are created by default in the user home directory. Alternatively, before running js-install, you can specify different locations for the keystore and keystore properties files via the environmental variables ks and ksp.

By default, database passwords are encrypted with the AES-128 algorithm in Cipher Block Chaining mode with PKCS5 padding. The AES algorithm is the current industry encryption standard. You can choose to modify the encryption strength by choosing either a different algorithm, a longer secret key size (for example AES-256), or a different encryption mode.

Edit the following properties in your default_master.properties and set these options. If a property is commented out, the default is used:

Property	Description	Default
build.key.algo	Algorithm used to encrypt the properties in configuration files.	AES
build.key.size	Size of the encryption key as in AES-128. To increase the key size, if it has not been done before, you might have to install "Unlimited Strength Jurisdiction Policy Files" from the Oracle site for your Java version. To install the files, download US_export_policy.jar and local_policy.jar. AFTER backing up the old files, extract the jars into %JAVA_HOME%/jre/lib/security directory. Alternatively, you may download one of the reputable providers such as Bouncy Castle (ships with JasperReports Server). You would need to add the Bouncy Castle provider to the list in %JAVA_HOME%/jre/lib/security/java.security file: security.provider.<seq number>=org.bouncycastle.jce.provider.BouncyCastleProvider	128 (bits)
enc.transformation	So-called encryption mode. See Java's `javax.crypto` documentation to understand the modes and padding better.	AES/CBC /PKCS5 Padding
enc.block.size	The size of the block that's encrypted. Encrypted text can contain many blocks. Usually the block is changed together with the encryption algorithm.	16 (bytes)
propsToEncrypt	A comma separated list of the properties to encrypt.	dbPassword

8.2 Configuring User Password Options

The user password options determine whether passwords can be remembered by the browser, whether users can change their own passwords, and whether password changes are mandatory or optional.

By default, passwords are stored in an encrypted format in the server's private database. For information about changing the way passwords are encrypted, see **"Encrypting User Passwords" on page 169**

8.2.1   Configuring Password Memory

As a general security policy, sensitive passwords should not be stored in browsers. Many browsers have a "remember passwords" feature that stores a user's passwords. Most browsers do not by default protect passwords with a master password. JasperReports Server can send the property `autocomplete="off"` to indicate that its users' passwords should not be stored or filled in automatically. This helps to ensure that your users don't store their passwords. Actual behavior depends on the browser settings and how the browser responds to the `autocomplete="off"` property.

Login encryption described in section **"Encrypting User Session Login" on page 172** is not compatible with password memory in the browser. Independent of the autocomplete setting, the JavaScript that implements the login encryption clears the password field before submitting the page. As a result, most browsers will not prompt to remember the password when login encryption is enabled, even if the user has password memory enabled in his browser.

> When `autoCompleteLoginForm` is true, as in the default installation, you should ensure that all of your users have a master password in their browser.

Password Memory in the Browser		
Configuration File		
.../WEB-INF/jasperserver-servlet.xml		
Property	**Value**	**Description**
`autoCompleteLoginForm`	`true` <default> `false`	When false, the server sets autocomplete="off" on the login page and browsers will not fill in or prompt to save Jaspersoft passwords. When true, the autocomplete property is not sent at all, and browser behavior depends on user settings.

8.2.2   Enabling Password Expiration

If your security policies require users to change their passwords at regular intervals, you can enable password expiration. This way JasperReports Server prompts users to change their passwords at your set interval. Users with expired passwords can't log in without changing their passwords. The default value is 0; in this case, passwords don't expire and users are never prompted.

When you enable this option, the server automatically enables the **Change Password** option on the Login page, even if `allowUserPasswordChange` is set to false.

> If your users are externally authenticated, for example with LDAP, do not enable this option.

Password Administration Option		
Configuration File		
.../WEB-INF/jasperserver-servlet.xml (controls the Login page) .../WEB-INF/applicationContext-security-web.xml (controls web services)		
Property	**Value**	**Description**
passwordExpirationInDays	0 <default> <any other value>	Set the value to any positive, non-zero value to specify the number of days after which a password expires.

8.2.3   Allowing Users to Change their Passwords

To allow users to change their passwords, this configuration makes the **Change Password** link appear on the Login page. By default, this option is turned off, and an administrator must define user passwords initially or reset a forgotten password. Enabling the password expiration option (described in the previous section) automatically enables users to change their passwords.

> ⚠ If your users are externally authenticated, for example with LDAP, do not enable this option.

Password Administration Option		
Configuration File		
.../WEB-INF/jasperserver-servlet.xml		
Property	**Value**	**Description**
allowUserPasswordChange	false <default> true	Set the value to true to enable the **Change Password** link. Any other value disables it.

8.2.4   Enforcing Password Patterns

If you allow or force users to change their passwords, you can enforce patterns for valid strong passwords, by requiring a minimum length and a mix of uppercase, lowercase, and numbers. The default pattern accepts any password of any length, including an empty password.

> ⚠ If your users are externally authenticated, for example with LDAP, do not enable this option.

Password Administration Option		
Configuration File		
.../WEB-INF/applicationContext.xml		
Property	**Bean**	**Description**
`allowedPasswordPattern`	`userAuthority Service`	A regular expression that matches valid passwords. The default pattern `^.*$` matches any password. Change the regular expression to enforce patterns such as: • Minimum and maximum password length • Both uppercase and lowercase characters • At least one number or special character Be sure that your pattern allows whitespace and international characters if needed by your users.

When you enforce a password pattern, you should set the following message to inform users why their password was rejected. Be sure to set the message in all your locales.

Password Administration Option	
Configuration File	
.../WEB-INF/bundles/jsexceptions_messages[_locale].properties	
Property	**Description**
`exception.remote.weak.password`	Message displayed to users when password pattern matching fails.

8.3   Configuring the User Session Timeout

The user session timeout is the length of time a user's session can remain inactive before the server automatically logs the user out. JasperReports Server displays a pop-up reminder that tells users their session is about to expire and gives them the chance to continue without being logged out.

User Session Timeout
Configuration File
.../WEB-INF/web.xml

User Session Timeout		
Property	**Value**	**Description**
`<session-config>` `  <session-timeout>`	20 <default>	Set the number of minutes that a user session can remain idle before automatic logout. Set the value to 0 (zero) to prevent sessions from ever timing out.

Note that the session timeout also applies to how long a session remains in memory after a web services call finishes. If there is another web service call with the same credentials within the timeout period, the server reuses the same session. If the timeout is too short for this case, you may have performance issues caused by a high load of web service calls.

If the timeout is too long, a user session may stay active for a long time (even indefinitely with a timeout of 0) if a user leaves his browser open. The risk of allowing long sessions is that the in-memory session is not updated with any role changes until the user logs out manually (ending the session) and logs in again (creating a new session).

8.4 Configuring CSRF Prevention

Cross-Site Request Forgery (CSRF) is an exploit where the attacker impersonates a valid user session to gain information or perform actions. In JasperReports Server, the security framework protects every page with a CSRF token in the post header, for example:

```
JASPER_CSRF_TOKEN: BVSY-UBBJ-K8E9-L4NZ-5866-Z4P2-ZG75-KKBW-U53Z-I833-V0OJ-BRK5-OFG5-
ZL6X
```

In the default configuration of the server, CSRF prevention is active. We recommend leaving this setting unchanged:

CSRF Prevention		
Configuration File		
.../WEB-INF/classes/esapi/security-config.properties		
Property	**Value**	**Description**
`security.validation.csrf.on`	`true` <default> `false`	Turns CSRF prevention on or off. By default, CSRF prevention is on. Any other value besides case-insensitive "false" is equivalent to true.

8.5 Configuring Input Validation

To block potential security threats like cross-site scripting and SQL injection, the security framework validates all user input and values passed to reports. Input validation prevents input with dangerous side-effects like malicious scripts and queries.

Administrators can monitor the server logs to search for evidence of attempted security breaches. However, input that was allowed in previous versions of the server may be blocked, and users may see errors when entering values. In particular:

- Parameter names and values can't have tags (< and >). If your business data contains tags, you need to update the security configuration to allow them.
- SQL queries should start with SELECT and can't have comments. Multiple queries separated by semi-colons (;) are also prohibited. If your reports or Domains have such queries, you need to either change them or update the security configuration to allow them.

If users see recurring errors, administrators can examine logs to determine what input is not allowed. Preferably, users should modify their input to remove special characters that are security risks. If that's not feasible, administrators can configure the security framework to modify security rules or turn off the security restrictions, based on their perceived threat level.

> Input validation rules were added to comply with security guidelines for web applications. Turning off input validation or modifying the validation rules may make the server more vulnerable to web attacks.

Input validation is complex and configured in the following files:

File	Contents
<js-webapp>/WEB-INF/classes/esapi/security-config.properties	Top-level configuration for enabling or disabling input validation.
<js-webapp>/WEB-INF/bundles/security.properties	Text of validation error messages shown to users.
<js-webapp>/WEB-INF/classes/esapi/security.properties	Defines the input validation rules for each field of the server's web pages and report input.
<js-webapp>/WEB-INF/classes/esapi/validation.properties	Defines the regular expressions used in security rules.

> Input validation is based on UTF-8 encoded input. Make sure your application server is configured for UTF-8 URIs as described in section **"UTF-8 Configuration" on page 266**.

Input validation is enabled by default when installing JasperReports Server. To turn off one or more of the protection features:

Input Validation
Configuration File
.../WEB-INF/classes/esapi/security-config.properties

Input Validation		
Property	**Value**	**Description**
`security.validation.input.on`	`true` <default> `false`	Turns field input validation on or off for the server web application. Any other value besides case-insensitive "false" is equivalent to true.
`security.validation.sql.on`	`true` <default> `false`	Turns SQL query validation on or off in the server. Any other value besides case-insensitive "false" is equivalent to true.

8.5.1   Customizing Security Error Messages

When input validation blocks input that violates a security rule, the server displays a generic error. By default, security messages are intentionally generic to avoid alerting potential attackers to security errors.

We highly recommend that external deployments customize the security error messages to be unique, yet still generic. You can change both the message and the error number. Choose any combination of numbers or letters so administrators can easily search the logs to detect security violations.

Input Validation Messages	
Configuration File	
.../WEB-INF/bundles/security.properties	
Property	**Value**
`message.validation.input`	`An error has occurred. Please contact your system administrator. (5321)` <default>
`message.validation.sql`	`An error has occurred. Please contact your system administrator. (6632)` <default>

Set these properties to messages and error codes that match the rest of your application. The goal is to display a message that attackers will not recognize as a security error, yet one that administrators can uniquely identify in the logs.

If you translate your application into other languages, be sure to create a locale-specific copy of this file and translate these messages as well.

8.5.2   Editing Input Validation Rules

Input validation rules determine what input is allowed when users send information to the server. This information generally consists of parameter-value pairs, for example the fields of an input form. For each known parameter-value pair, an input validation rule defines the following:

- What characters are allowed in the parameter name.

- What characters are allowed in the input value.
- The maximum allowable length for the parameter name and the input value (the same limit applies to both separately).
- Whether the value can be blank.

Allowed characters are determined by a regular expression called a validator. Validators are named regular expressions that can be used in any number of input validation rules. Although validators can be used in several rules, each validation rule should be as specific as possible to the allowable input.

The predefined input validation rules in JasperReports Server are designed to allow all data and normal user input, while blocking potential attacks on the server. If your data or your user input causes security errors (false positives), you may choose to modify the input validation rules to allow your input.

1. Locate the "SECURITY FAILURE" message in your logs. For more information about logs, see **"Configuring System Logs" on page 199**. The log message contains the name of the parameter and context where the parameter was used.
2. Make a backup copy of the file <js-webapp>/WEB-INF/classes/esapi/security.properties, then open it for editing.
3. Locate the parameter name and context. For example, this is the input validation rule for the entities parameter on the Manage Roles page:

    ```
    entities=Alpha,AlphaNumPunctuation,5000,true,entities-Manage_Roles_context
    ```

    The input validation rule has the following format:

    <parameter>=<nameValidator>,[!]<valueValidator>,<charLimit>,<blankAllowed>,<parameter>-<context>_context

4. Modify the rule to allow your input:

    a. Usually, you need to change the value validator to one that allows your input characters. Select a value validator from the file <js-webapp>/WEB-INF/classes/esapi/validation.properties that allows your input, or create one as described in the next section.

    b. If your input is atypically long, increase the character limit.

    c. Do not change any other part of the rule.

5. Save your changes and redeploy the JasperReports Server webapp, or restart your application server.

Recommendations:

- Try to keep the character limit as close to the expected value as possible.
- Try to use a validator as close to the expected values as possible. If a parameter's value is expected to be numbers only, use the Numeric validator.
- Most validators are whitelists that specify allowed character patterns. A validator may be preceded by an exclamation point (!) to indicate that everything but those values is permitted. When used with a validator that matches characters or words, this syntax implements a blacklist. Some rules are easier to define as whitelists, others as blacklists.
- If a parameter can have radically different values or the same parameter is used in different situations, you can apply more than one rule to that parameter. To do this, simply copy a parameter rule and add incremental integers to the parameter name. For example:

    ```
    standAlone=Alpha,Alpha,50,true,standAlone-Report_PopupMenu_context
    ```

    Updated to:

    ```
    standAlone=Alpha,AlphaNum,50,true,standAlone-Report_PopupMenu1of3_context
    standAlone2=Alpha,JSONObject,50000,true,standAlone-Report_PopupMenu2of3_context
    standAlone3=Alpha,JSONArray,500000,true,standAlone-Report_PopupMenu3of3_context
    ```

> With multiple rules for the same parameter, each rule is applied in the order listed until one passes (equivalent to a logical OR). If they all fail, the input is blocked and the user is notified with the generic error message. The rules that fail still appear as security warnings in the logs. Use numbering in the context names, as shown above, to easily identify these false-positive messages. When using multiple rules, define the most used rule or the most permissive rule first to optimize validation and reduce false-positive log messages.

8.5.3   Creating Validator Expressions

The validators are Java-based regular expressions that specify characters allowed (whitelist) or forbidden (blacklist), depending on how they're used in a validation rule.

> *Do not* modify the default validator expressions provided with the server. We have thoroughly tested these expressions to provide reasonable input validation security while allowing for the general use of the application. Also, a validator can be used in several input validation rules, so modifying them may have unintended consequences. You should *always* create new validators with new names.

1. Make a backup copy of the file <js-webapp>/WEB-INF/classes/esapi/validation.properties, then open it for editing.
2. Locate the validator used in the input validation rule you want to modify, for example the Alpha validator expression allows for any letters in any language:

    ```
    Validator.Alpha=^[\\p{L}\\p{M}]*$
    ```

3. Copy the entire rule on a new line and give it a new name with the following format:

    Validator.<validatorName>=<regularExpression>

    Remember to use double backslashes (\\) in properties files for single backslashes in the expression. You should also use the \p{} syntax to match international letters and their accent marks, for example:

    ```
    Validator.AlphaDotSpace=^[\\p{L}\\p{M}\\.\\s]*$
    ```

4. Use the new value validator name in your input validation rule, as described in the previous procedure.
5. Save your changes and redeploy the JasperReports Server webapp, or restart your application server.

8.5.4   Validating New Input Parameters

If you customize JasperReports Server to accept new input parameters, you must add the corresponding input validation rules to maintain server security.

1. Make a backup copy of the file *<js-webapp>/WEB-INF/classes/esapi/security.properties*, then open it for editing.
2. Create a new input validation rule in the following format:

    <parameter>=<nameValidator>,<valueValidator>,<charLimit>,<blankAllowed>,<parameter>-<context>_context

    The context is the string that will appear in the log when a security validation error occurs, so it should contain the exact parameter name.

3. Look at existing rules in the file <js-webapp>/WEB-INF/classes/esapi/validation.properties to find validators for the parameter name and value that allows your new input. If necessary, create new validator expressions as described in the previous procedure.
4. Save your changes and redeploy the JasperReports Server web app, or restart your application server.

8.5.5　Query Validation

Query validation is a special case of input validation, where the server ensures that all queries being issued by the server meet a preset pattern for a safe query. When query validation is enabled, all queries in reports and Domains use the following validator:

```
Validator.ValidSQL=^\\s*((?i)select)\\s+[^;]+$
```

As a result:

- SQL comments are forbidden.
- Ensure that you have only one executable query statement per query. Multiple queries separated by semi-colons (;) will be rejected. The following example will cause a security error:

  ```
  SELECT f1,f2 FROM tbl_1 WHERE f1=f2; SELCT f3 from tbl_2;
  ```

- Queries for reports must retrieve data only, in other words, use only the SELECT statement. The following statements are forbidden:

  ```
  DROP, INSERT, UPDATE, DELETE
  ```

- If you want to use stored procedures, you must add the following validator to the file <js-webapp>/WEB-INF/classes/esapi/validation.properties:

  ```
  Validator.ValidSPROC=^\\s*\\(((?i)call)\\s+[^;]+\\)$
  ```

  And then modify the validation rule for the corresponding parameter or field where you want to allow stored procedure calls. If you want to allow stored procedure calls in addition to select statements, specify multiple validation rules as shown in section **"Editing Input Validation Rules" on page 161**.

- When SQL validation fails, the logs contain the message described in **"Customizing Security Error Messages" on page 161**, such as the following:

  ```
  2011-11-21 13:54:28,007 ERROR ValidatorImpl,"http-bio-8090"-exec-12:48 - An error has
  occurred. Please contact your system administrator. (6632)
  org.owasp.esapi.errors.ValidationException: SQL_Query_Executor_context: Invalid
  input. Please conform to regex ^\s*((?i)select)\s+[^;]+$ with a maximum length of
  50000
  ```

8.5.6　Further Configuration

The configuration files contain some miscellaneous default settings for the security framework. In particular they define default action for input that has no validation rules. Changing these defaults is possible but not recommended:

Advanced Input Validation
Configuration File
.../WEB-INF/classes/esapi/security-config.properties

Advanced Input Validation		
Property	**Default Value**	**Description**
`log.msg.security.off`	SECURITY for [%s] is OFF	If security is turned OFF, this message will be logged. This message in the logs can alert administrators if the security configuration has been tampered with.
`msg.no.rule`	No rule for parameter [%s]. Using default validation on input=[%s].	If a request parameter is not previously known, this message is logged.
`msg.cannot.load`	Security configuration [%s] cannot be loaded.	If there is an error in the security configuration files, this message is logged. This is a severe error and should be resolved by the administrator.
Configuration File		
.../WEB-INF/classes/esapi/security.properties		
Property	**Default Value**	**Description**
`DEFAULT`	`Alpha,AlphaNumPunctuation Brackets,200000,true,DEFAULT`	If an input parameter does not have any defined validation rule, this validation rule is applied. The validator for values, `AlphaNumPunctuation-Brackets` is fairly permissive, and can be changed to something more restrictive. The DEFAULT property name is a keyword and should never be changed.

8.6 Restricting File Uploads

Several dialogs in JasperReports Server prompt the user to upload a file to the server. For performance and security reasons, you may want to restrict file uploads by name and size.

The following setting is the global file upload limit for the entire server. Any single upload that exceeds this limit will trigger an error and a stack trace message. It's intended to be an absolute maximum to prevent a worse out-of-memory error that affects the entire server.

Global File Size Upload Limit		
Configuration File		
…/WEB-INF/js.config.properties		
Property	**Value**	**Description**
file.upload.max.size	–1 <default>	Maximum size in bytes allowed for any file upload. The default value, -1, means there is no limit to the file size, and a large enough file could cause an out-of-memory error in the JVM. Some file uploads such as importing through the UI are necessarily large and must be taken into account. Set this value larger than your largest expected import and less than your available memory.

The following settings apply to most file upload dialogs in the user interface, such as uploading a JRXML or a JAR file to create a JasperReport in the repository. These settings in the fileResourceValidator bean restrict the file size and the filename pattern.

File Upload Restrictions		
Configuration File		
…/WEB-INF/flows/fileResourceBeans.xml		
Property	**Value**	**Description**
maxFileSize	–1 <default>	Maximum size in bytes allowed for a file uploaded through most UI dialogs. If an upload exceed this limit, the server displays a helpful error message. The default value, -1, means there is no limit to the file size, and an upload could reach the global limit if set, or an out-of-memory error. Usually, files required in resources are smaller, and a limit of 10 MB is reasonable.
fileNameRegexp	^.+$ <default>	A regular expression that matches allowed file names. The default expression matches all filenames of one or more characters. A more restrictive expression such as [a-zA-Z0-9]{1,200}\.[a-zA-Z0-9]{1,10} would limit uploads to alpha-numeric names with an extension.
fileName ValidationMessageKey	<null/> <default>	The name of a Java property key whose value is a custom message to display when the uploaded filename does not match fileNameRegexp. For example, you could add the following line to WEB-INF/js.config.properties: my.filename.validation=The name of the uploaded filename must contain only alphanumeric characters and have a valid extension.

The following setting restricts the extension of the uploaded file. The upload dialogs will browse only for files with the given extensions. Add or remove extensions to change the file type restrictions:

File Upload Extensions	
Configuration File	
<jasperserver-pro-war>/scripts/resource.locate.js	
Property	**Value**
`ALLOWED_FILE_` `RESOURCE_EXTENSIONS`	By default, the following extensions are allowed: `"css", "ttf", "jpg", "jpeg", "gif", "bmp", "png", "jar", "jrxml", "properties", "jrtx", "xml", "agxml", "docx", "doc", "ppt", "pptx", "xls", "xlsx", "ods", "odt", "odp", "pdf", "rtf", "html"`

8.7 Hiding Stack Trace Messages

By default, JasperReports Server displays stack traces in certain error messages. Stack traces reveal some information about the application, and security experts recommend that an application not display them.

The following setting turns off stack traces in error messages. However, more serious error messages still display Java exceptions without the stack trace.

Hiding Stack Trace Messages		
Configuration File		
.../WEB-INF/applicationContext-security-web.xml		
Property	**Bean**	**Description**
`showStacktrace` `Message`	`webAppSecurity` `Filter`	Set this value to false to prevent stack traces from appearing in error messages.

Restart your application server or redeploy the JasperReports Server web app for this change to take effect.

8.8 Defining a Cross-Domain Policy for Flash

For security reasons, a Flash animation playing in a web browser is not allowed to access data that resides outside the exact web domain where the SWF originated. JasperReports Server uses Flash for advanced Fusion-based charts such as gauges and maps.

As a result, even servers in subdomains cannot share data with a server in the parent domain unless they define a cross-domain policy that explicitly allows it. The file crossdomain.xml, located at the root of the server that contains the data, defines what domains may access the data without prompting the user to grant access in a

security dialog. Therefore, the server where the data is located determines which other servers may access the data.

The following crossdomain.xml sample allows access only from the example domain or any of its subdomains. This example is saying that the server with this file trusts only example.com to use its data.

```
<?xml version="1.0" ?>
    <!DOCTYPE cross-domain-policy SYSTEM
        "http://www.macromedia.com/xml/dtds/cross-domain-policy.dtd">

    <cross-domain-policy>
        <allow-access-from domain="example.com" />
        <allow-access-from domain="*.example.com" />
    </cross-domain-policy>
```

Behind a firewall, servers and users often refer to other computers in the same domain without using the domain name. Flash considers this a different domain and blocks access to data unless the computer name is given in the policy:

```
    <cross-domain-policy>
        <allow-access-from domain="myserver.example.com" />
        <allow-access-from domain="myserver" />
    </cross-domain-policy>
```

When using web services, use the `allow-http-request-headers-from` element so that actions encoded in the request header are allowed. The following example allows standard requests and web service requests from any subdomain of example.com.

```
    <cross-domain-policy>
        <site-control permitted-cross-domain-policies="master-only"/>
        <allow-access-from domain="*.example.com"/>
        <allow-http-request-headers-from domain="*.example.com" headers="*"
            secure="true"/>
    </cross-domain-policy>
```

For a description of all possible properties, see the cross-domain policy file specification.

To define a cross-domain policy for Flash-based reports, create a file such as the ones above on the server that contains the data being accessed. Be sure to place the crossdomain.xml file at the root of the filespace that's being served. For example, if you use Apache Tomcat, place your files in the following locations:

File	Location
crossdomain.xml	<website-B-tomcat-dir>/webapps/ROOT/crossdomain.xml
XML data (*.xml)	<website-B-tomcat-dir>/webapps/ROOT/<any-dir>/*.xml
Flash component (*.swf)	<website-A-tomcat-dir>/webapps/<appname>/<any-dir>

8.9   Encrypting User Passwords

User passwords are stored along with user profiles in JasperReports Server's own private database. By default, password encryption is enabled and passwords are stored as cipher text in the database. With the following procedure, system administrators can turn on or off user password encryption, or change the encryption algorithm and specify the salt key used to initiate the encryption algorithm.

To Configure User Password Encryption:

1. As a precaution, back up the server's private `jasperserver` database. To back up the default PostgreSQL database, go to the <js-install> directory and run the following command:

    ```
    pg_dump -U postgres jasperserver > js-backup.sql
    ```

    To back up DB2, Oracle, Microsoft SQL Server, and MySQL databases, refer to your database product documentation.

2. You can now stop your application server. You should leave your database running.

3. Export the entire contents of the repository, which includes user profiles and their passwords, with the following commands. Note that there are two dashes (--) in front of the command options:

    ```
    Windows:  cd <js-install>\buildomatic
              js-export.bat --everything --output-dir js-backup-catalog

    Linux:    cd <js-install>/buildomatic
              js-export.sh --everything --output-dir js-backup-catalog
    ```

    In the export operation, passwords are decrypted using the existing user password ciphers and re-encrypted with the import-export encryption key. This is a separate encryption that ensures that passwords are never in plain text, even when exported. For more information, see **"Setting the Import-Export Encryption Key" on page 136**

4. Edit the properties in the following table to configure different ciphers. Both the server and the import-export scripts access the user profiles and must be configured identically. Make the same changes in both files:

Table 8-1 User Password Encryption Configuration

Configuration Files		
<jasperserver-pro-war>/WEB-INF/applicationContext-security.xml <js-install>/buildomatic/conf_source/iePro/applicationContext-security.xml		
Property	**Bean**	**Description**
`allowEncoding`	`passwordEncoder`	With the default setting of `true`, user passwords are encrypted when stored. When `false`, user passwords are stored in clear text in JasperReports Server's private database. We do not recommend changing this setting.

keyInPlainText	passwordEncoder	When `true`, the `secretKey` value is given as a plain text string. When `false`, the secretKey value is a numeric representation that can be parsed by Java's Integer.decode() method. By default, this setting is false, and the `secretKey` is in hexadecimal notation (0xAB).
secretKey	passwordEncoder	This value is the salt used by the encryption algorithm to make encrypted values unique. This value can be a text string or a numeric representation depending on the value of `keyInPlainText`.
secretKeyAlgorithm	passwordEncoder	The name of the algorithm used to process the key, by default `DESede`.
cipher Transformation	passwordEncoder	The name of the cipher transformation used to encrypt passwords, by default `DESede/CBC/ PKCS5Padding`.

> You should change the `secretKey` value so that it is different from the default.

The `secretKey`, `secretKeyAlgorithm`, and `cipherTransformation` properties must be consistent. For example, the `secretKey` must be 24 bytes long in hexadecimal notation or 24 characters in plain text for the default cipher (DESede/CBC/PKCS5Padding). Different algorithms expect different key lengths. For more information, see Java's `javax.crypto` documentation.

5. Next, drop your existing `jasperserver` database, where the passwords had the old encoding, and recreate an empty `jasperserver` database. Follow the instructions for your database server:
   - **Dropping and Recreating the Database in PostgreSQL**
   - **Dropping and Recreating the Database in MySQL**
   - **Dropping and Recreating the Database in Oracle**
   - **Dropping and Recreating in the Database in Microsoft SQL Server**

6. Import your exported repository contents with the following commands. The import operation will restore the contents of JasperReports Server's private database, including user profiles. As the user profiles are imported, the passwords are encrypted using the new cipher settings.

   Note that there are two dashes (--) in front of the command options:

   ```
   Windows:  cd <js-install>\buildomatic
             js-import.bat --input-dir js-backup-catalog
   Linux:    cd <js-install>/buildomatic
             js-import.sh --input-dir js-backup-catalog
   ```

   During the import operation, passwords are decrypted with the import-export encryption key and then re-encrypted in the database with the new user password encryption settings. For more information, see **"Setting the Import-Export Encryption Key" on page 136**.

7. Using a database client such as the SQuirreL tool, check the contents of the `JIUser` table in the `jasperserver` database and verify that the password column values are encrypted.

8. Restart your application server. Your database should already be running.

9. Log into JasperReports Server to verify that encryption is working properly during the log in process.

8.9.1 Dropping and Recreating the Database in PostgreSQL

1. Change directory to `<js-install>/buildomatic/install_resources/sql/postgresql`.
2. Start psql using an administrator account such as postgres:
   ```
   psql -U postgres
   ```
3. Drop the `jasperserver` database, create a new one, and load the `jasperserver` schema:

   ```
   drop database jasperserver;
   create database jasperserver encoding='utf8';
   \c jasperserver
   \i js-pro-create.ddl
   \i quartz.ddl
   ```

8.9.2 Dropping and Recreating the Database in MySQL

1. Change directory to `<js-install>/buildomatic/install_resources/sql/mysql`.
2. Log into your MySQL client:
   ```
   mysql -u root -p
   ```
3. Drop the jasperserver database, create a new one, and load the jasperserver schema:

   ```
   mysql>drop database jasperserver;
   mysql>create database jasperserver character set utf8;
   mysql>use jasperserver;
   mysql>source js-pro-create.ddl;
   mysql>source quartz.ddl;
   ```

8.9.3 Dropping and Recreating the Database in Oracle

1. Change directory to `<js-install>/buildomatic/install_resources/sql/oracle`.
2. Log into your SQLPlus client, for example:
   ```
   sqlplus sys/sys as sysdba
   ```
3. Drop the `jasperserver` database, create a new one, and load the `jasperserver` schema:

   ```
   SQL> drop user jasperserver cascade;
   SQL> create user jasperserver identified by password;
   SQL> connect jasperserver/password
   SQL> @js-pro-create.ddl
   SQL> @quartz.ddl
   ```

8.9.4 Dropping and Recreating in the Database in Microsoft SQL Server

1. Change directory to `<js-install>/buildomatic/install_resources/sql/sqlserver`.
2. Drop the `jasperserver` database, create a new one, and load the `jasperserver` schema using the SQLCMD utility:

```
cd <js-install>\buildomatic\install_resources\sql\sqlserver
sqlcmd -S ServerName -Usa -Psa
1> DROP DATABASE [jasperserver]
2> GO
1> CREATE DATABASE [jasperserver]
2> GO
1> USE [jasperserver]
2> GO
1> :r js-pro-create.ddl
2> GO
1> :r quartz.ddl
2> GO
```

8.10 Encrypting User Session Login

By default, JasperReports Server does *not* enable the Secure Socket Layer/Transport Layer Security (SSL/TLS) to encrypt all data between the browser and the server, also known as HTTPS. Enabling HTTPS, as documented in the *JasperReports Server Ultimate Guide*, requires a certificate and a careful configuration of your servers. We recommend implementing HTTPS but recognize that it is not always feasible.

Without HTTPS, all data sent by the user, including passwords, appear unencrypted in the network traffic. Because passwords should never be visible, JasperReports Server provides an independent mechanism for encrypting the password values without using HTTPS. Passwords are encrypted in the following cases:

• Passwords sent from the login page.
• Passwords sent from the change password dialog (see **"Configuring User Password Options" on page 155**).
• Passwords sent from the user management pages by an administrator.

When a browser requests one of these pages, the server generates a private-public key pair and sends the public key along with the page. A JavaScript in the requested page encrypts the password when the user posts it to the server. Meanwhile, the server saves its private key and uses it to decrypt the password when it arrives. After decrypting the password, the server continues with the usual authentication methods.

Login encryption is not compatible with password memory in the browser. Independent of the autocomplete setting described in section **"Configuring Password Memory" on page 156**, the JavaScript that implements login encryption clears the password field before submitting the page. As a result, most browsers will never prompt to remember the encrypted password.

The disadvantage of login encryption is the added processing and the added complexity of web services login. For backward compatibility, login encryption is disabled by default. To enable login encryption, set the following properties. After making any changes, redeploy the JasperReports Server webapp or restart the application server.

> When login encryption is enabled, web services and URL parameters must also send encrypted passwords. Your applications must first obtain the key from the server and then encrypt the password before sending it. See the *JasperReports Server Web Services Guide* and *JasperReports Server Ultimate Guide*, respectively.

Login Encryption		
Configuration File		
.../WEB-INF/classes/esapi/security-config.properties		
Property	**Value**	**Description**
`encryption.on`	`truefalse` <default>	Turns login encryption on or off. Encryption is off by default. Any other value besides case-insensitive "false" is equivalent to true.
`encryption.type`	`RSA` <default>	Encryption algorithm; currently, only RSA is supported.
`encryption.key.length`	integer power of 2 `1024` <default>	The length of the generated encryption keys. This affects the strength of encryption and the length of the encrypted string.
`encryption.dynamic.key`	`true` <default> `false`	When true, a key will be generated per every single request. When false, the key will be generated once per application installation. See descriptions in **Dynamic Key Encryption** and **Static Key Encryption** below.

Encryption has two modes, dynamic and static, as determined by the `encryption.dynamic.key` parameter. These modes provide different levels of security and are further described in the following sections.

8.10.1   Dynamic Key Encryption

The advantage of encrypting the password at login is to prevent it from being seen, but also to prevent it from being used. For password encryption to achieve this, the password must be encrypted differently every time it is sent. With dynamic key encryption, the server uses a new public-private key pair with every login request.

Every time someone logs in, the server generates a new key pair and sends the new public key to the JavaScript on the page that sends the password. This ensures that the encrypted password is different every time it's sent, and a potential attacker won't be able to steal the encrypted password to log in or send a different request.

Because it's more secure, dynamic key encryption is the default setting when encryption is enabled. The disadvantage of dynamic keys is that generating keys slows down each login, though users may not always notice. Another effect of dynamic key encryption is that it doesn't allow remembering passwords in the browser. While this may seem inconvenient, it's more secure to not store passwords in the browser. See **"Configuring Password Memory" on page 156**.

8.10.2   Static Key Encryption

JasperReports Server also supports static key encryption. In this case, a unique key pair is generated automatically on the user's first login and remains the same for the entire server installation. Because the key is always the same, the encrypted value of a user's password is always the same. This means an attacker could steal the encrypted password and use it to access the server.

Static key encryption is very insecure and is recommended only for intranet server installation where the network traffic is more protected. The only advantage of static encryption over no encryption at all is that passwords can't be deciphered and used to attack other systems where users might have the same password.

Before setting `encryption.dynamic.key=false` to use static encryption, you must also configure the secure file called keystore where the key pair is kept. Be sure to customize the keystore parameters listed in the following table to make your keystore file unique and secure.

> For security reasons, always change the default keystore passwords immediately after installing the server.

Keystore Configuration (when encryption.dynamic.key=false)		
Configuration File		
.../WEB-INF/classes/esapi/security-config.properties		
Property	**Value**	**Description**
`keystore.location`	`keystore.jks` <default>	Path and filename of the keystore file. This parameter is either an absolute path or a file in the webapp classpath, for example <tomcat>/webapps/jasperserver-pro/WEB-INF/classes>. By default, the keystore.jks file is shipped with the server and doesn't contains any keys.
`keystore.password`	`jasper123` <default>	Password for the whole keystore file. This password is used to verify keystore's integrity.
`keystore.key.alias`	`jasper` <default>	Name by which the single key is retrieved from keystore. If a new alias is specified and does not correspond to an existing key, a new key will be generated and inserted into the keystore.
`keystore.key.password`	`jasper321` <default>	Password for the key whose alias is specified by keystore.key.alias.

When changing the key alias, the old key will not be deleted; it can be used again by resetting the key alias. Also, once the key has been created with a password, you can't change the password through the keystore configuration. To delete keys or change a keystore password, the server administrator must use the Java `keytool.exe` utility in the bin directory of the JRE or JDK. If you change the keystore password or the key password, the keystore configuration above must reflect the new values or login will fail for all users.

CHAPTER 9   SYSTEM CONFIGURATION

You can change the default behavior of JasperReports Server by editing the system's configuration. The configuration is defined by a set of properties and their values.

The properties are stored in configuration files located in various folders under the <js-install> directory, which is the root of your JasperReports Server installation. To change the configuration, you edit these files and then restart the server.

A few of the most commonly edited properties are available to the system admin through the user interface (UI). Changes to these properties take effect immediately, are stored in the repository, and override the equivalent values stored in files, even after the server restarts (as of version 5.0).

This chapter describes a subset of the properties in the configuration files. Settings that affect security are covered in **"Application Security" on page 149**. Configuration of the auditing feature is covered in **"Configuring Auditing and Monitoring" on page 231**. More options are described in the *JasperReports Server Installation Guide*.

Because the locations of files described in this chapter vary with your application server, the paths specified here are relative to the deployed WAR file for the application. For example, the applicationContext.xml file is shown as residing in the WEB-INF folder; if you use the Tomcat application server bundled with the installer, the default path to this location is:

> C:\Program Files\jasperreports-server-6.0\apache-tomcat\webapps\jasperserver-pro\WEB-INF

> Use caution when editing the properties described in this chapter. Inadvertent changes may cause unexpected errors throughout JasperReports Server that may be difficult to troubleshoot. Before changing any files, back them up to a location outside of your JasperReports Server installation.
>
> Do not modify settings that are not described in the documentation. Even though some settings may appear straightforward, values other than the default may not work properly and cause errors.

> This section describes functionality that can be restricted by the software license for JasperReports Server. If you don't see some of the options described in this section, your license may prohibit you from using them. To find out what you're licensed to use, or to upgrade your license, contact us.

> In addition to configuration described in this chapter, you can configure Liferay Portal or JBoss Portal to display the reports stored in your JasperReports Server instance. You can download the JasperReports Server portlets for these environments from the Jaspersoft Support Portal. For information about how to deploy the portlet, refer to the documentation in the portlet download package.

This chapter contains the following sections:

- **Configuration Settings in the User Interface**
- **Configuration for Using Proxies**
- **Configuration for Session Persistence**
- **Configuring Ad Hoc**
- **Enabling Data Snapshots**
- **Configuring System Logs**
- **Configuring Amazon Web Services**
- **Configuring Domains**
- **Configuring JasperReports Library**
- **Configuring Input Control Behavior**
- **Configuring the Scheduler**
- **Configuring the Heartbeat**
- **Configuring the Online Help**

9.1 Configuration Settings in the User Interface

As of JasperReports Server 5.0, changes to configuration settings in the user interfaces are persistent through server restarts. Previously, upon restart configuration settings would revert to values stored in the configuration files.

To make persistent configuration changes through the JasperReports Server user interface:

1. Log in as system administrator (superuser by default).

2. Select **Manage > Server Settings**:
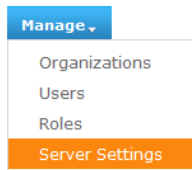
3. Choose a category of settings or administrator actions from the left-Settings panel.
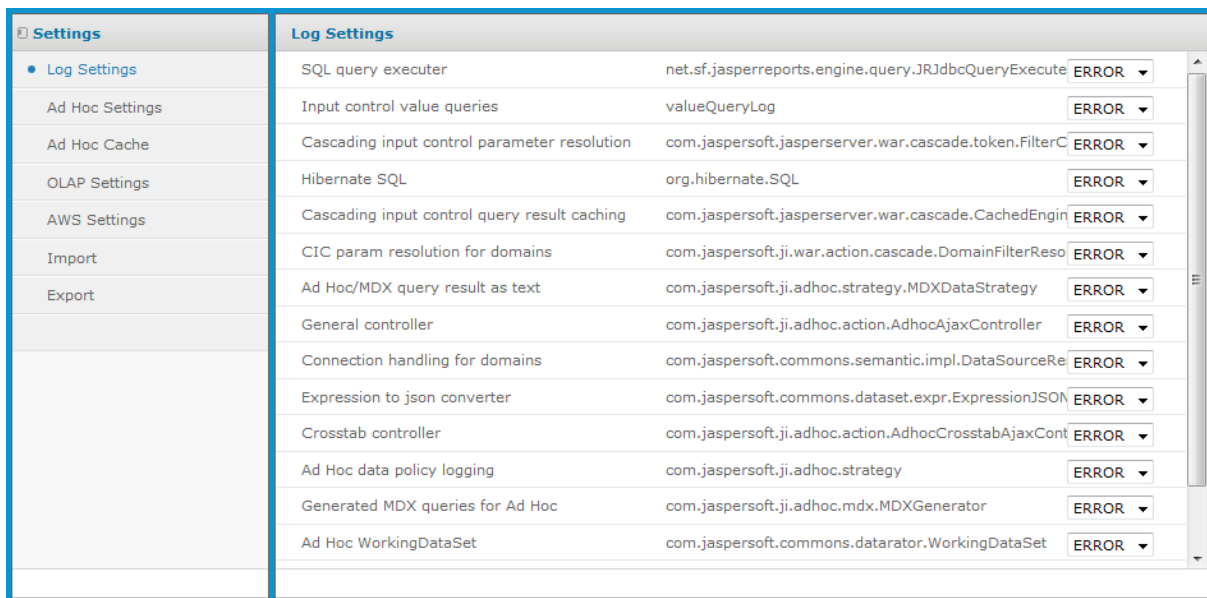
Figure 9-1 **The User Interface for Configuration Settings**

4. Find the configuration setting you want to change and edit its value. In the case of log levels, the new value takes effect immediately. In the case of other settings, click **Change** beside the individual setting.

These settings and administrator actions are documented in their respective sections:

Settings	Documentation
Log Settings	**"Configuring System Logs" on page 199**
Ad Hoc Settings	**"Ad Hoc Query Settings" on page 182**
Ad Hoc Cache	**"Ad Hoc Cache Management" on page 188**
OLAP Settings	*Jaspersoft OLAP User Guide*
AWS Settings	**"Configuring Amazon Web Services" on page 204**
Import	**"Importing From the Settings" on page 139**
Export	**"Exporting From the Settings" on page 138**

Here are some guidelines for modifying the configuration settings that appear in the user interface (Log Settings, Ad Hoc Settings, and OLAP Settings):

• The Settings pages display a subset of the settings available in configuration files. Therefore, all settings in the UI also exist in a configuration file.

• By default, the Settings pages display the values of settings that exist in the corresponding configuration file. If you modify the files and restart the server, your new file settings take effect on the server and are visible in the UI.

- When you change a value on the Settings pages, the new setting takes effect immediately, but the new value is *not* written to the corresponding configuration file. Instead, it's stored in a special resource in the repository, called the GlobalPropertiesList. The GlobalPropertiesList is located in the System Properties folder at the root of the repository and is visible only to the system admin (superuser).
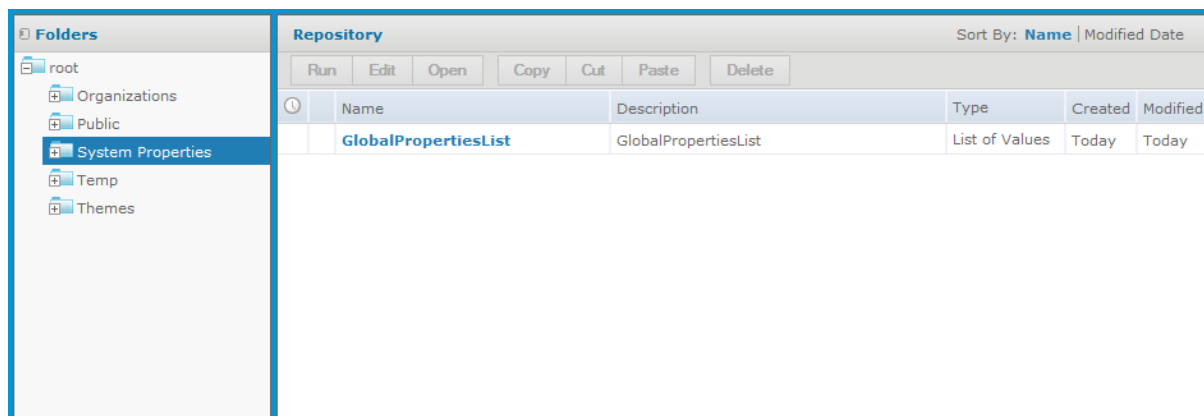
Figure 9-2 **The System Properties Folder Visible to System Admins**

Only configuration settings that have a value modified on the Settings pages of the UI are stored and made persistent in the GlobalPropertiesList.

- When the server restarts, the values in the GlobalPropertiesList take precedence over values for the same settings in the configuration files. However, each setting is independent, so a value that's not modified in the Settings UI is read from the configuration files.
- The Settings pages display the values of the settings in effect on the server. Therefore, settings in the GlobalPropertiesList appear on the Settings pages where they can be changed again if necessary.

Be aware that the configuration values that appear on the Settings pages are possibly a mixture of values loaded from configuration files and from the persistent GlobalPropertiesList.

- Right-click the GlobalPropertiesList and select Edit to view the configuration values it stores. If a setting has been modified in the UI, it appears in this list so that the setting is persistent. If you want to reset a setting so that it takes its value from the configuration file instead, click **Remove** next to that setting and then restart the server. Otherwise, we do not recommend modifying this list of values directly.

Figure 9-3 **The GlobalPropertiesList Containing Persistent Configuration Settings**

- The GlobalPropertiesList also includes JDBC drivers configured during the installation or through the data source creation wizard. For more information, see **"Managing JDBC Drivers" on page 70**.
- Storing configuration values in the GlobalPropertiesList in the repository makes it possible to export them to different servers or re-import them after a server upgrade. For more information, see **"Importing From the Settings" on page 139**,

9.2 Configuration for Using Proxies

When setting up JasperReports Server to use a proxy server, there are two additional settings to make in the server configuration. JasperReports Server exposes some URLs to itself through the UI and when using the scheduler to send emails. When using a proxy, those URLs must be configured so that they expose the proxy URL, not actual URL of the server instance.

Change the following settings so that JasperReports Server exposes the proxy URL.

Configuration for Using Proxies
Configuration File
.../WEB-INF/js.config.properties

Configuration for Using Proxies	
Property	**Description**
`deploy.base.url`	Set this property to the full URL for exposing the JasperReports Server UI through the proxy. This URL must include the application name, for example: `http://bi.example.com/jasperserver-pro`
Configuration File	
.../WEB-INF/js.quartz.properties	
Property	**Description**
`report.scheduler.web.deployment.uri`	This is the base URL used by the scheduler to generate links to reports in emails. Set this property to the full URL, including application name, that you expose through your proxy. It should be the same as `deploy.base.url`.

9.3   Configuration for Session Persistence

Many application servers can store user sessions while a web app is offline, for example when changing JasperReports Server configuration files. The app server remembers all the information about a user's session, such as the session ID and what page was being viewed; and when the web app restarts, the user session is restored. The user doesn't have to log in again, and often will not even notice that the server was temporarily unavailable. This is called session persistence.

> Sessions do not persist when redeploying a web app in the application server, only when restarting the web app.

JasperReports Server supports a limited form of session persistence. When session persistence is enabled in the app server, sessions can be restored in the following cases:
*   Browsing the repository, expanding folders in the repository tree and viewing folder contents.
*   Searching the repository, including all search filters and results.
*   Repository permissions dialog, including state and selections.
*   Add folder dialog.
*   Add resource dialogs, including adding or editing a data source, JasperReport, and other repository objects.
*   Copy, cut, and paste resources in the repository.
*   Scheduling a report, including all information such as a schedule and notifications.

If the server becomes unavailable when using the pages or dialogs above, the user will see a pause only when performing an action on these pages, such as submitting. When the server has finished restarting, the user can continue interacting with these UI elements. If the user takes no action while the server is unavailable, he may not even notice the server restart.

However, for other interactive dialogs in JasperReports Server, the state is too large to store in the user session. These features do not support session persistence and unsaved data can be lost:

- Ad Hoc editor - The state of the report layout and data in the Ad Hoc editor can't be restored.
- Dashboard designer - The contents and state of the canvas can't be restored.
- Interactive report viewer - The data in the report, as well as the state of column sorting and filtering can't be saved.
- Domain designer - Any tables, joins, filters, calculated fields, and display names can't be restored.
- OLAP viewer - The data in the OLAP view and current MDX expression can't be restored.
- Administration dialogs (when creating or editing an organization, user, or role) - The information entered in an administration dialog can't be restored if it was not submitted.

In the cases listed above, the user's work is interrupted, and any unsaved work is lost. However, when the server restarts, the user does not have to log in again, the server displays a message about the session that could not be fully restored, and redirects the user to the home page. The user must relaunch the interactive feature and recreate any unsaved work.

Session persistence also affects web service calls. The REST API supports a login to store a session ID, and with persistence enabled, that session ID will still be valid when the application server restarts. This simplifies the code you need to handle timeouts. In general, web service calls do not support interactive work like designing an Ad Hoc view, a dashboard, a Domain, or exploring data in OLAP, so they aren't affected by the lack of session persistence in those cases. However, web service calls are affected in the following case:

- Report execution - All asynchronous API calls for running and exporting reports rely on the large JasperPrint object that can't be persisted. When the server restarts, the asynchronous calls will return errors because the reports could not be saved in the session. Your application needs to detect this error and include code for re-running the report.

> JasperReports Server also supports session replication among multiple instances of the server in a cluster. However, session replication has the same limitations because it is based on session persistence. For more information, see the *JasperReports Server Ultimate Guide*.

The following procedure gives the configuration in JasperReports Server and the Apache Tomcat application server to enable session persistence. For another application server, refer to that server's documentation.

To configure JasperReports Server and Apache Tomcat for session persistence:

1. Edit the file <web-app>/META-INF/context.xml to comment out the Manager property as follows:

```
<!-- Manager pathname="" /-->
```

2. Edit the file <web-app>/WEB-INF/web.xml to make the following changes.

    a. Locate the listener of class `RequestContextListener` and replace it with the listener of class `TolerantRequestContextListener`. The new listener class is given in comments that you need to uncomment as follows:

```
<listener-class>com.jaspersoft.jasperserver.core.util.TolerantRequestContextListener
    </listener-class>
<!--listener-class>org.springframework.web.context.request.RequestContextListener
    </listener-class-->
```

    b. Locate the `ClusterFilter` given in comments and uncomment it as follows:

```
<filter>
    <filter-name>ClusterFilter</filter-name>
    <filter-class>com.jaspersoft.jasperserver.war.TolerantSessionFilter</filter-class>
</filter>
```

c.   Locate the corresponding mapping for the `ClusterFilter` and uncomment that, too. You must also uncomment the `<distributable>` element below it as follows:

```
<filter-mapping>
    <filter-name>ClusterFilter</filter-name>
    <url-pattern>/*</url-pattern>
</filter-mapping>
<distributable/>
```

3.   Add the following property to your JVM environment:

```
-Dorg.apache.catalina.session.StandardSession.ACTIVITY_CHECK=true
```

4.   Restart your Apache Tomcat application server.

9.4   Configuring Ad Hoc

Ad Hoc functionality is available only to JasperReports Server Professional edition users.

Ad Hoc settings help you fine-tune the Ad Hoc Editor and the views created in it. Users with `ROLE_SUPERUSER` can manage the Ad Hoc settings and cache. This section covers the following ways to configure Ad Hoc:

- **Ad Hoc Query Settings**
- **Ad Hoc Data Policies**
- **Ad Hoc Data Policies for Big Data**
- **Ad Hoc Templates and Report Generators**
- **Ad Hoc Configuration File**
- **Ad Hoc Cache Management**
- **Ad Hoc OLAP Filter Configuration**

9.4.1   Ad Hoc Query Settings

The Ad Hoc settings apply to Ad Hoc views based on Topics or Domains. Ad Hoc views based on OLAP connections use the OLAP settings described in the *Jaspersoft OLAP User Guide*.

The Ad Hoc settings include the following:

- General Settings to modify the Ad Hoc editor user interface:
    - **Configure View Query**. Determines whether users can see a button in the Ad Hoc Editor to display the SQL or MDX query generated for the view. This can be useful for advanced users, but you should consider your data security before enabling this. System admins can always view queries in the Ad Hoc Cache (see **"Ad Hoc Cache Management" on page 188**).
    - **Display Null as Zero**. Determines the appearance of null values in crosstabs and time-series charts. By default, this setting is disabled and null values are displayed as empty cells in crosstabs or missing points in time series charts (thus causing irregular intervals). When enabled, null values are displayed as

zero in crosstabs and in time-series (creating regular intervals). Regardless of this setting, null values are always displayed as zero in all other chart types.

- Query Limits to preserve resources used by queries when Ad Hoc views are designed and run:
  - **Ad Hoc Filter List of Values Row Limit**. The maximum number of items that should be displayed in the Condition Editor when a user defines filters for an Ad Hoc view based on a Domain. If this limit is exceeded when users define filters, JasperReports Server displays a message to that effect. Setting this to a lower value can improve performance.
  - **Ad Hoc Dataset Row Limit**. The maximum number of rows that an Ad Hoc view will request in a query. Be aware that JasperReports Server truncates the data displayed in the report when the limit is reached. Setting this to a lower number may improve performance, but your reports may not reflect the full data set.
  - **Ad Hoc Query Timeout**. The number of seconds the server should wait before timing out an Ad Hoc view while running its query. Setting this to a lower number may prevent exceptions when users run Ad Hoc views. Setting this to a higher number may prevent complex calculations from timing out, but may use more database connections.
- Data Policies that determine how JasperReports Server handles data loading and processing for certain kinds of Ad Hoc views. See **Ad Hoc Data Policies** in the next section.

To configure the Ad Hoc query settings:

1. Log in as system administrator (`superuser` by default).
2. Select **Manage > Server Settings** and choose **Ad Hoc Settings** in the left-panel.
3. Set the configuration values as described above.
4. Click **Change** beside each value you modified to make your changes effective. Or click **Cancel** to reset it to the previously saved value.

> As of JasperReports Server version 5.0, Ad Hoc settings made through the user interface are persistent, even when the server is restarted. For details, see **"Configuration Settings in the User Interface" on page 176**.

9.4.2   Ad Hoc Data Policies

> Data snapshots, described in **Data Snapshots** and **Enabling Data Snapshots**, apply only to reports displayed in the report viewer. This section covers data policies that apply only to views in the Ad Hoc Editor.

Data policies determine how data is cached and where certain calculations occur. All Ad Hoc work is based on a query, either from a Domain or Domain Topic, or from the JRXML of a plain Topic. Data policies determine whether the Ad Hoc engine uses the query as-is and process the data in memory, or rewrites the query so that the database processes data and returns only what Ad Hoc needs to display.

By default, the data accessed by Domain-based reports is grouped, sorted, and aggregated in the database, rather than processed in memory on the server. This way the server retrieves only the columns that appear in the report rather than the entire set of fields in the Domain. As of JasperReports Server 5.5, calculated fields are also processed by the database. For Topic queries based on JDBC (and JNDI) data sources, the default behavior is to request the entire result set and process the columns for display in memory. Note that independent check boxes control the behavior -- one for Domains and another for JDBC data sources.

When these check boxes are cleared, the server loads the entire set of fields associated with a Domain or Topic into memory, then applies the necessary calculations, grouping, sorting, and aggregation. This is also the case for Ad Hoc views that do not rely on Domains or JDBC data sources; in these cases, the server processes the data in memory.

Generally, we recommend that these settings be enabled, especially when working with large datasets. In deciding whether JasperReports Server should process the data in memory or push that processing to the database, consider these factors:

- The size and complexity of your reports. Reports with calculated fields, complex sorting, grouping, or aggregation may perform better when the server optimizes the queries so that the database performs the work.
- The amount of data in your data sources. If your data sources include a great deal of data, reports against them may perform better when the server optimizes the queries.
- The number of users editing and running Ad Hoc views. If you have a large number of users creating and running Ad Hoc views, performance may be better when the server optimizes the queries. Implementations with fewer users may perform better when the options are disabled.
- The performance characteristics of your data source. If the database or other data source is tuned for maximum performance, Ad Hoc views may perform better when the server optimizes the queries.
- If your data source is hosted by MySQL, we recommend that you keep the default (unchecked) for the **Optimize Queries for JDBC-based Reports** setting. MySQL has poor performance with the nested queries that this setting would generate.
- The amount of memory allocated to JasperReports Server's Java Virtual Machine (JVM). If the JVM of the application server hosting JasperReports Server is allocated plenty of memory, Ad Hoc views may perform better when JasperReports Server optimizes the queries. This is especially true if your data source tends to be slow.

To decide whether JasperReports Server should optimize queries for Ad Hoc views, we recommend disabling the settings, opening and saving some representative reports, and testing their performance. If the performance improves, leave the settings disabled and open and save the remaining reports.

The data polices you can set are:

- **Optimize Queries for JDBC-based Reports**. When checked, Ad Hoc rewrites the query to calculate, filter, group, sort, and aggregate columns when using Topics based on JDBC and JNDI data sources. Otherwise, the queries run unaltered and calculated fields, filtering, grouping, sorting, and aggregation take place in memory.
- **Optimize Queries for Domain-based Reports**. When checked, Ad Hoc rewrites the query to calculate filter, group, sort, and aggregate columns when using Domains or Domain Topics. Otherwise, the queries run unaltered and calculated fields, filtering, grouping, sorting, and aggregation take place in memory.

> These data policy settings do not retroactively update the existing reports created from Ad Hoc views in your repository. To change the data policy for an existing report, select the appropriate policy setting, open the corresponding view in the Ad Hoc Editor, and save the report again.

To configure the Ad data policies:

1. Log in as system administrator (`superuser` by default).
2. Select **Manage > Server Settings** and choose **Ad Hoc Settings** in the left-hand panel.
3. Select **Optimize Queries for JDBC-based Reports** to optimize and rewrite queries for JDBC-based Topics.

4. Select **Optimize Queries for Domain-based Reports** to optimize and rewrite queries for Domain-based reports.

5. Click **Change** beside each value you modified to make your changes effective. Or click **Cancel** to reset it to the previously saved value.

> As of JasperReports Server version 5.0, Ad Hoc settings made through the user interface are persistent, even when the server is restarted. For details, see **"Configuration Settings in the User Interface" on page 176**.

9.4.3   Ad Hoc Data Policies for Big Data

When handling large datasets (big data) from a Domain source in the Ad Hoc Editor, fields summarized by distinct count are computationally intensive. The server optimizes the distinct count by requesting distinct count calculations from the data source, as opposed to performing the calculations in memory. Database servers are usually optimized for these calculations, which improves the overall performance of the Ad Hoc Editor.

Ad Hoc for Big Data		
Configuration File		
.../WEB-INF/applicationContext-adhoc-dataStrategy.xml		
Property	**Bean**	**Description**
calcMethod	Distinct Count	This property's default is value="sqlUnionAll". The UnionAll is the optimized query that provides distinct count computed by the database. If this setting causes issues with your database, you can reset it to value="sqlGroupBy". If you make this change, redeploy the web app or restart the application server.

Performing distinct count aggregates in the database applies only in the following cases:
- Crosstabs based on Domains contain measures aggregated by distinct count.
- Tables based on Domains contain groups aggregated by distinct count, but no detail rows.

This setting has no effect when there is a row or column group involving a time, timestamp, or date. In this case, Ad Hoc performs the distinct count summary calculations in memory, regardless of the calcMethod setting.

9.4.4   Ad Hoc Templates and Report Generators

JasperReports Server5.5 introduced Ad Hoc templates and report generators. Ad Hoc templates are JRXML files in the repository that define the format for reports generated from Ad Hoc views. Report generators are custom beans that create custom output from Ad Hoc views.

In the user interface, users can select either the default template, browse for a different template, or select a report generator, if any are defined. For more information about the report template interface, see the JasperReports Server User Guide. This section explains how to configure these controls.

9.4.4.1 Changing the Default Ad Hoc Template

The default Ad Hoc template creates a report that resembles the Ad Hoc user interface. It's meant for online viewing and doesn't restrict the size of the contents for printing. To set a different template, modify the following property:

Default Ad Hoc Template		
Configuration File		
.../WEB-INF/applicationContext-adhoc.xml		
Property	**Bean**	**Description**
`defaultTemplateUri`	`util:map id="report Generator DefaultConf"`	Specify the repository path of your new default template. Make sure the template has permissions so it's accessible to all users. The default value is /public/templates/actual_size.510.jrxml.

9.4.4.2 Modifying the Ad Hoc Template Folders

Ad Hoc templates must be uploaded to specific folders in the repository. The default folders are /templates (in every organization) and /public/templates. To set different folders, modify the following property:

Ad Hoc Template Folders		
Configuration File		
.../WEB-INF/applicationContext-adhoc.xml		
Property	**Bean**	**Description**
`templateURIParent SQLPatternList`	`adhocEngine Service`	Add, change, or remove the values in this list to specify the folders in the repository where the server looks for Ad Hoc templates.

The default Ad Hoc template locations contain the default template. If you move or delete those folders in the repository, be sure to update the definition of the default Ad Hoc template as described in the previous section.

Also, organization templates include the adhoc/templates folder so it appears in every new organization. You should update all organization templates if you change or remove the %/templates value. For more information, see **"Default Folders for Organizations" on page 28**.

9.4.4.3 Adding Ad Hoc Report Generators

By default no custom report generators defined on the server, and none appear in the user interface.

To add a custom report generator:
1. Create a Java class that implements the com.jaspersoft.ji.adhoc.service.AdhocReportGenerator interface.
2. Compile the class and place the resulting JAR file in <js-webapp>/WEB-INF/lib.

3. Open the file .../WEB-INF/applicationContext-adhoc.xml for editing and register your class as a Spring bean as shown in the following example:

```
<bean id="myCustomReportGenerator" class="com.example.myCustomReportGenerator">
    <property name="id" value="my-custom-generator"/>
    <property name="..." value="..."/>
    ...
</bean>
```

4. In the same file, update the reportGeneratorFactory bean to include your custom generator bean:

```
<bean id="reportGeneratorFactory"
class="com.jaspersoft.ji.adhoc.service.ReportGeneratorFactoryImpl">
    <property name="reportGenerators">
        <list>
            <ref bean="myCustomReportGenerator" />
            <!--<ref bean="actualSizeReportGenerator" />-->
            <!--<ref bean="letterPortraitReportGenerator" />-->
            <!--<ref bean="letterLandscapeReportGenerator" />-->
            <!--<ref bean="a4PortraitReportGenerator" />-->
            <!--<ref bean="a4LandscapeReportGenerator" />-->
        </list>
    </property>
</bean>
```

5. Edit the .../WEB-INF/bundles/adhoc_messages.properties file to add a UI label for your custom generator. The key has the form ADH_REPORT_GENERATOR_<generator-id>. Add the same key to other language bundles if you want to support other languages.

```
ADH_REPORT_GENERATOR_my-custom-generator=Corporate Template
```
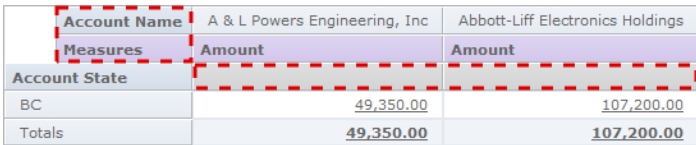
6. Restart the server or redeploy the JasperReports Server web app. The label for your custom generator appears in the list of report generators when users create and save a report from an Ad Hoc view.

9.4.5 Ad Hoc Configuration File

The following properties are among those that can be configured in the WEB-INF/applicationContext-adhoc.xml file:

Table 9-1 Configurable Properties in WEB-INF/applicationContext-adhoc.xml

Property	Description
JrxmlScriptURI	The location in the file system of the state2jrxml.js script, which generates the JRXML report based on the current Ad Hoc Editor selections. By default, this file is located in the /adhoc folder of the repository.
realmsURI and realmsURIParent SQLPatternList	The repository locations where Topics should reside. The defaults are /adhoc/topics and /public/adhoc/topics.

Property	Description
`defaultTheme`	The name of the default style for Ad Hoc views. This name must match a style defined in both a CSS and a JRXML file. The default is `default`.
`aruFolder`	The repository location where users are allowed to save their Ad Hoc views. The default is /. This allows your users to save Ad Hoc views anywhere. If you have a folder specifically for user content, specify this folder; for example, /userviews.
`tempFolderName`	The repository location where JasperReports Server saves reports created from Ad Hoc views. The default is /temp relative to root and to every organization. The server allows users with ROLE_ADMINISTRATOR or ROLE_SUPERUSER to view the temporary folders and their contents. The server manages these temporary files automatically, but files may accumulate in certain cases. As part of regular maintenance, you should periodically delete the files in these folders.
`maxSafeGroupMembers`	The maximum number of row groups or column groups a crosstab can display before the editor prompts the user for confirmation. This limit is a safeguard to avoid performance issues when grouping a field with too many values. The default is 100. Set it higher to allow more groups to appear without prompting users.
`createColumn CrosstabHeaders`	This property is located in the `actualSizeReportGenerator` bean. With the default setting of `false`, column group headers appear in the Ad Hoc view but not in the report generated from the view (this is the historical behavior). When set to `true`, column group headers appear in both the Ad Hoc view and in the generated report. When column group headers are included in a crosstab, an extra row is added below the column headers for spacing. ![Crosstab table showing Account Name, Measures, Account State columns with A & L Powers Engineering, Inc (Amount) and Abbott-Liff Electronics Holdings (Amount). BC row: 49,350.00 and 107,200.00. Totals: 49,350.00 and 107,200.00.]

The repository URI locations are relative to each and every organization in the server instance. For example, for a user in the default organization, the URI /adhoc actually refers to /organizations/organization_1/adhoc.

9.4.6 Ad Hoc Cache Management

The Ad Hoc cache applies to Ad Hoc views based on Topics or Domains, and any reports generated from those Ad Hoc views. Ad Hoc views based on OLAP connections use the OLAP cache. For a comparison of the two caches, see **"Comparison with Jaspersoft OLAP Cache" on page 192**. For instructions on setting the OLAP cache, see the *Jaspersoft OLAP User Guide*.

JasperReports Server can temporarily cache Ad Hoc query result sets for re-use. The cache is populated by the data that results from queries when creating or running Ad Hoc views. The datasets are uniquely identified by a key that references the query itself, the data source URI, and parameters used when the query was issued.

Caching reduces database loads and delivers frequently-used datasets to the user quickly. Caching applies when reports are created and when they're run. JasperReports Server version 5.0 introduces a new cache implementation called Ehcache that allows the administrator to view cache entry memory size and set limits on memory use. You can configure the Ad Hoc cache to optimize memory use and response time for your use patterns.

9.4.6.1 Setting the Cache Granularity

By default, datasets for each user are cached separately. A parameter in the cache key identifies the user. This per-user caching can result in duplicate datasets and impaired performance when different users run the same query. You can configure JasperReports Server to share cached datasets across users by editing the /WEB-INF/applicationContext-datarator.xml file.

The following code configures the `cacheKeyInterceptor` bean to ignore logged-in users' credentials when creating the cache keys:

```
<property name="ignoredParameters">
    <list>
        ...
        <value>LoggedInUser</value>
        <value>LoggedInUsername</value>
    </list>
</property>
```

Restart JasperReports Server after saving the modified file.

9.4.6.2 Configuring the Cache

Caching improves overall performance of data retrieval and sorting, but unused datasets can consume memory and cached data can become stale. To address these concerns, the cache automatically removes datasets periodically. By default, datasets are removed from the cache if they are not accessed for 30 minutes. They are also cleared after 90 minutes, regardless of how recently they were accessed.

To configure the frequency with which the cache is automatically cleared, edit the following configuration file:

Ad Hoc Cache Expiration
Configuration File
.../WEB-INF/adhoc-ehcache.xml

Ad Hoc Cache Expiration		
Property	**Default Value**	**Description**
maxBytesLocalHeap	400M	The maximum memory use of the entire cache, by default 400 MB. Use K for kilobytes, M for megabytes, and G for gigabytes. If left unbounded, the Ad Hoc cache can use up all available memory in your JVM. Set this value according to your server's available memory, the size of your datasets, and the number of cache entries you expect based on concurrent Ad Hoc use and the time settings below. We recommend setting this value to about half of the maximum heap size you configured for the JVM (-Xmx setting).
timeToIdleSeconds	1800	The number of seconds to wait after a dataset is has been accessed before removing it from the cache. The default is equivalent to 30 minutes. Use 0 (zero) for no limit.
timeToLiveSeconds	5400	The maximum time that a dataset is stored in the cache, even if it is being repeatedly accessed. Ensures that stale data is periodically replaced. The default is equivalent to 90 minutes. Use 0 (zero) for no limit.

Restart JasperReports Server after modifying these values.

9.4.6.3 Manually Clearing the Cache

Administrators can also use the server interface to view the queries whose datasets are in the cache. Administrators can see the full query but never the contents of the dataset. The Ad Hoc cache page also displays performance data about each query. This information can be helpful when trying to resolve performance issues. The interface displays several values for every query:

- Age (min:sec) – Time since the dataset was first stored in the cache.
- Query (msec) – Time in milliseconds from when query was sent to the data source (database) until the first row was received.
- Fetch (msec) – Time in milliseconds from when first row was received from the data source (database) until the last row was received.
- Memory used (MB) – Size in megabytes of the resulting dataset being stored in the cache entry.

The Ad Hoc cache page also allows administrators to manually remove datasets if necessary. Removing a dataset from the cache forces the server to get fresh data the next time a user creates or runs an Ad Hoc view with that query.

To view queries and manually clear the Ad Hoc cache:

1. In JasperReports Server, log in as system administrator (superuser by default).
2. Click **Manage > Server Settings** and choose **Ad Hoc Cache** in the left panel.

   The Ad Hoc Cache page appears, displaying all the datasets in the cache, sorted by age.

Figure 9-4 **Ad Hoc Dataset Caching Administration Page**

Each dataset is listed by its corresponding query and data source. Recall that Ad Hoc Topics have user-defined queries, so they tend to be short, whereas the query for Domains are generated from the design of the Domain and user selections in the Data Chooser dialog. The Ad Hoc Cache page displays only the first few lines of a query, as well as the data source.

3. To remove all datasets, click **Clear All** at the bottom of the Ad Hoc Cache page. This also clears the Teiid cache used by virtual data sources, including a virtual data source that wraps a data source for big data.

4. To remove a specific dataset from the cache, and click **Clear** beside the corresponding query.

5. To find a specific cache entry, you can change the sorting in the upper-left by clicking **Age**, **Last Used Time**, **Memory Used**, or **Data Source URI**.

6. To view the details of a specific query, including the full query string, click the query itself in the Query & Source column. The Detail page appears, displaying additional information for the selected query, such as the number of rows in the cached dataset.
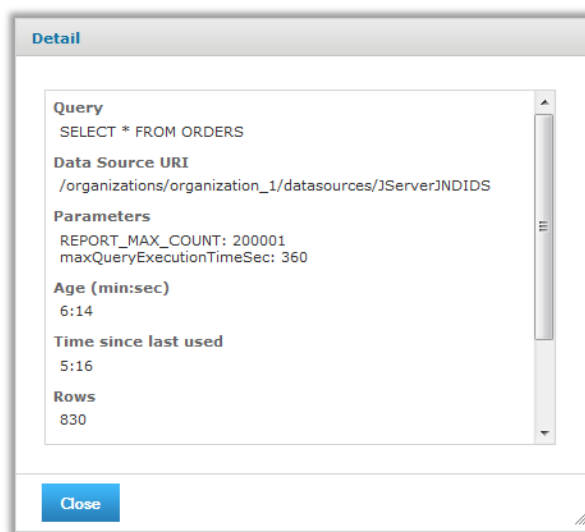
Figure 9-5 **Typical Dataset in Ad Hoc Cache**

9.4.6.4 Comparison with Jaspersoft OLAP Cache

The following table contrasts the key features of the Ad Hoc cache in JasperReports Server and Jaspersoft OLAP.

Table 9-2 Ad Hoc Caching in JasperReports Server and Jaspersoft OLAP

Cache Feature	JasperReports Server	Jaspersoft OLAP
Structure of cache	Result caches are held at the query level: query text and language, plus data source URI and query parameters.	Result caches are held at the analysis connection level: schema plus database connection.
Sharing	Not by default, but can be enabled as described in **Setting the Cache Granularity**, above.	There is only one cache; it is shared across all queries and users.
Security	Applied to cache control so that users are not allowed to see privileged data.	
Populating	Queries populate the cache. You can also schedule reports to pre-populate the cache during off-hours.	
Size	Limited by available JVM memory (heap). Not configurable. Running out of memory is unusual. It can only happen if a single query returns too many elements for available memory. The report fails with an out-of-memory error.	

Cache Feature	JasperReports Server	Jaspersoft OLAP
Automatic time-based cache policy	Configurable, as described in **Configuring the Cache** above.	In low-memory situations, cached items are removed automatically by JVM garbage collection; the least-recently-used items are cleared first. There is no way to remove data based on how long it has been in memory.
Clearing selected datasets manually	Configurable, as described in section **Manually Clearing the Cache** above.	Cache regions can be defined and cleared programmatically with APIs.
Clearing all datasets manually	Configurable, as described in **Manually Clearing the Cache** above.	In JasperReports Server, select **Manage > Server Settings**, then select **OLAP Settings** and click **Flush OLAP Cache**. For additional methods, see the *Jaspersoft OLAP Ultimate Guide*.

9.4.6.5 Disabling the Ad Hoc Cache

> Disabling either cache is a server-wide setting that applies to all data sources or connections used in any Ad Hoc view. Make sure that other views aren't negatively affected by this change.

There are two reasons to consider disabling the Ad Hoc cache:

- You have a high-performance database that returns results so fast that additional caching in the server does not improve response times. In this case, the slight overhead of the cache may actually impact performance.
- Your database manages real-time data, and you create Ad Hoc views that present up-to-the-minute information from this data source. In this case, you don't want to retrieve old data out of the cache.

To disable the Ad Hoc cache for Topics and Domains, set the value of `maxBytesLocalHeap` to 1 (1 byte). For instructions, see **"Configuring the Cache" on page 189**. This effectively turns off the cache so that every query is retrieved directly from the data source.

To disable the OLAP cache for OLAP connections used in the Ad Hoc Editor, check the `mondrian.rolap.star.disable-Caching` setting on the **Manage > Server Settings > OLAP Settings** page. For more information, see the *Jaspersoft OLAP User Guide*.

In addition, if you have modified any of these three properties in applicationContext-adhoc.xml, set them back to false:

```
<property name="applyQueryFilterInMemory" value="false"/>
<property name="applySecurityFilterInMemory" value="false"/>
<property name="applyDynamicFilterInMemory" value="false"/>
```

9.4.7   Ad Hoc OLAP Filter Configuration

When using filters in Ad Hoc OLAP, the server queries the database to display a list of values to select from. To avoid performance issues, the number of items in a filter is limited. By default, the limit is 250 possible values.

If your filters reach this limit and your list of values is truncated, you should first consider using a different filter operation. For example, instead of "city is one of <list>," use "City starts with <letter>." If you still need to change this limit, modify the following property:

Ad Hoc OLAP Filter Limit		
Configuration File		
.../WEB-INF/applicationContext-adhoc-dataStrategy.xml		
Property	**Bean**	**Description**
maxFilterValues	mdxDataStrategy	Set the value to the maximum number of filter values you expect. Setting this value higher than the default of 250 may cause performance issues.

9.5 Enabling Data Snapshots

The data snapshot feature was introduced in JasperReports Server release 4.7 to store report data in the server. Data snapshots create a significant change in the user experience:

- Without data snapshots – Whenever users run a report, the server queries the data source and displays the latest data. When the same report is run over and over, the data source is often returning the same data every time. This is the behavior of all releases before 4.7, and the default behavior of release 4.7.

- With data snapshots – The first time a report runs, it queries the data source and stores a copy of the data with the report in the repository. Users who view the report later see the data from the saved snapshot, not from querying the data source. Reports accessed through web service APIs are also based on the saved snapshot. For large reports or frequently viewed reports, the persisted snapshot provides a significant performance gain and reduces load on your data sources. Every user who has access to the report will see the data from the same snapshot. For users who require it, the report viewer provides a button to manually refresh the data snapshot anytime. In addition, when the scheduler runs a job on a report it always updates the snapshot. Data snapshots are implemented in JasperReports Server 4.7 and later, but must be enabled manually.

We encourage enabling data snapshots with the following recommendations:

- If you have a new installation of JasperReports Server, enable snapshots to get the full server functionality. In the future, persistent data snapshots may be enabled by default.

- If you are upgrading from a release earlier than 4.7, first proceed with the upgrade procedure and verify the outcome, as instructed in the *JasperReports Server Installation Guide*. Then, before enabling data snapshots, notify your users about this new functionality.

- Data snapshots are stored in the server's repository, which must be sized accordingly. If you have a large number of reports, or very large reports, consider the performance of your repository database before enabling snapshots. If your users rely on data that changes frequently or if they expect to see real-time data when opening a report, do not enable snapshots. Alternatively, you can enable snapshots selectively as described below.

9.5.1 Global Data Snapshot Configuration

The server-level settings determine whether the snapshot feature is available on the server:

Data Snapshots Server-Level Configuration		
Configuration File		
.../WEB-INF/applicationContext-data-snapshots.xml		
Property	**Bean**	**Description**
`snapshot Persistence Enabled`	`dataSnapshot Service`	When set to `true`, it allows the JasperReports Server report viewer to save data snapshots in the repository and open them the next time the report is run. By default, this is set to `false`.

There is also a property named `snapshotRecordingEnabled` that caches a snapshot in the report viewer memory when sorting and filtering columns interactively. This allows the report viewer to refresh the display without querying the database every time. Regardless of persistence, `snapshotRecordingEnabled` improves report viewing performance and decreases database load, so it should remain set to `true`.

9.5.2   Report-level Data Snapshot Configuration

You can disable snapshots on a specific report by setting the following property in the report's JRXML:

```
net.sf.jasperreports.data.cache.persistable=false
```

This report-level property depends on the snapshot mechanism in JasperReports Server. This property has no effect in other report viewers without such a mechanism, like the viewer integrated in Jaspersoft Studio.

There are two ways to control snapshots at the report level. In the case above:
• Data snapshots are enabled on the server, so most reports use them.
• Reports that don't benefit from data snapshots can explicitly disable snapshots in their own JRXML.

As with all report-level properties, you can set server-wide default values, as described in **"Configuring JasperReports Library" on page 212**:

Data Snapshots Default Report-Level Configuration	
Configuration File	
.../WEB-INF/classes/jasperreports.properties	
Property	**Description**
`net.sf.jasperreports.data.cache.` `persistable=false`	When set to `false`, the server-wide default for reports is to not use data snapshots, however, they are still available if a report overrides this value to true in its own JRXML.

Because the report-level property takes precedence over the server-level property, this enables a second way to control snapshots:

- Data snapshots are enabled on the server.
- But the server-wide default is set to false, so most reports don't use them.
- Reports that benefit from data snapshots can explicitly enable snapshots in their own JRXML with:
  `net.sf.jasperreports.data.cache.persistable=true`

9.5.3   Data Snapshots in the Scheduler

Scheduled jobs always run the report by accessing the data source, so the output has up-to-the-minute data. When data snapshots are enabled on reports, the job always updates the data snapshot with this new data after it runs. This way, when you schedule a report, it also refreshes the data snapshot periodically -- hourly, daily, weekly, or whatever suits your data requirements.

When data snapshots are enabled on the server, the scheduler interface has an extra option to output the data snapshot. This option, as shown in the following figure, generates a copy of the report with the new data snapshot. This copy is stored in the repository as a JasperReport, identical to the report being run. Over time, this will create an archive of your report data.

If you clear the Data Snapshot Output Format option, no copy of the report is saved with the new data snapshot, but the data snapshot on the original report is still updated when the job runs. Also, you must select at least one other output format to schedule the report.
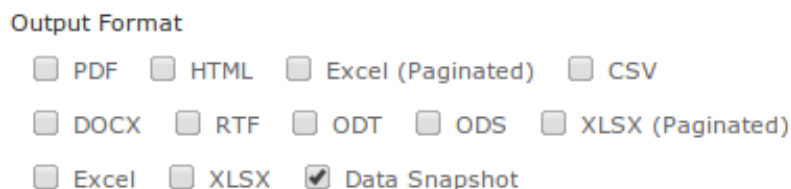

Figure 9-6 **The Data Snapshot Output Option in the Scheduler**

Finally, when data snapshots are enabled, you can also update them through REST web services calls. When specifying the report to run with the rest_v2/reportExecutions service, you can add arguments to explicitly update or not update the associated data snapshot. For more information, see the *JasperReports Server Web Services Guide*.

9.6   Enabling Data Staging

Data staging is a new feature in JasperReports Server 6.0 that can optimize the performance of certain Ad Hoc views and reports. With data staging, the entire dataset for a Domain Topic is indefinitely cached in the server's Ad Hoc cache. When the entire dataset is pre-loaded in the server, Ad Hoc views and reports from that Domain Topic run faster because they fetch data from the local cache, not by querying the production database. To keep data fresh, you can specify a reloading interval so that the server periodically accesses the database in the background to reload the entire dataset.

By default, data staging is not enabled on the server. You must enable data staging first, then each creator of a Domain Topic can choose to configure it for their Domain Topic.

When a Domain Topic uses data staging, it no longer uses any Domain-based optimizations (data policies) configured in the server because those apply only to database queries. Datasets that are staged are also exempt from the Ad Hoc cache time limits such as idle time and time-to-live. Data policies and Ad Hoc cache settings still apply to Domain Topics that do not use data staging.

When deciding to enable data staging, you should consider the nature of your data, its size, your database performance, and your access patterns to determine whether staging will help your views and reports run faster. You should also perform realistic usage tests to determine the amount of memory and size of data that give you the best performance. You can see staged datasets on the Ad Hoc cache page (**Manage > Server Settings > Ad Hoc Cache**), which displays their size and fetch time.

To get the full benefit of data staging, you must ensure your server has enough physical memory allocated to the Java virtual machine so that the cached dataset stays in memory. The cache for data staging will use disk storage when memory is full, but response times will suffer. Therefore, you must apply data staging to carefully selected Domain Topics so that the resulting sum of all staged datasets does not overwhelm your cache or your Java virtual machine (JVM) memory limits.

The cache for data staging also uses disk storage so that staged datasets are persistent during server restarts. Another advantage of staged datasets is that they allow fast reporting even when your original database has latency or downtime issues.

9.6.1   Global Data Staging Configuration

The following setting determines whether the data staging feature is available on the server:

Data Staging Server-Level Configuration		
Configuration File		
.../WEB-INF/applicationContext-adhoc-dataStrategy.xml		
Property	**Bean**	**Description**
enabled	stagingService	When set to `true`, it allows Domain Topic creators to configure data staging on Domain Topics. When a Domain Topic uses data staging, its entire data set is stored permanently in the Ad Hoc cache. By default, this is set to `false`.
Configuration File		
.../WEB-INF/adhoc-ehcache.xml		

Data Staging Server-Level Configuration		
Property	**Bean**	**Description**
maxBytesLocalHeap	ehcache	The total size of the heap shared between the Ad Hoc cache and the staging cache. The default is 400 MB. Use K for kilobytes, M for megabytes, and G for gigabytes. If you expect to have very large datasets being staged (hundreds of MB), you should increase this number to accommodate them. This value should still be about half of the maximum heap size you configured for the JVM (-Xmx setting).
maxBytesLocalDisk	ehcache	The maximum size of the staging cache stored on disk. The default is 2 GB. Use K for kilobytes, M for megabytes, and G for gigabytes. If you expect to have extremely large data sets being staged, you should increase this value as well.

9.6.2   Topic-level Data Staging Configuration

To use data staging, you must first create a Domain Topic by creating an Ad Hoc view and selecting a Domain as its source. After selecting a Domain, use the Data Chooser dialog to specify the sets and fields of the Domain in your Topic, along with any pre-filters and display names. The items chosen from the Domain, the filters in the Domain, along with any pre-filters in the Domain Topic all determine the dataset that data staging will store in the cache.

When data staging is enabled on the server as shown above, Domain Topics offer two new settings in the Save as Topic tab of the Data Chooser dialog.

☑ Enable staging
Time to keep query result in cache (in minutes):

    1440

Figure 9-7 **Data Staging Options in Domain Topics**

The two settings control data staging behavior:
- Enable staging – Turns on data staging for this Domain Topic. All Ad Hoc editor actions, Ad Hoc views, and Ad Hoc reports based on this Domain Topic will use the staged data in the cache.
- Time to keep query result in cache (in minutes) – This value determines the refresh interval in minutes of the staged data. At the end of every interval, the server refreshes the staged data by querying the data source for the entire dataset again. The new staged data replaces the old staged data in the cache so that the staging lasts indefinitely, but the dataset is renewed. Staged data is refreshed asynchronously every interval by the server, independently of any access to the staged data by an Ad Hoc view or report.

To ensure data security, data staging is disabled when your Domain is based on a data source that is defined with attributes. Server-level attributes are the exception and are allowed. Therefore, when a data source is defined with organization attributes or user attributes, the data staging settings are not availabe in the UI. For more information, see **4.1, "Attributes in Data Source Definitions," on page 66**.

9.6.3   Data Staging Dependencies

Data staging stores a dataset based on the items and filters selected in the Domain and Domain Topic. Views and reports based on the Domain Topic rely on this data being staged, that is, available in the cache.

When data staging has been turned on and views or reports have been created based on this Domain Topic, you can edit the Domain Topic as follows:

• You can add items from the Domain to the Domain Topic. When you save the Domain Topic, the server immediately retrieves the new dataset with the new items and stores it in the cache.

• You cannot remove items from the Domain Topic. If you try to remove items, the Data Chooser dialog will give an error. Look in the log file to find details about these dependencies.

• You cannot turn off data staging for this Domain Topic unless all Ad Hoc views and reports based on it have been deleted.

• You can change the refresh interval of the staged data. When you save the Domain Topic, the server immediately retrieves the dataset again and applies the new refresh interval in the future.

9.7   Configuring System Logs

This section describes the settings that control the information JasperReports Server writes to its logs.

The log files contain important information about how the server is running. JasperReports Server uses the Apache log4j package to generate log files. Jaspersoft uses the slf4j facade to wrap log4j.

• The default log file is WEB-INF\logs\jasperserver.log.

• The default log configuration file is WEB-INF\log4j.properties.

9.7.1   Managing Log Settings

To set the current logging levels:

1. Log in as system administrator (superuser by default).

2. Select **Manage > Server Settings** and choose **Log Settings** in the left-panel.

3. In the Log Settings panel, use the drop-down selectors to change the log level for each class being logged.

Figure 9-8 **System Log Settings**

The page lists some of the currently-enabled loggers (ones that typically need their logging levels adjusted from time to time) with their logging level. Any change to the logging levels takes effect immediately, without restarting JasperReports Server.

> These logging levels override the levels in the log4j.properties file. As of JasperReports Server version 5.0, these settings persist even when the server is restarted. Therefore the values on the Log Settings page that are in effect on the server may differ from settings in the log4j.properties file. For details, see **"Configuration Settings in the User Interface" on page 176**.

The four logging levels indicate the type of event recorded by a logger:

Setting	Level of Information
ERROR	Writes minimal information to the log describing serious program faults.
WARN	Writes error and warning messages to the log. Warning messages contain cautionary information to help you to decide whether the logged events require your attention.
INFO	Writes error, warning, and informational messages to the log describing significant events, such as those that affect application performance.
DEBUG	Writes error, warning, informational, and additional messages to the log. Debug messages are very detailed and often voluminous. Use this setting only to diagnose a problem. DEBUG can impact system performance and should not be used in production environments. If several loggers are set to DEBUG, the server may generate huge logs, and performance can suffer.

JasperReports Server's default root logger setting is ERROR, as configured in log4j.properties. A logger without an assigned value inherits the setting of its parent in log4j.properties.

The following table lists each logger name as it appears on the Log Settings page, the identifier used to find it in the log file, and a description of the logger.

Logger Name	Identifier in Log	Description
SQL query executer	JRJdbcQueryExe cuter	Logs SQL text and parameter values for queries that are run by the SQL query executer.
Input control value queries	valueQueryLog	Logs SQL text and parameter values for queries associated with input controls.
Cascading input control parameter resolution	FilterCore	Logs activity associated with cascading input controls. Query-driven input controls can cascade when a query has a parameter whose value comes from another input control. When the parameter value is changed, the query is automatically rerun, possibly changing the list of values for its input control.
Cascading input control query result caching	TokenControlLogic	Logs use of the cache for results of cascading input control queries.
Hibernate SQL	SQL	Logs SQL run by the Hibernate layer to access the JasperReports Server repository database. This logger generates a large volume of logging that could affect performance.
Ad Hoc data policy logging	CommonDomain-DataStrategy SubFilterInputControl-Generator Others	Logs various activities of the Ad Hoc data policy implementations, which use SQL queries or in-memory operations to get datasets for Ad Hoc views.
SQL generated for Domain queries	JdbcBaseDataSet	Logs SQL queries generated from queries using a Domain.
Connection handling for Domains	DataSourceResolverImpl	Logs use of JDBC connections used by Domains to run SQL queries.
Expression to JSON converter	ExpressionJSON Converter	Logs information about the conversion between DomEL and JSON, which is used by Ad Hoc filters.
Domain-based security tests	SemanticLayerSecurityResolver Impl	Logs activity related to Domain column- and row-level security.

Logger Name	Identifier in Log	Description
Cascading input control resolution for Domains	DomainFilterResolver	Logs the same activity as the FilterCore logger (Cascading input control parameter resolution) above, but adds information specific to Domain queries.
Ad Hoc cache activity	CachedData	Logs information about the life cycle of datasets that are cached in memory when Ad Hoc views are accessed.
Timing for SQL queries run for reports	JsControlledJdbcQueryExecuter	Logs the time it takes a query run by the SQL query executer to return data to a report.
Ad Hoc WorkingDataSet	WorkingDataSet	Logs activity for the WorkingDataSet, used by the Ad Hoc Editor to perform in-memory dataset transformations of query results.
General controller	AdhocAjaxController	Logs activity of the Ad Hoc Editor.
Crosstab controller	AdhocCrosstabAjaxController	Logs additional activity of the Ad Hoc Editor specific to crosstab reports.
Groovy code generation for memory datasets	GroovyGenerator	Logs Groovy classes generated from DomEL expressions used by the Ad Hoc Editor for filters and calculated fields.
Ad Hoc AJAX requests	adhocAjaxRequests	Logs information about AJAX requests made by Ad Hoc Editor and dashboard designer, including report parameters and response times. Enable this setting when you want to understand the Ad Hoc Editor and dashboard designer or if you've encountered an error or slow response times.
Ad Hoc cache activity	com.jaspersoft.commons. datarator.CachedData	Tracks the life cycle of datasets managed by the Ad Hoc cache as they transition between states. This log output includes information from the Ad Hoc cache in a format that lends itself to troubleshooting. Use this setting to understand how query response times contribute to the performance and responsiveness of the Ad Hoc Editor. Because it doesn't log the queries themselves, use it in conjunction with the SQL Query Executer log setting.

You can add other loggers to the Log Settings page if you know their classnames.

To add a logger to the page from the web interface:

1. Log in as system administrator (`superuser` by default).

2. Select **Manage > Server Settings** and choose **Log Settings** in the left panel.

3. Scroll to the bottom of the page.

4. Enter the logger's classname in the text field. See the other properties on the page for guidance, for example:

   ```
   com.jaspersoft.ji.adhoc.action.AdhocCrosstabAjaxController
   ```

5. Use the drop-down to set the logging level.

The logger setting is persistent even when the server is restarted. However, the logger setting may not appear on the Log Settings page again. For information about adding loggers to this page permanently, see **"Adding a Logger to the Log Settings Page" on page 204**.

9.7.2   Log Configuration Files

You can edit the log configuration file to set loggers, logging levels, and log output, but you must restart the server for your changes to take effect.

If you've made modifications in the Log Settings UI, those settings are persistent in the repository, and take precedence over the configuration files. However, these changes are *not* written to the configuration files. Each setting is independent, so a value that's not modified in the Log Settings UI is taken from the corresponding file. For more information, see **"Configuration Settings in the User Interface" on page 176**.

Logger names are defined in the Java source. Loggers can have any name, but the Jaspersoft convention is to give them their full class names. In the log4j properties file, the classname must be preceded by `log4j.logger`. For example, the classname `org.acegisecurity.intercept` is represented in the log4j.properties file `as log4j.logger.org.acegisecurity.intercept`. If you want to add a new logger, find its classname in the source.

The file you edit depends on whether you're configuring server logging or logging during import and export.

Functionality to Log	File Location
Import/Export	<js-install>\buildomatic\conf_source\iePro
JasperReports Server	WEB-INF\log4j.properties in the JasperReports Server installation

If the logger is defined in the configuration file but commented out, simply remove the comment character (#) to add the logger. Otherwise, add the logger's classname and set the logging level.

The form of a logger definition should be:

```
log4j.logger.<logger-classname> = <log-level>, <output-type>
```

where:

- `<logger-classname>` is the name of the class you want to monitor.
- `<log-level>` is ERROR, WARN, INFO, or DEBUG
- `<output-type>` is a standard output type, such as stdout. For example:
  ```
  log4j.logger.org.springframework.webflow=DEBUG, stdout, fileout
  ```

Restart the server for your changes to take effect.

9.7.3 Adding a Logger to the Log Settings Page

If you know of a log4j logger that JasperReports Server uses, you can add it to the Log Settings page available to the superuser. To add a logger, edit a configuration file.

> Because editing text files can be error-prone, we recommend that you add loggers from the web interface by entering them into the text field on the Log Settings page. Edit the configuration file only if you need to permanently add the logger.

To edit the list of loggers to be displayed on the page:

1. Edit the logger_descriptions_pro.properties file found under WEB-INF/bundles in your JasperReports Server installation.
2. Add a new line and specify the logger's classname and a brief description of it.

   Entries should be in the form `<logger-classname> = <description>`.

   See the other properties in the file for guidance, for example:

   ```
   com.jaspersoft.ji.adhoc.action.AdhocCrosstabAjaxController = Crosstab controller
   ```
3. Restart the server for your changes to take effect.

> The logger_descriptions_pro.properties file controls the labels for the English locale. You can specify labels for other locales by editing the logger description property files for those locales. For example, to add the label in French, add an entry to the logger_descriptions_pro_fr.properties file. For more information on supporting other languages, refer to **"Localization" on page 263**.

9.8 Configuring Amazon Web Services

The following settings control how JasperReports Server interacts with your Amazon environment:
- The AWS settings page enables you to change Security Group settings without restarting the server.
- The AWS configuration file allows you to change the JDBC driver used with AWS data sources.

For more information about AWS data sources, see **"AWS Data Sources" on page 75**.

To change AWS Security Group settings:

1. Log in as system administrator (superuser by default).
2. Click **Manage > Server Settings**. The **Log Settings** page appears.
3. Click **AWS Settings** in the left-menu. The AWS Settings panel appears:

Figure 9-9 AWS Settings Page

We set up one AWS DB Security Group (using IP address) in each RDS region, per JasperReports Server instance. The security group allows connections from the specific JasperReports Server instance to the specified AWS database instance.

4. Modify the following settings and click **Change** after each modification. Changes are effective immediately on the server:

- **Enable AWS Security Group Changes**: This checkbox is generally left checked. When checked the JasperReports Server will use the instance credentials it assumes from the IAM role to grant itself access to RDS and Redshift data services. For example, you stop your EC2 instance with JasperReports Server on Friday. You restart it on Monday, and the instance gets a new IP address. JasperReports Server then re-grants itself access to RDS. If you want to manage the security groups manually, un-check this box.

- **Security Group Name**: When JasperReports Server creates security groups to support AWS data sources on this instance, it will use this name as the basis of the security group name. When the JasperReports Server instance is running on EC2, the EC2 instance ID will be appended. When running outside of EC2, you must make sure that name is unique among JasperReports Server instances (*i.e.* each instance should have its own name), so the IP addresses are properly granted access to the appropriate database instances.

- **Security Group Description**: This text will be used for the description field next to security group or groups in the AWS console.

- **Security Group Ingress Public IP:** Most users on EC2 should leave the this field empty. JasperReports Server determines the IP address automatically. If you are running JasperReports Server outside of EC2, you must determine your IP address manually and enter it in this field. It is also possible with complex EC2 topology involving Virtual Private Clouds (VPCs) that you need to provide your IP address manually.

- **Suppress EC2 Credentials Warning**: If your JasperReports Server instance was created with no IAM role, when you go to the data source wizard to add a data source with EC2 credentials there will be a warning message saying there is no proper role set. Checking this box suppresses the warning and disables the option..

To change the JDBC driver used with AWS data sources:

1. Open the file .../WEB-INF/applicationContext-webapp.xml for editing.
2. Locate the `jdbcConnectionMap` bean and the key of your AWS database type within it. Modify this key to specify a different JDBC driver. For example, the default driver for MySQL databases is set to the MariaDB driver:

```
<entry key="mysql">
<util:map>
    ...
    <entry key="jdbcUrl" value="jdbc:mysql://$[dbHost]:$[dbPort]/$[dbName]"/>
    <entry key="jdbcDriverClass" value="org.mariadb.jdbc.Driver"/>
    ...
</util:map>
```

3. Save the file and restart JasperReports Server.

9.9   Configuring Domains

Advanced uses of Domains may consider these configurations:

- **Disabling the Domain Validation Check**
- **Optimizing Snowflake Schema Joins**
- **Configuring Domain Dependency Behavior**

When you use Domains with certain database constructs, you may need to configure JasperReports Server:

- **Enabling Oracle Synonyms**
- **Enabling CLOB Fields**
- **Enabling Proprietary Types**
- **Extending JDBC Type Mapping**
- **Accessing Materialized Views**

9.9.1   Disabling the Domain Validation Check

By default, JasperReports Server validates a Domain against its data source to ensure that the Domain design maps properly to the underlying tables. This validation occurs when a Domain design file is uploaded to the server. If your data source is very large and complex, this validation can take time. If the validation takes too long, you can disable it. In this case, JasperReports Server assumes the Domain design is valid, and simply uploads it without the check. To disable the validation edit the following configuration file:

Configuring Domain Validation Check
Configuration File
.../WEB-INF/applicationContext-semanticLayer.xml

Configuring Domain Validation Check		
Property	**Bean**	**Description**
`skipDomainDatabase Validation`	`slConfig`	Set this property to `TRUE` to disable the validation check.

> If the tables and fields referenced in the Domain design don't exist in the data source when `skipDomainDatabaseValidation` is set to TRUE, the Domain wizard won't detect the problem, but the Choose Data wizard returns errors when your end users work uses the Domain.

9.9.2   Optimizing Snowflake Schema Joins

When creating a Domain on top of a snowflake schema, the default joins generated when using the Domain in the Ad Hoc editor may take a long time and include dimensions not used in the report. For example, a schema with over a hundred dimension tables mostly connected to a subset of 5-10 fact tables may cause such behavior. The following setting can be enabled to optimize the joins generated for such a snowflake database schema. The default setting has better performance in the more common cases with fewer tables.

Configuring Domain Join Optimization		
Configuration File		
.../WEB-INF/applicationContext-semanticLayer.xml		
Property	**Bean**	**Description**
`specialOptimizationOn`	`graph Operations`	The default setting of `false` handles typical cases of Domains based on 10-100 tables. For snowflake schemas that typically have 100 or more tables, or for database topologies that cause slow join performance in Ad Hoc views, set this property to `true` to optimize the joins in the Ad Hoc editor.

9.9.3   Configuring Domain Dependency Behavior

When modifying a Domain that's already in use by Ad Hoc views, JasperReports Server checks to see if any of these dependent views are affected. For example, if an Ad Hoc view uses a field, it should not be deleted from the Domain. However, if that field is not in use by any dependent view, it can be removed from the Domain, and from the list of fields in every view.

For dependency purposes, a field is "in use" by an Ad Hoc view if that field appears in the display manager (in any row, column, or group), if it's used in any filter, or if it's used in any calculated field formula, whether the calculated field is in use or not.

When modifying fields in a Domain, JasperReports Server always checks all views that depend on the Domain and always notifies the user of any fields in use. There are two settings to configure the Domain dependency behavior after the check:

- `defaultDomainDependentsBlockAndUpdate` - This setting turns on or off the behavior that ensures Ad Hoc views remain consistent with their Domains. When on, it ensures consistency by either blocking Domain changes that affect fields in use by any dependencies, or by updating the dependencies when there are no fields in use. When updating a dependency, it removes the field from the list of available fields (left-panel in the Ad Hoc editor). When off, it allows changes to Domains that will cause errors in any dependent view. If a field is in use by a view, and this setting is off and allows the Domain to delete that field, the view will cause an exception when opened. If a field is not in use by a view, and this setting is off, the field is not removed from the list of available fields. If a deleted field appears in the list of available fields, the view can still be opened, but any action on that field will cause an error.
You should turn this setting off only when a field is deleted from the data source, and the Domain can't be modified because the field is being used by some dependencies. You then need to manually edit the dependencies so they don't cause errors.

- `defaultAddToDomainDependents` - This setting determines whether a field being added to a Domain is automatically added to each dependent Ad Hoc view. When on, any new field added to a Domain is added to the list of available fields in every dependent view. When this setting or the previous setting is off, dependent views are not updated with new Domain fields.

Turning Domain Dependency Checks On or Off		
Configuration File		
.../WEB-INF/applicationContext.xml		
Property	**Bean**	**Description**
`defaultDomainDependents BlockAndUpdate`	`configuration Bean`	The default setting of `true` means that Domain modification will be blocked if any Ad Hoc view uses the modified fields. If no Ad Hoc view uses the modified fields, the Domain modification proceeds, and all Ad Hoc views are updated with the change. When set to `false`, Domain modifications are not blocked when the field is in use, and dependent views are not updated when fields are not in use. Any Ad Hoc view may become unusable if it references a field that's deleted from a Domain while this setting is `false`.

Configuring Domain Dependency Additions		
Configuration File		
.../WEB-INF/applicationContext.xml		

Configuring Domain Dependency Additions		
Property	**Bean**	**Description**
`default AddToDomainDependents`	`configuration Bean`	When this setting is `true`(the default), AND `defaultDomainDependents BlockAndUpdate` is set to `true`, any field being added to a Domain is added to each dependent Ad Hoc view. It might not be possible to add a field to an Ad Hoc view if the field being added does not belong to the data island being used in the Ad Hoc view. When this setting is set to `false` OR `defaultDomainDependents BlockAndUpdate` is `false`, dependent Ad Hoc views are not updated with fields that are added to a Domain. The new fields will not be available when a user opens a dependent Ad Hoc view.

9.9.4   Enabling Oracle Synonyms

By default, Domains can't access synonyms in an Oracle database. Set the following property to enable them. If you access your Oracle database through JNDI, you also need to configure the JNDI connection.

Be aware that the Oracle metadata service is significantly slower when synonyms are in scope.

Enabling Oracle Synonyms		
Configuration File		
.../WEB-INF/applicationContext-semanticLayer.xml		
Property	**Bean**	**Description**
`includeSynonyms ForOracle`	`jdbcMeta Configuration`	Set the value to true: `<value>true</value>`
Configuration File		
...\META-INF\context.xml		
Property	**Bean**	**Description**
`accessToUnderlying ConnectionAllowed`	`<Resourcename= "jdbc/oracle"...`	If you use JNDI, add the following property: `accessToUnderlying ConnectionAllowed="true"`

9.9.5   Enabling CLOB Fields

Support for CLOB (Character Large Object) fields is dependent on your database and must be enabled manually. If you want to access CLOB fields in JasperReports Server, set the following options according to your database.

The Oracle JDBC driver implementation uses the `CLOB` JDBC type for CLOB fields.

CLOB Support for Oracle		
Configuration File		
.../WEB-INF/applicationContext-semanticLayer.xml		
Property	**Bean**	**Description**

`jdbc2JavaType Mapping`	`jdbcMeta Configuration`	This property contains a map of database field types to Java types. Find the line for CLOB that is commented out:
		
		`<!--entry key="CLOB" value=""/-->`
		
		Modify it as follows:
		
		`<entry key="CLOB" value="java.lang.String"/>`

The MySQL JDBC driver implementation uses either the `CLOB` JDBC type, the `LONGVARBINARY` JDBC type, or both to represent CLOB fields, depending on their length.

CLOB Support for MySQL		
Configuration File		
.../WEB-INF/applicationContext-semanticLayer.xml		
Property	**Bean**	**Description**

`jdbc2JavaType Mapping`	`jdbcMeta Configuration`	This property contains a map of database field types to Java types. Find the following lines:
		
		`<!--entry key="CLOB" value=""/-->`
		`<!--entry key="LONGVARBINARY" value=""/-->`
		
		And modify them as follows:
		
		`<entry key="CLOB" value="java.lang.String"/>`
		`<entry key="LONGVARBINARY" value="java.lang.String"/>`

9.9.6   Enabling Proprietary Types

JasperReports Server provides a JDBC-to-Java type mapping for all standard JDBC column types for use in Domains. However, some databases have proprietary types, such as NVARCHAR2 in Oracle. You can map these types with a special configuration.

As a prerequisite, the proprietary type must be logically equivalent to one of following Java classes:

```
java.lang.Boolean        java.lang.Float        java.lang.String        java.sql.Timestamp

java.lang.Byte           java.lang.Integer      java.math.BigDecimal    java.util.Date

java.lang.Character      java.lang.Long         java.sql.Date

java.lang.Double         java.lang.Short        java.sql.Time
```

There are two ways to create a mapping for a proprietary type, as shown in the following table:

- Modify the generic mapping for NUMERIC types. By default, any numeric type that doesn't match one of the other types is mapped to `BigDecimal`.
- Create a secondary mapping under the special OTHER key, where the secondary key can be your custom type name.

Proprietary Database Type Mapping		
Configuration File		
.../WEB-INF/applicationContext-semanticLayer.xml		
Properties	**Bean**	**Description**

`jdbc2Java`
`TypeMapping`
	`jdbcMeta`
	`Configuration`
		• To modify the generic mapping, edit this line:
		`<entry key="NUMERIC" value="java.math.BigDecimal"/>`
		
		• Add any secondary key to the OTHER key, following this example:
		`<entry key="OTHER">`
		`    <map>`
		`        <entry key="NVARCHAR2" value="java.lang.String"/>`
		`    </map>`
		`</entry>`
		
		• Java 1.6 supports java.sql.Types.NVARCHAR, therefore you should add it as a generic mapping, not under OTHER:
		`<entry key="NVARCHAR" value="java.lang.String"/>`

9.9.7 Extending JDBC Type Mapping

Some database types are not even mapped to a JDBC type. In particular, Oracle uses the TIMESTAMP WITH TIME ZONE and TIMESTAMP WITH LOCAL TIME ZONE that must be mapped in order to appear in JasperReports Server. If there are other types in your database, you can override or extend the JDBC type mapping with the following configuration:

Extending JDBC Type Mapping
Configuration File
.../WEB-INF/applicationContext-semanticLayer.xml

Extending JDBC Type Mapping		
Property	**Bean**	**Description**

`codeToJdbcType Mapping`	`jdbcMeta Configuration`	This property contains a map of database type codes to JDBC types. By default the codes for Oracle TIMESTAMP types are mapped:
		
		`<entry key="-101" value="TIMESTAMP"/>`
		`<entry key="-102" value="TIMESTAMP"/>`
		
		Add or replace these entries to map additional types from your database.

9.9.8 Accessing Materialized Views

Domains access tables and views by default, but some databases support other structures such as materialized views. These alternate table structures don't show up by default, but you can often configure Domains to display and access them.

If the JDBC driver for your database assigns a standard table type identifier to the materialized view, you can access it in Domains, Ad Hoc views, and reports. To find the table type, use a JDBC client such as the SQuirreL tool to view your database schema. In SQuirreL, use the "Objects" tab to browse the tables and views organized by table type. Look for your materialized view and note its table type.

The table type values are defined in the DatabaseMetaData.html.getTables() documentation. When you know the string corresponding to your table type, add it to the following configuration value:

Accessing Materialized Views		
Configuration File		
.../WEB-INF/applicationContext-semanticLayer.xml		
Property	**Bean**	**Description**
`tableTypes`	`jdbcMeta Configuration`	Uncomment the JDBC table type corresponding to the materialized view or other table structure in your databases, for example: `<value>LOCAL TEMPORARY</value>`

9.10 Configuring JasperReports Library

JasperReports Server's reporting features are built on the JasperReports Library, which is embedded in the server. Many of the options you can configure to change the server's functionality are actually JasperReports Library options. The configuration options can control many aspects of the server's behavior, from the way reports are exported into different file formats, to the default font.

These options can be set at different levels:

- Global – Applies to all reports generated by the server. Global JasperReports properties are defined in the .../WEB-INF/classes/jasperreports.properties file.
- Report – Defined in the JRXML of the report and applies to that specific report.
- Element – Defined in the JRXML and applies to specific elements of the report.

For more information about JasperReports Library configuration, see http://jasperreports.sourceforge.net/config.reference.html.

The following sections highlight a few of the available options:

- **Extending JasperReports Library**
- **Changing the Crosstab Limit**
- **Setting a Global Chart Theme**
- **Disabling Interactivity in the Report Viewer**
- **Enabling the XHTML or HTML Exporters**
- **Enabling Flash or HTML5 for Pro Charts**
- **Configuring a JavaScript Engine for Graphical Report Rendering**

9.10.1   Extending JasperReports Library

You can extend JasperReports Library by implementing the public interfaces it exposes.

Such an implementation is usually stored in a JAR (Java Archive) that contains a file called jasperreports_extension.properties, and specifies a factory class used to instantiate an extension registry. The extension registry specifies one or more extension objects, each of which corresponds to a JasperReports Library extension point represented by a Java interface.

Place this JAR on the JasperReports Library classpath, and your extension is automatically available.

For more information, refer to *JasperReports Library Ultimate Guide*.

9.10.2   Changing the Crosstab Limit

If you use crosstab reports, you may experience Out of Memory errors if the reports are very large or complex. You can configure JasperReports Server to return a message instead of memory errors when users run such crosstabs. To do so, enable the `net.sf.jasperreports.crosstab.bucket.measure.limit` property and set its maximum value. To do so, edit the following configuration file:

Crosstab Report Configuration Option
Configuration File
.../WEB-INF/classes/jasperreports.properties

Crosstab Report Configuration Option	
Property	**Description**

`net.sf.jasperreports.`
`crosstab.bucket.`
`measure.limit`
	This value represents the maximum number of cells multiplied by the number of measures in the crosstab. The default value is `100000`.
	
	Enter large values to allow your users to create larger, more complicated crosstabs; enter small values to restrict them.
	
	If you experience OutOfMemoryExceptions after changing this value, try setting it to a smaller number, or configure your JVM to allow more memory to be used.

9.10.3   Setting a Global Chart Theme

Chart themes control the look and feel of the charts generated by JasperReports Server. Chart themes can be applied at the level of either the server or the individual report:

- To apply a theme at the report level, select it when designing the report in Jaspersoft Studio. Note that you can also apply a theme to individual chart elements. Note that a chart theme can be included in a report unit as a resource. In this case the theme is available only to charts in that report unit.
- To apply a theme at the server level, copy the chart theme JAR to the correct location and edit its configuration file.

A chart theme is a JAR file that defines the look and feel of a chart. Once you have created the chart theme JAR file, copy it to the WEB-INF\lib directory. Chart themes in this location are available to any chart in the instance of the server. They can also be set as the global chart theme.

To set a theme as the default chart theme, edit the following configuration file:

Global Report Theme	
Configuration File	
.../WEB-INF/classes/jasperreports.properties	
Property	**Description**
`net.sf.jasperreports.` `chart.ChartTheme`	The name of a chart theme that is in the .../WEB-INF/lib directory.

We recommend that you create your chart themes in Jaspersoft Studio. Click **File** > **New** > **Other**>**Chart Theme**, then use Jaspersoft Studio to archive the new chart theme as a JAR.

Chart themes do not apply to Ad Hoc chart views.

9.10.4   Disabling Interactivity in the Report Viewer

By default, the report viewer's interactivity is enabled, and reports with interactive elements (such as the table component) are interactive when run in the web server and displayed in the viewer. If you don't want your reports to be interactive, you can disable interactivity across the entire server by editing a configuration file.

Interactivity in the Report Viewer	
Configuration File	
.../WEB-INF/classes/jasperreports.properties	
Property	**Description**
`net.sf.jasperreports.` `components.table.interactive`	By default, this property is set to `true`; in this case, interactivity is enabled in the report viewer. Set it to `false` to disable interactivity.

> Changing this setting in this configuration file changes the behavior for the entire server. To configure this behavior at the report, table, or column level, edit the report's JRXML properties.

9.10.5   Enabling the XHTML or HTML Exporters

By default, JasperReports Server exports HTML format using an HTML-based exporter. Unlike the default exporters from previous versions of the server (`html` and `xhtml`), the new HTML exporter (`html2`) is more forgiving when exporting reports that have overlapping elements while still preventing text from being cut off due to font metrics issues.

This setting affects all cases when HTML is exported, including when reports are exported from the report viewer and when they are scheduled to produce HTML output.

To use an older HTML exporter in JasperReports Server, edit the following configuration file:

HTML Exporters
Configuration File
.../WEB-INF/classes/jasperreports.properties

HTML Exporters	
Property	**Description**
`com.jaspersoft.jasperreports.export.html.type`	Determines which of the HTML exporters is used. Valid values are: • `html2` is the default HTML exporter. It handles overlapping report elements more gracefully than the other exporters. • `xhtml` was the default HTML exporter in versions from 4.5 to 5.0. We continue to support this exporter. It handles overlapping report elements more gracefully that the `html` exporter. However, it is subject to font metric mismatches between client browsers, which can result in text being cut off. • `html` was the default HTML exporter in versions of JasperReports Server before 4.2. Jaspersoft continues to support this exporter. It doesn't handle overlapping elements as gracefully as the other exporters.

Note that the properties are mutually exclusive; you can have only one uncommented at a time.

As of JasperReports Server version 5.5, if your reports include interactive elements such as the table component (which supports sorting and filtering in the HTML viewer), you must use the `html2` exporter to enable the interactive features; the `html` and `xhtml` exporters don't support them.

9.10.6   Enabling Flash or HTML5 for Pro Charts

By default, JasperReports Server renders Pro Charts (those based on Fusion Charts) using Adobe Flash. If Flash isn't found in the client environment, the server renders the chart using HTML5 instead. For example, Pro Charts displayed on devices that run Apple's iOS are rendered using HTML5 because Flash isn't available. Note that not all browsers support HTML5.

Note that Pro Charts are available only in the JasperReports Server professional edition.

You can configure the server to default to HTML5 when rendering Pro Charts. In this case, if your browser doesn't support HTML5, the chart won't be rendered.

To render Pro Charts using HTML5, edit the following configuration file:

Pro Charts Renderer
Configuration File
.../WEB-INF/classes/jasperreports.properties

Pro Charts Renderer	
Property	**Description**
`com.jaspersoft.jasperreports.` `fusion.charts.render.type`	Determines which of the following renderers is used: • `flash` is the default renderer for Pro Charts. If Flash isn't available, the server tries to render the chart in HTML5. • `html5` is the newest renderer for Pro Charts. Use it if you can't support Flash.

Note that this property applies only to reports that rely on Pro Charts and affects only the HTML preview and export.

Typically, this property is set at the server level; to override the server-level setting for a specific Pro Chart report, set this property at the report level, and also specify a second property as shown:

```
net.sf.jasperreports.print.transfer.fusion=com.jaspersoft.jasperreports.fusion
```

This allows the reporting engine (JasperReports Library) to recognize the Fusion settings. If this property isn't set, the `com.jaspersoft.jasperreports.fusion.charts.render.type` property is ignored at the report level.

9.10.7   Configuring a JavaScript Engine for Graphical Report Rendering

Depending on the circumstances, a given graphical element (such as a chart, a map, or a widget) in a report can be rendered in two ways:
•   When run directly in the web UI, the browser itself renders the chart.
•   When scheduled to run later or run in the background, an internal engine renders the chart.

By default, JasperReports Server's internal JavaScript engine is Rhino, which is an excellent solution for most cases; most JasperReports Server users can accept this default. However, you may want to investigate other engines if you encounter any of the following issues when scheduling chart-based reports or running them in the background:
•   Poor performance when generating complex charts or charts that contain large volumes of data.
•   Out of memory messages.
•   Incorrect scaling when certain Pro Chart reports are printed.
•   Results that don't match those generated when the report is run directly in the web UI. For example, text elements incorrectly sized or placed.

In such cases, we recommend using PhantomJS as the engine to execute JavaScript when generating graphical reports that are run in the background or scheduled. PhantomJS is a headless WebKit with JavaScript API. To use PhantomJS, download the correct version for your environment. Download PhantomJS and install it on the computer hosting JasperReports Server. For installation instructions, refer to the documentation provided with PhantomJS.

Once PhantomJS is installed, point JasperReports Server to its location. You can configure several options independently: HighCharts generation, Pro Charts generation, Pro Widgets generation, and Pro Maps generation.

> These are a server-wide settings. In a given server, all charts of the same type (HighCharts or Fusion (Charts Pro, Maps Pro, or Widgets Pro)) must use the same JavaScript engine.
>
> You can't use PhantomJS to render JFreeCharts. Such reports are always generated by Rhino when run in the background or scheduled.

To configure JasperReports Server to use PhantomJS for HighCharts, edit the following properties:

JavaScript Engine Configuration for HighCharts	
Configuration File	
.../WEB-INF/classes/jasperreports.properties	
Property	**Description**
`com.jaspersoft.jasperreports.highcharts.phantomjs.executable.path`	This property points to the engine the server should use to generate HighCharts-based charts in reports run in the background or scheduled. For example, if you're using Windows and you expanded the PhantomJS 1.8.1 ZIP file into the root of your C: drive: `com.jaspersoft.jasperreports.highcharts.phantomjs.executable.path=C:\\phantomjs-1.8.1-windows\\phantomjs.exe`
`com.jaspersoft.jasperreports.highcharts.phantomjs.tempdir.path`	The temporary directory where PhantomJS stores its output. By default, JasperReports Server expects this output in the location defined by Java's `java.io.tmpdir` system property.
`com.jaspersoft.jasperreports.highcharts.phantomjs.executable.timeout`	The maximum number of milliseconds to wait for output from PhantomJS before the chart times out. The default is 3000.

To configure JasperReports Server to use PhantomJS for Pro Charts (Fusion), edit the following properties:

JavaScript Engine Configuration for Pro Charts (Fusion)
Configuration File
.../WEB-INF/classes/jasperreports.properties

JavaScript Engine Configuration for Pro Charts (Fusion)	
Property	**Description**
`com.jaspersoft.jasperreports.` `fusion.phantomjs.` `executable.path`	This property points to the engine the server should use to generate Pro Charts (based on Fusion) in reports run in the background or scheduled. For example, if you are using Windows and you expanded the PhantomJS 1.8.1 ZIP file into the root of your C: drive: `com.jaspersoft.jasperreports.fusion.` `phantomjs.executable.path=C:\\phantomjs-1.8.1-` `windows\\phantomjs.exe`
`com.jaspersoft.jasperreports.` `fusion.phantomjs.` `tempdir.path`	The temporary directory where PhantomJS stores its output. By default, JasperReports Server expects this output in the location defined by Java's `java.io.tmpdir` system property.
`com.jaspersoft.jasperreports.` `fusion.phantomjs.` `executable.timeout`	The maximum number of milliseconds to wait for output from PhantomJS before the chart times out. The default is 3000.

By default, when Fusion-based reports are viewed in the web UI, they are generated as Flash elements. You can configure the web UI to generate your reports using HTML5 instead. For details, see **"Enabling Flash or HTML5 for Pro Charts" on page 216**.

9.11 Configuring Input Control Behavior

When defining text input controls, the default server behavior allows empty strings, even if you've configured a regular expression and made the input control mandatory. Use this setting to enforce the regular expression even on empty strings. This forces the user to provide a conforming value for the input control.

Input Control Behavior	
Configuration File	
.../WEB-INF/applicationContext-cascade.xml	
Bean	**Description**
`applyRegexpToEmptyString`	The default value of `false` gives the traditional behavior: even if a regular expression is defined, it is not applied to empty strings. If you want to strictly enforce the regular expression, even on empty input strings, set this property to `true`.

9.12 Configuring the Scheduler

The scheduler runs reports in the background according to a user-defined schedule (also called a job). You can configure the following aspects of the scheduler:

- **Configuring the Scheduler Misfire Policy**
- **Configuring Scheduler Failure Notifications**
- **Restricting File System Output**
- **Removing Report Scheduling Interval Options**
- **Adding a Holiday Exclusion Calendar**

9.12.1 Configuring the Scheduler Misfire Policy

A scheduler misfire occurs when the scheduler cannot run a report at the designated time, for example because JasperReports Server is offline, its database is offline, or the number of threads is limited. In this case, you can configure the behavior of the scheduler to retry the report or skip the scheduled run.

You can set a different misfire policy for each kind of job schedule: single job, repeating job, and calendar job. Misfire policies are defined in the Quartz Scheduler documentation and other online resources:

http://quartz-scheduler.org/documentation/quartz-2.x/tutorials/tutorial-lesson-05
http://quartz-scheduler.org/documentation/quartz-2.x/tutorials/tutorial-lesson-06
http://nurkiewicz.blogspot.com/2012/04/quartz-scheduler-misfire-instructions.html

Configuring Scheduler Misfire Policy	
Configuration File	
.../WEB-INF/js.quartz.properties	
Property	**Description**
`report.quartz.misfirepolicy. singlesimplejob`	Sets the misfire policy for single jobs to one of the following: • SMART_POLICY • MISFIRE_INSTRUCTION_FIRE_NOW • MISFIRE_INSTRUCTION_IGNORE_MISFIRE_POLICY

Configuring Scheduler Misfire Policy	
`report.quartz.misfirepolicy.repeatingsimplejob`	Sets the misfire policy for repeating jobs to one of the following values: • SMART_POLICY • MISFIRE_INSTRUCTION_FIRE_NOW • MISFIRE_INSTRUCTION_IGNORE_MISFIRE_POLICY • MISFIRE_INSTRUCTION_RESCHEDULE_NEXT_WITH_EXISTING_COUNT • MISFIRE_INSTRUCTION_RESCHEDULE_NOW_WITH_EXISTING_REPEAT_COUNT • MISFIRE_INSTRUCTION_RESCHEDULE_NOW_WITH_REMAINING_REPEAT_COUNT
`report.quartz.misfirepolicy.calendarjob`	Sets the misfire policy for jobs with calendar recursion to one of the following values: • SMART_POLICY • MISFIRE_INSTRUCTION_IGNORE_MISFIRE_POLICY • MISFIRE_INSTRUCTION_FIRE_ONCE_NOW • MISFIRE_INSTRUCTION_DO_NOTHING

9.12.2 Configuring Scheduler Failure Notifications

By default, if a scheduled report runs but causes an error, the scheduler sends an email to the schedule owner and to all JasperReports Server administrators in the same organization. This is in addition to any failure notification addresses specified on the **Notifications** tab of the scheduler wizard. To receive these scheduler failure alerts, administrators must have valid email addresses defined in their user accounts.

You can also configure the scheduler to send failure notifications to different users based on roles, or turn off failure notifications.

Configuring Scheduler Failure Notifications		
Configuration File		
.../WEB-INF/applicationContext-report-scheduling.xml		
Entry Key	**Bean**	**Description**
`administrator Role`	`quartz Scheduler`	This setting determines the role to which the scheduler failure notifications will be sent. All users in the organization with this role and a valid email address defined in their user profile will receive the email notification. By default, this setting is `ROLE_ ADMINISTRATOR`.

Configuring Scheduler Failure Notifications		
disableSending AlertToAdmin	quartz Scheduler	Disables or allows failure notifications to be sent to the role in the previous setting. By default, this setting is `false`, meaning that notifications are sent. Set this value to `true` to disable scheduler failure notifications being sent to administrators (or the role defined above).
disableSending AlertToOwner	quartz Scheduler	Disables or allows failure notifications to be sent to the schedule owner. By default, this setting is `false`, meaning that notifications are sent. Set this value to `true` to disable scheduler failure notifications to the schedule owner.

9.12.3   Restricting File System Output

The scheduler outputs reports through several channels. Most reports are emailed, but reports can also be written to FTP folders. You can also configure the scheduler to write reports to the server's local file system. This option is disabled by default for security reasons.

> If you turn on scheduler file system output, make sure you have configured user and folder access rights to make sure that malicious files cannot be written to your file system. The process that writes the files is the same user that runs the application server hosting JasperReports Server.

Scheduler File System Output
Configuration File
.../WEB-INF/applicationContext.xml

Scheduler File System Output	
Property to Update	**Description**
enableSaveToHostFS	Set the value from "false" (the default) to "true". When true, the user interface for the scheduler displays active fields that allow the schedule creator to specify a folder in the server's file system. The scheduler will write files to this location every time it runs the schedule for this report. **Output Destination** ☑ Output To Repository /public/Samples/Reports Browse ☑ Output To Host File System /Archives/reports/daily This property also determines the scheduler's overall access to the file system. When true, any schedule configured with a file system folder will write to the file system. When false, no scheduled reports will write output to the file system (FTP and email output are not affected). However, any file system output specified in a schedule remains defined and will again trigger file system output when this property is true again.

9.12.4 Removing Report Scheduling Interval Options

Users can schedule reports to run at regular intervals. For simple recurrence, the default interval can be expressed in days, hours, or minutes. To prevent users from scheduling frequent reports, you can limit the intervals to days or hours by editing the following configuration file:

Scheduling Interval Options	
Configuration File	
.../WEB-INF/flows/reportJobBeans.xml	
Section to Update	**Description**
recurrenceIntervalUnits	Comment out the intervals you want to disable.

To remove a temporal interval, enclose the corresponding bean in comment characters. For example, to keep users from scheduling reports at minute intervals, comment out the bean containing the INTERVAL_MINUTE field:

```
<!--
<bean class="com.jaspersoft.jasperserver.war.dto.ByteEnum">
    <property name="code">
        <util:constant static-field="com.jaspersoft.jasperserver.api.engine.scheduling.
            domain.ReportJobSimpleTrigger.INTERVAL_MINUTE"/>
```

```
    </property>
    <property name="labelMessage">
        <value>job.interval.unit.minute.label</value>
    </property>
</bean>
-->
```

9.12.5   Adding a Holiday Exclusion Calendar

The scheduler supports exclusion calendars to specify days or times when no report should be run, even if scheduled. For example, you might not want a report to run on a bank holiday when the financial data would be meaningless.

The scheduler maintains a list of named calendars, and the user interface allows the schedule creator to select a calendar whose dates will be excluded from the schedule.

Currently, the only method to define a holiday calendar is through the REST API. You can use any browser plug-in that acts as a REST client and can send PUT requests to JasperReports Server. Using such a plug-in, compose and send the following REST request (header and body) to your server:

```
PUT http://<host>:<port>/jasperserver[-pro]/rest_v2/jobs/calendars/2014FrenchHolidays
Content-Type: application/xml

<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<reportJobCalendar>
    <calendarType>holiday</calendarType>
    <description>2014 French Holidays</description>
    <excludeDays>
        <excludeDay>2014-01-01</excludeDay>
            <excludeDay>2014-04-18</excludeDay>
        <excludeDay>2014-04-21</excludeDay>
        <excludeDay>2014-05-01</excludeDay>
        <excludeDay>2014-05-08</excludeDay>
        <excludeDay>2014-05-29</excludeDay>
        <excludeDay>2014-06-09</excludeDay>
        <excludeDay>2014-07-14</excludeDay>
        <excludeDay>2014-08-15</excludeDay>
        <excludeDay>2014-11-01</excludeDay>
        <excludeDay>2014-11-11</excludeDay>
        <excludeDay>2014-12-24</excludeDay>
        <excludeDay>2014-12-25</excludeDay>
    </excludeDays>
    <timeZone>GMT+01:00</timeZone>
</reportJobCalendar>
```

For example, using the Poster plug-in for Firefox, you can submit this request as shown in the following figure. The figure also shows the successful reply from the server.
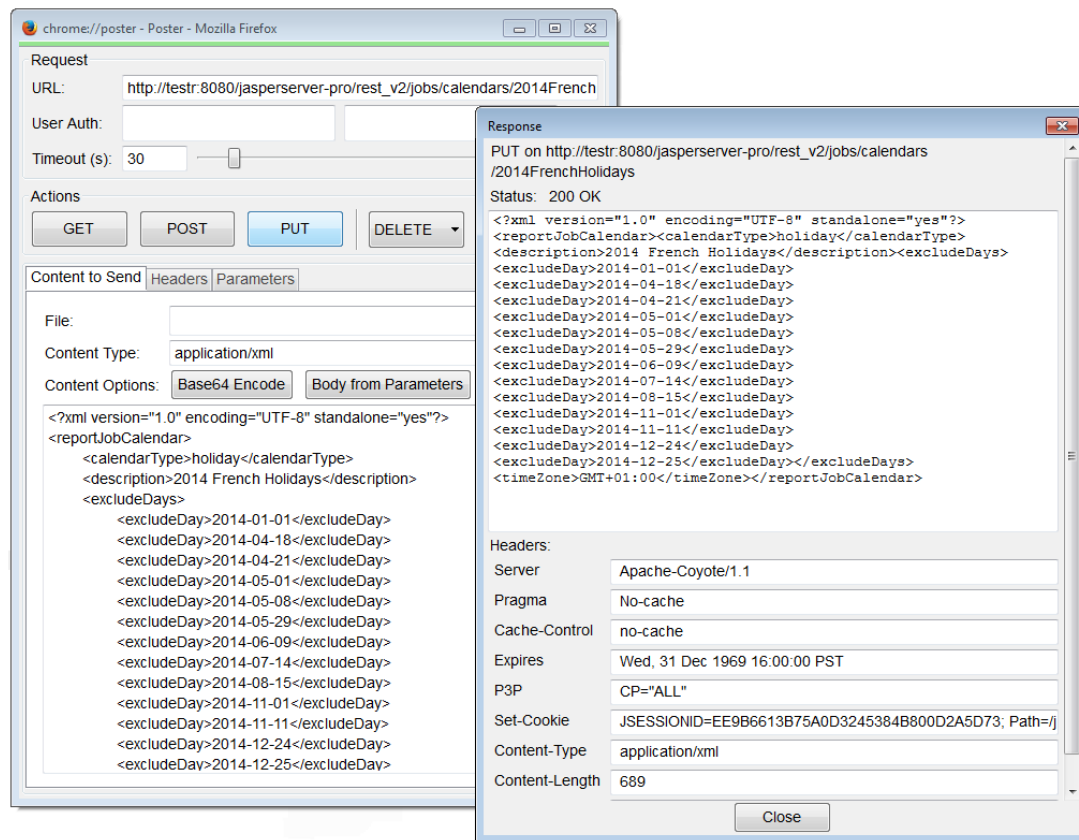
Figure 9-10 Creating a Holiday Calendar with REST Web Services

Then you should see your new calendar in the list of calendars in the Schedule tab.
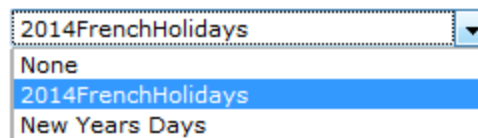
Figure 9-11 Selecting a Custom Holiday Calendar in the Scheduler

The REST API supports other types of calendars, however, the user interface lists only calendars of type `holiday`. Using the REST API, you can create and manage any number of calendars and update any schedule to use them. For more information, see the *JasperReports Server Web Services Guide*.

9.13 Configuring the Heartbeat

During installation (or the first time you log in as administrator), you're prompted once to participate in our Heartbeat program, which reports technical information to us about your implementation, such as the operating system, JVM, application server, database (type and version), data source types, and JasperReports Server edition and version number.

If you change your mind, you can change the heartbeat behavior by editing the following configuration file:

Heartbeat Options	
Configuration File	
.../WEB-INF/js.config.properties	
Property	**Description**
`heartbeat.enabled=true`	When this property is set to `true`, JasperReports Server reports information about your environment to us once a week. When it is set to `false`, information is not sent.
`heartbeat.askForPermission.enabled`	Determines whether the administrator is prompted (the next time he logs into the web UI) about whether to allow heartbeat data to be sent. Typically, there is never cause to edit this property directly.
`heartbeat.permissionGranted.enabled`	Indicates whether a user has granted the server permission to send Heartbeat data. Setting this property to `false` prevents data from being sent.

All of these settings are properties that are substituted into the `heartbeatBean` in the .../WEB-INF/applicationContext-heartbeat.xml file.

9.14 Configuring the Online Help

JasperReports Server professional edition includes an online help system that describes the web interface. If your users don't have Internet connectivity, or if you don't want to provide access to this system, you can configure the server to hide the help links completely.

Online Help Configuration Options	
Configuration File	
.../WEB-INF/applicationContext-webHelp.xml	

Online Help Configuration Options		
Property	**Bean**	**Description**
showHelp	webHelp	Determines whether the help links are displayed in the JasperReports Server web UI. Valid values are true and false. The **Help** link appears at the top right corner of the web UI's pages.
hostURL	webHelp	Indicates the name of the computer hosting the web server where the help is running. The value depends on the version of JasperReports Server. Do not change this value.
pagePrefix	webHelp	Defines the default page name to pass to the web server hosting the help system. The only valid value is Default_CSH.htm for this property.
helpContextMap	webHelp	Maps contexts in the application to topic identifiers in the help system. Many pages in the web application are configured for context-sensitivity. When a user clicks **Help** on such a page, JasperReports Server loads a specific topic in the help system. The topic that appears is determined by a map in the applicationContext-webHelp.xml file. The only valid values are the defaults.

CHAPTER 10 SERVER DIAGNOSTICS

JasperReports Server provides comprehensive diagnostics so administrators can monitor the health and performance of the running server. As of version 5.0, the server augments the audit logging with monitoring data that's easier to report against. Version 5.0 also introduced diagnostics that provide real-time information about the running server, such as memory use. All of these tools can help you troubleshoot issues and analyze performance.

- Auditing stores key events of interest to system administrators, such as login/logout time, user, report generated, report details, and object sizes. The audited events can be saved and moved to an archive automatically after a specified number of days.
- Monitoring is based on audit events and provides a multi-dimensional Domain to create Ad Hoc views. This helps you generate reports that contain key metrics such as which reports use the most resources. This allow you to find places to optimize your data sources and your reports.
- Diagnostics provide run-time data that shows the overall health of your server instance. For example, it includes values such as logged in users, currently running reports, scheduled reports, and memory use and performance in the Ad Hoc cache. The diagnostic data is exposed in a custom data source and can be easily viewed in a report or integrated with industry standard management consoles.

> This section describes functionality that can be restricted by the software license for JasperReports Server. If you don't see some of the options described in this section, your license may prohibit you from using them. To find out what you're licensed to use, or to upgrade your license, contact us.

This chapter contains the following sections:
- **Events Being Audited and Monitored**
- **Configuring Auditing and Monitoring**
- **Using the Audit Data**
- **Using the Monitoring Data**
- **Importing and Exporting Event Data**
- **Real-Time Diagnostics**
- **Exposing Diagnostics Through Jaspersoft's JMX Agent**
- **Using the Diagnostic Data in Reports**
- **Excluding Diagnostic Attributes**
- **Disabling Real-Time Diagnostics**

10.1 Events Being Audited and Monitored

In broad terms, an audit event is any atomic operation that can be recorded by the audit system. Event properties and attributes are features of the event; they can be defined internally or in custom code. Auditing and monitoring rely on the same record of events, so the audit events are also available in monitoring data sources, Domains, and reports.

The following table lists the defined audit events and the information collected about them. For every recorded event, JasperReports Server logs the time it occurred and the user who initiated it. See the configuration file applicationContext-audit.xml for complete specification of the events.

Event	Information Collected
Log in or log out	Time and user ID (recorded for every event)
Log in as	User logged in as
Run report or run a subreport within any other report	• Report referenced • Data source referenced • Report parameters and values • Report queries (such as SQL, Domain, HQL, generated SQL) • Execution start, end • Query execution time (in milliseconds) • Report rendering time (in milliseconds) • Caching parameters • Errors that occurred
Report schedule created, deleted, or updated	• Report referenced • Scheduling parameters and values
Scheduled report run	• Report output • Report delivery parameters (such as email) • Same parameters as when report is run
Creating report in Ad Hoc Editor	• Field added as a column • Field added as a group
Resource accessed for any reason (such as view, used in report, etc.)	• Resource referenced • Resource type
Resource added or updated	• Resource referenced • Resource type
Resource or folder deleted	Resource or folder referenced
Permissions added, updated, or deleted	• Resource or folder referenced • Previous permissions (before update)

Event	Information Collected
User added, updated, or deleted, also user password change	• User ID • User name • Email • Enabled flag • External flag • User attributes
Role added, updated, or deleted	• Role ID • Role name • Role organization
Organization added, updated, or deleted	• Organization ID • Organization description

10.2 Configuring Auditing and Monitoring

The auditing and monitoring features both rely on the same subsystem in JasperReports Server. The settings are located in the WEB-INF/applicationContext-audit.xml file. The beans and properties of this file are used to enable auditing and monitoring, set the archive options, and select the events to be logged.

10.2.1   Enabling Auditing and Monitoring

Because auditing and monitoring share the same subsystem, there is a master configuration setting to turn on the subsystem, and individual settings for auditing and monitoring. This allows you to turn on each feature independently, or turn off the whole subsystem to avoid any overhead from an unused feature.

By default, the auditing and monitoring subsystem is off.

Enabling Auditing and Monitoring Options	
Configuration File	
.../WEB-INF/js.config.properties	
Property	**Description**
`audit.records.enabled=false`	Set this property to `true` to enable the auditing and monitoring subsystem. Given the default value of the other settings, only auditing is turned on when this setting is first set to `true`.
`monitoring.records.enabled=false`	Once `audit.records.enabled=true`, set this property to `true` to enable monitoring. More precisely, this setting controls the logging of events in the tables used by the monitoring Domain.

Enabling Auditing and Monitoring Options	
`audit.sizeof.enabled=false`	By default, the large print objects in memory are not measured because that can impact performance. If you want to temporarily enable auditing for the memory use of print objects in reports, set this value to `true`.

Configuration File		
.../WEB-INF/applicationContext-audit.xml		

Property	Bean	Description
`isAuditPersistence Enabled`	`audit Service`	Set this property to `false` to disable logging of events in the tables used by the audit Domain. This allows you to turn off auditing data while keeping monitoring data. Avoid turning off both monitoring and audit persistence. Instead, turn off the entire subsystem with `audit.enabled=false`.

10.2.2   Auditing Archive Options

Archiving moves audit data to separate database tables. For example if you create reports for weekly or monthly server use based on the audit Domain, you'll never access audit logs older than a week or a month. Because audit data can be quite large, the old data can increase the time needed to write new data and query data for your report. Archiving automatically moves audit data to separate database tables after a certain time. If you want to create reports on historical audit data, we provide a Domain to access the archived audit logs. You can also configure the audit feature to delete old archive data if you no longer need it.

You should set the archiving interval to a level that balances your need to record and access audit data with your server's performance on large audit tables. Use the following configuration settings to change the archiving behavior.

Auditing Archive Options		

Configuration File		
.../WEB-INF/applicationContext-audit.xml		

Property	Bean	Description
`maxAudit EventAge ToArchive`	`auditService`	The number of days to keep audit data in the active database. The default is 30. Older data is moved to the archive.
`maxAudit EventAge`	`auditService`	The number of days to keep the data. Older data is deleted. The default is 0 (zero), meaning that old data is never deleted.

Auditing Archive Options		
`cronExpression`	`auditEvent ArchiverTrigger`	Defines the frequency of the archiving job in `cron` syntax. The default, `0 0 5* * ?`, is every day at 5a.m.
`cronExpression`	`auditEvent PurgerTrigger`	Defines the frequency of the audit delete (purge) job in `cron` syntax. The default, `0 0 3 * * ?`, is every day at 3 a.m.

The `cronExpression` properties use a <u>Quartz scheduler cron expression</u> that specifies the repeating trigger as seconds, minutes, hour, day of month, month, day of week.

10.2.3   Disabling Events and Properties

By default, all events and properties are logged. To enable or disable logging of a given event or property, use the applicationContext-audit.xml configuration file. Events that are logged are used by both the audit and monitoring subsystems, therefore disabling an event makes it unavailable for both auditing and monitoring.

In the file, event types and their properties are listed under `<util:map id="enabledEventsMapping">`. The map has three parts:

• WEB_SERVICES – Event types related to accessing JasperReports Server through a web service.
• GUI – Event types for access through the user interface.
• INTERNAL – Event types used by the server itself, such as when running a scheduled report.

To disable an event, comment it out. For example:

```
<!-- <entry key="createFolder" value="folderName,folderLabel,folderDescription" /> -->
```

To disable a property, use any of these measures:

• Delete the property. For example, remove `folderDescription`, resulting in:
```
<entry key="createFolder" value="folderName,folderLabel,exception" />
```

• Disable it with the `|` syntax. For example:
```
<entry key="createFolder" value="folderName,folderLabel,|folderDescription,exception" />
```

• Use the "all except" `*|` syntax to specify only the disabled property. All others are recorded. For example:
```
<entry key="createFolder" value="*|folderDescription" />
```

10.3 Using the Audit Data

JasperReports Server makes the audit data available to administrators through Domains and several prepared views and reports. These are located in the /Public/Audit folder of the repository.

Figure 10-1 **Audit Reports in the Repository**

There are two Domains and two sets of reports created for accessing audit data:

- Audit Domain and Audit Reports – Use these to view the current audit data; they run against the active audit database.
- Audit Archive Domain and Archived Audit Reports – Use these to run reports on archived data; they run against the archive database.

The contents of both Domains and reports are identical—they differ only in the database tables accessed in each case.

To create an Ad Hoc View based on the audit Domains, select **Create > Ad Hoc View**, select the **Domains** tab in the Data Chooser, and expand the folders to select one of the audit Domains.
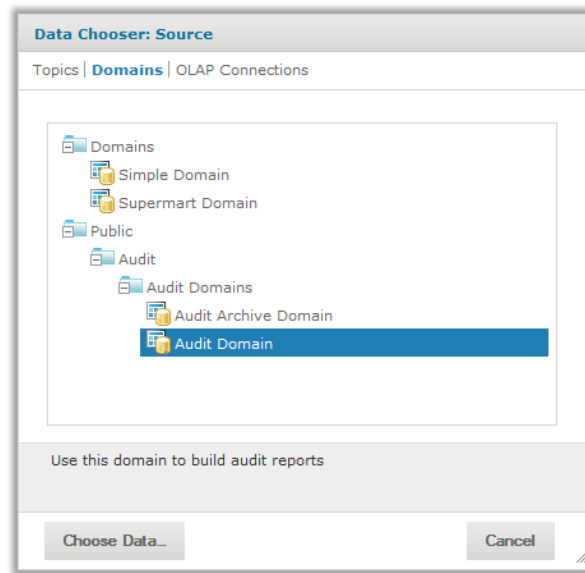
Figure 10-2 **Selecting an Audit Domain to Create an Ad Hoc View**

For instructions on using Domains in reports, see the Ad Hoc chapter in *JasperReports Server User Guide*. For documentation of Domains in general, see the Domains chapter in the same manual.

The following sections explain the contents of the Domains and the reports provided.

10.3.1   Audit Domain Items

The audit Domains expose the fields of the audit logging tables stored in the server's internal database. As with all Domains, the database tables are joined, and the fields are presented as items that can be used in Ad Hoc views.

The following tables describe the items in both audit Domains (Audit Domain and Audit Archive Domain). These items correspond to the information recorded for each event. When creating a view based on either Domain, choose the items that correspond to the type of event you want to report on.

These domain items are used in both general events and repository events:

Domain Item	Description
Date	Date event occurred.
Prop Long Value	`clob` value of event property, such as query.
Prop Type	Type of event property, such as destination folder, as per event map in configuration file.
Prop Value	`string` value of event property, such as folder name.

Domain Item	Description
Time	Time event occurred.
Event Type	Type of event, such as save resource, as per event map in configuration file.
Request Type	Repository request type of event.
Resource Type	Repository type of resource accessed in event.
Resource URI	URI of repository resource.

Domain items in the following table are recorded for user events:

Domain Item	Description
E-mail	E-mail address of event user (user at time of event).
Enabled	Whether event user is currently a user.
External	Whether event user was an external user.
Full Name	Full name of event user.
Password Changed	Whether event was a change of password
Organization	Organization of event user.
User Name	UserID of event user.

Domain items in the following table are recorded for role events:

Domain Item	Description
External	Whether role was defined in an external system.
Role Name	Name of the role in the event.
Organization	Organization of the role in the event.

Domain items in the following table are recorded when a report is generated:

Domain Item	Description
Date	Date the report is generated.
Time	Time the report is generated.

Domain Item	Description
Resource URI	URI of repository resource accessed for report.
Resource Type	Repository type of resource accessed for report.
Datasource URI	URI of data source accessed for report.
Query Execution Time	Time to execute the query in the database.
Report Rendering Time	Time to prepare query results for display.
Report Execution Time	Total time to execute the report (query execution + report rendering + overhead). Overhead includes tasks such as loading repository resources (report unit, data source, etc) and obtaining a DB connection from the data source.
Query	Specification of the report query.
User Name	UserID of event user.
Organization	Organization of event user.
Crosstab Group Field	Field name used in crosstab

10.3.2 Audit Reports and Ad Hoc Views

> Auditing is disabled by default, so the audit reports and their views are initially blank. To view audit reports, enable auditing as described in **"Configuring Auditing and Monitoring" on page 231**, then wait for user activity to generate events.

A number of Ad Hoc views and reports based on the audit Domains are provided in the Public/Audit/Audit Reports folder. The same views and reports are also provided in the Archived Audit Reports subfolder. These reports are identical, except they use the Audit Archive Domain and run against the archived audit data. These reports are visible only to administrators.

The reports are designed to cover common audit needs and can be used as-is. When auditing is enabled, audit events are recorded, the reports contain an up-to-the-minute record of events on your server. You can run the reports or schedule them as needed.

The Ad Hoc views used to create each report are also included. You can open them in the Ad Hoc editor to explore the audit data in real-time. You can also modify these views in the Ad Hoc Editor to generate new reports to suit your auditing requirements.

The following views and reports are provided:
• Audit Report – Generic example of a an audit report showing commonly audited events.
• Performance Crosstab Report – A crosstab that shows average performance of reports.
• Performance Report – Generates a list of reports, sorted by run-time to identify slow reports.
• Repository Resources Report – Shows repository resources and their associated events.
• Resource Execution Report – Generates a list of executed reports.

- User Activity Report – Generates a list of reports run by a specified user.

10.4 Using the Monitoring Data

JasperReports Server makes the monitoring data available to administrators through a Domain and several prepared views and reports. These are located in the /Public/Monitoring folder of the repository.
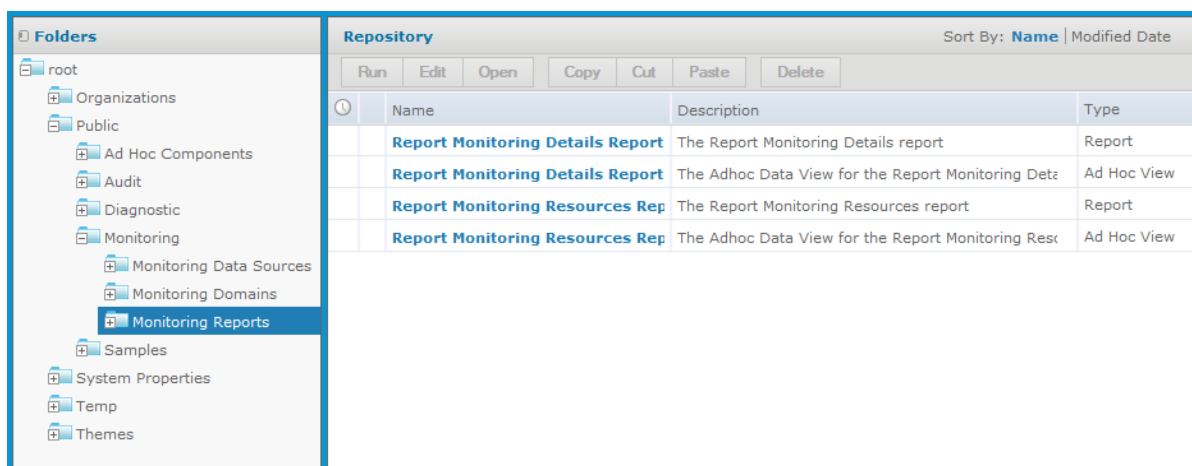

Figure 10-3 **Monitoring Reports in the Repository**

To create an Ad Hoc View based on the audit Domains, select **Create > Ad Hoc View**, select the **Domains** tab in the Data Chooser, and select the monitoring Domain. For instructions on using Domains in reports, see the Ad Hoc chapter in *JasperReports Server User Guide*. For documentation of Domains in general, see the Domains chapter in the same manual.

The following sections explain the contents of the Domains and the reports provided.

10.4.1 Monitoring Domain Items

The monitoring Domain exposes the fields of the monitoring tables stored in the server's internal database. As with all Domains, the database tables are joined, and the fields are presented as items that can be used in Ad Hoc views.

In this release of JasperReports Server, the monitoring fields are limited to those that record report execution events:

Domain Item	Description
Day	Day of the month the report finished running.

Domain Item	Description
Editing Action	The Ad Hoc editing step the user just performed (null for report execution): • insertDimensionInAxisWithChild – A dimension was added. • addMeasure – A measure was added. • setProperty – A property was set. • moveDimension – A dimension was moved.
Event Context	The context that triggered the report execution. Possible values are: • ui – The report ran interactively from the user interface. • web services – The report was run via web services. • internal – The report ran from an internal process, usually the scheduler.
Event Type	The type of report that was executed. Possible values are: • report execution – A report that ran from the repository. • ad hoc editing – A report that ran from an Ad Hoc view being edited.
Hour	Hour the report finished running.
id	ID number of the monitoring event.
Minute	Minute the report finished running.
Month	Month report finished running.
Query Execution Time	The time spent executing the SQL query in the database.
Report Rendering Time	The time spent rendering the report after receiving the query results (dataset).
Report URI	Repository path of the report that was run.
Time Stamp	Full time and date when the report was finished running, including milliseconds.
Total Report Execution Time	The total time spent running the report. Typically this is a little more than the sum of the query execution and report rendering times, due to overhead. such as loading repository resources (report unit, data source, etc) and obtaining a DB connection from the data source.
User Name	The user who ran the report.
User Organization	The organization of the user who ran the report.
Year	Year the report finished running.

10.4.2   Monitoring Reports and Ad Hoc Views

> The monitoring reports and their views are blank by default, because the audit subsystem that monitoring depends on is disabled by default and no audit data exists. To view these reports, first enable auditing as described in section **"Configuring Auditing and Monitoring" on page 231**, then wait for user activity to generate events.

A number of Ad Hoc views and reports based on the monitoring Domain are provided in the Public/Monitoring/Monitoring Reports folder.

The reports are designed to cover common monitoring needs and can be used as-is. When monitoring is enabled, audit events are recorded, and the reports contain an up-to-the-minute record of events on your server. You can run the reports or schedule them as needed.

The Ad Hoc views used to create each report are also included. You can open them in the Ad Hoc editor to explore the monitoring data in real-time. You can also modify these views in the Ad Hoc Editor to generate new reports to suit your monitoring requirements.

The following views and reports are provided:
- Report Monitoring Resources Report – Gives a list of all reports and shows their average and high-low runtimes.
- Report Monitoring Details Report – A crosstab that shows report runtimes on one axis and many dimensions such as a time hierarchy, user and organization, and event type on the other axis,.
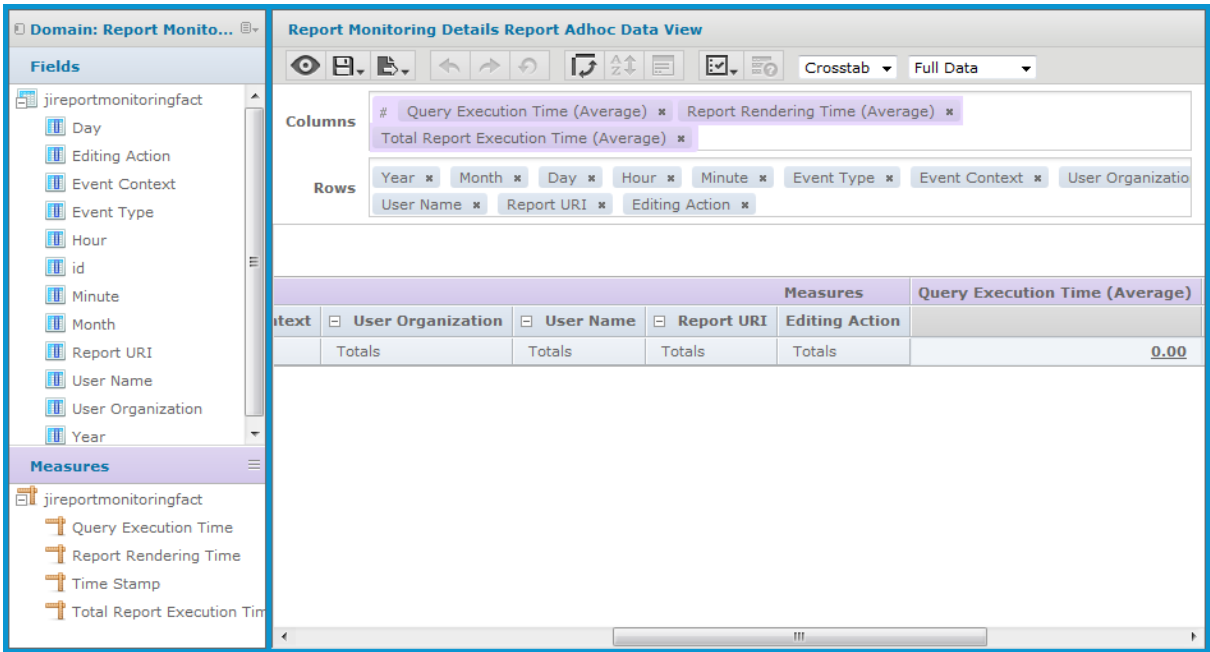
Figure 10-4 **Monitoring Ad Hoc View with Multi-dimensional Analysis**

10.5 Importing and Exporting Event Data

Audit and monitoring data can be imported and exported with the utilities described in the chapter **"Import and Export" on page 135**.

- To export audit data, use the `--include-audit-events` option in the export command.
- To export monitoring data use the `--include-monitoring-events` option in the export command.
- To import audit data, import the catalog containing audit data with the `--include-audit-events` option.
- To import monitoring data, import the catalog containing audit data with the `--include-monitoring-events` option.

> Data in temp folders is not exported.

10.6 Real-Time Diagnostics

With JasperReports Server 5.0, we introduced full instrumentation of the server so it can provide real-time diagnostic information such as memory use, concurrently running reports, cache statistics, and much more. Whereas auditing and monitoring provide a historical record of events on the server, diagnostics display the current status of the server and its Java Virtual Machine (JVM) environment.

For example, you can use the diagnostic information to adjust your JVM settings to optimize memory based on the size of typical datasets in your Ad Hoc cache. And because diagnostic data is available through a Topic, you can create Ad Hoc views that help you analyze your server performance and identify performance bottlenecks before they happen.

The diagnostic feature relies on the Java Management Extensions (JMX) industry standard to gather run-time data and expose it to other management interfaces. JasperReports Server implements several JMX elements:

- New JMX managed beans used to gather and hold diagnostic data such as the number of concurrent reports.
- Its own JMX agent, like a server, that responds to JMX requests and exposes the information from the managed beans.
- Connections to existing JMX managed beans in other components such as Ehcache to include their diagnostic data.
- Connections to other JMX agents such as JVM agent to gather system diagnostics.

For more information about JMX, see http://docs.oracle.com/javase/tutorial/jmx/.

There are two ways to use diagnostic data from JasperReports Server:

- If you have an external management console that can connect to a JMX agent, you can use it to connect to the server's JMX agent. You'll probably also want to connect to the JVM's JMX agent. You can then view the real-time diagnostic data from JasperReports Server integrated with the data from your other servers in your management console. Management consoles often offer further functionality like thresholds and alarms you can set on data from any JMX managed bean.
- We provide a custom data source within the server that collects the same information from the managed beans in the server and in the JVM. You can then create JasperReports that access this data source and present this diagnostic data in a report that suits your needs. We also provide a report and an Ad Hoc Topic to demonstrate this.

10.7 Exposing Diagnostics Through Jaspersoft's JMX Agent

By default, the diagnostic feature is configured to allow access to the diagnostic data through a remote management console that supports JMX. This allows you to integrate JasperReports Server diagnostics with your overall IT management strategy. Alternatively, the Java Development Kit provides the jConsole tool that uses JMX and can be used to monitor your server remotely. All of these management consoles access the JMX agent provided with JasperReports Server, and access is secured by the same users, passwords, and roles used by the server's web interface.

10.7.1 Connecting to the JMX Agent

To connect your management console to the JasperReports Server's JMX agent, initiate a new connection from your console and specify the following connection string:

```
service:jmx:rmi://localhost/jndi/rmi://<host>:<port>/<connectionName>
```

where:

- `<host>` is the computer where JasperReports Server is running
- `<port>` is the JMX port, by default `10990`
- `<connectionName>` is the name of the JMX agent, by default `jasperserver`

Therefore, the default connection string is:

```
service:jmx:rmi://localhost/jndi/rmi://<host>:10990/jasperserver
```

> If you have a firewall implemented on the computer hosting JasperReports Server, you'll need to open the JMX port (10990 by default) before connecting.

When prompted to enter a username and password, give a user with the following roles:

Commercial editions:	ROLE_SUPERUSER, thus by default the `superuser` user
Community project:	ROLE_ADMINISTRATOR, thus by default the `jasperadmin` user

The following sections explain how to modify the default connection values for:

- The connection name
- The JMX port
- The required roles

10.7.2 Configuring the Port and Connection Name

The connection name is the name registered for the JMX agent with the remote method invocation (RMI) service. If you have more than one JasperReports Server instance in a single app server, you need to change the registered connection name for all but the first one.

There are times you may want to change the port from default (10990) to another port, for example, if another service is configured to use the default port. For example, change the port if you have JasperReports Server running in a second app server on the same computer.

To change the port or connection name, edit one of following files:

- In the WAR file distribution before installation: <js-install>/buildomatic/default_master.properties
- In the installed WAR file before starting the server: <js-war>/WEB-INF/js.diagnostic.properties

Find the following lines and edit the values to the port number and connection name you want:

```
diagnostic.jmx.port = 10990
diagnostic.jmx.name = jasperserver
```

10.7.3   Configuring Roles for JMX Connections

If you want to allow other users to establish the JMX connection from a remote management console, configure
JasperReports Server to allow other roles:

- For commercial editions, edit the WEB-INF/applicationContext-Diagnostic-pro.xml file and modify the
  following setting:

```
<util:list id="diagnosticAllowedRolesPro">
    <value>ROLE_SUPERUSER</value>
</util:list>
```

- For community projects, edit the WEB-INF/applicationContext-Diagnostic.xml file and modify the
  following setting:

```
<util:list id="diagnosticAllowedRolesCe" value-type="java.lang.String">
    <value>ROLE_ADMINISTRATOR</value>
</util:list>
```

In both cases, you can change the existing role or add additional lines containing alternate `<value>ROLE_
name</value>`.

10.7.4   Disabling Remote Connections to the JMX Agent

By default, remote connections to JMX Agent are enabled and configured as described in **"Connecting to the
JMX Agent" on page 242**.

.To disable remote connections, edit the applicationContext-diagnostic.xml file and make the following
changes:

- Comment out the entire `jMXAuthenticator` bean.
- Comment out the entire `jMXregistry` bean.
- Comment out the entire `jasperJMXServerConnector` bean.
- Comment out or remove the property `depends-on="jMXregistry"` in the `jasperJMXServer` bean.

To comment out a section of this XML file, surround it with `<!--` and `-->` tags.

10.7.5   Alternative Connection Through App Server JMX Service

Most app servers also have a JMX agent, and if configured properly, they can discover the JasperReports Server
JMX agent and expose the diagnostic information. However, this connection has some significant security
implications:

- When the app server discovers and connects to the JMX agent, it has full access to the diagnostic
  information.
- If the app server's JMX agent is configured for remote access, the remote manager also has access to the
  JasperReports Server JMX agent--without needing to provide the JasperReports Server username and
  password. In other words, by trusting the app server to access the diagnostic information, you're also
  allowing anyone the app server trusts to connect.

- The connection between the two JMX agents is reciprocal. If remote access is still enabled on the JasperReports Server JMX agent, a remote manager who connects to it also sees the contents of the app server JMX agent. In other words, the app server JMX agent is also trusting anyone who we trust to connect to our JMX agent (with our username and password).

To connect automatically to the app server's JMX agent, assuming one is available, edit the default_master.properties file before you deploy the JasperReports Server web app, and add the following line:

```
diagnostic.jmx.usePlatformServer = true
```

If the app server is Apache Tomcat for example, a local JMX connection named Catalina appears to anyone accessing the JasperReports Server JMX agent.

10.8 Using the Diagnostic Data in Reports

In addition to remote JMX connections, JasperReports Server makes the diagnostic data available to administrators for internal reports. The following resources are visible to the system admin (superuser by default) in the repository:

- /Public/Diagnostic/Diagnostic Data Source – A custom data source that returns all of the diagnostic fields.
- /Public/Diagnostic/Diagnostic Report – A report that displays nearly every single diagnostic field.
- /Public/Ad Hoc Components/Topics/Diagnostic Topic – A JRXML resource that can be used as a Topic to create an Ad Hoc view using diagnostic fields.
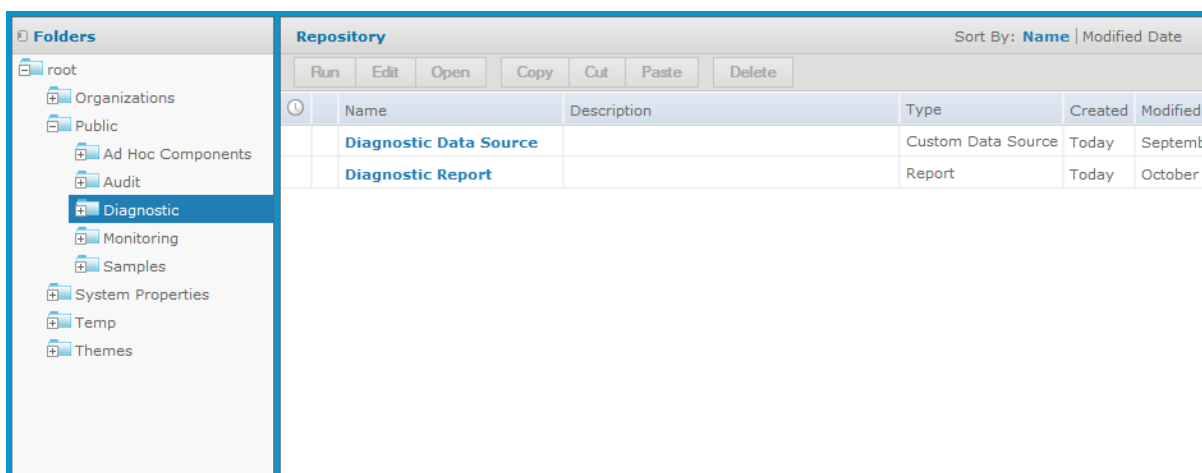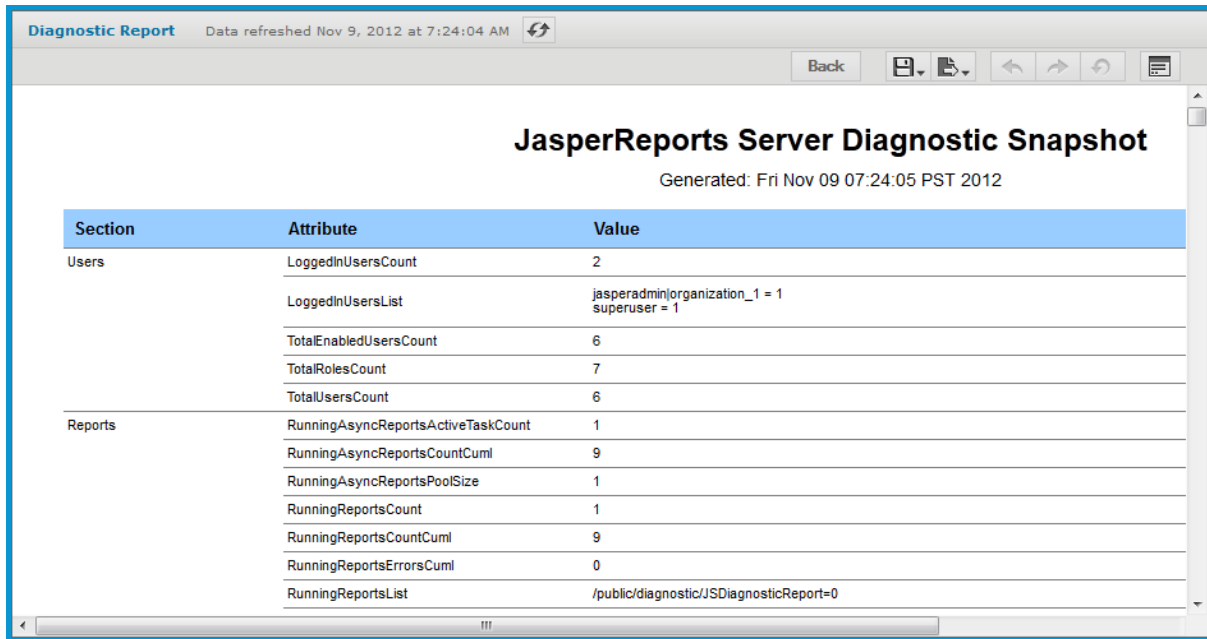
Figure 10-5 **Diagnostic Report in the Repository**

The diagnostic topic and report are based on the diagnostic data source. This is a custom data source that encapsulates the JMX information and makes it available as a data source for reports. In addition, there is an Internal Diagnostic Data Source type that can be used to create a new data source. If you accidentally delete the diagnostic data source, select **Create > Data Source**, set the type to Internal Diagnostic Data Source, and set the name and location to re-create it.

The following figure shows the beginning of the diagnostic report. This report displays nearly every field available through JMX and the diagnostic subsystem. This report serves as a reference for the diagnostic data. The report is also very useful when configuring your server, because it shows many server configuration settings

as well as system information such as memory use. You can use this information to fine tune the performance of your JasperReports Server instance.

Figure 10-6 **Contents of the Diagnostic Report**

10.9 Excluding Diagnostic Attributes

The JasperReports Server diagnostic feature exposes a set of managed beans (MBeans), each with a number of JMX attributes. By default, all attributes of all MBeans are available. If you would like to limit what is exposed, you can exclude any of the attributes or entire beans through the following configuration file:

.../WEB-INF/applicationContext-diagnostic.xml

- To exclude an attribute, locate the MBean where it's defined and uncomment it from the excludedDiagnosticAttributes property. For example, if you want to hide sensitive information about your internal database, modify the diagnosticRepositoryDatabaseInfoCe MBean as follows:

```
<bean id="diagnosticRepositoryDatabaseInfoCe" class=
    "com.jaspersoft.jasperserver.api.logging.diagnostic.jmx.DiagnosticDynamicMBean">
    <property name="diagnosticServices">
        <set>
            <ref bean="repositoryDatabaseInfo"/>
        </set>
    </property>
    <property name="excludedDiagnosticAttributes">
        <set>
            <value>DatabaseProductName</value>
            <value>DatabaseProductVersion</value>
            <value>DriverName</value>
            <value>SQLKeywords</value>
            <value>URL</value>
            <value>UserName</value>
            <value>JDBCMajorVersion</value>
            <value>JDBCMinorVersion</value>
            <!--
            <value>MaxRowSize</value>
            <value>MaxStatementLength</value>
            <value>MaxConnections</value>
            <value>MaxCharLiteralLength</value>
            <value>MaxColumnsInTable</value>
            <value>MaxColumnsInSelect</value>
            <value>MaxColumnsInGroupBy</value>
            <value>MaxColumnNameLength</value>
            -->
        </set>
    </property>
</bean>
```

- To exclude an entire bean, comment it out or remove it from the list of beans in the `diagnosticExportingMBeansMap`. For example, instead of excluding selected attributes, you could remove the entire repository database MBean as follows:

```
<util:map id="diagnosticExportingMBeansMap" >
    <entry key="jasperserver:name=Users" value-ref="${bean.diagnosticUsers}"/>
    <entry key="jasperserver:name=Reports" value-ref="${bean.diagnosticReports}"/>
    <entry key="jasperserver:name=Scheduler" value-ref="${bean.diagnosticScheduler}"/>
    <entry key="jasperserver:name=Settings" value-ref="${bean.diagnosticSettings}"/>
    <entry key="jasperserver:name=Repository" value-ref="${bean.diagnosticRepository}"/>
    <entry key="jasperserver:name=About" value-ref="${bean.diagnosticJSAbout}"/>
    <entry key="jasperserver:name=HibernateStatics"
            value-ref="${bean.diagnosticHibernate}"/>
    <entry key="jasperserver:name=EhCache"
            value-ref="${bean.diagnosticEhCache}"/>
    <entry key="jasperserver:name=ExternalInfo"
            value-ref="${bean.diagnosticExternalInfo}"/>
    <!--
    <entry key="jasperserver:name=RepositoryDatabase"
            value-ref="${bean.diagnosticRepositoryDatabaseInfo}"/>
    -->
</util:map>
```

10.10 Disabling Real-Time Diagnostics

By default the JMX diagnostic subsystem is always enabled, but external access is password-protected and requires opening the diagnostic port in your firewall as described in **"Exposing Diagnostics Through Jaspersoft's JMX Agent" on page 242**. If you wish to disable all external access, see **"Disabling Remote Connections to the JMX Agent" on page 243**.

Internally, the diagnostics subsystem is passive and has no performance impact until it is accessed in a report through the diagnostic data source. However, if you wish to disable real-time diagnostics entirely, rename or remove the following files:

• applicationContext-diagnostic.xml
• applicationContext-diagnostic-pro.xml

In that case, the diagnostic data source, the sample report, and the sample Topic described in **"Using the Diagnostic Data in Reports" on page 244** will not function either. They can be deleted from the repository.

APPENDIX A   TROUBLESHOOTING

This section describes functionality that can be restricted by the software license for JasperReports Server. If you don't see some of the options described in this section, your license may prohibit you from using them. To find out what you're licensed to use, or to upgrade your license, contact us.

This appendix contains the following sections:

- **Number of Users Exceeded**
- **Running Out of Database Connections**
- **Fields Not Listed in Ad Hoc Editor**
- **Field Names Disappear in Ad Hoc Editor**
- **Ad Hoc Filter With All Values Causing Error**
- **Ad Hoc Dimensions Too Large**
- **Custom URLs Not Loading in Dashboards**
- **Print View Not Displaying in Dashboards**
- **Scheduler Sending Multiple Emails**
- **Scheduler Running Deleted Jobs**
- **Charts Not Appearing in Excel Export**
- **Working With Data Sources**
- **Hadoop-Hive Reports Not Running**
- **Reverting to the Old Home Page**

A.1   Number of Users Exceeded

When you have more users defined in the server than your license allows, the login page displays a warning; users can still log in. After a 24-hour grace period, an email is sent to the administrator and users can no longer log in. Most server functionality is disabled. To re-enable server functionality:

- Contact Jaspersoft sales to purchase additional user licenses. When you install the new license, the server becomes fully functional for all users.
- Delete user accounts until the number of user accounts on the server is in accord with your license. When server functionality is disabled, administrators can still log on and select **Manage > Users** to delete user accounts. For more information see **"Managing Users" on page 32**.

A.2   Running Out of Database Connections

JasperReports Server manages a pool of connections for each JDBC data source. The default number of connections is 20, but if you run many reports concurrently against the same data source you may reach the connection limit and see degraded performance. In particular, using the web service APIs, REST clients can easily launch many report executions at the same time and reach the limit.

The connection pool size is limited to avoid having too much memory permanently allocated to connections. But if you need more concurrent connections on a regular basis, you can increase the limit with the following configuration:

Reducing the Size Limit for Ad Hoc Dimensions		
Configuration File		
.../WEB-INF/applicationContext.xml		
Property	**Bean**	**Description**
`<constructor-arg type="int" value="20"/>`	`dataSource ObjectPool Factory`	Change the default value to match your concurrent connections. Make sure you have enough memory to handle the connections and the concurrent report executions.

If you are using JNDI data sources, you can configure the number of connections in your application server. For more information, see the sections on JNDI in **"Working With Data Sources" on page 256**.

A.3   Fields Not Listed in Ad Hoc Editor

The Ad Hoc editor supports only certain data types. If a Topic contains a field with an unsupported type, the field does not appear when you open the Topic in the Ad Hoc editor. These are the data types supported in the Ad Hoc editor:

- java.lang.String
- java.lang.Byte
- java.lang.Short
- java.lang.Integer
- java.lang.Long
- java.lang.Float
- java.lang.Double
- java.lang.Number

- java.util.Date
- java.sql.Date
- java.sql.Time
- java.sql.Timestamp
- java.math.BigDecimal
- java.math.BigInteger
- java.lang.Boolean
- java.lang.Object

Unsupported data types may occur when editing Topics manually, and sometimes with data sources for big data, in particular MongoDB. The connector for MongoDB uses the data type of a given value in the last document containing that value, and errors in input files may cause unexpected types. For example, omitting the single quotes in the JSON format causes a string type to be interpreted as a numeric type.

If your Topic or Domain fields do not appear in the Ad Hoc editor, you can enable logging on the following class to see details of fields with unsupported data types:

com.jaspersoft.ji.adhoc.metadata.AdhocTopicMetadata

For information about enabling logging, see **"Configuring System Logs" on page 199**.

A.4 Field Names Disappear in Ad Hoc Editor

Some fields with international characters in their display names disappear when the field is dragged into the canvas of the Ad Hoc Editor. This is caused by non-Unicode characters used in the field name in the JRXML underlying the selected Topic.

To make the international characters appear in Ad Hoc view field labels, use the resource bundle mechanism:

1. Create a resource bundle (*.properties) and associate each of your field labels with a unique key. Use Unicode escape sequences such as `\u0153` for the œ character to insert international characters in your label values.
2. Use the $R syntax in the Topic to specify the appropriate key for the label of each desired field.
3. Upload the resource bundle as a resource of the Topic.

When you open the Topic in the Ad Hoc Editor, the labels are displayed correctly from the resource bundle.

This method has the advantage that you can create a resource bundle for each language the Topic needs to support, and users see the labels for the locale they set in their browser.

For more information about localizing JasperReports Server, see **"Localization" on page 263**.

A.5 Ad Hoc Filter With All Values Causing Error

When using filters in the Ad Hoc editor, your browser sends lists of values to the server with a POST operation. If you filter a field with tens or hundreds of thousands of distinct values, and then select all values, your browser will send megabytes of data in the POST operation. Some application servers are configured to reject such large input by default.

For example, if you select 100 000 values in an Ad Hoc filter on a default installation on Tomcat, Tomcat will log an error and redirect the user to the JasperReports Server home page. The Tomcat error log may contain the following entry:

```
2013-09-30 15:12:33,847 ERROR errorPage_jsp,http-8080-6:559 - stack trace of
exception that redirected to errorPage.jsp
java.lang.NullPointerException
```

If you apply filters to fields with large numbers of distinct values, make sure your app server is configured to accept large input. The following table shows how to configure Apache Tomcat. For other app servers, refer to your app server's documentation about POST operations.

Configure Apache Tomcat to Accept Large Filter Values
Configuration File
<tomcat>/conf/server.xml

Configure Apache Tomcat to Accept Large Filter Values	
Property	**Description**

`<Connector port="8080" protocol="HTTP/1.1"`
`        connectionTimeout="20000"`
`        redirectPort="8443"`
`        URIEncoding="UTF-8"`
`        maxPostSize="0" />`
	Add the `maxPostSize` parameter to set the number of bytes accepted by the app server; `"0"` indicates there is no limit. For more information, see the Tomcat documentation.

A.6 Ad Hoc Dimensions Too Large

A column or row dimension in an Ad Hoc crosstab is the equivalent of a row group in an Ad Hoc table. The members of a dimension or group are the unique values used to aggregate the data. Some dimensions and groups can have hundreds or thousands of members. Even if the dimensions are collapsed, the internal engine must calculate and store the values for all dimension members and all cells.

Some datasets contain fields that make huge dimensions on the order of tens of thousands or hundreds of thousands of members. These fields are usually meant to be filtered, but if they are inserted into the crosstab before filtering or by accident, they will trigger a large database operation and a large processing load in the Ad Hoc editor. To limit this impact, you can configure a limit to the number of members in any dimension.

The internal process that calculates groups, dimensions, and members is called the categorizer. Change the following setting to reduce this limit and avoid performance issues with large cardinality dimensions.

Reducing the Size Limit for Ad Hoc Dimensions		
Configuration File		
.../WEB-INF/applicationContext-catFactory.xml		
Property	**Bean**	**Description**
`maxMembers`	`baseCategorizer`	By default, the limit is set to 100,000 members per dimension. When the limit is reached, all other members are grouped in a member named "Other." Set this property to a lower value based on your typical data and reporting needs.

In addition, dimensions and groups may be nested on several levels, for example Country, Province, and City. If your row data has 100 countries, and each country has 10 provinces, and each province has 10 cities or towns, there will be 100 x 10 x 10 = 10,000 rows in your full crosstab. If you also have two column dimension, each with 10 members, there will be 100 columns in your crosstab and one million cells when all dimensions are fully expanded. This scenario has several implications:

• Nested dimensions and high cardinality dimensions quickly create huge crosstabs.
• Huge crosstabs have a performance impact and take a long time to display and update.
• Consider whether it's possible for users to actually read and interpret such a large crosstab.
• Avoid dimensions with more than 10 members and avoid nesting many levels on each axis.
• Use filters as input controls instead of hiding and expanding dimensions in the crosstab.

For example, it's unlikely that a user can read the expanded data for more than one country at a time. The large report in this example can be replaced with two reports, one that has only the country dimension and allows the user to compare aggregate values from all countries, and another that displays all provinces and cities for a single country selected by a drop-down filter list. Both reports will run much faster than the single large report, and the user will not be blocked waiting for the report to refresh.

A.7 Custom URLs Not Loading in Dashboards

Dashboards allow you to specify frames that contain web pages loaded from custom URLs specified when designing the dashboard. These URLs can even include parameters from input controls. If the URL takes too long to load, JasperReports Server will display an error message instead of the content.

If you expect the custom URL to take longer than 10 seconds, you can change the default timeout as follows:

Configuring the Dashboard URL Loading Timeout	
Configuration File	
.../scripts/dashboard.designer.js	
Property	**Description**
CUSTOM_URL_IFRAME_TIMEOUT	Time in milliseconds that the server will allow a custom URL to load in a dashboard before displaying an error. The setting takes effect immediately when the file is saved, there is no need to restart the server.

A.8 Print View Not Displaying in Dashboards

If you cannot display the Print view for a dashboard, there might be an issue with the size of the input control values. Input control values are passed as URL parameters on this page, and the application server can limit the length of the URL that includes the parameters.

To avoid this limit and allow large numbers of input control values in dashboard print view, edit the following configuration file or the equivalent in your application server.

Configure Apache Tomcat to Accept Large Filter Values
Configuration File
<tomcat>/conf/server.xml

Configure Apache Tomcat to Accept Large Filter Values	
Property	**Description**

`<Connector port="8080" protocol="HTTP/1.1"`
`            connectionTimeout="20000"`
`            redirectPort="8443"`
`            URIEncoding="UTF-8"`
`            maxPostSize="0"`
`            maxHttpHeaderSize=65535 />`
	Add the `maxHttpHeaderSize` parameter to set the number of bytes accepted in the URL by the app server; `"65535"` is equivalent to 64 KB. For more information, see the Tomcat documentation.

A.9 Scheduler Sending Multiple Emails

In cases where you have a cluster of JasperReports Server instances accessing the same repository, the schedulers in each instance can sometimes conflict and send multiple emails. The behavior depends on the run-time of the scheduled reports, because a long report may cause the scheduler on another node to attempt to run the same report before the first node finishes.

To change this behavior, set the following parameter in <WAR-file>/WEB-INF/js.quartz.base.properties:

```
org.quartz.jobStore.clusterCheckinInterval = 900000
```

In case a job fails on the first node, the check-in interval is meant to ensure that the job runs on a second node after this delay. Because the schedulers do not communicate directly, the second scheduler cannot distinguish between a node that had a failure and a node that is still running a job. The default value corresponds to 15 minutes.

This parameter can be adjusted as follows:
- If you have scheduled reports that take a long time to run, longer than 15 minutes, you may see multiple emails. Increase this parameter to an interval longer than your longest report's expected run-time.
- On the other hand, if you have small reports that finish quickly, the default value means that any scheduler or node problem isn't detected by the other scheduler before 15 minutes. If you have time-critical reports scheduled, you can lower this parameter, but the value should still exceed your longest expected report run-time.

Restart all of your server instances after changing this parameter.

A.10 Scheduler Running Deleted Jobs

In some cases, old versions of JasperReports Server did not delete the scheduled jobs when deleting a report. These jobs cause errors when the scheduler tries to run them, but you can't remove the jobs through the user interface. The server no longer creates such "orphan" jobs, but they may appear again when you upgrade or import a catalog that contains them.

If you accidentally imported orphan jobs, make the configuration change shown below and restart your server.

Automatically Deleting Orphan Jobs		
Configuration File		
.../WEB-INF/applicationContext-report-scheduling.xml		
Entry	**Bean**	**Description**
`autoDeleteBroken UriReportJob`	`quartz Scheduler`	Change this property from `false` (the default) to `true`. Orphan jobs are detected and deleted just before they run, so all orphan jobs will be deleted gradually over time.

A.11 Scheduler Timezones in Excel Output

When scheduling a report, you can specify the timezone that should be applied to the output instead of using the data source's timezone. This works for all outputs except Excel (XLS), both paginated and non-paginated. To overcome this limitation, you must explicitly enable timezone usage in XLS output.

Setting Timezones in Scheduled Excel (XLS) Reports		
Configuration File		
.../WEB-INF/classes/jasperreports.properties		
Property	**Value**	**Description**
`net.sf.jasperreports.export. xls.use.timezone`	`true`	Define this property as `true` to apply timezones in scheduled Excel (XLS) reports.

This property can also be defined on individual elements in the report's JRXML if only certain fields should apply the timezone.

A.12 Charts Not Appearing in Excel Export

When exporting a report to Excel, JasperReports Server usually removes images that decorate the report and that do not fit in the Excel data-centric layout. However, JasperReports Server also converts any charts to images and uses the special property `net.sf.jasperreports.export.xls.ignore.graphics` set to `false` to make the image appear. If your report does not set this property explicitly, the chart images do no appear in your reports when exported to Excel.

If you have a lot of reports with this issue, you can set the property on the server:

Charts Images in Excel Export	
Configuration File	
.../WEB-INF/classes/jasperreports.properties	
Property	**Description**
`net.sf.jasperreports.` `export.xls.ignore.graphics`	By default, this property is set to `true`; in this case, images and chart images from the report do not appear when exported to Excel. Set this property to `false` to make chart images appear in Excel exports.

A.13 Working With Data Sources

When adding a data source to JasperReports Server, several things can cause errors. Start by looking at the following general connectivity issues:

- Check that your database server is available and accepting TCP/IP connections from the host where JasperReports Server is installed.
- Check in your RDBMS that the username and password you're using are correct and have access to the selected database.
- Check for firewalls or network connectivity errors.

Many databases, including MySQL, also require the user grants to include the specific host from which connections are allowed. Otherwise, when testing the JDBC connection, a connection may not be allowed even though the username and password are correct. For more information, refer to the MySQL documentation for setting up users.

An easy way to test connectivity from the server to the database with a particular user is to use a tool such as SQuirreL or another DB query tool to connect to the database from the host of your JasperReports Server instance.

A.13.1 Logging JDBC Operations

You can enable additional logging to help you find the cause of the error. Set any or all of the following loggers in the server settings interface or in the .../WEB-INF/log4j.properties file:

- log4j.logger.com.jaspersoft.jasperserver.api.engine.jasperreports.service.impl.JdbcDataSourceService
- log4j.logger.com.jaspersoft.jasperserver.api.engine.jasperreports.service.impl.JndiJdbcDataSourceService
- log4j.logger.com.jaspersoft.jasperserver.war.action.ReportDataSourceAction
- log4j.logger.com.jaspersoft.commons.datarator.JdbcDataSet
- log4j.logger.com.jaspersoft.jasperserver.war.common.JasperServerUtil
- log4j.logger.com.jaspersoft.commons.semantic.dsimpl.JdbcDataSetFactory
- log4j.logger.com.jaspersoft.commons.semantic.metaapi.impl.jdbc.BaseJdbcMetaDataFactoryImpl
- log4j.logger.com.jaspersoft.jasperserver.war.validation.ReportDataSourceValidator

A.13.2   JDBC Drivers

As of JasperReports Server 5.1, JDBC driver management is automated and simplified. JasperReports Server ships with drivers for some databases, as listed in the dialog for creating data sources. If the JDBC driver for your database is not included, the system administrator can easily upload the driver and use it immediately in a data source.

For instructions on updating JDBC drivers without restarting the server, see **"Managing JDBC Drivers" on page 70**. For instructions on adding JDBC drivers when using JBoss, see the next section.

A.13.3   JDBC Drivers on JBoss

For modularity reasons, the JBoss application server does not allow web applications such as JasperReports Server to change executable files, such as JDBC drivers, on the fly. In order for JDBC drivers to appear in the list of available drivers and be selectable, you must configure JBoss before you use the drivers to create a JDBC data source.

To add JDBC drivers on JBoss:

1.   Download or copy the JDBC driver JAR file to the .../WEB-INF/lib/ directory.

2.   Edit the jboss-deployment-structure.xml file as described in the following table:

Configuring JBoss for JDBC drivers	
Configuration File	
.../META-INF/jboss-deployment-structure.xml	
Property	**Description**
`resource-root path="<driver>.jar"`	Locate the resource root element for the JDBC driver you added and uncomment it. The name of the JAR file must match exactly the name of the JDBC driver that you upload. You can add a new resource root element if the JDBC driver of your choice is not given in the commented list.

3.   Restart JBoss.

A.13.4   Database Permissions

When creating database users, you must ensure that they have the appropriate privileges to access data, as well as permission to connect from the server that JasperReports Server is running on.

•   The database user that you specify in your data source definition should have the appropriate select permissions to query the tables within your database for the reports you want to generate.

•   If you accept the defaults during installation of JasperReports Server on Linux from an RPM using apt-get, rpm, or yum, the bundled PostgreSQL allows only the user who owns PostgreSQL to connect. Enter the following commands to connect:

```
su - postgres
psql -U postgres
```

- Many databases, including MySQL, also require that the user permissions name the specific host from which connections are allowed. Otherwise, when testing the JDBC connection, a connection may not be allowed even though the user name and password are correct. For example, see the MySQL documentation for setting up users.

A fairly easy way to test permissions and connectivity is to use a tool such as SquirrelSQL or another DB query tool to connect to the database from the same host as JasperReports Server and to run typical queries against your database.

A.13.5   JDBC Database URLs

When you choose a JDBC driver, the data source creation wizard prompts you for the elements of the URL that are required for your database. In some cases, you may need to add certain arguments to the JDBC URL. Ensure that the database URL you entered when defining your JDBC data source is consistent with what is required for your specific database and database driver. The following table gives the default URLs and port numbers and examples of optional arguments supported by the most common databases:

Database	Default JDBC Database URL
PostgreSQL	jdbc:postgresql://<host>:5432/<db-name>
MySQL and MariaDB	jdbc:mysql://<host>:3306/<db-name>?useUnicode=true&characterEncoding=UTF-8
Ingres	jdbc:ingres://<host>:II7/<db-name>;CURSOR=READONLY;auto=multi
Oracle	jdbc:oracle:thin:@<host>:1521:orcl
SQL Server	jdbc:sqlserver://<host>:1433;databaseName=<db-name>;SelectMethod=cursor
SQL Server jTDS driver	jdbc:jtds:sqlserver://<host>:1433/<db-name>
DB2	jdbc:db2://<host>:50000/<db-name>:driverType=4;currentSchema=<schema-name>;fullyMaterializeLobData=true;fullyMaterializeInputStreams=true; progressiveStreaming=2;progresssiveLocators=2
Vertica	jdbc:vertica://<host>:5433/<db-name>
Informix	jdbc:informix-sqli://<host>:1526/<db-name>:INFORMIXSERVER=<server-name>
Vertica	jdbc:sybase:Tds:<host>:5433?ServiceName=<service-name>

A.13.6   SQL Functions with TIBCO JDBC Drivers

As of JasperReports Server 5.6.1, TIBCO provides a set of JDBC drivers in the installed server. The TIBCO JDBC drivers are based on the Progress DataDirect Connect drivers and support a slightly different SQL syntax

than the vendors' drivers. If you see errors like the one below when running reports or creating domains that use scalar functions, you'll need to modify your queries.

```
com.jaspersoft.commons.semantic.metaapi.MetaDataException: Cannot execute JDBC Query.
java.sql.SQLDataException: [TIBCO][SQLServer JDBC Driver][SQLServer]Conversion failed
when converting the varchar value 'Assets' to data type int.
```

For example, the following query does not work with the TIBCO JDBC drivers:

```
SELECT account_id+account_description AS account_concat from account
```

The correct syntax for the TIBCO JDBC drivers is as follows:

```
SELECT {fn CONCAT(account_id,account_description)} AS account_concat from account
```

For more information, see the Progress DataDirect page on scalar functions.

A.13.7   JNDI Services on Apache Tomcat

If you have trouble with a JNDI connection, you need to look at the JNDI definition for your database on your application server. This section gives common issues with JNDI definitions on Apache Tomcat connecting to MySQL. If you use a different application server or database server, refer to its documentation.

A JNDI connection on Tomcat is defined in two different files. Make sure both have the following information:

*   <tomcat>/webapps/jasperserver[-pro]/META-INF/context.xml

```
<Resource name="jdbc/<db-name>" auth="Container" type="javax.sql.DataSource"
maxActive="100" maxIdle="30" maxWait="10000"
username="<db-user>" password="<db-user-password>"
driverClassName="org.postgresql.Driver"
validationQuery="SELECT 1" testOnBorrow="true"
url="jdbc:mysql://<host>:3306/<database>?autoReconnect=true&amp;autoReconnect
ForPools=true"/>
```

*   <tomcat>/webapps/jasperserver-pro/WEB-INF/web.xml

```
<resource-ref>
<description>JNDI Example</description>
<res-ref-name>jdbc/<db-name></res-ref-name>
<res-type>javax.sql.DataSource</res-type>
<res-auth>Container</res-auth>
</resource-ref>
```

Also check the following points:
*   Ensure the driver for your database connection is in the <tomcat>/lib folder.
*   If you installed JasperServer from a WAR file, Tomcat may have created a separate copy of context.xml in <tomcat>/conf/Catalina/Localhost/jasperserver-pro.xml. See the corresponding section in the troubleshooting appendix of the *JasperReports Server Installation Guide*.
*   See the Apache Tomcat documentation for JNDI datasources.
*   For Oracle databases, you may need to specify additional parameters in the context.xml file. For example, in order to support in Oracle, add the following line

```
driverClassName="oracle.jdbc.OracleDriver"
validationQuery="SELECT 1 FROM dual"
accessToUnderlyingConnectionAllowed="true"
```

A.13.8   JNDI Services on JBoss

After defining JNDI Services on the JBoss application server, JasperReports Server does not automatically detect the new services. To use the new JNDI services as data sources in the server, follow these steps:

1. Define and deploy a JNDI data source in the JBoss administrator console.
2. Modify the file <jboss>/webapps/jasperserver-pro/WEB-INF/web.xml to include a data source reference to this new JNDI service.
3. Modify jboss-web.xml to include a reference to this data source.
4. Because the deployment configuration files such as web.xml were modified, redeploy the JasperReports Server application.

Now you can define JNDI data source in the repository, as described in **"JNDI Data Sources" on page 73**.

A.13.9   JNDI Services on WebLogic

Follow these steps to configure JasperReports Server to use JNDI data sources with WebLogic:

1. Append the following definition to the `<reference-descriptor>` node of WEB-INF/weblogic.xml:

```
<resource-description>
    <res-ref-name>TestDatabase</res-ref-name>
    <jndi-name>jdbc/testDatabase</jndi-name>
</resource-description>
```

2. Append the following definition to WEB-INF/web.xml:

```
<resource-ref>
    <description>TestDatabase database</description>
    <res-ref-name>TestDatabase</res-ref-name>
    <res-type>javax.sql.DataSource</res-type>
    <res-auth>Container</res-auth>
</resource-ref>
```

3. In the WebLogic Admin Console, add a datasource with TestDatabase as the JNDI name.
4. Restart the jasperserver-pro instance using the WebLogic Admin Console.

A.13.10   Creating a Data Source on SQL Server Using Windows Authentication

If your database is Microsoft SQL Server and you use Windows Authentication (also called Integrated Security), use the following procedure to create a data source.

1. Go to the download page for Microsoft SQL Server JDBC Driver 3.0.

    Do not use version 4.0 of this driver; it will not work.
2. Download and run the self-extracting executable: 1033\sqljdbc_3.0.1301.101_enu.exe
3. Open the extracted folder sqljdbc_3.0\enu\auth, and then either the x64 or the x86 subfolder, depending on whether your version of Windows is 64-bit or 32-bit, respectively.

4. Copy the file sqljdbc_auth.dll to the folder your app server automatically searches for DLLs.

    For Tomcat, this is the <tomcat>\bin folder.

5. Restart your app server.

6. Log into JasperReports Server as and administrator.

7. Select **Create > Data Source** from the main menu.

8. In the Type field, select **JDBC Data Source**. The page refreshes to show the fields necessary for a JDBC data source.

9. Enter a name and optional description for your data source.

10. From the dropdown field, select com.microsoft.sqlserver.jdbc.SQLServerDriver

11. Enter the database hostname and database name of your SQL Server instance.

12. In the URL field, add the following string to the end of the generated URL:

    `;integratedSecurity=true`

13. In the User Name field, enter any non-blank string you want, for example "none."

14. In the Password field, enter any non-blank string you want, for example "none."

15. Set the Time Zone and Save Location fields if necessary.

16. Click **Test Connection** and verify that the connection works.

17. Click **Save** to save the data source in the repository.

A.14 Special Characters in Database Schemas

Some databases allow you to use spaces and special characters in their database schema. These spaces and symbols can thus appear in table names and column names. However, in order to define Domains, JasperReports Server relies on a restricted set of symbols. Databases having table names and column names with forbidden symbols will cause Domain errors and be unusable with Domains.

As of JasperReports Server 6.0, Domains allow the following symbols:

Symbols Allowed in Table Names	Symbols Allowed in Column Names
`<space> @ # $ % & _ /`	`<space> @ # $ % & ( ) _ [ ] : , + - /`

These symbols have been tested with the following databases:
* SQL Server
* DB2
* PostgreSQL

A.15 Hadoop-Hive Reports Not Running

If you created a Hadoop-Hive data source in JasperReports Server 5.6 or earlier, it may not run when imported to a more recent version of the server, for example after an upgrade. The reason is the changes to the Hadoop-Hive connectors.

To update Hadoop-Hive data sources:

1. Create a new Hive data source using the same URL as the old Hadoop-Hive data source used by your reports. If you're using Cloudera 5, create a CDH 5 Hive-Impala data source. For all other hive servers,

create a JDBC data source that uses the Hive JDBC driver. For instructions, see **"Hadoop-Hive Data Sources" on page 81**.

2. Update your Hadoop-Hive-based reports to use the new data source.

3. Delete the old, imported data source.

A.16 Cassandra Reports Not Running

The Cassandra architecture introduces some restrictions that other databases do no have. Certain filters such as is-one-of (IN), <, and > cannot be used with all fields. For example, you may see the following error:

```
Caused by: com.datastax.driver.core.exceptions.InvalidQueryException: Cannot use IN operator on column
not part of the partition key
```

Cassandra clusters use partition keys to split data between nodes, but this imposes a certain segmentation of the data. This segmentation is not compatible with all filtering, because it would be very inefficient. Therefore, the definition of the partition keys in your Cassandra schema limits you to certain comparison operations on certain fields.

In general you should avoid comparison operators other than strict equality in filters on Cassadra data sources. The Cassandra reference documentation recommends *not* using the IN comparison "under most conditions."

For more details, see the Cassandra documentation of the SELECT WHERE clause in CQL (Cassandra Query Language).

A.17 Reverting to the Old Home Page

JasperReports Server 5.5 introduced a new home page with more functionality and access to more features in the server. If your installation or customizations require the old home page with the large buttons, you can specify that the server use the old home page.

1. Open the <js-webapp>/WEB-INF/flows/homeFlow.xml file.

2. Locate the following line:

```
<view-state id="homeView" view="modules/home/home">
```

3. Replace the view value as shown in the following sample:

```
<view-state id="homeView" view="modules/old_home/home">
```

4. Restart the server or redeploy the JasperReports Server web app.

APPENDIX B   LOCALIZATION

By default, JasperReports Server is presented in the English language (US version), but it supports other languages with translations that include data formats and resource bundles. Supported languages are Brazilian Portuguese, Chinese (Simplified), French, German, Italian, Japanese, and Spanish. The translations are included in your JasperReports Server instance by default; to view the application in a specific locale, select it before logging in.

If you need to support a language other than the supported ones, you can localize JasperReports Server, including translating it into a different language by providing labels and messages in the preferred language. For other locales, you may also need to change the default locale and time zone. This chapter describes the procedures and gives a few examples.

For information about localizing Domains, Topics, and reports, refer to the *JasperReports Server User Guide*.

This chapter contains the following sections:
- **Configuring JasperReports Server for Multi-byte Fonts**
- **UTF-8 Configuration**
- **Changing Character Encoding**
- **Creating a Locale**
- **Configuring JasperReports Server to Offer a Locale**

B.1   Configuring JasperReports Server for Multi-byte Fonts

Although translation packs for Chinese and Japanese ship with JasperReports Server, the fonts that it uses by default do not support those languages. Therefore, if your organization requires those fonts, you need to configure JasperReports Server for them.

While the fonts JasperReports Server uses are generally dictated by the JRXML files that define your reports, some font configuration is required for special circumstances. For example, you can configure Jaspersoft OLAP to offer different options in the **Chart Default Font** field in the Chart Options dialog. But note that, in order to use a font, the font must be available to the host's operating system. This section describes steps you may need to take, depending on the functionality you use and the locales you support.

The tasks in this section require you to edit these files:

File Name	Location	Purpose of Edits
jpivot_internal_messages.properties	WEB-INF/internal	Specifying chart fonts for Jaspersoft OLAP Community
Ja_pro_internal_messages.properties	WEB-INF/internal	Specifying chart fonts for Jaspersoft OLAP Professional and Enterprise
userConfig.xml	WEB-INF/jpivot/print	Embedding fonts in PDF

B.1.1   Enabling East Asian Fonts

The default configuration of the Java Runtime Environment (JRE) does not support East Asian fonts. If your locale requires such a font, you need to configure your users' computers for the fonts and update their JRE.

To configure a Microsoft Windows computer (XP and later) for East Asian fonts:

Details of this procedure vary, depending on your version of Windows.

1. In the Control Panel, click **Region and Language**.
2. In the Region and Language dialog, select the **Keyboards and Languages** tab.
3. On the tab, install the language(s) that you need.
4. If necessary, install the related keyboard modifications.
5. Close the control panel.
6. Locate the fontconfig.properties.src file in the C:\Program Files\<JRE_directory>\lib folder.
7. In the file, locate the following line:
   ```
   sequence.allfonts=alphabetic/default,dingbats,symbol
   ```
8. Change the line to include the East Asian fonts that you need, such as the following:
   ```
   sequence.allfonts=alphabetic/default,dingbats,symbol,korean,japanese,chinese-
   ms936,chinese-ms950
   ```
9. At the end of the file, check to be sure that the fonts you selected are listed, as in the following:
   ```
   filename.Gulim=gulim.TTC
   ```
   If the fonts are not listed, add them.
10. Save and close the file.
11. Rename the file to fontconfig.properties

B.1.2   Configuring Ad Hoc Charts for Asian Fonts

The default font for the legend of an Ad Hoc chart does not support some Asian characters, such as Japanese kana glyphs. Default fonts for Ad Hoc reports are defined in the following file:

Changing Ad Hoc Chart Legend Fonts		
Configuration File		
.../WEB-INF/adhoc/themes/default.new.jrxml		
Property	**Value**	**Description**

`style name=`
`    "ChartLegend"`
`fontName=`
	`"DejaVu Sans"`
	<default>
	
	`"SansSerif"`
		The default value, DejaVu Sans, does not include kana glyphs, and thus the legend contains blank symbols.
		
		If the server machine has system Japanese fonts available, setting the value to SansSerif should work in most Java font configurations.

> After making changes to the default.new.jrxml file, you must rerun all Ad Hoc reports that contain Japanese characters for them to appear.

If Japanese fonts are not installed in the server machine, you can add it in the following manner:

1. Package a Japanese font as a font extension JAR.
2. Add the new JAR to the WEB-INF/lib folder of the JasperReports Server WAR file.
3. Edit the default.new.jrxml file as described above to specify the new font name.
4. Redeploy JasperReports Server or restart its app server.
5. Rerun the affected reports. Reports saved from Ad Hoc should now display correctly in the Report Viewer, but the corresponding view may not display chart legends correctly within the Ad Hoc editor.

B.1.3   Configuring OLAP Options for Chart Default Fonts

If you implement Jaspersoft OLAP and support a locale with special font requirements, you can configure Jaspersoft OLAP to offer different options in the **Chart Default Font** field in the Chart Options dialog of the OLAP view. This may be necessary if you implement locales that Latin 1 doesn't support.

An OLAP view's Chart Options dialog includes the **Chart Default Font** field, which allows users to select the font to use in charts. The default options are SansSerif, Serif, and MonoSpaced. JasperReports Server reads these values from a properties file and attempts to map them to fonts available in the server host's operating system. You can configure the server to offer different fonts if these fonts don't support the locales you implement.

To change the Chart Default Font field's options:

1. Save the jpivot_internal_messages.properties file with a new name that reflects the new locale. For example, for Japanese, the new file would be called jpivot_internal_messages_ja.properties.
2. Open the new file and locate the following keys:

```
JAJ_000_jsp.jpivot.chartpropertiesform.sansSerif=SansSerif
JAJ_000_jsp.jpivot.chartpropertiesform.serif=Serif
JAJ_000_jsp.jpivot.chartpropertiesform.monospaced=Monospaced
```

If you are using Jaspersoft OLAP Community Edition, the name of the file and the keys that you edit are different. For the Community Edition, open the jpivot_internal_message.properties file and edit these keys:

```
jsp.wcf.chart.sansserif=SansSerif
jsp.wcf.chart.serif=Serif
jsp.wcf.chart.monospaced=Monospaced
```

3. Change one or more of the strings to the name of a font available in the host's operating system. For example, if you wanted to change the SansSerif font to the SimHei font, edit the value specified by `jsp.wcf.chart.sansserif`. For example:

```
jsp.wcf.chart.sansserif=SimHei
```

4. Save the file.
5. Restart JasperReports Server.

B.1.4   Embedding Fonts in PDF Output

By default, JasperReports Server can create PDF (Portable Document Format) files with many different fonts. However, if you experience font problems in the PDF output, you may need to take the steps described in this section to make the fonts available to JasperReports Server's XSL Formatting Object (XSL-FO) processor.

You must have distribution rights to a font in order to embed it in a PDF file.

When users save reports in PDF format, JasperReports Server generates the PDF output using Apache FOP (Formatting Objects Processor). In order for FOP to render fonts properly, you must install the font itself (for example, a TTF file) on the server host, create a font metrics file (using Apache's `org.apache.fop.fonts.apps.TTFReader` utility), and update the userConfig.xml file to associate the font with its metrics. For more information, refer to the Apache FOP documentation.

You can embed any Unicode font using this procedure, though larger font files may have significantly larger memory footprints. To keep memory requirements small, we recommend you use the smallest font file you can, such as SimHei to support Chinese, Japanese, and Korean.

B.2   UTF-8 Configuration

JasperReports Server uses UTF-8 (8-bit Unicode Transformation Format) character encoding. If your database server or application server uses a different character encoding form, you may have to configure them to support UTF-8. This section provides information for configuring the character encoding for several application servers and database servers. If you use a different application server or database, and its default character encoding isn't UTF-8, you may need to make similar updates to support certain locales. For more information, refer to the documentation for your application server or database.

B.2.1   Tomcat

By default, Tomcat uses ISO-8859-1 (ISO Latin 1) character encoding for URIs, which is sufficient for Western European locales, but does not support many locales in other parts of the world.

If you plan to support locales that Latin 1 does not support, you must change Tomcat's URI encoding format.

> If you chose the instance of Tomcat that is bundled with the installer, you don't need to make this change. The bundled Tomcat is pre-configured to support UTF-8. If you installed the WAR file distribution with your own instance of Tomcat and want to support UTF-8, follow this procedure.

To configure Tomcat to support UTF-8:

1. Open the conf/server.xml file and locate the following code:

```
<!-- Define a non-SSL HTTP/1.1 Connector on port 8080 -->
<Connector>
    port="8080" maxHttpHeaderSize="8192"
    maxThreads="150" minSpareThreads="25" maxSpareThreads="75"
    enableLookups="false" redirectPort="8443" acceptCount="100"
    connectionTimeout="20000" disableUploadTimeout="true"
</Connector>
```

2. At the end of this section, insert the following line before the closing tag:

```
URIEncoding="UTF-8"
```

3. For example, after your changes, the section might read:

```
<!-- Define a non-SSL HTTP/1.1 Connector on port 8080 -->
    <Connector>
    port="8080" maxHttpHeaderSize="8192"
    maxThreads="150" minSpareThreads="25" maxSpareThreads="75"
    enableLookups="false" redirectPort="8443" acceptCount="100"
    connectionTimeout="20000" disableUploadTimeout="true"
    URIEncoding="UTF-8"
</Connector>
```

4. Save the file.
5. Restart Tomcat.

B.2.2   JBoss

Because JBoss uses Tomcat as its web connector, the configuration changes in **"Tomcat" on page 266**) also have to be made for JBoss. The only difference is that the server.xml file is located in the Tomcat deployment directory, typically server/default/deploy/jbossweb-tomcat55.sar. Make the same configuration changes, then restart JBoss.

B.2.3   PostgreSQL

JasperReports Server requires PostgresSQL to use UTF-8 character encoding for the database that stores its repository as well as for data sources. A simple way to meet the requirement is to create the database with a UTF-8 character set. For example, enter the following command:

```
create database jasperserver encoding='utf8';
```

B.2.4   MySQL

By default, MySQL uses ISO-8859-1 (ISO Latin 1) character encoding. However, JasperReports Server requires MySQL to use UTF-8 character encoding for the database that stores its repository as well as for data sources. The simplest way to meet the requirement is to create the database with a UTF-8 character set. For example, enter the following command:

```
create database jasperserver character set utf8;
```

To support UTF-8, the MySQL JDBC driver also requires that the `useUnicode` and `characterEncoding` parameters be set as in this startup URL:

url="jdbc:mysql://localhost:3306/jasperserver?useUnicode=true&characterEncoding=UTF-8"

If the MySQL database is a JNDI data source managed by Tomcat, such as the JasperReports Server repository database, the parameters can be added to the JDBC URL in WEB-INF/context.xml. The following is a sample resource definition from that file:

```
<Resource name="jdbc/jasperserver" auth="Container" type="javax.sql.DataSource"
    maxActive="100" maxIdle="30" maxWait="10000"
    username="root" password="password" driverClassName="com.mysql.jdbc.Driver"
    url="jdbc:mysql://localhost/jasperserver?useUnicode=true&amp;characterEncoding=UTF-8" />
```

JBoss ignores the context.xml file, instead requiring an XML file to define JNDI data sources in the deployment directory, which is typically server/default/deploy. The following is an example of a resource definition in one of those XML files:

```
<local-tx-datasource>
    <jndi-name jdbc/jasperserver />
    <connection-url>
        jdbc:mysql://localhost/jasperserver?useUnicode=true&amp;characterEncoding=UTF-8
    </connection-url>
    <driver-class com.mysql.jdbc.Driver />
    <user-name jasperadmin />
    <password jasperadmin />
    <min-pool-size 5 />
    <max-pool-size 20 />
    <idle-timeout-minutes 0 />
    <metadata>
        <type-mapping mySQL />
    </metadata>
</local-tx-datasource>
```

If the database is a JDBC data source configured in the repository, change the JDBC URL by editing the data source in the JasperReports Server repository. The following is an example of the JDBC URL (note that the ampersand isn't escaped):

```
jdbc:mysql://localhost:3306/foodmart_ja?useUnicode=true&characterEncoding=UTF-8
```

B.2.5   Oracle

Oracle databases have both a default character set and a national character set that supports Unicode characters. Text types beginning with N (NCHAR, NVARCHAR2, and NCLOB) use the national character set. As of JasperServer 1.2, all the text data used by the JasperReports Server repository (when stored in Oracle) is stored

in NVARCHAR2 columns, so that JasperReports Server metadata can use the full Unicode character set. For more information about Unicode text support, refer to the Oracle white paper (PDF).

To work properly with Unicode data, the Oracle JDBC driver requires you to set a Java system property by passing the following argument to the JVM:

```
-Doracle.jdbc.defaultNChar=true
```

In Tomcat, add the variable to JAVA_OPTS in bin/setclasspath.sh (Linux) or bin/setclasspath.bat (Windows):

1. Locate the following line in the script:

| Linux | # Set the default -Djava.endorsed.dirs argument |
| Windows | rem Set the default -Djava.endorsed.dirs argument |

2. Add the following line before it:

| Linux | JAVA_OPTS="$JAVA_OPTS "-Doracle.jdbc.defaultNChar=true |
| Windows | set JAVA_OPTS=%JAVA_OPTS% -Doracle.jdbc.defaultNChar=true |

Because JBoss also uses JAVA_OPTS to pass options to the JVM, you can add the same JAVA_OPTS line to bin/run.sh (Linux) and bin/run.bat (Windows). Add it before this line:

| Linux | # Setup the java endorsed dirs |
| Windows | rem Setup the java endorsed dirs |

B.3   Changing Character Encoding

Depending on the third-party software you use and the locales you support, you may also have to configure JasperReports Server and its host. The steps described in this section are necessary only under certain circumstances, such as if you plan to use a character encoding form that UTF-8 cannot handle.

To use a character encoding form other than UTF-8, you must configure JasperReports Server, your application server, and your database server.

The tasks in this section require you to edit the following file:

File Name	Location	Purpose of Edits
applicationContext.xml	WEB-INF	Changing character encoding

B.3.1   Configuring JasperReports Server

To configure JasperReports Server for a different encoding form, you must edit the applicationContext.xml file.

To specify a different encoding form:

1. Open the applicationContext.xml file and locate the following bean. It's configured for UTF-8:

```
<bean id="encodingProvider"
      class="com.jaspersoft.jasperserver.api.common.util.
      StaticCharacterEncodingProvider">
   <constructor-arg value="UTF-8" />
</bean>
```

2. Change "UTF-8" to the encoding type your database server and application server use. For example:

```
<bean id="encodingProvider"
      class="com.jaspersoft.jasperserver.api.common.util.
      StaticCharacterEncodingProvider">
   <constructor-arg value="UTF-16" />
</bean>
```

3. Save the file.
4. Restart JasperReports Server.

B.3.2 Configuring the Application Server and Database Server

If you want to use a character encoding other than UTF-8, you may need to configure the third party software that JasperReports Server relies on. For more information, refer to the documentation for your application server and database server. For Tomcat, you can specify a different character encoding by following steps similar to those described in **"Tomcat" on page 266** and **"UTF-8 Configuration" on page 266**.

> This step is necessary only if you plan to support locales that require a different character encoding, such as UTF-16. In addition to this change, your application server and database must be configured to use the character encoding you require. For more information, refer to the documentation for your third party software.

B.3.3 Configuration for Localized Analysis Schemas

If you plan to use localized OLAP views, you must take additional steps to configure JasperReports Server.

To configure JasperReports Server for localized OLAP views:

1. Every Unicode database that JasperReports Server interacts with (whether it's the repository database or a database accessed through a data source defined in JasperReports Server) must be created to support UTF-8. For example, to create the Foodmart database in PostgreSQL, you might give a command similar to the following:

```
create database foodmart_ja encoding='utf8';
```

2. The URL of any OLAP data source that JasperReports Server accesses must be properly configured in the /ji-pro/META-INF/context.xml file. For example, the URL definition for the Foodmart sample database might be similar to the following:

```
<Resource name="jdbc/MondrianFoodMart_ja"
    auth="Container" type="javax.sql.DataSource"
    maxActive="100" maxIdle="30" maxWait="10000"
    username="postgres" password="postgres" driverClassName="org.postgresql.Driver"
    url="jdbc:postgresql://localhost:5432/foodmart_ja" />
```

3. Encoding options must be added to the JDBC connection string for any data source that points to an OLAP database. For example, when creating a data source in JasperReports Server that points to an OLAP database, use the following connection string:

```
jdbc:postgresql://localhost:5432/foodmart_ja
```

B.4   Creating a Locale

When you want to create other locales for JasperReports Server, translation is only one aspect of localization. Creating a locale includes these tasks:

- Translating labels and messages
- Changing date formats
- Changing format masks

The tasks in this section require you to edit these files:

File	Location	Purpose of Edits
*.properties files	WEB-INF/bundles buildomatic/conf_source/iePro/bundles	Translating labels and messages
jasperserver_ config.properties	WEB-INF/bundles	Changing date formats
adhoc_masks	WEB-INF/bundles	Changing format masks

B.4.1   About Properties Files

Resource bundles for JasperReports Server and Jaspersoft OLAP are Java properties files found in the .../WEB-INF/bundles directory. The properties files contain all the labels and messages used in JasperReports Server and Jaspersoft OLAP.

A bundle includes a default locale (for example, jasperserver_messages.properties), which is written in U.S. English. Then it consists of all the properties files with the same base name, but different locale (such as jasperserver_messages_fr.properties). Each file translates all of the strings of the default file into the language given by the locale. The Java programming language has rules for specifying locales and alternate locales and determining which locale in the bundle to use.

Default Resource Bundles in JasperReports Server	
File in .../WEB-INF/bundles	**Description**
AdHocFiltersBundle.properties	Labels and messages for the Ad Hoc Filters panel.
adhoc_masks.properties	Masks (data formats) for values appearing in the Ad Hoc Editor.
adhoc_messages.properties	Labels and messages for the Ad Hoc Editor.
calendar.properties	Labels and messages used by the pop-up calendar dialog.
CommonBundle.properties	Bundle for tests.

Default Resource Bundles in JasperReports Server	
File in .../WEB-INF/bundles	**Description**
createsldatasource_messages .properties	Labels and messages for the Create Domain dialog.
domain_designer_messages .properties	Labels and messages for the Domain Designer UI.
HomeBundle.properties	Labels for the new home page introduced in version 5.5.
image_descriptions_messages .properties	Labels and messages for the AWS (Amazon Web Services) machine images.
jasperreports_highcharts_ messages.properties	Messages for charts in the report viewer.
jasperreports_messages .properties	Labels and messages for the report viewer.
jasperserver_config.properties	Configuration properties for dates and date-times.
jasperserver_messages.properties	Labels and messages used in the main JasperReports Server user interface.
jsexceptions_messages.properties	Messages used in errors and exceptions, both in the UI and in the log messages.
LicenseMessages.properties	Labels and messages used when validating licenses.
logger_descriptions.properties logger_descriptions_pro.properties	Internal log messages.
pro_nav_messages.properties	Labels and messages for the menu bar and old Home page.
querybuilder_messages.properties	Labels and messages for the Choose Data dialog (for creating a Domain Topic before using Ad Hoc editor).
report_option_messages .properties	Labels and messages for the report options dialog.
ScalableInputControlsBundle .properties	Labels for lists of values in input controls.
scheduling_ws.properties	Validation messages for the report scheduler.
security.properties	Messages used by input validation.

Default Resource Bundles in JasperReports Server	
File in .../WEB-INF/bundles	**Description**
semanticLayer.properties	Labels and messages for the Domain designer and Ad Hoc data policies.
ValidationBundle.properties	Messages for dashboard dialogs.

Default Resource Bundles in Jaspersoft OLAP	
File in .../WEB-INF/bundles	**Description**
ja_mondrian.properties	Labels and messages in the OLAP settings UI.
ja-pro_messages.properties	Labels and messages in the commercial edition OLAP viewer.
jpivot_messages.properties	Labels and messages in the community edition OLAP viewer.
mondrian_exception_messages .properties	MDX validation error messages specific to the internal analysis engine.

The standalone import and export tools have their own bundles located outside of the web app. The following bundles are located in the WAR file distribution package, under the buildomatic folder. This location also holds a copy of all bundles listed in the table above.

Default Resource Bundles for js-export.sh and js-import.sh	
File in buildomatic/conf_ source/iePro/bundles	**Description**
ji-export-messages.properties	Labels and messages for js-export --help.
multi-tenancy-export-messages.properties	Labels and messages for export in commercial editions.
multi-tenancy-import-messages.properties	Labels and messages export in commercial editions.

If you use the JasperReports Server portlet to display JasperReports Server content in a portal (such as Liferay), the deployed portlet includes properties files, as well:

File	Location	Description
jaspersoft_portlet_message.properties	WEB-INF\bundles under the location where the portlet is deployed. For example: C:\liferay\webapps\<portlet_context_name>\WEB-INF\bundles (where <portlet_context_name> is the name specified when the JasperReports Server portlet was deployed).	Labels and messages that appear in the portlet, including the help text. Note that this does not include text in specific reports; to localize the report's content, you must upload its resource bundle to the repository.

Jaspersoft iReport Designer (iReport) and the iReport Plug-in for JasperReports Server (iReport plug-in) have their own resource bundles, including:

Component	File	Description
iReport	ireport.properties	Labels and messages used in Jaspersoft iReport Designer.
iReport plug-in	irplugin.properties	Labels and messages used in iReport Plug-in for JasperReports Server.

B.4.2 Creating a Resource Bundle

Create a resource bundle by making a copy of each *.properties file, using the following syntax for the copy's file name:

```
<default_file_name>_<locale>.properties
```

where

`<default_file_name>` is the name of the default version of the properties file, and

`<locale>` is a Java-compliant locale identifier.

For example, consider the core JasperReports Server resource bundle. For various locales, it might be named as follows:

File Type	File Name
Default resource bundle	jasperserver_messages.properties
English	jasperserver_messages_en.properties
French	jasperserver_messages_fr.properties
French in Switzerland	jasperserver_messages_fr_CH.properties

For a list of Java-compliant locales, please refer to the Java web site.

The resource bundles described in this document consist of locale-specific Java properties files. Java properties files use the ISO-8859-1 (Latin-1) encoding that is the same as ASCII for all English non-accented characters. For international characters that are not in ISO-8859-1, use Unicode escape sequences (for example `\u00e9` is é).

To create a new JasperReports Server resource bundle:

1. Copy each of the properties files (keeping them in the same directory as the originals) and rename them according to your locale.
2. Translate each *.properties file's labels and messages into the new language.

    Some of the strings in the properties files are date formats and format masks that may need to be edited for the new locale. For more information, refer to **"Setting Date and Datetime Formats" on page 275**.
3. Save the files.
4. If the new locale requires specific character encoding or fonts, ensure that JasperReports Server and the third party software it relies on are configured to support them. For more information, refer to **"Configuring JasperReports Server for Multi-byte Fonts" on page 263**.

The new locale is not available in JasperReports Server until you follow the steps described in **"Specifying Additional Locales" on page 277**.

B.4.3   Setting Date and Datetime Formats

Each locale may have its own rules for displaying dates and datetime values. System date and datetime formatting is controlled by four patterns that are specified in the jasperserver_config_<locale>.properties file associated with a particular locale.

For example in the English resource bundle, the four entries are:

```
date.format=dd-MM-yyyy
datetime.format=dd-MM-yyyy HH:mm
calendar.date.format=%d-%m-%Y
calendar.datetime.format=%d-%m-%Y %H:%M
```

The first two keys are used to parse and format dates and datetime values using an internal `java.util.DateFormat` object across the whole application. These patterns should be non-localized date patterns, in accordance with the Java Development Kit (JDK) syntax.

The other two keys are used by the calendar control, which formats the user-selected date and datetime values in accordance with its own pattern syntax.

To change the system date and datetime formatting for a new locale, edit the strings specified by these keys.

B.4.4   Setting Data Format Masks

Each locale may have its own format masks that determine how numbers appear in the Ad Hoc Editor. To make the data format masks vary by locale, you must create an adhoc_masks file for the new locale. To do so, copy the file adhoc_masks.properties to a new name that specifies the new locale and change the masks defined in the new file. For example, the French file would be named adhoc_masks_fr.properties.

> The data format masks described in this section are used in the Domains and in the Ad Hoc Editor; they appear in Ad Hoc views as well as JRXML reports based on Domains. They're not applicable to Jaspersoft OLAP.

Customize the available data format masks for dates, integers, and decimals by editing the existing masking entries or adding new ones. The default entries are given in the following table:

Data Format Mask Properties	Appearance in en_US Locale
ADH_100_MASK_date_0 = short,hide ADH_100_MASK_date_1 = long,hide ADH_100_MASK_date_2 = short,medium ADH_100_MASK_date_3 = medium,medium ADH_100_MASK_int_0 = #,##0 ADH_100_MASK_int_1 = 0 ADH_100_MASK_int_2 = $#,##0;($#,##0) ADH_100_MASK_int_3 = #,##0;(#,##0) ADH_100_MASK_dec_0 = #,##0.00 ADH_100_MASK_dec_1 = 0 ADH_100_MASK_dec_2 = $#,##0.00;($#,##0.00) ADH_100_MASK_dec_3 = $#,##0;($#,##0)	3/31/09 Mar 31, 2009 March 31, 2009 Mar 31, 2009 23:59:59 -1,234 -1234 ($1,234) (1234) -1,234.56 -1234 ($1,234.56) ($1,234)

The data format masks for each type are numbered consecutively from zero; create new masks by adding new entries. The keys of the new entries must follow the convention established in the default entries. For example, a new decimal data format mask might have this ID:

```
ADH_100_MASK_dec_4
```

Date format masks are implemented using `java.text.SimpleDateFormat` and JasperReports extensions that provide access to predefined localized data format masks. New datetime masks must be specified in one of the following formats:

- A style for the date part of the value and a style for the time part (separated by comma) or a single style for both parts. A style is one of Short, Medium, Long, Full, Default (which correspond to `java.text.DateFormat` styles) and Hide.
- A pattern that can be supplied to `java.text.SimpleDateFormat`. In this case, internationalization support is limited.

Both integer and decimal data format masks are implemented with `java.text.DecimalFormat`, which localizes characters in the format specification. For example, consider the case of the digit grouping symbol (thousands separator): in French, it is a space; in U.S. English, it is a comma. `DecimalFormat` handles both cases: if the number pattern #,##0 is used, the number 6000 appears as 6 000 in the French locale and as 6,000 in the U.S. English locale.

For more information about Java's handling of decimal and date format masks, see:
- http://download.oracle.com/javase/6/docs/api/java/text/DecimalFormat.html
- http://download.oracle.com/javase/6/docs/api/java/text/DateFormat.html

> By default, monetary values in Ad Hoc views are masked as USD (United States Dollars). Depending on your data, you may need to support a different currency, support more than one currency, or support currency conversion. These are three very different cases:
>
> • Supporting a different currency than USD involves changing the monetary masks to use the correct symbol for your currency (for example, replace the `$` symbol in the `ADH_100_MASK_dec_2` and `ADH_100_MASK_dec_3` masks). However, changing this symbol does not actually convert currencies in your reports.
>
> • Supporting other currencies in addition to USD involves adding new masks. However, adding data formats does not actually convert currencies in your reports.
>
> • Supporting currency conversion is more complicated; you must consider such issues as fluctuations in conversion rates. Oftentimes, a third-party service can be used to perform currency conversion

B.5   Configuring JasperReports Server to Offer a Locale

After creating a locale, you must configure JasperReports Server to offer it to your users, along with any new time zones.

The tasks in this section require you to edit these files:

File Name	Location	Purpose of Edits
applicationContext-security.xml	WEB-INF	Specifying additional locales
jasperserver-servlet.xml	WEB-INF	Specifying additional time zones

B.5.1   Specifying Additional Locales

By default, JasperReports Server appears in the locale selected in the end user's browser. The Login page allows users to specify the locale they want to use. The list of available locales is defined in applicationContext-security.xml. Edit this file to add a new locale.

To add a new locale:

1. Edit the applicationContext-security.xml file and locate the bean named `userLocalesList`. For example:

```
<bean id="userLocalesList"
        class="com.jaspersoft.jasperserver.war.common.LocalesListImpl">
    <property name="locales">
        <list>
            <value type="java.util.Locale">en</value>
            <value type="java.util.Locale">fr</value>
            <value type="java.util.Locale">it</value>
            <value type="java.util.Locale">de</value>
            <value type="java.util.Locale">ro</value>
            <value type="java.util.Locale">ja</value>
            <value type="java.util.Locale">zh_TW</value>
        </list>
    </property>
</bean>
```

2. Add the new locale to the end of the list. For example, add the following line for Dutch (Java's nl_NL locale):

```
<value type="java.util.Locale">nl_NL/value>
```

3. Save the file.

4. Restart JasperReports Server, and log into the web application to test your translation. Reviewing the translated strings in context can help you improve your word choices.

For a list of Java-compliant locales, please refer to the Java web site.

B.5.2   Specifying Additional Time Zones

By default, JasperReports Server assumes the user's time zone is that of the JasperReports Server host. The Login page allows users to choose a different time zone. The available list is defined in applicationContext.xml file.

To add a time zone:

1. Open the applicationContext.xml file and locate the `userTimeZonesList` bean. For example:

```
<bean id="userTimeZonesList"
        class="com.jaspersoft.jasperserver.war.common.JdkTimeZonesList">
   <property name="timeZonesIds">
      <list>
           <value>America/Los_Angeles</value>
           <value>America/Denver</value>
           <value>America/Chicago</value>
           <value>America/New_York</value>
           <value>Europe/London</value>
           <value>Europe/Berlin</value>
           <value>Europe/Bucharest</value>
      </list>
   </property>
</bean>
```

2. Add the new time zone to the bottom of the list. Specify each time zone as the standard Java time zone values so that JasperReports Server adjusts for daylight savings time when appropriate. For example, add the following line for Tokyo:

```
<value>Asia/Tokyo</value>
```

3. Save the file.

4. Restart JasperReports Server.

For more information about Java-complaint time zones, please refer to the Java web site.

B.5.3   Setting a Default Time Zone

If you want JasperReports Server to use a time zone other than the host computer's, you can set a specific time zone in Java. It becomes the default time zone for all users, but they can still select a different time zone when they log in.

To set a default time zone, set the `user.timezone` property in the JVM as shown in the tables below. Locate the file containing JVM settings for your platform and application server. The value for the property must be a Java-compliant time zone, for example, `Europe/Bucharest`.

You must restart your application server for this setting to take effect. The time zone is set for all applications in your application server, including JasperReports Server.

JVM Settings for Default Time Zone			
Operating System	**App Server**	**File**	**Setting**

Windows	Tomcat	\<apache-tomcat\>\bin\setenv.bat	Add this line of code:
			`set JAVA_OPTS=%JAVA_OPTS%`
			
			`-Duser.timezone=<timezone>`

	JBoss	\<jboss\>\bin\run.bat	

Linux	Tomcat	\<apache-tomcat\>/bin/setenv.sh	Add this line of code:
			`export JAVA_OPTS="$JAVA_OPTS`
			
			`-Duser.timezone=<timezone>"`

	JBoss	\<jboss\>/bin/run.sh	
Both	GlassFish	\<glassfish\>/domains/domain1/config/domain.xml	Add this line of code to \<jvm-options\> section: `-Duser.timezone=<timezone>`

INDEX

U

Unicode Transformation Format (UTF-8). See character
encoding. 266
Universal Resource Identifiers (URI) 56
user attributes
and auditing 231
and query-based input controls 107
User attributes 35
user interface samples 133
users
administering 17, 19, 32
authenticating 19-20, 57
changing passwords 157
default users 32
logging in as 62
object-level permissions 60
roles 38
searching for 33
session timeout 158

V

version of the software 23
views 21
virtual data sources 86

W

web services 233

X

XHTML exporter 215