Ariba Analysis
Configuration Guide

Version 3.0
May 2004

Table of Contents

Preface

The *Ariba Analysis Configuration Guide* describes how you can tailor Ariba Analysis to meet the specific analytical needs of your company to best manage the effectiveness of its spend. The book also describes the mechanics of configuring and administering Ariba Analysis. It includes the following topics:

Audience and Prerequisites

The intended audience of this book is anyone who configures or administers Ariba Analysis:

- System administrators

Anyone administering Ariba Analysis should be familiar with the following:

- Using Web browsers and HTML pages
- How to install software on Windows NT or Windows 2000, Sun Solaris, HP-UX, or IBM AIX
- System administration of those operating systems
- Administration of Web servers, application servers, and databases

This book is **not** a replacement for the documentation from the supplier of your Web, application, or database server. The book assumes that you already have such documentation or know how to obtain it.

Integrating Ariba Analysis with ASM Applications

See the *Ariba Spend Management Integration Guide* for details about integrating Ariba Analysis with other Ariba Spend Management applications.

Information contained in the *Ariba Spend Management Integration Guide* includes how to integrate Ariba Analysis in a conventional, simple configuration and specific procedures for integrating earlier releases of other Ariba Spend Management applications with Ariba Analysis. Also included is information about programming the Ariba Unified Dashboard.

Understanding Ariba Analysis

Implementing Ariba Analysis may involve many people:

- System administrators to maintain the Ariba Analysis computers and software
- Database administrators to make sure the databases powering Ariba Analysis run smoothly
- Data warehouse designers, whose task is to make the components of your company's analytical reports useful
- Business analysts and managers, the end-users of Ariba Analysis

Before customizing Ariba Analysis, you must understand the users of this software, their goals, and their needs. One prerequisite to this understanding is becoming familiar with how Ariba Analysis itself works: how to create an analytical report and use it to investigate spend data. At a minimum, you should look over the end-user documentation to gain this familiarity. This book assumes that you know the definitions and concepts presented in the following:

- The Ariba Analysis Quick Tours
- The printed book for users, *Ariba Analysis Advanced User Guide*

Typographic Conventions

The following table describes the typographic conventions used in this document:

Typeface or Symbol	Meaning	Example
AaBbCc123	Text you need to change is italicized.	`http://server:port/app/inspector`
AaBbCc123	The names of user interface controls, menus, and menu items.	Choose **Edit** from the **File** menu.
`AaBbCc123`	Files and directory names, parameters, fields in CSV files, command lines, and code examples.	There is one line in `ReportMeta.csv` for each report in the system.
AaBbCc123	The names of books.	For more information, see the *Ariba Analysis Configuration Guide*.

Ariba Technical Support

For assistance with Ariba products, Ariba Technical Support is available by phone, email, or over the Web. For information on how to contact Ariba Technical Support, refer to the following page on the Ariba Technical Support Website:

http://connect.ariba.com/TechSupport_Contacting.htm

Ariba Analysis Documentation

The Ariba Analysis documentation set contains the following books.

Title	Audience	Purpose and contents
Ariba Analysis Installation Guide	System and database administrators	**Planning and installing Ariba Analysis** • Architectural overview, software components, deployment configurations • Step-by-step installation
Ariba Analysis Configuration Guide	System and database administrators	**Configuring and administering Ariba Analysis** • Configuration files, command reference, parameters, administration console • System security • Managing users and reports
Ariba Analysis Customization Guide	Systems integrators and data warehouse designers	**Customizing Ariba Analysis** • Design goals, methodologies, and best practices • Working with and extending the data model. Adding facts, measures, dimensions, and materialized views. • Controlling data visibility • Tailoring the user interface
Ariba Analysis Data Load Guide	Data warehouse designers and database administrators	**Loading data from Ariba applications and external systems into Ariba Analysis** • Configuration files • Data-loading metadata XML • command reference
Ariba Analysis Advanced User Guide	Procurement and sourcing business analysts, systems integrators	**Setting-up Ariba Analysis for daily use** • Ariba Analysis prepackaged report models • Designing compound reports, multi-source reports, and template dashboards by role • Customizing Microsoft Excel templates for use with Ariba Analysis • Overview to Ariba integrated Supplier Performance Management

Chapter 1
Overview

This chapter is a high-level overview of how to configure and administer Ariba Analysis. It contains the following sections:

Essential Guides

You will get the most from the *Ariba Analysis Configuration Guide* by familiarizing yourself with some other essential guides:

These guides are all available from Connect at the following location:

`https://Connect.Ariba.com/`

Other Ariba Analysis Guides

Use this guide in conjunction with the *Ariba Analysis Customization Guide* and *Ariba Analysis Data Load Guide*. The three go hand-in-hand.

The *Ariba Analysis Customization Guide* is for a data warehouse designer or programmer who modifies Ariba Analysis to meet an organization's specific, custom needs. It explains the basic concepts of Ariba Analysis, in particular, OLAP, and metadata XML. The *Ariba Analysis Data Load Guide* describes all aspects of data loading. You need to be familiar with these topics to configure and administer Ariba Analysis effectively.

Ariba Spend Management Application and Web Server Guide

You might find it helpful to keep this guide on hand. It describes the Ariba-specific aspects of configuring application and web servers.

Ariba Spend Management Integration Guide

To see how Ariba Analysis can be integrated with other Ariba applications, refer to the *Ariba Spend Management Integration Guide*.

Overview of Administrative Tasks

The following table categorizes the tasks of an administrator of Ariba Analysis and correlates them to chapters in this guide.

Task	Pertinent Documentation
1 Maintain Web server and application server hardware and software. Keep up with server housekeeping, such as log file rotation and configuration updates	Chapter 2, "Components of Ariba Analysis"
2 Start and stop Ariba Analysis	"Starting and Stopping Ariba Analysis" on page 21
3 Enforce proper security	"Securing Ariba Analysis as an Administrator" on page 24
4 Schedule and run tasks to load data into Ariba Analysis from Ariba applications and ERP systems	*Ariba Analysis Data Load Guide*

Task	Pertinent Documentation
5 Support data warehouse designers in readying Ariba Analysis for use in production by end-users	Chapter 5, "Managing Users, Reports, and Downloads"
6 Set language preference, configure currencies, work with the online help	Chapter 6, "Miscellaneous Configuration"
7 Use the Ariba Analysis administrative console for common tasks	Chapter 7, "Ariba Analysis Administrative Console"
8 Move Ariba Analysis from development into production	"Moving Ariba Analysis into Production" on page 56

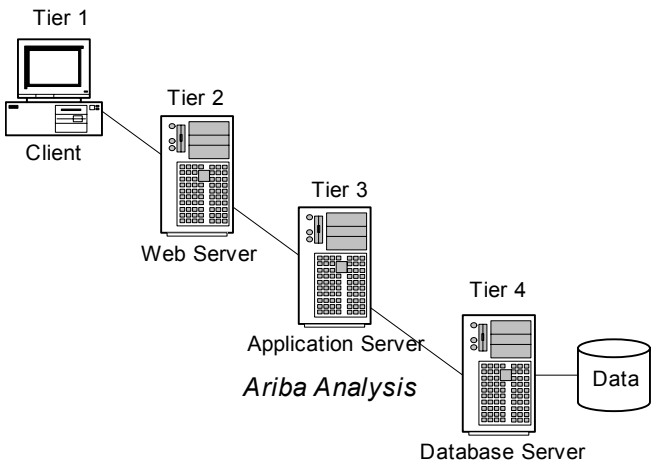
Chapter 2
Components of Ariba Analysis

This section describes Ariba Analysis software components, how to start and stop Ariba Analysis, and how to secure it. It includes the following sections:

Note: This chapter does **not** discuss in detail how to administer your Web, application, or database servers, except as those topics relate to the maintenance of Ariba Analysis itself. If you need such details, consult the appropriate documentation for the specific software in question.

Architectural Tiers of Ariba Analysis

Ariba Analysis has several architectural tiers, as depicted in the following drawing.



The Client tier consists of end-user browsers and spreadsheet programs.

You can combine the Web Server tier, Application Server, and Database tiers onto a single set of hardware or distribute them across a network depending on your needs. However, you must install Ariba Analysis on the same hardware as the J2EE application server.

Servers and Client Software

This section includes the following discussions:

Client Software

Two considerations of the client software requirements for using Ariba Analysis are discussed here: Web browsers and spreadsheet applications.

Web Browser: Internet Explorer

To get the most from Ariba Analysis, you may want to use Microsoft's Internet Explorer Web browser. The only reason for this recommendation is that Ariba Analysis relies on ActiveX controls to invoke and manipulate the Microsoft Excel spreadsheet application during data export from Ariba Analysis.

You can use other Web browsers with Ariba Analysis, but exported data is limited to comma-separated value (CSV) files, not Microsoft's proprietary Excel format.

Types of Servers in Ariba Analysis

Regardless of your hardware configuration, Ariba Analysis is composed of four software servers, corresponding to the architectural tiers (excluding the client tier) described in "Overview of Administrative Tasks" on page 16:

Server Type	Description
Web Server	The Web server is the front end that receives requests from clients and passes them to the application server for processing.
Application Server	The J2EE-compliant application server runs the Java Virtual Machine (JVM) for your selected hardware and operating system and hosts the Ariba Analysis Server. For more information about application servers, see the *Ariba Spend Management Application and Web Server Guide*.
Database Server	The database server holds the repository for all facts, measures, dimensions, and materialized views of the data. The database server is rarely accessed directly.
Ariba Analysis Server	Ariba Analysis runs as an application called the *Analysis Server* on top of the application server. Physically, the Analysis Server is a set of Java classes and supporting files.

Starting and Stopping Ariba Analysis

This chapter details how to start the servers associated with Ariba Analysis. Steps include the following:

Start Database and Web Server

Before attempting to start Ariba Analysis, first ensure that you have started your database server and that your Ariba Analysis server host can connect to it.

Likewise, make sure that you have started the Web server through which you will access Ariba Analysis.

Start Application Server

Before starting Ariba Analysis, you must start its associated application server's administration server. Precise steps depend on the type of application server you use.

For details about starting and stopping your application server, see the *Ariba Spend Management Application and Web Server Guide*.

Start Ariba Analysis

The exact steps you use to start Ariba Analysis depend on which application server you use or if you want to start Ariba Analysis from the command line:

Note: On UNIX, Ariba Analysis needs an X Window System buffer. Therefore, you must start Analysis while running an X Window manager. If you do not run an X Window system manager, use the X Window virtual frame buffer software provided with the Sun Solaris version of Ariba Analysis:

```
<AnalysisInstallRoot>/Server/bin/startxvfb
export DISPLAY=<hostname>:1.0
```

After running `startxvfb`, you must export your `DISPLAY` environment variable. `startxvfb` uses `1.0` as its default `DISPLAY`. If that display is already in use, set a different one with the following option on the `startxvfb` command:

```
startxvfb -display <desiredDisplay>
```

If you are not running Sun Solaris, you must start your own X server software.

startanalysis Command

Use the `startanalysis` command to start Ariba Analysis:

AnalysisServerRoot/bin/startanalysis

The startanalysis command connects to the application server's administration server, loads Ariba Analysis, and then reports its success.

In the window in which you run the command, you see messages that indicate a successful startup. The following example shows a successful start up with BEA WebLogic Server:

```
C:\Ariba\app\Analysis\Server\bin>startanalysis
Preparing to start Node(s):
Node1: WebLogic Server analysisserver1 on Machine alange
WebLogic Node Manager on alange successfully contacted...
Starting Node1...

To monitor the status of Analysis:
Analysis log files, WebLogic stdout, and WebLogic stderr
are located under the Analysis installation area.
```

Stop Ariba Analysis

The exact steps you use to stop Ariba Analysis depend on which application server you use or if you want to stop it from the command line.

For information about stopping Ariba Analysis with the application server, see the *Ariba Spend Management Application and Web Server Guide*.

stopanalysis Command

To stop Ariba Analysis from the command line, use the stopanalysis command:

AnalysisServerRoot/bin/stopanalysis

The following is an example of stopping an Ariba Analysis implementation using BEA WebLogic on Microsoft Windows:

```
C:\Ariba\app\Analysis\Server\bin>stopanalysis
Stopping node Node1...
analysisserver1 shut down successful
```

Securing Ariba Analysis as an Administrator

This section details important steps you must take to ensure the security of your Ariba Analysis installation. It contains the following sections:

The Administrative User in Development: aribasystem

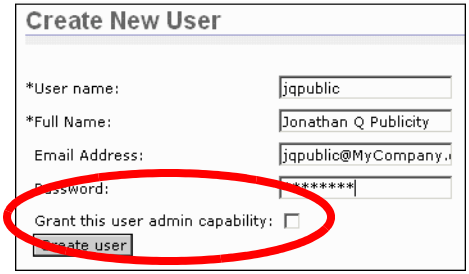
For greater security, when Ariba Analysis is running in development, you must create user names for your test users.

To create a user name, you must login as follows:

User name	`aribasystem`
Password	`ariba`

Creating Test Users in Development

After you have logged in as the `aribasystem` user, use the navigation panel item **Create a New Test User** to see the form by which you can add user names and passwords to Ariba Analysis in development.



The user name and full name are required. All other fields are optional.

To grant the newly created test user administrative privileges, click the checkbox.

Getting Administrative Permission and Menu Hyperlinks

Becoming an Ariba Analysis administrator has two parts:

- Getting the `AnalysisAdmin` permission
- Adding administrative hyperlinks to your navigation panel

The user `aribasystem` has the administrative permission called `AnalysisAdmin`. Any user with the `AnalysisAdmin` permission can use the Ariba Analysis administrative console, which aids in some common administrative tasks. See Chapter 7, "Ariba Analysis Administrative Console.".

Getting AnalysisAdmin Permission

There are several ways a user can obtain the `AnalysisAdmin` permission:

- In development, a user can be granted the permission when his account is created by the `aribasystem` user.
- In production, by permissions granted in Ariba Buyer or Ariba Enterprise Sourcing.

Adding the Analysis Admin Module to the Navigation Panel

After a user has been given `AnalysisAdmin` permission, he must complete the following steps to add the administrative hyperlinks to his navigation panel.

1. Login to Ariba Analysis.
2. On the dashboard, click **Customize**.
3. From the displayed list, drag the **Analysis Admin (Nav)** object to the navigation panel on the left.
4. Drop the object onto the navigation panel.

The administrative hyperlinks are now displayed in the navigation panel of the user's dashboard.

Securing the Inspector

By default at installation, the Inspector is enabled and unprotected so that development can begin quickly.

In *AnalysisServerRoot*/config/Parameters.table, the values in System.Debug relating to the Inspector are as follows:

```
Inspector = true;
InspectorAuthenticate = false;
InspectorPassword = "";
InspectorUsername = "";
```

To disable the Inspector, set the Inspector parameter as follows:

```
Inspector = false;
```

If you want to restrict access to the Inspector, set the InspectorAuthenticate parameter, which causes a prompt for user name and password when someone attempts to access it:

```
InspectorAuthenticate = true ;
```

You should also change the Inspector's default user name and password to something more secure.

Chapter 3
Server Configuration Files

This chapter describes some considerations about maintaining your servers.

Ariba Analysis Directory Structure

This section details the various directories that make up Ariba Analysis and introduces some terminology for them.

Ariba Analysis Install Root

The top level directory where Ariba Analysis is stored is referred to throughout this document as:

AnalysisInstallRoot

In the Ariba Analysis installation program, the default locations for *AnalysisInstallRoot* are as follows:

Microsoft Windows	C:\Ariba\app\Analysis
UNIX	/opt/ariba/app/analysis

The *AnalysisInstallRoot* is further subdivided into the following directories:

Directory	Common Term	Description
AnalysisInstallRoot/Server	*AnalysisServerRoot*	**Analysis Server components**
AnalysisInstallRoot/WebComponents	*AnalysisWebRoot*	Ancillary HTML and graphics files

All configuration files are in the *AnalysisServerRoot* directory and its subdirectories. *AnalysisWebRoot* contains the Ariba online help.

Analysis Server Root and Description of Subdirectories

The *AnalysisServerRoot* directory is your primary focus as an administrator or designer. Here is a description of its subdirectories and their purposes.

Directory	Description and Purpose	See Also
_jvm	Java virtual machine for the appropriate operating system	*Ariba Analysis Installation Guide*
_uninst	Program and support files for uninstalling Ariba Analysis	
3rdParty	Third-party programs used internally by Ariba Analysis	
ariba	Basic definitions of objects and other aspects of Ariba Analysis and customizations to them, in the form of metadata XML files and extensions	*Ariba Analysis Customization Guide*
bin	Executable programs for starting Ariba Analysis, running tasks, and other functions	
classes	Compiled Java class files	
config	Ariba Analysis primary configuration files, such as the Parameters.table. The config subdirectory also contains definitions and configuration files for data loading.	"Configuration Files in AnalysisServerRoot/config" on page 30 *Ariba Analysis Customization Guide*
configureOptions	Template configuration files for use with the addsourcesystem script to create files in the config/sourceTypes directory	"addsourcesystem" on page 73
doc	Ariba Analysis documentation set	
docroot	Support files for Ariba Analysis web pages	
etc	Ariba Analysis ancillary files	
etc/certs	SSL certificates	"Moving Ariba Analysis into Production" on page 56

Directory	Description and Purpose	See Also
`lib`	Support Perl modules	
`logs`	Ariba Analysis logfiles	"Ariba Analysis Logs" on page 34
`sample`	Skeletal files of sample configurations of Ariba Analysis.	"About the Sample Configuration" on page 29
`temp`	Ariba Analysis temporary working space	
`uploadedFiles`	Excel templates users upload Ariba Analysis. This directory is present only if at least one file has been uploaded.	"Managing Uploaded User Files" on page 49
`wizard`	Support files for the Analysis Wizard	*Ariba Analysis Advanced User Guide* and online help
`xmlcache`	A caching directory for processing metadata XML	

About the Sample Configuration

If you installed the demonstration configuration (refer to the *Ariba Analysis Installation Guide* for details), you have a sample configuration in the following directories. Study these files to understand how to modify or extend Ariba Analysis:

AnalysisServerRoot`/ariba/variants/`
AnalysisServerRoot`/ariba/sourceTypes/`
AnalysisServerRoot`/config`

Note: The subdirectory *AnalysisServerRoot*`/sample` contains skeletal files that are altered before being installed. Do not use these files.

The sample configuration includes many examples. Use them as a basis for your own extensions to Ariba Analysis:

- Extensions to the core data model of Ariba Analysis
- Sample data loading events

Configuration Files in AnalysisServerRoot/config

The directory *AnalysisServerRoot*/config holds several configuration files and subdirectories.

File or Directory Name	Purpose	See
ConnectionInfo.table	Describes how to connect to ERP systems or databases. **Note:** Connections to Ariba applications, except Ariba Buyer 7.x, are defined in the AppInfo.xml file.	*Ariba Analysis Data Load Guide*
LoadDB.txt	Lists scheduled tasks or data load events to be executed by initdb -loaddb	*Ariba Analysis Data Load Guide*
Parameters.table	Contains centralized parameters affecting all aspects of Ariba Analysis	"Parameters.table" on page 30
DataLoadTasks.table	Names definitions of tasks related to data loading: class names and parameters	*Ariba Analysis Data Load Guide*
ScheduledTasks.table	Names definitions of tasks run by Ariba Analysis on a schedule you specify	Appendix C, "Scheduled Task Reference"
sourceSystemConfig.txt	Lists source systems created with the addsourcesystem script or from initial configuration	"addsourcesystem" on page 73
sourceTypes Directory	Holds data loading definition files	*Ariba Analysis Customization Guide*
variants Directory	Holds extensions to Ariba Analysis metadata XML	*Ariba Analysis Customization Guide*

Parameters.table

The file *AnalysisServerRoot*/config/Parameters.table contains settings that control many features and functions of Ariba Analysis. This section discusses the structure of config/Parameters.table.

Ariba Analysis shares a common set of parameters with Ariba Buyer, as documented in the *Ariba Buyer Configuration Reference Guide*. A complete list of all parameters specific to Ariba Analysis and their meanings is in "Ariba Analysis Parameters.table" on page 83. In addition, individual parameters affecting various features of Ariba Analysis are discussed in the section pertinent to that feature.

Structure of Parameters.table and Full Parameter Names

`Parameters.table` is divided into sections or groups of parameters. For example, the `System` section of the table includes parameters related to the functions of the Ariba Analysis system (as opposed to the Ariba Analysis application). For example:

```
System {
  Analysis {
    /* parameters go here */
    someParameterName = "someParameterValue";
    }
  }
```

Likewise, sections in the `Parameters.table` can have subsections, and these subsections may in turn include other subsections or the parameters themselves.

The full name of a parameter specifies the entire path to uniquely identify the parameter, with the portions of the full name delineated by periods (“.”). For example, the full name of the `BaseCurrency` parameter in the `Analysis` subsection of the `Application` section of `Parameters.table` is as follows:

```
Application.Analysis.BaseCurrency = USD;
```

Debugging

The Ariba Analysis logging category called `analysis`. By default, Ariba Analysis logs only informational messages, as indicated by the `info` setting in the following snippet. To set logging to the maximum, change `info` to `debug` on the `Categories` lines in `Parameters.table`, as shown below, or use the Ariba Analysis administration console to do so. See"Set Logging Levels" on page 60.

```
Logging = {
  Console = {
    Categories =
    "util/info:general/info:startup/warning:analysisMaster/info";
    DisableLogListener = false;
    LocaleForLogMessages = en_US;
    };
```

```
MainLogFile = {
   Categories = "util/info:general/info:startup/info:analysisMaster/info";
   DisableLogListener = false;
   LocaleForLogMessages = en_US;
   };
};
```

AppInfo.xml

The file *ASMsharedDirectory*/config/asmshared/AppInfo.table defines connections to other Ariba applications. AppInfo.xml is discussed in detail in the *Ariba Spend Management Integration Guide*.

ConnectionInfo.table

The file *AnalysisServerRoot*/config/ConnectionInfo.table is an essential part of data loading, which is discussed in detail in the *Ariba Spend Management Configuration Guide*. It contains definitions for how Ariba Analysis can connect to databases: either Ariba Buyer 7.1 or other, non-Ariba-Buyer databases.

Note: Connection definitions for Ariba Buyer are required only if you are working with Ariba Buyer version 7.1. No definitions are required in ConnectionInfo.table if you are using Ariba Buyer 8.x.

For each remote database, in ConnectionInfo.table you must specify a label and associate with that label the particulars of how to connect to that remote database, using a Java database connection (JDBC) string. The allowable parameters and values in ConnectionInfo.table are identical to those in the Ariba Buyer ConnectionInfo.table. See the *Ariba Buyer Configuration Guide*.

The default ConnectionInfo.table packaged with Ariba Analysis contains the following definitions for connecting to an Oracle database and a DB2 database:

```
{
    DBConnections = {
        sampleOracle = {
            DBType = Oracle;
            Driver = oracle.jdbc.driver.OracleDriver;
            Password = password;
            URL = “jdbc:oracle:thin:@HOSTNAME:PORTNUMBER:SID”;
            User = user;
        };
```

```
        sampleDB2 = {
            DBType = DB2;
            Driver = "COM.ibm.db2.jdbc.app.DB2Driver";
            Password = password;
            URL = "jdbc:db2:SERVER";
            User = user;
        };
    };
}
```

Logfile Descriptions and Locations

Ariba Analysis has several logfiles in the *AnalysisServerRoot*/logs directory.

Logfile	Purpose and Location
Web server logs	These logs record all HTTP or HTTPS requests and Web server errors. Precise location depends on the Web server you use and how it is configured.
Database server logs	These reside on your database server. Precise location varies depending on which database software you use and how it is configured. Very rarely do you need to consult the database server logs. Important messages related to the database are recorded in the Ariba Analysis logs.
Application server logs	These logs record the application server start and stop messages and some Java exceptions. Precise location depends on which application server you use and how it is configured.
Analysis Server logs	Ariba Analysis logs are in *AnalysisServerRoot*/logs and are described more fully in "Ariba Analysis Logs" on page 34.

Ariba Analysis Logs

Ariba Analysis has different logs in *AnalysisServerRoot*/logs to record messages about different aspects of the server. You can use the Ariba Analysis administration console's options **Log Files** and **Log Settings** to control the level of detail recorded in the logs and to find and view specific logs. See "View Log Files" on page 59 and "Set Logging Levels" on page 60.

Logfile Name	Purpose
archive	A directory to hold old logs.
AnalysisLoadDBLog.txt	Records the running of data loading events. See the *Ariba Analysis Data Load Guide*. This log can become large if data loading debugging is enabled. See "Debugging" on page 31.
AnalysisNode1Log.txt	The main log for the Ariba Analysis. Records server starts and stops and other important messages, such as possible Java exceptions.
AnalysisRunTaskLog.txt	Records details from using the runtask command.
classMetrics-Node1.txt	Records performance data about the Ariba Analysis Java classes.
configuration.txt	Records parameters processed by the configuration program configure. See "configure" on page 75.
db2stats.txt	Records details about calculation of database statistics. Present only if you are running IBM DB2 database.
dbinit.txt	Records the running of the initdb program. See "initdb" on page 76.
j2eeSetupLog.txt	Records the running of the j2eesetup script, which is invoked by the configure command. This log is in *AnalysisServerRoot*/etc/install/logs.
DBOwnerReset.log	Records the running of the resetdatabaseowner command. See "resetdatabaseowner" on page 78.

Chapter 4
Scheduled Tasks

This chapter introduces scheduled task configuration files and describes how you use these files to set the schedule and parameters for scheduled tasks. It includes the following sections:

Introduction to Scheduled Tasks

Scheduled tasks are processes that run on a regular basis, in the background. Ariba Buyer uses scheduled tasks for regular system maintenance and administration, such as archiving log files.

Scheduled tasks usually run on a regular schedule, which is defined in the scheduled task configuration file. You can also run scheduled tasks on from the command line, with the `runtask` command.

Scheduled Task Configuration Files

You configure scheduled tasks by editing a *scheduled task configuration file*. A scheduled task configuration file is a list of scheduled tasks, together with a schedule for each and the parameters that apply to that task.

There is a small set of parameters recognized by all scheduled tasks. Most scheduled tasks also recognize additional task-specific parameters. For example, a scheduled task that sends notification messages might take additional parameters to specify the frequency for sending out those notifications.

Location of Scheduled Task Configuration Files

By default, the scheduled task configuration file for global data is in the following location:

```
config/ScheduledTasks.table
```

Scheduled Task Configuration File Format

A scheduled task configuration file is in table file format and consists of a list of task definitions, one for each task. For example:

```
TaskName = {
    ScheduledTaskClassName = "ariba.analytics.util.FailedDocumentMessages";
    Schedules = {... };
}
```

Each entry has a name (`TaskName`, in this example) and a collection of parameters. This example shows two parameters, `ScheduledTaskClassName` and `Schedules`.

Scheduled task names must be unique.

Scheduled Task Properties

Each entry in a scheduled task configuration file has at least one required property, `ScheduledTaskClassName`, which is the name of a Java class file that runs the scheduled task. For example:

```
ScheduledTaskClassName = "ariba.analytics.util.FailedDocumentMessages";
```

Most scheduled tasks also have a `Schedules` property, which defines when and how often the task will run. If there is no `Schedules` property, the task does not run on a regular basis. The following example is an entry for a scheduled task that runs every Monday and Thursday at 9 PM:

```
Task = {
    ScheduledTaskClassName = ariba.procure.server.TaskName;
    Schedules = {
        Schedule1 = {
            DayOfWeek = Monday;
            Hour = 21;
        };
        Schedule2 = {
            DayOfWeek = Thursday;
            Hour = 21;
        };
    };
}
```

If the task does not run at its scheduled time for any reason (for example, if there is a power failure), it will not run again until the next scheduled time. Ariba Buyer does not attempt to rerun tasks that did not run on schedule.

Most scheduled tasks take additional properties, which are specific to that task.

Syntax of Schedules

The Schedules property defines when and how often the scheduled task runs.

The Schedules key is a hash table, built from the following components:

```
DayOfWeek = Weekday | Everyday | Monday | Tuesday …
DayOfMonth = 0..31;
Hour = 0..24;
Minute = 0..59;
Second = 0..59;
```

You can define a schedule in two ways: DayOfWeek or DayOfMonth.

After you have specified DayOfWeek or DayOfMonth, you can define the time with Hours, Minutes, and Seconds. The times are optional. If you do not supply any times, the default is 0. For example, to run a task every Monday at midnight, you would set the following schedule:

```
Schedules = {
    Schedule1 = { DayOfWeek = Monday};
};
```

The schedule names (Schedule1 in this example) are just names. They can be anything you like. Ariba Buyer determines the schedules for a given task by enumerating all schedules under the Schedules key, whatever their names might be.

You can schedule the same task to run at several specific times. For example:

```
Task = {
    ScheduledTaskClassName = ariba.procure.server.TaskName;
    Schedules = {
        Schedule1 = {
            DayOfWeek = Monday;
            Hour = 21;
        };
        Schedule2 = {
            DayOfWeek = Thursday;
            Hour = 21;
        };
    };
}
```

The keys (such as DayOfWeek and Everyday) and the values are case-sensitive.

You can also schedule a task to run periodically, with the Period parameter. For example:

```
Task = {
    ScheduledTaskClassName = ariba.procure.server.TaskName;
    Schedules = {
        Schedule1 = {
            DayOfWeek = Weekday;
            Hour = 0;
            Period = {
                Unit = Hours;
                Quantity = 1;
        };
    };
}
```

In this example, the task runs every weekday at midnight, as specified by the DayOfWeek and Hour parameters, and then again every hour, as specified by the Period parameter.

Each Period is defined with Unit and Quantity parameters. The Unit parameter can be Seconds, Minutes, Hours, Days, or Weeks. The Quantity parameter indicates how many units to skip before running the task again. For example, to run the task every 30 seconds after the task is first run, you would specify Seconds for the unit and 30 for the quantity. To run the task every month, use the DayOfMonth parameter. The value for Quantity is interpreted as an integer. If you specify a decimal, such as 3.6, the value is truncated down to an integer (3, in this example).

Reference List of Properties

The following table describes properties that are common to all scheduled tasks.

Property	Description
ScheduledTaskClassName	Names the Java class for this scheduled task. For example: `ScheduledTaskClassName = "ariba.analytics.util.FailedDocumentMessages";` This parameter is required.

Property	Description
Schedules	Defines when and how often the task runs. This parameter is optional. If not specified, the task does not run on a regular basis. For information on the syntax of this parameter, see “Syntax of Schedules” on page 38.

Chapter 5

Managing Users, Reports, and Downloads

This chapter describes administrative tasks associated with managing users of Ariba Analysis, their analytical reports, and associated files.

This chapter contains the following sections:

Security of Ariba Client Automation

With ActiveX controls signed by Ariba, Inc., the Ariba Client Automation software secures the exporting of data from Ariba Analysis to Microsoft Office products, such as Word or Excel.

This section includes the following topics:

How Ariba Client Automation Works

The Ariba Client Automation software is a signed ActiveX control. It securely verifies the authenticity of data exported from Ariba Analysis to a user's desktop.

Information Technology organizations have two deployment options:

1. They can distribute the signed ActiveX controls to all users' personal computers in the organization.
2. They can allow users to answer "Yes" when prompted automatically to download the controls during the normal use of Ariba Analysis.

Different web browser settings can also be employed to take advantage of even greater security.

Server Set-up

This section details the necessary steps to set-up the Ariba Client Automation ActiveX software on your Ariba Analysis servers. It includes the following sections:

- "Acquire Digital Certificate for Use with ActiveX" on page 42
- "Set Parameters" on page 43

Acquire Digital Certificate for Use with ActiveX

For maximum security, you should acquire and install a digital certificate from a well-known Certification Authority (CA), not Ariba, Inc., to ensure the safety of data export and the authenticity of the ActiveX controls. This certificate can be the same as you use to secure your web or other servers, or for the most security, it can be separate from them.

Set Parameters

Three parameters in the Ariba Analysis `config/Parameters.table` file secure the Ariba Client Automation software.

Parameter	Description
`Application.Base.CertificateUrl`	The URL by which the ActiveX control can download the server certificate to verify signature of the control. This parameter can be either a URL relative to server's `ResourceURL` or an absolute URL.
`Application.Base.SSLKeyStoreFile`	File path to a file that contains the server's private key. The default for this parameter is `etc/aribakeystore`. For information about generating a keystore file, see the information about `keytool` in the *Ariba Buyer Configuration Guide* although any of a number of freely available tools can be used to do create the keystore.
`Application.Base.SSLKeyStorePassword`	Password to access the keystore specified in `SSLKeyStoreFile`

Client Set-up

This section describes the steps necessary to set-up users' web browsers to secure data export with the Ariba Client Automation ActiveX control.

Deployment Options: Predistribute or User Acceptance

You have two deployment options for the Ariba Client Automation ActiveX controls:

1. You can predistribute the signed control to users' computers.

   This option, although it requires more effort, allows you to retain control of exactly what is distributed.

2. You can allow users to answer prompts during data export to accept the download.

Predistribution

Microsoft documents how to distribute software or files over a network to users' personal computers. Consult that documentation for precise detailed steps.

The Ariba-signed ActiveX control is located in the following file.

```
WebComponents/ariba/resource/en_US/lib/clientautomation.cab
```

To install this control, you must first extract the dynamic load library `clientautomation.dll` out of the cab file.

Here are some sample installation commands for reference.

Installation

Copy the Ariba-signed ActiveX Control to a directory where Internet Explorer searches for installed ActiveX controls:

```
copy clientautomation.dll "c:\WINDOWS\Downloaded Program
Files\clientautomation.dll"
```

Register the control with the Windows operating system:

```
regsvr32 -s "c:\WINDOWS\Downloaded Program Files\clientautomation.dll"
```

Optional step to install the web server certificate so that users don't even see the certificate challenge:

```
certmgr -add -c -n [commonNameInCertificate] [certificate] -s TrustedPeople
```

For example:

```
certmgr -add -c -n "analysis.ariba.com" ariba.cer -s TrustedPeople
```

Note: `certmgr` is not avaiable on Microsoft Windows by default. It is part of the Microsoft .NET framework SDK., which is downloadable from Microsoft.

Uninstallation

Unregister `clientautomation.dll` with the Windows operating system:

```
regsvr32 -s /u "c:\WINDOWS\Downloaded Program Files\clientautomation.dll"
```

Optional step to remove the web server certificate from certificate store.

```
certmgr -del -c -n [commonNameInCertificate] -s TrustedPeople
```

For example:

```
certmgr -del -c -n analysis.ariba.com -s TrustedPeople
```

User Acceptance

During data export, if the Ariba Client Automation ActiveX control is not present on the users computer, Ariba Analysis automatically downloads it and prompts the user to accept the download.

▼ **To see if Ariba ActiveX controls have been installed on a particular computer:**

Check the browser settings.

1. In Internet Explorer, in the **Tools** menu, choose **Internet Options**.
2. On the **General** tab of **Internet Options**, click **Settings** in the **Temporary Internet Files** area.
3. In the **Settings** window, click **View Objects**.
4. Look for **Ariba Client Automation ActiveX Control**.

   If that file is labelled as **Installed** in the **Downloaded Program Files** directory, the Ariba-signed ActiveX control has been installed on the computer.

User Web Browser Security Settings

The Internet Explorer security settings for use with the Ariba Client Automation signed ActiveX control depend on how the control has been installed on a user's computer: predistributed or accepted for download by the user.

- With predistribution, the single required setting is to allow only **Administrator approved** ActiveX controls.
- With user-accepted download, the two settings are as follows.

Setting	Choices
Download signed ActiveX controls	Either **Prompt** or **Enabled**
Run ActiveX controls and plug-ins	Either **Prompt** or **Enabled**

For more details about the settings corresponding to these deployment options, consult the Ariba Analysis online help.

Permissions to Publish or Schedule Reports

Ariba Analysis has the following permissions that you can grant to privileged users:

- `AnalysisScheduleReport`: schedule reports.
- `AnalysisPublishReports`: share reports with other users.

In the user authentication source for Ariba Analysis, assign these permissions to the appropriate users. The user authentication source is either Ariba Buyer or Ariba Enterprise Sourcing, unless Ariba Analysis is standalone, in which case Ariba Analysis itself is the user authentication source. For more information about user authentication sources, see the *Ariba Spend Management Integration Guide*.

Dashboard Templates and Report Models

The dashboard of the user named `TemplateUser` can be used as the basis for the default dashboard for all users. However, you can set-up custom dashboards for specific business roles and functions.

The models of reports and Microsoft Excel templates are owned by the user `ashell`.

For more information about `TemplateUser` and the reports, see the *Ariba Analysis Advanced User Guide*.

About User Profiles and Default Currency

In production, a user profile in Ariba Analysis is an exact duplicate of the profile in Ariba Buyer or Ariba Enterprise Sourcing. The only part of the profile that the user can edit in Ariba Analysis is their default currency. The rest of the user information is a duplicate of the information stored in Ariba Buyer or Ariba Enterprise Sourcing.

Importing and Exporting User and Report Definitions

You can export and import the definitions of users' dashboards and folder structures, which is useful when you want to upgrade Ariba Analysis or move Ariba Analysis from development to production.

Use the scheduled task `ObjectEncoding` to import or export user or report definitions. You can import or export the definitions of the dashboard and folders for either all users or a single user. Moving user and report definitions from your Ariba Analysis development system to production requires this scheduled task.

See "Importing and Exporting Users: ObjectEncoding" on page 103 for details on this scheduled task.

Managing User Login

How to control user access to Ariba Analysis and its data depends in general on what mode Ariba Analysis is running in: development or production. This section contains the following topics:

Controlling User Session Timeout

Ariba Analysis disconnects users' Web browsers that have been idle for a specified period of time. The users must login again.

You control session timeout with settings in your application server. For detail instructions, see the *Ariba Spend Management Application and Web Server Guide*.

Resetting User Passwords

How user passwords are reset depends on whether Ariba Analysis is running in development or in production.

Development

▼ **In development, to reset a user's password with the Inspector:**

1. Access the Inspector.

   By default, the Inspector is accessed at the following URL:

   `http://yourServer:yourPort/Analysis/inspector`

   Access to the Inspector might be protected. See "Securing the Inspector" on page 25.

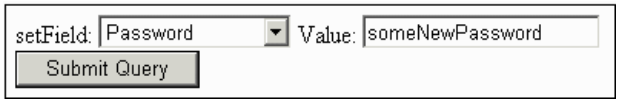
2. To locate a particular user, select the following cluster root class:

   `User-ariba.analytics.core.User`

3. From the list of user names, click the hyperlink associated with the particular user whose password must be reset.

4. Use the **setField** function to specify a new password for this user.

   For example:



5. Click **Submit Query** to reset the password to the new value.

Production

In production, passwords are changeable only by the users themselves by way of your Ariba Buyer or Ariba Enterprise Sourcing implementation.

Controlling User Access to Data

You can block users from viewing particular data by relying on the default data access control manager supplied by Ariba or on your own custom access control. You must write your access control manager using the Ariba Java API for this purpose. See "Work with the Example Data Access Profile Editor" on page 63 and the *Ariba Analysis Customization Guide*.

In addition, metadata XML can control the visibility of hierarchies or specific levels in hierarchies. See the *Ariba Analysis Customization Guide*.

Managing Uploaded User Files

Ariba Analysis allows users to upload Excel templates and other files that they may customize for special purposes. Ariba Analysis stores these files in the *AnalysisServerRoot*/uploadedFiles directory.

Do not manually delete files from the uploadedFiles directory. Users who want to delete uploaded files must do so themselves using the folder management features in Ariba Analysis.

Controlling Microsoft Excel Export Behavior

Some organizations might want to disable the automatic launching of Microsoft Excel during data export from Ariba Analysis, for example, to comply with a security policy about the automatic launching of applications from web browsers.

To disable Excel downloading, set the ExcelExportDisabled parameter in *AnalysisServerRoot*/config/Parameters.table:

```
Application {
  Analysis {
    ExcelExportDisabled=true ;
    }
  }
}
```

Note: In addition, Ariba Analysis has parameters to control comma-separated value (CSV) files, such as setting a delimiter for each locale and setting the character encoding. For more information, see "Application.Analysis.CSVDelimiters" on page 84 and "Application.Analysis.CSVExportEncoding" on page 85.

Disabling the Public Folder

You can disable Ariba Analysis public folders either for all users or user-by-user.

Each user object has an extrinsic field called SharedFolderDisabled. If this field is set to `true`, the public folder is not displayed.

By default, the TemplateUser's public folder is enabled. All newly created users inherit the setting from TemplateUser. To disable public folders by default for all users, use the Inspector to set the field SharedFolderDisabled to `true` for TemplateUser.

To disable shared folders for a single user, in the Inspector use the setField function to set the SharedFolderDisabled field to `true`. To re-enable, set the field `false`.

To enable shared folders only for a few users, like reports (the owner of the prepackaged reports) use SQL as follows:

1. Disable shared folder for all users with this SQL statement:

   ```
   update UserTab set Usr_SharedFolderDisabled = 1;
   ```

2. For the users whose public folders should be enabled, set SharedFolderDisabled to `false` with the inspector or the value 0 with SQL.

Chapter 6
Miscellaneous Configuration

This chapter details configuring various aspects Ariba Analysis not discussed in other chapters of this book.

It contains the following sections:

Managing Report Cache and Background Queries

Ariba Analysis caches reports in both main memory and on disk, saving the results of database queries from generating analytical reports. Any user who creates or runs an analytical report with the same results benefits from this cache. Reports are displayed with little or no delay.

In addition, users can run reports in the background.

Parameters in `config/Parameters.table` for controlling the report cache and background queries are discussed in the following sections:

Maximum Rows for In-Memory Cache

The total number of rows (database records) in the Ariba Analysis in-memory report cache cannot exceed the limit set by the parameter `InMemCacheMaxTotalRowLimit` in `config/Parameters.table`.

The default setting is as follows:

```
System.Analysis.OLAP.InMemCacheMaxTotalRowLimit = 200000 ;
```

Maximum Number of Database Tables for Secondary Report Cache

The maximum number of database tables for the secondary report cache is specified in `config/Parameters.table` with the `DBCacheMaxTotalEntriesLimit` parameter.

By default `DBCacheMaxTotalEntriesLimit` is as follows:

```
System.Analysis.OLAP.DBCacheMaxTotalEntriesLimit = "140";
```

Secondary cache table names have the prefix `CACHE_ANALYSIS`.

Disabling the Report Cache

You can completely disable to the report cache with the `CacheDisabled` parameter in `config/Parameters.table`.

By default `CacheDisabled` is as follows:

```
System.Analysis.Debug.CacheDisabled = "false";
```

To disable the cache, set `CacheDisabled` to true, and restart Ariba Analysis.

Background Query Timeout

The timeout for database queries for reports run in the background cannot exceed the the limit set by the parameter `BackgroundQueryTimeOut` in `config/Parameters.table`.

The default setting is as follows:

```
System.Analysis.OLAP.BackgroundQueryTimeOut = 20 ;
```

This value is the number of seconds that can elapse before a database query times out.

Setting Language Preference

You can set your language preference by editing the value of the following parameter in `config/Parameters.table`:

```
System.Base.DefaultLocale = locale ;
```

The value for *locale* can be any of the following languages Ariba Analysis supports. The default setting is `en_US`.

Language	Locale
US English	`en_US`
French	`fr`
German	`de`
Italian	it
Spanish	es

The language preference for data loading from Ariba Buyer is determined by the following parameter:

```
Application.Base.Data.DefaultLanguage = language ;
```

The value for *language* can be any of the languages Ariba Buyer supports. The default setting is `English`.

Administering Multiple Currencies

Ariba Analysis supports multiple currencies. Currencies are defined as any valid three-character currency code as specified by the standard ISO 4217.

This section describes the following topics:

For information about how multiple currencies affect data loading, see the *Ariba Analysis Data Load Guide*.

MultiCurrency Data Type

A `<measure>` that is to represent a currency amount (such as `Amount`) must have a data type of `MultiCurrency`. Define this with the type attribute. For more information about managing multiple currencies, see the *Ariba Analysis Customization Guide*.

Currency-related Parameters

Two Ariba Analysis system parameters relate to currencies:

- "Application.Analysis.BaseCurrency" on page 84

  The default base currency is the US dollar, `USD`.

- "Application.Analysis.SupportedCurrencies" on page 87

  The default supported currencies include the default base currency (`USD`), the Euro (`EUR`), and the Japanese yen (`JPY`).

Impact of Multiple Currencies on Using Ariba Analysis

Users of Ariba Analysis are allowed to set a currency preference, which can be any single valid currency code from the combined values of the parameters `BaseCurrency` and `SupportedCurrencies`.

Ariba Analysis computes materialized views for every supported currency you specify, including the base currency. In addition, Ariba Analysis fact tables also store values in each of the supported currencies. For these reasons, you should not set the number of supported currencies to more than the essential few you need for analysis. The more supported currencies you specify, the greater the performance impact and the greater the database storage requirements.

Constructing a Login URL

The *login URL* is the URL by which your users access Ariba Analysis in development or the URL you must specify in certain configuration files when you are integrating Ariba Analysis with other products of Ariba Spend Management. The login URL is composed of the values of several different parameters in the `System.Base` section of *`AnalysisServerRoot`*`/config/Parameters.table` and in the global Ariba Spend Management configuration file `AppInfo.xml`:

- IncomingHttpServerURL in AppInfo.xml
- ContextRoot in Parameters.table
- MainServletName in Parameters.table

For a description of AppInfo.xml, see the *Ariba Spend Management Integration Guide*. For a description of the other parameters, see "System.Base" on page 101.

The login URL is constructed as follows:

IncomingHttpServerURL/ContextRoot/MainServletName

Using the default values of these parameters at installation and as recorded in Parameters.table, you have a URL like this:

http://*yourHostName*:8100/Analysis/Main

In the actual URL that you provide to users or use in configuration files on hosts other than your Ariba Analysis server, you must substitute the machine and domain name for *yourHostName*.

Examples of IncomingHttpServer URL for Different Configurations

The IncomingHttpServerURL parameter in AppInfo.xml specifies the name and port of the front-end to Ariba Analysis, which can be either your Web server or your application server. For example, if you prefer that access be directly through the application server, IncomingHttpServerURL is as follows:

http://*appServerHost*:*appServerPort*

However, if you have decided to deploy Ariba Analysis in a three-tier model, with distinct machines for the Web server (with a proxy plug-in), the application server, and database server, IncomingHttpServerURL is as follows:

http://*webServerHost*:*webSeverPort*

If you have enabled SSL on your Web or application server, your protocol specification in IncomingHttpServerURL must begin with https, not http.

Working with the Online Help

Ariba online help is located in *AnalysisWebComponents*. The full path to the English version of the help is:

AnalysisWebComponents/ariba/analytics/en_us/help

Modifications to Help

If you want to change the online help, be sure to centralize your changes into as few files as possible. Doing so will make upgrades to Ariba Analysis much easier than otherwise.

Localized Help

Localized versions of the online help are available from Ariba Connect.

▼ **To install a localized version of the Ariba online help:**

1. Download the desired version from the Ariba Connect site at http://connect.Ariba.com.
2. Unzip the downloaded file into the *AnalysisWebComponents*/ariba/analytics/ directory.
3. Restart Ariba Analysis.

Moving Ariba Analysis into Production

This section contains the following topics about moving Ariba Analysis into production:

Transferring Development Reports and Users

Use the scheduled tasks `ObjectEncoding` to export any report or user definitions from your Ariba Analysis development system and then import them on the production system. See "Importing and Exporting Users: ObjectEncoding" on page 103.

SSL in Production

Ariba strongly recommends that you use SSL with production instances of Ariba Analysis.

In the default configuration, Ariba Analysis performs various checks to make sure that SSL is enabled in a production instance. For example, in a production instance, Ariba Analysis checks the value of the parameter `IncomingHttpServerURL` in the global Ariba Spend Management configuration file `AppInfo.xml` to be sure it uses `https`.

Providing secure services can impede the performance of your Web server. The data encryption and decryption routines used as part of SSL consist of low-level, CPU-intensive mathematical functions. Most Web servers are not designed to perform these calculations in addition to providing normal page retrieval services. Installing your Web server and Ariba Analysis on different computers can improve SSL performance.

Modifying config/Parameters.table

This section describes the parameters to modify in `config/Parameters.table` so that Ariba Analysis uses the Secure Socket Layer (SSL) protocol.

System.Base.Production

Set to `true`.

In the default configuration, this parameter is set to `false`.

System.Base.DisableRPCSecurity

Set to `true`.

System.Base.IncomingHttpServerURL

In the global Ariba Spend Management configuration file `AppInfo.xml`, set to a URL that starts with `https` if you are using SSL or `http` if you are not.

`https://`*`webserver_name:port`*

For example:

`https://myAnalysisFrontEnd.myCompany.com:8100`

For more information about `AppInfo.xml`, see the *Ariba Spend Management Integration Guide*.

System.Base.ResourceURL

Set to a URL that starts with `https` if you are using SSL or `http` if you are not.

`https://`*`webserver_name/virtual_directory`*

For example:

`https://myAnalysisFrontEnd.myCompany.com/AribaAnalysis`

System.Performance.SSLRPC

Set to `false`.

Modifying config/cXML/AMFConfiguration.table

To move Ariba Analysis into production you must change the value of the parameter `DeploymentMode` in the following file:

`AnalysisServerRoot``/config/cXML/AMFConfiguration.table`

Valid values for this parameter are either `test` or `production`. The default at installation is `test`.

Chapter 7

Ariba Analysis Administrative Console

Effective use of the Ariba Analysis web-browser-based administrative console requires that you have at least a basic understanding of Ariba Analysis Ariba Analysis metadata XML, data loading, and scheduled tasks.

- "Work with the Example Data Access Profile Editor" on page 63
- "XML Rules for Limiting Data Access" on page 64

Effective use of the Ariba Analysis web-browser-based administrative console requires that you have at least a basic understanding of Ariba Analysis Ariba Analysis metadata XML, data loading, and scheduled tasks.

Access the Console

Ariba Analysis displays the **Admin Console** menu in the navigation panel for any logged-in user who has the `AnalysisAdmin` permission.

Click **Admin Console** to display the main page of the console.

View Log Files

Ariba Analysis writes information to log files that can help you track or troubleshoot problems. With the administrative console, you can search for particular logs and view their contents.

For details about how to use the administration console to increase or decrease the amount of information recorded in the logs, see "Set Logging Levels" on page 60.

▼ **To see the contents of a specific log file:**

1 On the administrative console main page, click **Log Files**.

2 Use **Search** to locate a log file whose name matches the search string you enter.

Use the pull-down menu to limit your search to **Archive** files (files in the *AnalysisServerRoot*/logs/archive directory) or **Active** files or both.

3 Click **View** to see the contents of the desired log file.

Set Logging Levels

With the **Log Settings** selection in the administrative console navigation panel, you can set the level of detail that Ariba Analysis records in its logs.

How Logging Categories are Grouped

You can use the **Log Category Name** field and **Search** to locate a particular logging category.

For a description of all the logging categories, see the *Ariba Buyer Configuration Reference Guide*.

Logging Levels

The following table explains logging levels. Each level is less detailed than the preceding level.

Log Level	Records...
Debug	Debugging, informational, warning, and error messages
Info	Informational, warning, and error messages
Warning	Warnings and errors
Off	Only errors

Controls for Setting Log Levels

The controls labelled **All On**, **All Off**, and **Reset** have the following effects:

All On	Set all logging categories to Warning level
All Off	Turn off logging, except for errors
Reset	Resets all settings to their original state when Ariba Analysis first displayed the page

With the control **Apply to Sublogs**, you can apply a logging level to an entire group of categories. First, on the main category, set the desired logging level. Then, click **Apply to Sublogs**. Ariba Analysis applies the setting of the main category to all its subcategories.

View Scheduled Tasks and Status

Scheduled tasks are groups of class names and parameters you define to perform operations relating to Ariba Analysis data structures. Scheduled tasks are divided into two types:

1. Data loads, which import data from different sources into the Ariba Analysis database
2. Scheduled tasks, which are run by Ariba Analysis according to schedules you specify

Data load events are grouped by source system, which define a particular source of data. Source systems are represented by subdirectories in *AnalysisServerRoot*`/config`. For information about data loading, see the *Ariba Analysis Data Load Guide*.

▼ To access the Scheduled Tasks page:

- Click **Scheduled Tasks** in the table of contents or on the main administrative console page.

▼ To locate a specific task by name, type, or source system:

1. In the field labelled **Scheduled Task**, enter a string you believe matches the name of the scheduled task.
2. From the **Type** pull-down menu, select either **Data Loads** or **Scheduled Tasks**, or use the default **All Tasks** if the distinction is not important to you.
3. If you are looking for a data load and know the source system to which it belongs, select the name of that source system from the **Source System** pull-down menu.

   Note: Scheduled tasks proper are not grouped into source systems.

4. Click **Search**.

▼ To alter the appearance of the Scheduled Tasks list:

1. Right-click the purple icon on the right side of the page.
2. From the top half of the pull-down menu, select the column of the list you want to show or hide:
   - Task name
   - Type
   - Source System
   - Status
   - End Time
   - Exec Time

▼ To group scheduled tasks for easier viewing:

1. Right-click the purple icon on the right side of the page.
2. From the bottom half of the pull-down menu, select the way you want to group the tasks:
   - Task name
   - Type
   - Source System
   - Status
   - End Time
   - Exec Time

▼ **To view configuration details about a task or source system:**

1 Click the hyperlinked name of the task or source system

2 Examine the details.

- Configuration details about scheduled tasks are from the file *AnalysisServerRoot*/config/DataLoadTasks.table.
- Configuration details about source systems are from the file *AnalysisServerRoot*/config/Parameters.table.

3 Click **Done** to return to the main **Scheduled Tasks** page.

Work with the Example Data Access Profile Editor

This section describes how to use the example Data Access Profile Editor from the administration console. With it, you apply XML-based rules that restrict users' view of data. For details about the configuration of this example access control, see the *Ariba Analysis Customization Guide*. For information about and an example of rules for access control, see "XML Rules for Limiting Data Access" on page 64.

Note: The User Access example menu option is displayed in the Ariba Analysis administrative console only if Ariba Analysis has not been moved into production.

▼ **To get to the Data Access page:**

- Click **Data Access** in the administrative console's table of contents or main page.

  From the **Data Access** page you can locate a user whose access you wish to limit.

▼ **To limit a user's view of data with the Data Access Profile Editor:**

1 Select the user whose profile you wish to edit. You have several ways to do this:

   a Enter the full names, user names, or email addresses of the user you want to locate, and click **Search**.

   b Use the page control to advance to the page that lists the user name.

2 Click the **Edit Access** button that corresponds to the user whose view you wish to limit.

   Ariba Analysis displays the Data Access Profile Editor.

3 Enter XML-based rules into this box to limit this user's data access.

See "XML Rules for Limiting Data Access" on page 64 for details about and an example of such rules.

4 Click **Save** to apply these rules or **Cancel** to discard them.

XML Rules for Limiting Data Access

The Ariba Analysis example rules in XML for access control are inclusive. That is, a rule specifies what data a user is allowed to see, rather than prevents him from seeing it.

If no rule is specified for a user, that user can view all data. If a rule is specified, the user can view all data except the hierarchies to which the rule apply. In those hierarchies, the user can view only the objects specified in the rules. In addition, the user profile rules are subject to any general visibility conditions that might be applied with metadata XML.

The following is an example of some rules:

```
<Extrinsic name="AccessControl">
  <Fact name="ariba.analytics.fact.POLineItem">
    <Field path="Commodity.CommodityName">
       <Value>Computers</Value>
       <Value>Services</Value>
    </Field>
    <Field path="CostCenter.DepartmentName">Marketing</Field>
  </Fact>
</Extrinsic>
```

These rules restrict a user's view of the `POLineItem` fact in three ways:

1 The user has unlimited view of all hierarchies except `Commodity` and `CostCenter`.

2 The user can see data relating to the `Computers` and `Services` categories of the `Commodity` hierarchy, but no other data in the `Commodity` hierarchy.

3 In the `CostCenter` hierarchy, the user can view only data related to the `Marketing` department.

The following simple example allows the user `jqpublic` to view purchase order data from the year 2001.

Browse Metadata XML and Related Tasks

The administrative console's Schema Browser is for looking at relationships among facts, dimensions, their related elements (measures and hierarchies), and the scheduled tasks and data loads related to them. It provides a direct view into these relationships as they are represented in the Ariba Analysis metadata XML. With the Schema Browser you can also export dimension data for editing.

The Schema Browser is similar to the Inspector but pertains only to the interrelationships among Ariba Analysis OLAP data structures, scheduled tasks, and data loading definitions.

To fully appreciate the Schema Browser, you must be familiar with metadata XML. Familiarity with data loading is also helpful. See the *Ariba Analysis Customization Guide*.

The Schema Browser allows you to view relationships among some metadata XML elements, but to change those elements, you must edit the files in which they are defined.

General Navigation in the Schema Browser

In general, the Schema Browser displays pages that contain the hyperlinked names of facts, measures, dimensions, hierarchies, and their associated scheduled tasks and data loads. You can navigate through many different paths to arrive at the same location.

Examining Facts and Measures

This section describes how to examine facts and measures using the administrative console.

▼ To examine the facts:

1 Click **Facts** in the administrative console's navigation panel or main page. To go directly to a specific fact, click the right-point arrow to the left of the word **Facts** to display a list of all fact names directly in the navigation panel.

Ariba Analysis displays the main page listing all facts, a description of them, and the number of database records for each.

- You can change the sort order of the fact names by clicking the arrow to the right of the word **Name** in the column headings.

2 To examine a fact, click the hyperlinked name of the fact.

For example, click **Expense Report**.

General Information for a Fact

The general information for a fact includes the following:

- Fact name
- Class name for this fact
- Description of the fact
- Number of records in the database
- Database table name
- Lists of measures, dimension references, and other fields for this fact

With the list of measures, you can see information about any of this fact's related measures. Information about measures includes their operator, type, and description, as expressed in metadata XML. The detailed page for any single measure also lists the data loads that populated this measure in the database and the source system with which those data loads are associated.

With the list of dimension references, you can see information about any of the dimensions to which this fact refers. See "Examining Dimensions and Hierarchies" on page 67 for information about dimensions and "Exporting, Editing, and Importing a Data Source" on page 69 for details about how to export dimension data.

Fact-related Data Loads

The **Data Loads** tab displays the data load definitions related to the fact you are examining. Click the name of the data-loading event file to see the mapping between the data source and the fields in the fact.

See "View Scheduled Tasks and Status" on page 61 for details about the information about data loads. For information about running data loads and defining and customizing data loading, see the *Ariba Analysis Data Load Guide*.

Examining Dimensions and Hierarchies

▼ This section describes how to examine dimensions and hierarchies using the administrative console. **To examine dimensions, do any of the following:**

1 Click **Dimensions** in the administrative console's navigation panel or main page. To go directly to a specific dimension, click the right-pointing arrow to the left of the word **Dimensions** to display a list of all dimension names directly in the navigation panel.

   Ariba Analysis displays a list of all dimensions, a description of them, the number of database records for each, and **Export** for each.

- You can change the sort order of the dimension names by clicking the arrow to the right of the word **Name** in the column headings.

2 To examine a dimension, click the hyperlinked name of the dimension.

   For example, click **Account**.

General Information for a Dimension

The general information for a dimension includes the following:

- Dimension name
- Class name for this dimension
- Description of the dimension
- Number of records in the database
- Database table name
- List of fact references, that is, facts on which this dimension is based

With the list of fact references, you can see information about any of this dimension's related facts.

Hierarchies and Levels and Fields Tabs

The **Hierarchies** tab displays information about the levels and descriptions of the hierarchies defined for the dimension.

The **Levels & Fields** tab displays important information about the levels of the hierarchies for this dimension, including the data type of the level and which fields are lookup keys.

The name of a field is a hyperlink that displays detailed information about the field, including related data load definitions and the following attributes:

- Label
- Name
- Description
- Type
- Database column name in dot notation, indicating table name and column name

Dimension-related Data Loads

The **Data Loads** tab displays the data load definitions and source systems related to the dimension you are examining. Click the name of the data-loading event file to see the mapping between the data source and the fields in the dimension.

See "View Scheduled Tasks and Status" on page 61 for details about the information about data loads. For information about running data loads and defining and customizing data loading, see the *Ariba Analysis Data Load Guide*.

Data Sources to Modify Dimensions

Use the **Data Sources** screen in the administrative console to edit your dimension data. A *data source* in Ariba Analysis is a database table for exporting the contents of a dimension, hierarchy, or level, and for importing your changes to them.

With a data source you can edit your existing dimension data or insert new dimension data, but you cannot delete data. To delete data, you must reinitialize your database with `initdb -initdb` and reload it from its source.

Using **Data Sources** to edit your dimension data is a multi-step process:

1. Create a data source associated with a dimension, hierarchy, or level.
2. Export the data matching that dimension, hierarchy, or level to a comma-separated value (CSV) file.
3. Edit the CSV file.
4. Import the edited CSV file back to the original data source.
5. Test your changes by viewing them in a pivot table.
6. Run the scheduled task `DataSourceMerger` to make the changes permanent.

Creating a Data Source

Click **Create New Data Source** to associate a data source with a dimension, hierarchy, or level.

1. Enter a descriptive name for your data source.
2. Select either a hierarchy or a level of the dimension you wish to change.
3. Click **Create** or **Cancel**.

Exporting, Editing, and Importing a Data Source

After you've created a data source, export it to a comma-separated value (CSV) file, so you can edit the contents as you desire, and then import those changes back to the data source.

Exporting

On the main **Data Source** page, select the data source and click **Export and select either Export Only Unclassified Data or Export All**.

Ariba Analysis prompts you save or open the file, displays informative messages about the progress of your download, and finally the total number of records exported and the look-up keys.

Editing

Change the data fields of the exported CSV file as is necessary to correct the dimension data.

Importing

On the **Data Source** main page, select the data source and click Import. Ariba Analysis displays a file browse button.

Click **Browse** to locate your edited CSV file, and click **Import**.

After the importing is complete, click **Done** to return to the main **Data Source** page.

Note: Ariba Analysis validates your imported data in the background, because such validation can take some time. The current status of the data source is displayed on the main **Data Sources** page.

Order Your Data Sources: Move Up and Down

Your data sources may be order-dependent. That is, one data source might change the dimension in a way that another data source requires to happen first.

To change the order of your data sources, select the data source you wish to move and click either **Move Up** or **Move Down**.

By default, data source are sorted by their load order. After you **Move Up** or **Move Down**, be sure to look at the rightmost column to see your changes.

Delete a Data Source

Select the data source you want to delete, and click **Delete**. On the displayed screen, click either **Delete** to remove the data source or **Cancel**.

Testing Your Changes

First, select the validated data source to which you have imported some changes. Then click **Test Changes**.

In test mode, you can view the effect of your changes in an Ariba Analysis pivot table.

Making Dimension Edits Permanent

See the *Ariba Analysis Data Load Guide* for a description of the scheduled task `DataSourceMerger`, which you must run in order for your imported edits to become permanent.

Backing Up Data Sources to Disk

See the *Ariba Analysis Data Load Guide* for a description of the `DataSourceDump` scheduled task you can use to dump your data sources to comma-separated value files for safe-keeping.

User Sessions

The administrative console's **User Sessions** feature displays a list of who is currently logged into Spend Management Dashboard.

The information displayed includes the following:

- Name: logged-in user's full name
- Username
- Email Address
- Login Time

Use the **Search** button to search for users' full names, user names, or email addresses.

Appendix A
Command and Tools Reference

This appendix describes programs that you need to configure Ariba Analysis:

All programs are located in *AnalysisServerRoot*/bin and are discussed in detail in this section.

addsourcesystem

Use addsourcesystem to create skeletal files for your source systems, which you can then edit. The command bin/addsourcesystem uses template files in the directory *AnalysisServerRoot*/configureOptions to create source system directories.

Syntax

The syntax of the addsourcesystem command is as follows.

AnalysisServerRoot/bin/addsourcesystem -configOption *templateDirectory* -baseType *templateType* -typeName *nameOfSourceTypeToCreate* -systemName *nameOfSourceSystemToCreate* -connKey *buyerKey* -partition *partitionName* -version *versionNumber*

Options

The addsourcesystem command has the following options.

Option	Description
-configOption	Indicates which set of template files you want to use. There are two allowable values: • baseConfig, a sample of an Ariba Buyer integration • demoConfig, which is a sample set of CSV loading definitions. For this option, you must have chosen to install the demonstration configuration files when Ariba Analysis was installed.
-baseType	Indicates which template sourceType in the *AnalysisServerRoot*/configureOptions directory you want to use. **Warning:** Use the Global baseType with caution. Doing so resets all values in your config/Parameters.table.
-connKey	The business key you want to use in the Ariba application instance's config/cXML/CredentialMap.xml file. This key is need to connect to that other application. The key you specify here is written to the source system definition in Parameters.table. You may then need to modify Ariba Buyer configuration to match this key.
-typeName	The name of the new sourceType directory you want to create
-systemName	The name of the new sourceSystem subdirectory you want to create. This name must be unique among all source system names.
-partition	This optional argument is used only in integrating with Ariba Buyer or other Ariba application that uses partitions.

Configuration Names and Types

The arguments to the addsourcesystem script's -baseType option represent sample implementations of various configurations of data, metadata XML, and definitions of data-loading XML and data load events.The example configurations are represented by directories in *AnalysisServerRoot*/configureOptions, as described here.

Directory Name	Description
ACM	A sample integration of Ariba Category Management
buyer-csv	Comma-separated Value files for simulating an integration with Ariba Buyer
buuyer-None	
buyer-oracle	An Ariba Buyer and Oracle integration
buyer-psoft	An Ariba Buyer and PeopleSoft integration
buyer-sap	An Ariba Buyer and SAP integration
Global	**Warning:** Use the Global baseType with caution. Doing so resets all values in your config/Parameters.table.
initialConfig	A basic configuration

Example

The following command is an example of making a copy of the basic configuration, which is a sample integration with Ariba Buyer, as a basis for your own modifications. The new source type is named PSOFT, and the new source system is buyer1.

```
addsourcesystem -configOption baseConfig -baseType Global -typeName PSOFT
-sourceName buyer1 -connKey buyer1
```

configure

You usually run the program *AnalysisServerRoot*/bin/configure only at installation. It prompts you for various configuration details, such as the name of your Web server, your database connection information, and other parameters. See the *Ariba Analysis Installation Guide* for an example of its use.

The details that the configure program gathers from you are written to Analysis Server configuration files, particularly config/Parameters.table, which is discussed in “Ariba Analysis Parameters.table” on page 83. The file *AnalysisServerRoot*/logs/configuration.txt is a record of the most recent details.

initdb

The program *AnalysisServerRoot*/bin/initdb initializes the Ariba Analysis database and runs data loading events. You run initdb at installation to initialize the database, and you rely on it in daily use for data loading, which is discussed in the *Ariba Analysis Customization Guide*.

The initdb command has a counterpart command runtask, which is used for executing scheduled tasks. See “runtask” on page 79.

Syntax

The usage and syntax of initdb is as follows.

AnalysisServerRoot/bin/initdb *options>*

Options

The initdb command takes the following options.

Option	Description
-emptydb	Drops and recreates all known tables
-emptydbbuiltin	Drops and recreates only built-in tables
-db2stats	Creates database statistics for IBM DB2. Not usually needed.
-debug	Runs initdb inside the Java debugger jdb
-file *file*	Specifies the *file* from which to read load events. Must be used with -loaddb option. Location of *file* is relative to *AnalysisServerRoot*/config. If -file is not used, then config/LoadDB.txt is read.

Option	Description
`-incremental`	Evaluates and runs the clauses contained in `incrementalClause` attributes in the data-loading XML definitions being run. See the *Ariba Analysis Customization Guide*. Must be used with `-loaddb` option.
`-initdb`	The same as: `initdb -emptydb -loadmeta -loaddb` If your database is IBM DB2, the command `initdb -initdb` automatically computes your database statistics with the DB2 command `REORGCHK`.
`-initdbbuiltin`	The same as: `initdb -emptydbbuiltin -loadmeta -loaddb`
`-loaddb`	Loads data by running data load events specified in the file `config/LoadDB.txt`. See the *Ariba Analysis Customization Guide*.
`-loadmeta`	Reloads metadata from XML files
`-readfromdump` *`directory`*	Specifies the *`directory`* from which to read data load definitions previously written with the `-writetodump` option. Must be used with the `-loaddb` option.
`-reshapeDB`	Applies database changes based on changes in metadata XML files. This option updates the database schema after loading extrinsic metadata that affects the shape of the data, such as adding a new fact or dimension. Changes that require the `reshapeDB` option include the following: • Adding a new fact or dimension • Adding a new field to an existing fact or dimension • Creating an index on a table • Adding the unique attribute to a field
`-writetodump` *`directory`*	Specifies a *`directory`* to which a copy of the data being loaded into the Ariba Analysis database is written. This directory can be used to load data again with the `-readfromdump` option. Must be used with the `-loaddb` option.

The Inspector

The Inspector allows you to examine objects in the database, reset some of their values, and see relationships among data objects. For example, one use of the Inspector is to reset users' passwords during development of your Ariba Analysis implementation.

Syntax

To access the inspector, point your web browser to the following URL:

`http://yourAnalysisHost/Analysis/inspector`

You might need to use the `https` instead of `http` if your server is protected by a digital certificate.

The inspector might be password protected. See "Securing the Inspector" on page 25 for more information.

Options

The inspector has no options.

resetdatabaseowner

The program `bin/resetdatabaseowner` is run by the `configure` program when you first install Ariba Analysis.

If you change the IP address of your Ariba Analysis hardware or change the database user name and password, you will receive an exception when the Analysis Server starts that advises you to run `resetdatabaseowner`:

```
Assert failed: The database is initialized by another process.Please confirm
that this machine is the proper owner, run the 'resetdatabaseowner'script, and
rerun this program
```

You may want to check the `config/Parameters.table` to be sure it includes the proper user name and password for connecting to the database. The following example shows only the pertinent parameters:

```
System.Database = {
   AribaDBHostname = some_host_name_here;
   AribaDBUsername = some_DB_user_name;
   AribaDBPassword = some_DB_password;
}
```

After running `resetdatabaseowner`, you should be able to start Ariba Analysis.

Syntax

The `resetdatabaseowner` command has the following syntax:

AnalysisServerRoot`/bin/resetdatabaseowner`

Options

The `resetdatabaseowner` command has no options.

runtask

The `runtask` command executes a single data load event or scheduled task.

For data loading, you should prefer the `initdb` command over the `runtask` command. `initdb` (which is multi-threaded) is more performant than `runtask` (which is single-thread) and is designed to execute many data loading events, whereas `runtask` only executes a single data load event.

Syntax

The runtask command has the following syntax.

AnalysisServerRoot/bin/runtask [-dataload *dataloadEventName* -sourceSystem *sourceSystemName* | -task *taskName*]

Options

The initdb command takes the following options. The -dataload and -task options are mutually exclusive.

Option	Description
-dataload *dataloadEventName*	Runs the *dataload_event_name* defined in *AnalysisServerRoot*/config/*sourceSystemName*/DataLoadEvents.table. Must be used with -sourceSystem option.
-sourceSystem *sourceSystemName*	Indicates the source system directory in *AnalysisServerRoot*/config that contains the DataLoadEvents.table defining the data load event. Must be used with -dataload option.
-task *taskName*	Runs the task defined in *AnalysisServerRoot*/config/DataLoadTasks.table and *AnalysisServerRoot*/config/ScheduledTasks.table.

Examples

The following examples illustrate the use of runtask.

▼ runtask Syntax for executing a task

```
runtask -task LoadFromStaging
```

Runs the LoadFromStaging task defined in *AnalysisServerRoot*/config/DataLoadTasks.table to apply versioning to a slowly changing dimension.

▼ runtask Syntax for data loading

```
runtask -dataload BuyerPOLoad -sourceSystem buyer-psoft
```

Runs the BuyerPOLoad data load event defined in *AnalysisServerRoot*/config/buyer-psoft/DataLoadEvents.table.

startanalysis and stopanalysis

The `startanalysis` and `stopanalysis` command start and stop Ariba Analysis.

Syntax

AnalysisServerRoot`/bin/startanalysis`

AnalysisServerRoot`/bin/stopanalysis`

Options

The `startanalysis` and `stopanalysis` commands have no options.

Appendix B
Ariba Analysis Parameters.table

This appendix contains a list of the parameters specific to the Ariba Analysis `config/Parameters.table`. Ariba Analysis shares a common set of parameters with Ariba Buyer. The *Ariba Buyer Configuration Guide* documents the full set of allowable parameters. See the *Ariba Buyer Configuration Guide* for details on any parameters in the following sections of `Parameters.table` that are not discussed in this appendix:

- `System.Base`
- `System.Debug`
- `System.Logging`
- `System.Nodes`
- `System.PasswordAdapters`
- `System.Performance`

Ariba Buyer 7.x Parameters for Ariba Analysis

There are specific parameters that relate to enabling single sign-on through Ariba Buyer 7.1 for Ariba Analysis users. These are grouped under `System.Analysis` and must be added to the Ariba Buyer `config/Parameters.table`. These are discussed in the *Ariba Spend Management Configuration Guide*.

Application.Analysis Parameters

This section describes the parameters grouped under `Application.Analysis`:

Application.Analysis.BaseCurrency

The default currency for your implementation of Ariba Analysis. If an Ariba Analysis user has not specified any currency preference, Ariba Analysis displays money amounts in base currency.

The default setting is as follows. All three-letter ISO 4217 currency codes, such as FRF, GBP, JPY, or USD, are allowed. Your setting for `BaseCurrency` in Ariba Analysis must match your Ariba Buyer setting of the `System.Base.Data.BaseCurrency` parameter.

Application.Analysis.BaseCurrency = USD;

A related parameter is "Application.Analysis.SupportedCurrencies" on page 87.

Application.Analysis.CSVDelimiters

When data are exported from Ariba Analysis to a personal computer, the default field delimiter in a comma-separated value (CSV) file is a comma. However, different locales require different field delimiters. You can specify the delimiter you want for a locale with the `CSVDelimiters` parameter. For example, to specify that a semi-colon be used for the Swiss locale, set the following:

```
Application = {
  Analysis = {
    CSVDelimiters = {
        de_CH = ";" ;
    }
  }
}
```

For any locale not specified, the comma is the default delimiter.

Note: This parameter has no effect on CSV data loading into Ariba Analysis and only concerns data export.

Application.Analysis.CSVExportEncoding

Microsoft Excel has known issues with reading comma-separated value (CSV) files encoded in UTF-8, especially if the file has non-English characters like umlauts and accents. The parameter `CSVExportEncoding` can be used to control the default character encoding for data exported in CSV files. Any of the following IANA-supported character encoding names is allowed:

- ASCII
- ISO-8859-1
- Big5
- GB2312
- KS_C_5601-1987
- UTF-8
- Shift_JIS

For example, the following entry in Parameters.table sets the character encoding to UTF-8:

```
Application = {
  Analysis = {
    CSVExportEncoding = UTF-8 ;
  }
}
```

Application.Analysis.ExcelExportDisabled

Some organizations might want to disable the automatic launching of Microsoft Excel during data export from Ariba Analysis, for example, to comply with a security policy about the automatic launching of applications from web browsers.

To disable the automatic launching of Excel, set the `ExcelExportDisabled` parameter to `true`. (The default value is `false`.) Users can still save downloaded data as comma-separated values (CSV) files.

```
Application = {
  Analysis = {
    ExcelExportDisabled = true ;
  }
}
```

Note: If Excel downloading is disabled, users cannot upload Excel templates.

Application.Analysis.FiscalMonthShift

`FiscalMonthShift` denotes the offset between a company's calendar months and its fiscal months.

Valid values for `FiscalMonthShift` are any integer from -11 to positive 11. No sign is necessary to indicate a positive integer.

By default, the fiscal month is shifted three months **ahead** of the calendar month. this means that the first month of fiscal year 2002 is October, 2001:

```
Application.Analysis.FiscalMonthShift = 3;
```

The following example indicates that the start of the fiscal year is June:

```
Application.Analysis.FiscalMonthShift = 6;
```

Application.Analysis.MaxChartDataPoints

The maximum number of data points allowed on an Ariba Analysis chart.

The default is as follows:

```
Application.Analysis.MaxChartDataPoints = 50;
```

Application.Analysis.MaxChartDataSeries

The maximum number of data series allowed on an Ariba Analysis chart. A data series is the number of values to be included from a column field on the pivot table. For example, the years 2002, 2003.

The default is as follows:

```
Application.Analysis.MaxChartDataSeries = 5;
```

Application.Analysis.MaxUIStringLength

The maximum length of any string displayed in the Ariba Analysis user interface. Any string exceeding this setting is truncated.

The default is as follows:

```
Application.Analysis.MaxUIStringLength = 50;
```

Application.Analysis.MaxHierarchyPathLength

The maximum length of a path displayed in the pull-down menu on a parameterized report (Analysis Wizard, Step 3) to traverse a hierarchy to a specific level or value.

The default is as follows:

```
Application.Analysis.MaxHierarchyPathLength = 50;
```

Application.Analysis.MaxLineLevelRowsPerLevel

The maximum number of transaction records returned by a database query for any single level in a hierarchy. The default is as follows:

```
Application.Analysis.MaxLineLevelRowsPerLevel = 100;
```

Application.Analysis.ReportsUser

The name of the Ariba Analysis user who owns the Ariba Analysis prepackaged reports. For more information about the prepackaged reports, see the *Ariba Analysis Advanced User Guide*.

The default values are as follows:

```
Application = {
  Analysis = {
     ReportsUser = {
         UniqueName = "ashell";
         PasswordAdapter="PasswordAdapter1";
         };
  }
}
```

Application.Analysis.SupportedCurrencies

A comma-delimited list of currencies supported by your implementation of Ariba Analysis. In addition to `BaseCurrency` (see page 84), users are allowed to choose which of the supported currencies Ariba Analysis should use to display money amounts and other currency-related values.

You must indicate the values for supported currencies as a comma-delimited list of valid currency codes as defined by the standard ISO 4217. For a list of codes, see http://www.unece.org/etrades/codesindex.htm. The default setting allows both the Euro and the Japanese yen:

```
Application.Analysis.SupportedCurrencies = ( EUR, JPY ) ;
```

Ariba Analysis computes materialized views for every supported currency you specify, including the base currency. In addition, Ariba Analysis fact tables also store values in each of the supported currencies. For these reasons, you should not set the number of supported currencies to more than the essential few you need for analysis. The more supported currencies you specify, the greater the performance impact and the greater the database storage requirements.

Application.Base

This section discusses only those parameters in the Application.Base section whose function differs from their use in Ariba Buyer.

Application.Base.Data.DefaultLanguage

The language desired in loading data from Ariba Buyer. Ariba Analysis loads data in only one language. If Ariba Buyer supports multiple languages, use this parameter to specify the language of the data you want to load.

The default value of this parameter is as follows:

```
Application.Base.Data.DefaultLanguage = English ;
```

You can override this value for any particular data load event or source system. See "Default Language from Ariba Buyer" in the *Ariba Analysis Data Load Guide*.

Application.Base.TemplateUser

The name of the user after which new user accounts are modelled.

The default is:

```
TemplateUser = "aribasystem" ;
```

Application.Base.TemplateUsers

In Parameters.table you can define role-based dashboard templates in Application.Base.TemplateUsers. The dashboard of the user named TemplateUser is used as the basis for the default dashboard for all users with the role Senior Analyst:

```
TemplateUsers = {
  Default = {
    PasswordAdapter = PasswordAdapter1;
    UniqueName = TemplateUser;
    };
  SeniorAnalystTemplate = { /* This is predefined in Ariba Analysis */
    PasswordAdapter = PasswordAdapter1;
    Rank = 1;
    Role = "Senior Analyst";
    UniqueName = TemplateUser;
    };
  TestUserTemplate = { /* my own test template user*/
    PasswordAdapter = PasswordAdapter1;
    Rank = 1;
    Role = "Senior Analyst";
    UniqueName = reportTester;
    };
  };
```

Application.ClassificationCodes

The parameters discussed in this section relate to classification codes (also know as commodity codes).

Application.ClassificationCodes.CommodityCodeUIDomainImpl

Specifies an implementation of the interface ariba.basic.util.ClassificationCodeDomainDefaulter. Returns a commodity code domain, which is used for displaying commodity codes in the user interface, and for defaulting when no code is specified (for example, with an ad-hoc item).

In the default configuration, this parameter is set to ariba.basic.core.DefaultCommodityCodeUIDomainImpl, which returns the domain specified as Application.ClassificationCodes.SystemCommodityCodeDomainName.

Application.ClassificationCodes.SystemCommodityCodeDomainName

Defines the default domain for commodity codes, which is the domain used in the user interface and in application logic. To specify additional logic for choosing the default domain, set the parameter `Application.ClassificationCodes.CommodityCodeUIDomainImpl`.

In the default configuration, this parameter is set to `unspsc`.

Application.ClassificationCodes.UnspscAuditFile

Specifies the location of the UNSPSC audit file. This file contains a list of all UNSPSC versions and how they map to each other, so that data can be mapped correctly if an incoming catalog uses a different UNSPSC version.

In the default configuration, this parameter is set to `config/standards/unspscaudit122.csv`.

For more information on the audit file, see *Ariba Buyer Catalog Guide*.

Application.ClassificationCodes.UnspscPinnedVersion

Specifies the UNSPSC version used by Ariba Analysis. All versioned UNSPSC codes in catalogs are translated to this version.

In the default configuration, this parameter is set to `8.01`.

For more information on UNSPSC versions, see *Ariba Buyer Catalog Guide*.

Application.UI

The following are the parameters grouped under Application.UI.

Application.UI.DashboardRefreshInterval

The frequency with which the dashboard is refreshed, expressed in minutes. The default is 5 minutes.

Application.UI.DashboardRefreshTimeOutInSeconds

The maximum amount of time (in seconds) Ariba Analysis takes to refresh dashboard content before finishing the refreshing as a background process. This prevents delay to the user.

The default is 20 seconds.

Application.UI.DashboardForcedRefreshTimeoutInSeconds

SourceSystems Parameters

You must define these parameters for use in data loading, as discussed in the *Ariba Analysis Data Load Guide*

In the group of parameters in `Parameters.table` identified as `SourceSystems`, you must define a label as a way to refer to a specific source system. This label contains path definitions to specify the precise location of the `DataLoadEvents.table` and `ScheduledTask.table` associated with a particular source system.

For example, the following entry in `Parameters.table` comes with Ariba Analysis and indicates the location of the data load event and scheduled task files for the global, default source system:

```
SourceSystems = {
  /* This is the out-of-the-box source system. */
  Default = {
    DataLoadEventsFile =
    "config/sourceTypes/Global/sourceSystems/Default/DataLoadEvents.table";
  };
```

System.Analysis

The `System.Analysis` group of parameters is subdivided into parameters controlling the following:

- `AccessControlManager`
- `DataLoading`
- `Debug`

- OLAP
- ResourceURL
- Server

System.Analysis.AccessControlManager

The name of a custom Java class you write to control which users can view which data in Ariba Analysis. The value for AccessControlManager is a path relative to <AnalysisServerRoot>. For example:

```
System.Analysis.AccessControlManager =
“classes.MyClasses.CustomAccessControl” ;
```

Details about writing a custom access control manager are in “Custom Java for Access Control” on page 70.

System.Analysis.DataLoading

The parameters grouped under DataLoading control the underlying mechanics of how data is retrieved from systems external to Ariba Analysis.

For information about running data loads and defining and customizing data loading, see the *Ariba Analysis Data Load Guide*.

System.Analysis.DataLoading.AQLQueryTimeout

The number of minutes Ariba Analysis waits for data during data loading before timing out the AQL query to the data source. If no data is received within this time, the execution of the data load event is stopped.

The default is 30 minutes.

System.Analysis.DataLoading.BatchSize

The number of database records written to or updated in the Ariba Analysis database with a single COMMIT.

The default is 20 records.

Warning: Because this parameter can adversely affect data-loading performance, Ariba recommends that you change this parameter only on the advice of Ariba Technical support or Ariba Solutions consultants.

System.Analysis.DataLoading.DateFormat

The format of dates.

Dates are specified as follows. The delimiter between the components of the date can be any character other than `d`, `m`, or `y`.

Format	Description
`d`	Day. Must be at least `dd`. The day field is left-padded with zeroes if necessary.
`m`	Month. Must be at least `mm`. The month field is left-padded with zeroes if necessary.
`y`	Year. For example, a four-digit year is specified as `yyyy` and a two-digit year is `yy`.

By default, the date format is the following:

```
DateFormat = "yyyy-mm-dd";
```

System.Analysis.DataLoading.IncrementalParams

Contains parameters that specify the interval and unit for incremental data loading: `Interval` and `Type`.

System.Analysis.DataLoading.IncrementalParams.Interval

Specifies the date interval for incremental data loading. The value of `Interval` can be any integer.

The following example, which is the default setting, indicates that incremental data for the preceding six days will be loaded. If a data load event is run on 10/31/2002, data from 10/24 through 10/30 are selected.

```
IncrementalParams = { Interval = 7; Type = Days;};
```

System.Analysis.DataLoading.IncrementalParams.Type

Specifies the unit of the interval for incremental data loading. The valid values for `Type` are as follows: `Days`, `Weeks`, `Months`, and `Years`. The values do **not** include the current day, week, month, or year.

For the `Weeks` value, Ariba Analysis assumes by default that the end of the week is Friday. If the end of the week is some other day, use the `EndOfWeek` parameter to specify the three-character name of the day:

```
EndOfWeek = Sun;
```

System.Analysis.DataLoading.NoNewSCDRowInterval

The `NoNewSCDRowInterval` parameter in `config/Parameters.table` specifies a time interval in days during which multiple data loading to the same dimension does **not** result in versioned data. This allows you to load partial data without creating new versions.

The default is one day:

```
NoNewSCDRowInterval = 1;
```

System.Analysis.DataLoading.QueryParams.IncrementalPullWindow

Specifies the window of time used in database queries by way of the data-loading XML attribute `incrementalClause`. The valid values for `IncrementalPullWindow` are any integer.

After defining `IncrementalPullWindow` in `Parameters.table`, you use the symbolic notation `IncrementalPullWindow` in your data-loading XML definitions, rather than a hardcoded integer.

The following example defines an offset of 7 units.

```
DataLoading = {
  QueryParams = { IncrementalPullWindow = 7;};
  };
```

See the *Ariba Analysis Data Load Guide* for details about using `IncrementalPullWindow` to construct the actual query used in incremental data loading.

Note: In your data load event definitions for individual events, you can override the system-wide `IncrementalPullWindow` from `Parameters.table`. See the *Ariba Analysis Data Load Guide*.

System.Analysis.Cache

The parameters in `Cache` control the report cache in memory and on disk.

Maximum Cells and Entries for In-Memory Cache

The size of the Ariba Analysis in-memory report cache cannot exceed the limits set by the parameters `InMemCacheMaxTotalCellLimit` and `InMemCacheMaxTotalEntriesLimit` in `config/Parameters.table`.

The default setting is as follows:

```
System.Analysis.Cache.InMemCacheMaxTotalCellLimit = 1000000 ;
System.Analysis.Cache.InMemCacheMaxTotalEntriesLimit = 100 ;
```

Maximum File Cache Limit

The size of the Ariba Analysis file-system-based report cache cannot exceed the limit set by the parameters `InMemCacheMaxTotalCellLimit` and `InMemCacheMaxTotalEntriesLimit` in `config/Parameters.table`.

The default setting is as follows:

```
System.Analysis.Cache.FileCacheMaxTotalEntriesLimit = 150 ;
```

Screening Cache Entries

The `CacheIfLessThanPercentageOfCellLimit` parameter specifies that a query will be cached only if its row count as a percentage of the `InMemCacheMaxTotalRowLimit` is less than the value of `CacheIfLessThanPercentageOfCellLimit`.

This parameter helps prevent a single large result set (such as line-item detail, for example) from clearing out a number of cache entries so that the `InMemCacheMaxTotalRowLimit` threshold can be enforced.

By default, the parameter is set as follows:

```
System.Analysis.Cache.CacheIfLessThanPercentageOfCellLimit = "12";
```

System.Analysis.Debug.CacheDisabled

You can completely disable to the report cache with the `CacheDisabled` parameter in `config/Parameters.table`.

By default `CacheDisabled` is as follows:

```
System.Analysis.Debug.CacheDisabled = "false";
```

To disable the cache, set `CacheDisabled` to true, and restart Ariba Analysis.

System.Analysis.OLAP

The parameters in `OLAP` define characteristics of the hypercube underlying an analytical report, such as limits on the amount of data or labels displayed in the user interface, and other OLAP-related behavior. The main purpose of this group of parameters is to provide a fast user experience.

System.Analysis.OLAP.BackgroundQueryTimeOut

The timeout for database queries for reports run in the background cannot exceed the the limit set by the parameter `BackgroundQueryTimeOut` in `config/Parameters.table`.

The default setting is as follows:

```
System.Analysis.OLAP.BackgroundQueryTimeOut = 20 ;
```

This value is the number of seconds that can elapse before a database query will timeout.

System.Analysis.OLAP.CubeInitialWaitTime

How long Ariba Analysis waits for a response for the first database query against the hypercube underlying any given analytical report before refreshing the HTML page from which the query was issued.

The default initial wait time is as follows:

```
System.Analysis.OLPA.CubeInitialWaitTime = 10;
```

By default, for any query against the hypercube, Ariba Analysis waits up to 10 seconds. If the query is done within 10 seconds, the HTML result is displayed, which could be the analytical report pivot table, pie chart or bar chart. Otherwise, Ariba Analysis runs the query in the background.

The user is informed that a long-running query is in progress until the query is done, the operation is cancelled, or the query times out.

System.Analysis.OLAP.CubeMaxRowLimit

The maximum number of records Ariba Analysis allows on the row edge of the hypercube underlying an analytical report.

By default, `CubeMaxRowLimit` is as follows:

```
System.Analysis.OLAP.CubeMaxRowLimit = 5000;
```

System.Analysis.OLAP.CubeQueryTimeout

The number of seconds allowed in the database for any query against the hypercube, which by default is 180 seconds:

```
System.Ananlysis.OLAP.CubeQueryTimeout = 180;
```

For example, suppose a user modifies his analytical report such that Ariba Analysis will return an extremely large amount of data. Furthermore, suppose that such a query takes the database 185 seconds to return the first row.

During the first 10 seconds of processing of this query, Ariba Analysis displays the refresh page described under “System.Analysis.OLAP.CubeInitialWaitTime” on page 96. The refresh page polls the Analysis Server and continues to inform the user that a long-running query is being executed.

After 180 seconds, the background query operation times out, and the next time refresh page polls the server, the user is notified that query has timed out.

System.Analysis.OLAP.DashboardGenerationTime

The number of seconds to generate a user's dashboard items before processing queries in the background.

Some users' dashboards might contain analytical reports whose queries take more time to return results from the database than the users are willing to wait. If a dashboard item is not already in cache, Ariba Analysis tries to generate it in less than the value of this parameter. Otherwise, if this threshold is exceeded, the item is generated in the background, and Ariba Analysis displays a message in the dashboard to inform the user.

By default AnalysisDashboardGenerationTime in seconds is as follows:

```
System.Analysis.OLAP.DashboardGenerationTime= 10 ;
```

System.Analysis.OLAP.DatabaseConnections

The number of connections to the database that Ariba Analysis must keep open to process queries against hypercubes.

By default DatabaseConnections is as follows:

```
System.Analysis.OLAP.DatabaseConnections= 7;
```

System.Analysis.OLAP.DBCacheMaxTotalEntriesLimit

The maximum number of database tables for the secondary report cache.

By default DBCacheMaxTotalEntriesLimit is as follows:

```
System.Analysis.OLAP.DBCacheMaxTotalEntriesLimit = "140";
```

Secondary cache table names have the prefix CACHE_ANALYSIS.

System.Analysis.OLAP.LineLevelQueryRowLimit

The maximum number of transaction records returned by a query. The default is as follows:

```
System.Analysis.OLAP.LineLevelQueryRowLimit = 8000;
```

System.Analysis.OLAP.MaxColumnEdgeCells

The maximum number of cells Ariba Analysis allows on the column edge of the hypercube underlying an analytical report.

By default, `MaxColumnEdgeCells` is as follows:

```
System.Analysis.OLAP.MaxColumnEdgeCells = 25;
```

System.Analysis.OLAP.MaxColumnsOnDashboard

The maximum number of columns displayed for an analytical report or summarized view that a user adds to the Spend Management Dashboard.

By default is `MaxColumnsOnDashboard` as follows:

```
System.Analysis.OLAP.MaxColumnsOnDashboard = 5;
```

System.Analysis.OLAP.MaxRowEdgeCells

The maximum number of cells Ariba Analysis allows on the row edge of the hypercube underlying an analytical report.

By default, `MaxRowEdgeCells` is as follows:

By default is `MaxColumnsOnDashboard` as follows:

```
System.Ananlysis.OLAP.MaxColumnsOnDashboard = 200;
```

System.Analysis.OLAP.MaxRowsOnDashboard

The maximum number of rows displayed for an analytical report or summarized view that a user adds to the Spend Management Dashboard.

By default is `MaxRowsOnDashboard` as follows:

```
System.Analysis.OLAP.MaxRowsOnDashboard = 20;
```

System.Analysis.OLAP.MemberQueryTimeout

The number of seconds allowed in the database to retrieve all possible members for display on a pull-down menu. The term *members* in this context means the selectable values a user of an analysis can pick from a pull-down menu to drill down into a column, page, or row field (that is, a hierarchy).

To conserve storage space and ensure fast performance, not all possible members are necessarily stored in the analytical report hypercube. If the hypercube does not have all members, the hypercube must execute a database query to retrieve the members.

`MemberQueryTimeout` specifies how much time Ariba Analysis allows for this type of query. If the database server does not process the query in the number of seconds specified by `MemberQueryTimeout`, no drill down members are displayed.

The default value is as follows:

```
System.Ananlysis.OLAP.MemberQueryTimeout = 10;
```

System.Analysis.ResourceURL

The URL for Ariba Analysis to display its HTML and graphics. The value of this parameter equates to a web server directory mapping (also called an "alias" or "virtual directory," depending on your web server) that points to the directory *<AnalysisInstallRoot>*`/WebComponents`.

`System.Ananlysis.ResourceURL` has no default, because it depends on a value you specify at installation.

System.Analysis.Server

`System.Analysis.Server` parameters control the functioning of the server-related aspects of Ariba Analysis.

System.Analysis.Server.RecycleDayOfWeek

Controls the day of the week when Ariba Analysis restarts itself. Valid values for `RecycleDayOfWeek` are either the word `Everyday` or the names of the days of the week: `Monday` through `Sunday`.

If `RecycleDayOfWeek` is null, Ariba Analysis restarts itself daily.

Note: The time of the restart is set by the parameters RecycleWindowEnd and RecycleWindowStart. The RecycleDayOfWeek parameter is ignored if the parameters RecycleWindowEnd and RecycleWindowStart are null.

The following example restarts Ariba Analysis between 6:10PM and 6:13PM each Tuesday:

```
RecycleDayOfWeek = "Tuesday";
RecycleWindowEnd = "18:13";
RecycleWindowStart = "18:10";
```

System.Analysis.Server.RecycleWindowEnd, RecycleWindowStart

These parameters control when Ariba Analysis restarts itself.

The default values are null, which disables the automatic restart.

```
System.Analysis.Server.RecycleWindowStart = “”;
System.Analysis.Server.RecycleWindowEnd = “”;
```

Between Recycle.Window.Start and Recycle.Window.End, Ariba Analysis does not accept any new incoming requests so that it can finish processing those it has already received. The server restarts when either all requests received prior to Recycle.Window.Start have been serviced or when Recycle.Window.End is reached, whichever occurs first.

The format of the value of these parameters is wall-clock time specified in 24 hour (military) notation: HH:MM where HH is the two-digit hour and MM the two-digit minute.

For example, with the default settings, the Analysis Server will stop accepting new requests at 2:30AM while it still continues to process requests it received before that time. Once all requests have been processed, or at 4:30AM, the Analysis Server will restart itself.

System.Base

The parameters grouped under System.Base define the specifics about the basic workings of the system.

System.Base.DefaultLocale

Establishes the default language location system-wide. The default value is as follows:

```
System.Base.DefaultLocale = en_US;
```

System.Base.MainServletName

The name of the primary servlet that constitutes the Ariba Analysis application. The default value is as follows:

```
System.Base.MainServletName = Main;
```

System.Database

The parameters grouped under `Database` define some specifics about the database you have chosen to use with your Ariba product.

System.Database.DB2Optimization

Specifies the optimization level of queries in IBM DB2.

The default value is:

```
System.Database.DB2Optimization = 5;
```

Note: Do not change this value without express consultation with Ariba.

System.J2EEServer

The parameters grouped under `J2EEServer` define the specifics about the application server you have chosen to use with your Ariba product.

This group of parameters is fully documented in the *Ariba Buyer Application Server Guide*.

Appendix C
Scheduled Task Reference

Tasks are compiled Java classes you can invoke to affect the data stored in Ariba Analysis. There are two kinds of tasks: data load tasks and scheduled tasks. Data load tasks are described in the Ariba Analysis *Data Load Guide*. Scheduled tasks, which are run by Ariba Analysis itself based upon user-specified schedules, are described in this appendix.

Frequency of scheduled tasks is defined in the file `config/ScheduledTasks.table`, located in the *`AnalysisServerRoot`*`/config` directory. For information about the structure of `ScheduledTasks.table`, see "Scheduled Tasks" on page 35.

This appendix divides tasks into the following major categories:

Class Packaging

Unless otherwise detailed in the discussion of individual tasks, the task classes described here come with Ariba Analysis and are in the following Java package:

`ariba.analytics.tasks.`*`taskNameHere`*

Importing and Exporting Users: ObjectEncoding

The `ObjectEncoding` task allows you to import or export the definitions of a users' dashboard and folders. It is useful for moving user definitions from one Ariba Analysis installation to another.

The details about a users are stored in an XML-based format in files and folders whose names and structure mirror the hierarchy of the folder user interface in Ariba Analysis. You can edit these XML files with care. Although the encoding format is based on the Simple Object Access Protocol (SOAP), it is not intended for extension by customers.

In the following example, which is from the global, default `DataLoadTasks.table`, the definitions of the dashboard and folders for several users are imported from the directories in *`AnalysisServerRoot`*`/sample/dumpUsers`:

```
LoadUsers = {
  ScheduledTaskClassName = "ariba.analytics.tasks.ObjectEncoding";
  Operation = "Import";
  Directory = "sample/dumpUsers";
  };
```

Parameters and their definitions for the `ObjectEncoding` task are as follows.

Parameter	Allowable Values	Description
`Operation`	`Import` `Export`	Specifies whether you import or to export dashboard and folder definitions.
`Directory`	`Import` `Export`	The directory from which or to which the definitions are imported or exported. This directory name is relative to *`AnalysisServerRoot`*.

`ObjectType`	`User`
	`Folder`
	`FolderItem`
		Specifies the type of object being imported or exported. If not included, all users and their folders and folder contents are acted on.

`UserName`	*`username`*	The name of the user whose dashboard and folder definitions are being imported or exported when `ObjectType=User`.
`FolderPath`	`pathToFolder`	

Creating New Users with Import

You can use the import function of the `ObjectEncoding` task to create new users in Ariba Analysis. To do so, export one user first, and then edit the exported data in the XML-based definition file, or look at the definitions in the *`AnalysisServerRoot`*`/sample/dumpUsers` directories.

The following are some considerations about creating new users with `ObjectEncoding`.

- Be careful when you edit the XML defining a user. Corrupted definitions cannot be loaded. You will have to correct them before you can successfully load.
- Be sure to specify a password for the user in the definition file in clear text. The XML definition looks like the following:

```
<Password>yourPasswordInCleartextHere</Password>
```

Importing Reports Previously Exported

You can use the import function of the `ObjectEncoding` task to import previously exported reports into Ariba Analysis. To do so, use the following parameters in your task definition:

- `ObjectType=Folder` or `ObjectType=FolderItem`
- `FolderPath=`*`pathToFolder`*

Look at the definitions in the *`AnalysisServerRoot`*`/sample/dumpUsers` directories. When analytical reports are imported, if the data to which those analytical reports refers is not present in the Ariba Analysis database, the importing succeeds, but the hierarchy or construct to which the analytical report refers is removed silently from the analytical report. Ariba Analysis writes warning messages to the loading log to indicate which constructs it removed.

Exporting User Definitions and Reports: DumpSystem

The task `DumpSystem` relies on the `ObjectEncoding` task to write a SOAP-encoded representation of all user data and report definitions. One use of `DumpSystem` is to capture all user data prior to migrating to a new release.

`DumpSystem` is defined in the global, default source system. as follows:

```
DumpSystem = {
  ScheduledTaskClassName = "ariba.analytics.tasks.SystemDump";
  Operation = "Export";
  Directory = "systemDump";
};
```

ArchiveLog

```
ArchiveLog = {
  ScheduledTaskClassName = "ariba.util.log.scheduler.ScheduledLogArchive";
    Schedules = {
    Schedule1 = {
       DayOfWeek = Weekday;
       Hour = 23;
       Minute = 59;
       Second = 30;
       };
    };
    ExecutionNode = "AllNodes";
  };
```

The `ArchiveLog` task recycles the log files for Ariba Analysis. When it runs, it takes the current log file, moves it into the archive directory, and starts logging to a new file. It recognizes one additional parameter:

The `ExecutionNode` parameter specifies that the task will run on all nodes (`AllNodes`) in your configuration.

FailedDocumentMessages

Users of Ariba Analysis can create alerts that trigger notifications of exceptions (violations of defined thresholds) to both Ariba Enterprise Sourcing andAriba Category Management. Ariba Analysis sends notifications to users of Ariba Category Management about triggered alerts to interested parties who have associated analytical reports with their projects in Ariba Category Management. For more about creating alerts, see the Ariba Analysis online help and the *Ariba Analysis Advanced User Guide*.

In Ariba Analysis, the `FailedDocumentMessages` scheduled task is an error-recovery task that attempts to retransmit alert notifications to Ariba Category Management.

```
FailedDocumentMessagesTask = {
  ScheduledTaskClassName = "ariba.analytics.tasks.FailedDocumentMessages";
  Schedules = {
    Schedule1 = {
       DayOfWeek = Everyday;
       Hour = 15;
       }
    }
```

```
}
```

LoadDBDonePoller

The `LoadDBDonePoller` scheduled task monitors the progress of data loading. It is primarily used internally by Ariba Analysis to invalidate the user report cache when data loading is complete.

The default polling interval is 10 minutes,.

```
LoadDBDonePoller = {
  ScheduledTaskClassName = "ariba.analytics.tasks.LoadDBDonePoller";
  Schedules = {
    Schedule1 = { Period = { Quantity = 10; Unit = Minutes;};};
    };
  };
```

ScheduledQueryOperation

The `ScheduledQueryOperation` scheduled task runs an analytical report according to a schedule you specify. There is no limit to the number of reports that you can schedule, but each report must have a separate entry in `config/ScheduledTasks.table`.

Note: This task is in the class package **`ariba.analytics.core`**.

`ScheduledQueryOperation` accepts two forms of the name of a report to schedule: the unique name (`UniqueReportName`) or a directory path (`ReportPath`).

UniqueReportName

The task parameter `UniqueReportName` is the database unique name of a report. The unique name of a report can be viewed with the Inspector.

Example:

```
Report_104 = {
  ScheduledTaskClassName =
    ariba.analytics.core.ScheduledQueryOperation;
  UniqueReportName = Report_104;
```

```
Schedules = { Schedule1 = { DayOfWeek = Everyday; Hour = 15; }; };
}
```

ReportPath

The value for the task parameter `ReportPath` is a directory path to the desired report.

Note: The directory path must begin with a user name, followed by the literal string ^c (an up-arrow or caret and the letter "c"), followed by the rest of the path to the report and the report's name. The path to a report is a displayed in the Ariba Analysis folder UI.

Example:

```
ReportPath_Spend_By_Category = {
  ScheduledTaskClassName = ariba.analytics.core.ScheduledQueryOperation;
  ReportPath = "//ashell^cPasswordAdapter1/Personal Folders/Public/PO
  Analysis/Spend by Category" ;
  Schedules = { Schedule1 = { DayOfWeek = Everyday; Hour = 15; }; };
  }
```

Migration Aid: DumpReport

Use the `DumpReport` scheduled task to extract report data from the database into CSV files.

You can dump report data for a single user. By default, CSV output is written to the *AnalysisServerRoot*/`dumpReports` directory. The following example is from the Ariba Analysis 3.0 `DataLoadEvents.table` configuration file:

```
DumpReports = {
  ScheduledTaskClassName = "ariba.analytics.migration.DumpReport";
  User = "userLoginNameHere";
  Directory = "dumpReports";
  };
```

Appendix D
Ariba Analysis Glossary

These are definitions of commonly used terms in Ariba Analysis.

Term	Definition
80/20 rule	A widely used Pareto analysis that divides data into a ratio of most/least. For example, 80% of your spend may be accounted for by only a handful of suppliers, while the remaining 20% may represent a large number of suppliers.
AML	See *metadata XML*.
analysis	See *analytical report*.
analytical report	An organized collection of measures and hierarchies designed to investigate particular aspects spend-related facts. An analytical report is manifested as a *pivot table*. See also *reports*.
Applied Filters	The area across the top of the *pivot table* of an analytical report that shows what constraints have been imposed on the data and what level of data is currently displayed in the pivot table. Click any level in the Applied Filters to return the pivot table to trat level.
bucket	A defined grouping of data. For example, a range of dates or of amounts.
calculated field	See *derived measure.*
collapse	To view less data by hiding a *level* in a *hierarchy.*
column edge, column fields	The area across the top of a *pivot table* directly above the data fields where *data fields* or *line-level detail* are placed. Column fields correspond to the columns in a traditional spreadsheet.

Term	Definition
compound report	In Ariba Analysis, a compound report is a user-defined composite of pivot tables, charts, and summarized views from existing analytical reports. Compound reports allow you to view many different charts and tables at a single glance.
computed field	See *derived measure.*
constrain	To reduce the amount of data actively being investigated in an analytical report by applying some exclusion, for example, selecting a particular level of a hierarchy or a particular value in a hierarchy, or selecting a range of dates.
data fields	The area in the middle of a *pivot table* holds the data fields. Also called *measures*, data fields are created in several ways: • *Pre-defined data fields* are created by summing or aggregating *line-level details*, which are transaction data. • *User-defined fields* are created by you by calculating them based on other data fields. The data field area of the pivot table also holds *line-level details,* the basic accounting transactions from which all other fields are derived.
derived measure	A measure computed from some other measures, called *user-defined fields*.
dimension	An aspect of a fact that is of some interest and has some useful purpose in an analytical report. For example, **Supplier** and **Commodity** are dimensions of purchase orders and invoices, just as start date is a dimension of a project or end date is a dimension of a contract.
drill-down	To show finer detail by going to the next level of a hierarchy.
edge	Term referring to the underlying structure of an *OLAP* hypercube represented by the *pivot table*. A hypercube has *row*, *column*, and *page* edges.
expand	To view more data by exposing the next lower *level* of a *hierarchy.*

Term	Definition
export	To extract data from Ariba Analysis and download it to your personal computer.

fact	In traditional OLAP terminology, facts represent the subject—the interesting pattern or event in the enterprise that must be analyzed to understand its behavior. They represent the subject of your investigation. The following are the pre-defined facts in Ariba Analysis:
	• Purchase orders and PO delivery information: commodity purchasing patterns
	• Expense reports, expense report lines and violations: travel expense spending patterns
	• Invoices, invoice lines and exceptions: supplier billing patterns
	• Contracts and contract clauses: agreements established with suppliers from Ariba Contract Workbench
	• Requests: a general term for employee requisitions, travel requests, and similar data, including collaborative requisitions
	• User activity: Ariba Buyer usage and information
	• Projects: Ariba Category Management- and Ariba Contract Workbench-related facts
	  • Contract projects
	  • Sourcing projects
	  • Supplier Performance Management projects and tasks:
	• RFX summaries and awards: Ariba Enterprise Sourcing requests for proposal or information
	• Surveys and scorecards Ariba Enterprise Sourcing supplier evaluation data
	• Temporary labor and time sheets: services spend information
	• Proposals
	•

Field Browser	The palette on the left of an Ariba Analysis pivot table that lists fields that have been added to or can be edited from an analytical report.

Term	Definition
grade	A specialized form of *user-defined field* that calculates a numerical score based on the value of the data.
hierarchy	A stratified collection of data characterized by *levels*. Any particular level has finer-grained data than the level above it and coarser-grained data than the levels beneath it, unless the level is at the top or bottom of the hierarchy.
inline dimension	A *fact* in Ariba Analysis that can be placed on the row, column, or page edges of the Ariba Analysis pivot table, just like a dimension.
interval	A period of time between two events. For example, Ariba Analysis calculates the PO-to-Invoice Interval as the number of days between the PO or ER date and the corresponding invoice date. A negative number means the invoice was created before the PO, which is not desirable. This calculated interval of days is used to classify invoices into several buckets of time ranges, such as 3 to 6 weeks.
KPI	Key Performance Indicator
level	A stratum of a certain granularity of the data in a hierarchy. For example, “month” is a level lower in the time hierarchy than “year.”
line-level detail	The lowest level of detail in a fact. That is, the individual transactions. See also *data field*.
measure	A numerical property of a fact or a calculation based on such a property. For example, Amount is a measure of a purchase order. Measures are also known as *data fields*.
metadata XML	Data structures and data loading are defined in Ariba Analysis with metadata XML. For data structures, this XML is contained in files ending with `*.aml`. For data-loading definitions, files end with `*.xml`.
non-rollup hierarchy	A *hierarchy* in which the total is not represented by a straight sum of the levels, such as a score of “Pass” or “Fail.”

Term	Definition
OLAP	On Line Analytical Processing, a specialized architecture of databases for dealing with aggregation of large datasets. OLAP systems are characterized by *facts*, *measures*, *dimensions*, and *hierarchies*, represented by fields on a hypercube or *pivot table*. For a comprehensive discussion of OLAP, see (for example): `http://infolab.usc.edu/csci585/Spring2002/den_ar/pederson_p40.pdf`
page edge, page field	The area across the top of a *pivot table* directly above the column fields where *data fields* (or *measures*) are placed that can act as a filter on the other fields.
parameterized report	An analytical report that allows parameters to constrain the data. You set parameters such as dates, money amounts, or hierarchies and levels to specific values you are interested in.
pivot table	A spreadsheet-like structure consisting of page fields, row fields, column fields, and data fields that allows you to manipulate the view of the data. It is the end-product of creating an analytical report. See also *Field Browser.*
ragged hierarchy	A *hierarchy* in which the top-level contains data that must be summed to make an accurate total.
range	In a spreadsheet, a collection of cells.
report	Ariba Analysis has three kinds of reports: • *analytical* • *parameterized* • *compound* See the definitions of these terms for distinctions.
row edge, row fields	The area down the left side of a *pivot table* next to the data fields where *dimensions* are placed. Row fields correspond to the rows in a traditional spreadsheet.
slice and dice	See *constrain.*

Term	Definition
slowly changing dimension	A dimension whose individual values might change over time. Also called *versions* or *versioning*. The values of the dimension that change are recorded for auditing.
split line-item	A single line of a purchase order or expense report that is charged to more than one General Ledger account.
summarized view	In an Ariba Analysis compound report, a summarized view is a user-created table that extracts specific values or totals of values from analytical reports to reveal relationships about multiple facts that might not otherwise be readily visible.
UNSPSC	The Universal Standard Products and Services Classification code is a scheme that classifies and identifies commodities. It is used in sell-side and buy-side catalogs and as a standardized account code in analyzing expenditure (Spend Analysis). See `http://eccma.org/unspsc/`
UOM	Unit (or units) of measure
user-defined field	See *derived measure.*
versioning	See *slowly changing dimension*.

Index

Third-Party Software

Apache Software

The Apache Software License, Version 1.1

Copyright (c) 2000 The Apache Software Foundation. All rights reserved.

Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met:

1. Redistributions of source code must retain the above copyright notice, this list of conditions and the following disclaimer.

2. Redistributions in binary form must reproduce the above copyright notice, this list of conditions and the following disclaimer in the documentation and/or other materials provided with the distribution.

3. The end-user documentation included with the redistribution, if any, must include the following acknowledgment: "This product includes software developed by the Apache Software Foundation (http://www.apache.org/)." Alternately, this acknowledgment may appear in the software itself, if and wherever such third-party acknowledgments normally appear.

4. The names "Apache" and "Apache Software Foundation" must not be used to endorse or promote products derived from this software without prior written permission. For written permission, please contact apache@apache.org.

5. Products derived from this software may not be called "Apache," nor may "Apache" appear in their name, without prior written permission of the Apache Software Foundation.

THIS SOFTWARE IS PROVIDED "AS IS" AND ANY EXPRESSED OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE APACHE SOFTWARE FOUNDATION OR ITS CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

This software consists of voluntary contributions made by many individuals on behalf of the Apache Software Foundation. For more information on the Apache Software Foundation, please see <http://www.apache.org/>.

org.apache.bcel 5.0

4. The names "Apache," "Apache Software Foundation," and "Apache BCEL" must not be used to endorse or promote products derived from this software without prior written permission. For written permission, please contact apache@apache.org.

org.apache.commons.collections 2.1

4. The names "The Jakarta Project," "Commons," and "Apache Software Foundation" must not be used to endorse or promote products derived from this software without prior written permission. For written permission, please contact apache@apache.org.

org.apache.lucene 1.2

4. The names "Apache," "Apache Software Foundation," and "Apache Lucene" must not be used to endorse or promote products derived from this software without prior written permission. For written permission, please contact apache@apache.org.

5. Products derived from this software may not be called "Apache," or "Apache Lucene," nor may "Apache" appear in their name, without prior written permission of the Apache Software Foundation.

org.apache.oro 2.0

4. The names "Apache," "Apache Software Foundation," and "Jakarta-Oro" must not be used to endorse or promote products derived from this software without prior written permission. For written permission, please contact apache@apache.org.

5. Products derived from this software may not be called "Apache" or "Jakarta-Oro," nor may "Apache" or "Jakarta-Oro" appear in their name, without prior written permission of the Apache Software Foundation.

Portions of this software are based upon software originally written by Daniel F. Savarese. We appreciate his contributions.

org.apache.poi.hssf 1.5.1

Portions of this software are based upon public domain software originally written at the National Center for Supercomputing Applications, University of Illinois, Urbana-Champaign.

org.apache.regexp 1.2

4. The names "The Jakarta Project," "Tomcat," and "Apache Software Foundation" must not be used to endorse or promote products derived from this software without prior written permission. For written permission, please contact apache@apache.org.

org.apache.soap 2.2

4. The names "SOAP" and "Apache Software Foundation" must not be used to endorse or promote products derived from this software without prior written permission. For written permission, please contact apache@apache.org.

org.apache.tomcat 3.2.4

4. The names "The Jakarta Project," "Tomcat," and "Apache Software Foundation" must not be used to endorse or promote products derived from this software without prior written permission. For written permission, please contact apache@apache.org.

org.apache.xerces-1.4

4. The names “Xerces” and “Apache Software Foundation” must not be used to endorse or promote products derived from this software without prior written permission. For written permission, please contact apache@apache.org.

This software consists of voluntary contributions made by many individuals on behalf of the Apache Software Foundation and was originally based on software copyright (c) 1999, International Business Machines, Inc., http://www.ibm.com. For more information on the Apache Software Foundation, please see <http://www.apache.org/>.

com.jclark 1.0

CONTAINS software developed by James Clark

Copyright (c) 1997, 1998 James Clark

Permission is hereby granted, free of charge, to any person obtaining a copy of this software and associated documentation files (the “Software”), to deal in the Software without restriction, including without limitation the rights to use, copy, modify, merge, publish, distribute, sublicense, and/or sell copies of the Software, and to permit persons to whom the Software is furnished to do so, subject to the following conditions:

The above copyright notice and this permission notice shall be included in all copies or substantial portions of the Software.

THE SOFTWARE IS PROVIDED “AS IS” WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL JAMES CLARK BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE.

Infozip.zip 2.3

CONTAINS software developed by Info-ZIP

Copyright (c) 1990-2003 Info-ZIP. All rights reserved.

For the purposes of this copyright and license, “Info-ZIP” is defined as the following set of individuals:

Mark Adler, John Bush, Karl Davis, Harald Denker, Jean-Michel Dubois, Jean-loup Gailly, Hunter Goatley, Ian Gorman, Chris Herborth, Dirk Haase, Greg Hartwig, Robert Heath, Jonathan Hudson, Paul Kienitz, David Kirschbaum, Johnny Lee, Onno van der Linden, Igor Mandrichenko, Steve P. Miller, Sergio Monesi, Keith Owens, George Petrov, Greg Roelofs, Kai Uwe Rommel, Steve Salisbury, Dave Smith, Christian Spieler, Antoine Verheijen, Paul von Behren, Rich Wales, Mike White

Infozip.zip 2.3, continued

This software is provided “as is,” without warranty of any kind, express or implied. In no event shall Info-ZIP or its contributors be held liable for any direct, indirect, incidental, special or consequential damages arising out of the use of or inability to use this software.

Permission is granted to anyone to use this software for any purpose, including commercial applications, and to alter it and redistribute it freely, subject to the following restrictions:

1. Redistributions of source code must retain the above copyright notice, definition, disclaimer, and this list of conditions.

2. Redistributions in binary form (compiled executables) must reproduce the above copyright notice, definition, disclaimer, and this list of conditions in documentation and/or other materials provided with the distribution. The sole exception to this condition is redistribution of a standard UnZipSFX binary (including SFXWiz) as part of a self-extracting archive; that is permitted without inclusion of this license, as long as the normal SFX banner has not been removed from the binary or disabled.

3. Altered versions--including, but not limited to, ports to new operating systems, existing ports with new graphical interfaces, and dynamic, shared, or static library versions--must be plainly marked as such and must not be misrepresented as being the original source. Such altered versions also must not be misrepresented as being Info-ZIP releases--including, but not limited to, labeling of the altered versions with the names “Info-ZIP” (or any variation thereof, including, but not limited to, different capitalizations), “Pocket UnZip,” “WiZ” or “MacZip” without the explicit permission of Info-ZIP. Such altered versions are further prohibited from misrepresentative use of the Zip-Bugs or Info-ZIP e-mail addresses or of the Info-ZIP URL(s).

4. Info-ZIP retains the right to use the names “Info-ZIP,” “Zip,” “UnZip,” “UnZipSFX,” “WiZ,” “Pocket UnZip,” “Pocket Zip,” and “MacZip” for its own source and binary releases.

ARIBA, INC.

807 11th Avenue
Sunnyvale, CA 94089 USA
Telephone: 650.390.1000
Fax: 650.390.1100

www.ariba.com