



Teradata Aster R User Guide

Release Number: 6.20
Product ID: B700-2010-620K
August, 2015

The product or products described in this book are licensed products of Teradata Corporation or its affiliates.

Teradata, Active Data Warehousing, Active Enterprise Intelligence, Applications-Within, Aprimo Marketing Studio, Aster, BYNET, Claraview, DecisionCast, Gridscale, MyCommerce, QueryGrid, SQL-MapReduce, Teradata Decision Experts, "Teradata Labs" logo, Teradata ServiceConnect, Teradata Source Experts, WebAnalyst, and Xkoto are trademarks or registered trademarks of Teradata Corporation or its affiliates in the United States and other countries.

Adaptec and SCSISelect are trademarks or registered trademarks of Adaptec, Inc.

AMD Opteron and Opteron are trademarks of Advanced Micro Devices, Inc.

Apache, Apache Avro, Apache Hadoop, Apache Hive, Hadoop, and the yellow elephant logo are either registered trademarks or trademarks of the Apache Software Foundation in the United States and/or other countries.

Apple, Mac, and OS X all are registered trademarks of Apple Inc.

Axeda is a registered trademark of Axeda Corporation. Axeda Agents, Axeda Applications, Axeda Policy Manager, Axeda Enterprise, Axeda Access, Axeda Software Management, Axeda Service, Axeda ServiceLink, and Firewall-Friendly are trademarks and Maximum Results and Maximum Support are servicemarks of Axeda Corporation.

Data Domain, EMC, PowerPath, SRDF, and Symmetrix are registered trademarks of EMC Corporation.

GoldenGate is a trademark of Oracle.

Hewlett-Packard and HP are registered trademarks of Hewlett-Packard Company.

Hortonworks, the Hortonworks logo and other Hortonworks trademarks are trademarks of Hortonworks Inc. in the United States and other countries.

Intel, Pentium, and XEON are registered trademarks of Intel Corporation.

IBM, CICS, RACF, Tivoli, and z/OS are registered trademarks of International Business Machines Corporation.

Linux is a registered trademark of Linus Torvalds.

LSI is a registered trademark of LSI Corporation.

Microsoft, Active Directory, Windows, Windows NT, and Windows Server are registered trademarks of Microsoft Corporation in the United States and other countries.

NetVault is a trademark or registered trademark of Dell Inc. in the United States and/or other countries.

Novell and SUSE are registered trademarks of Novell, Inc., in the United States and other countries.

Oracle, Java, and Solaris are registered trademarks of Oracle and/or its affiliates.

QLogic and SANbox are trademarks or registered trademarks of QLogic Corporation.

Quantum and the Quantum logo are trademarks of Quantum Corporation, registered in the U.S.A. and other countries.

Red Hat is a trademark of Red Hat, Inc., registered in the U.S. and other countries. Used under license.

SAP is the trademark or registered trademark of SAP AG in Germany and in several other countries.

SAS and SAS/C are trademarks or registered trademarks of SAS Institute Inc.

SPARC is a registered trademark of SPARC International, Inc.

Symantec, NetBackup, and VERITAS are trademarks or registered trademarks of Symantec Corporation or its affiliates in the United States and other countries.

Unicode is a registered trademark of Unicode, Inc. in the United States and other countries.

UNIX is a registered trademark of The Open Group in the United States and other countries.

Other product and company names mentioned herein may be the trademarks of their respective owners.

THE INFORMATION CONTAINED IN THIS DOCUMENT IS PROVIDED ON AN "AS-IS" BASIS, WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESS OR IMPLIED, INCLUDING THE IMPLIED WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE, OR NON-INFRINGEMENT. SOME JURISDICTIONS DO NOT ALLOW THE EXCLUSION OF IMPLIED WARRANTIES, SO THE ABOVE EXCLUSION MAY NOT APPLY TO YOU. IN NO EVENT WILL TERADATA CORPORATION BE LIABLE FOR ANY INDIRECT, DIRECT, SPECIAL, INCIDENTAL, OR CONSEQUENTIAL DAMAGES, INCLUDING LOST PROFITS OR LOST SAVINGS, EVEN IF EXPRESSLY ADVISED OF THE POSSIBILITY OF SUCH DAMAGES.

The information contained in this document may contain references or cross-references to features, functions, products, or services that are not announced or available in your country. Such references do not imply that Teradata Corporation intends to announce such features, functions, products, or services in your country. Please consult your local Teradata Corporation representative for those features, functions, products, or services available in your country.

Information contained in this document may contain technical inaccuracies or typographical errors. Information may be changed or updated without notice. Teradata Corporation may also make improvements or changes in the products or services described in this information at any time without notice.

To maintain the quality of our products and services, we would like your comments on the accuracy, clarity, organization, and value of this document. Please email: teradata-books@lists.teradata.com.

Any comments or materials (collectively referred to as "Feedback") sent to Teradata Corporation will be deemed non-confidential. Teradata Corporation will have no obligation of any kind with respect to Feedback and will be free to use, reproduce, disclose, exhibit, display, transform, create derivative works of, and distribute the Feedback and derivative works thereof without limitation on a royalty-free basis. Further, Teradata Corporation will be free to use any ideas, concepts, know-how, or techniques contained in such Feedback for any purpose whatsoever, including developing, manufacturing, or marketing products or services incorporating Feedback.

Copyright © 2000-2015 by Teradata. All Rights Reserved.

Table of Contents

Preface	7
Conventions Used in This Guide	7
Typefaces	7
Text Conventions	7
Command Shell Text Conventions	8
Contact Teradata Global Technical Support (GTS)	8
Third Party Licenses	8
About Teradata Aster	8
About This Document	9
Version History	9

Chapter 1: Introduction	10
R Programming Language Overview	10
Teradata Aster R Product Overview	10
Terminology	12
Open Source R on Aster Cluster	13
Teradata Aster R Package	14

Chapter 2: Open Source R Installation	16
R Installation Process Overview	16
R Installation	16
Requirements	17
Prerequisites	17
Obtaining R Software	17
Adding a Permanent Repository	17
Installing R	18
Installing R on RHEL With Internet Access	18
Installing R on RHEL Without Internet Access	20
Installing R on SLES With Internet Access	22
Installing R on SLES Without Internet Access	23

Installing R on Aster Express VM With Internet Access	25
Installing R on Aster Express VM Without Internet Access	30
Running a Sample R Script	34
Managing R Packages	35
Installing and Upgrading OS Packages	39
Upgrading R	40
Uninstalling R	44
R Package Installation	44
Installing the R package 'nloptr' on RHEL or SLES	44
Installing the R package 'caret' on RHEL or SLES	45
Installing the R package 'rJava' on SLES	45
Installing the R package 'rjags' on RHEL	45
Installing the R package 'rjags' on SLES	46

Chapter 3: Aster R Package 48

Aster R Package Installation	48
Requirements	48
Prerequisites	48
Obtaining Aster R package	49
Installing or Upgrading Aster R package on Linux x86_64	49
Installing or Upgrading Aster R package on Linux i686	49
Installing or Upgrading Aster R package on Windows 32-bit and 64-bit	49
Installing or Upgrading Aster R package Using RStudio	50
Uninstalling Aster R package	50
Aster R Package Overview	50
Accessing Aster R Inline Documentation	51
Accessing Aster R Vignette Documentation	51
Aster R Quick Start Reference	51
Checking R Installation	52
Attaching Aster R Package Library	52
Creating Connection to Aster Database	52
Exporting Data from Aster to R	53
Loading Data from R to Aster	53
Using Aster Virtual Data Frames	55
Creating Tables in Aster Database	56
Using In-Database R Function Runner	56
Aster R Package Function Reference	58
Utility and Management Functions	58
Data Structure Management Functions	62
Data Exploration Functions	64

Data Transformation Functions	66
Basic Statistical Functions	68
R Map Reduce Runner Functions	73
Parallel Stats and Big Data Analytics Functions.....	75
Aster R Package Limitations	80
Statistics Functions Behavior for bool, char, bit, and varbit Data Types.....	80
Large NUMERIC Values not Displayed From a Virtual Data Frame	80
In-Database R Function Runners Support Maximum of 1599 Columns	80
Aster R Results Differ from Native R for - and ! Operators	80
Aster R functions are not Supported in Expressions.....	80
Aster R Functions With External C Calls are not Uploaded.....	81
ta.write.csv() Encoding Limitation When Exporting to Windows	81
Changing Column Type of Virtual Data Frame in Aster R Limitation	82
ta.cbind Reordering Inputs Limitation.....	82
ta.subset Limitation When Using an R Object or an Aster R Object.....	82
ta.create Limitation for UTF-8 Characters.....	82
bytea Data Type not Supported for Aster R Functions	82
Aster R Package Considerations	82
ta.head and ta.tail Functions can Generate Random Order of Results	83
Using the DataDirect Driver Manager	83
SSL/SSO Authentication is not Supported	83
Saving Your Workspace is not Supported	83
Data Access from Native R Data Frames and Aster Virtual Data Frames	83
Maximum and Minimum Value Ranges for Double Data Types.....	83
Timestamp and Timestamptz	83
Mapping Between integer64 in R and bigint in the Aster Database.....	83
"BC" in Date or Timestamp Type Value is not Recognized by RODBC	84
Numeric Type Double Precision for Empty Table not Consistent with R	84
Type Conversion Behavior for Numeric Values in Quotes in an Expression	84
Include methods Library to Print Virtual Data Frame to Console from R Script	84
General SQL Wrapper Workaround When Executing a SQL Query.....	84
Data Type Mapping.....	85

Chapter 4: Invoke R Using SQL-MR Stream Module

Execution Model For R	90
R Program Invocation	90
Writing an R Program to Run Inside the Aster Database	91
Writing an Output File from an R Program to the File System	92
Using the PARTITION BY Clause in Queries that Invoke R Programs	93
Data Type Mapping Between R and Aster Database	94

Character	95
Bit Data Type.....	95
Bytea Data Type	95
BIGINT Data Type	95
Null (Missing) Value Handling	97
Hash(#) Character Handling	97
Troubleshooting.....	97

Index	100
--------------------	------------

Preface

This guide provides introductory, installation, and usage information for users of the Teradata Aster R product.

The following additional resources are also available:

- Aster Database upgrades, clients and other packages:
<http://downloads.teradata.com/download/aster>
- Documentation for existing customers with a Teradata @ Your Service login:
<http://tays.teradata.com/>
- Documentation that is available to the public:
<http://www.info.teradata.com/>

Conventions Used in This Guide

This document assumes that the reader is comfortable working in Windows and Linux/UNIX environments.

This document uses these typographical conventions.

Typefaces

Command line input and output, commands, program code, filenames, directory names, and system variables are shown in a `monospaced` font. Words in *italics* indicate an example or placeholder value that you must replace with a real value. **Bold type** is intended to draw your attention to important or changed items. Menu navigation and user interface elements are shown using the `User Interface Command` font.

Text Conventions

In the Teradata Aster R synopsis sections, we follow these conventions:

- Square brackets ([and]) indicate one or more optional items.
- Curly braces ({ and }) indicate that you must choose an item from the list inside the braces. Choices are separated by vertical lines (|).
- An ellipsis (...) means the preceding element can be repeated.

- A comma and an ellipsis (, ...) means the preceding element can be repeated in a comma-separated list.
- In command line instructions, SQL commands and shell commands are typically written with no preceding prompt, but where needed the default Aster Database SQL prompt is shown: beehive=>

Command Shell Text Conventions

For shell commands, the prompt is usually shown. The \$ sign introduces a command that's being run by a non-root user:

```
$ ls
```

The # sign introduces a command that's being run as root:

```
# ls
```

Contact Teradata Global Technical Support (GTS)

For assistance and updated documentation, contact Teradata Global Technical Support (GTS):

- **Support Portal:** <http://tays.teradata.com/>
- **International:** 212-444-0443
- **US Customers:** 877-MyT-Data (877-698-3282)

Third Party Licenses

An Aster installation includes a number of open source products. The license text for these products is available in the *Aster Database User Guide*, in the chapter "Licenses" and on the Aster queen, as a set of text files in the `/home/beehive/licenses` directory.

About Teradata Aster

Teradata Aster provides data management and advanced analytics for diverse and big data, enabling the powerful combination of cost-effective storage and ultra-fast analysis of relational and non-relational data. Teradata Aster is a division of Teradata and is headquartered in San Carlos, California.

For more information, go to <http://www.teradata.com/Teradata-Aster/overview/>

About This Document

This document is the initial release of the *Teradata Aster R User Guide*. This edition supports the AC6.20 version of the Teradata Aster R package.

Version History

Product ID	Date	Description
B700-2010-620K	August 2015	Initial Release

CHAPTER 1 Introduction

This section provides information on these topics:

- [R Programming Language Overview](#)
- [Teradata Aster R Product Overview](#)
- [Open Source R on Aster Cluster](#)
- [Teradata Aster R Package](#)

R Programming Language Overview

R is an open source programming language and software environment for statistical data analysis and graphics. The R language is used by data scientists for advanced data analysis and model development.

R provides a wide range of analytic functions covering areas such as time series analysis, classification, clustering, data smoothing, linear and generalized linear models, nonlinear regression models, resampling methods, classical parametric, and nonparametric tests. Currently, the library of over 6,700 add-in packages is available from the Comprehensive R Archive Network (CRAN). For a listing of all R packages, visit the cran.r-project website, <http://cran.r-project.org>, and select Packages.



Teradata does not ship the R environment with the Teradata Aster database because R is Open Source and under the GPL license. Also, Teradata does not provide support for the R Open Source environment. As described in this document, Teradata provides mechanisms for facilitating the installation, administration, and integration of R, which you can download from the Open Source community, on the Aster Database cluster.

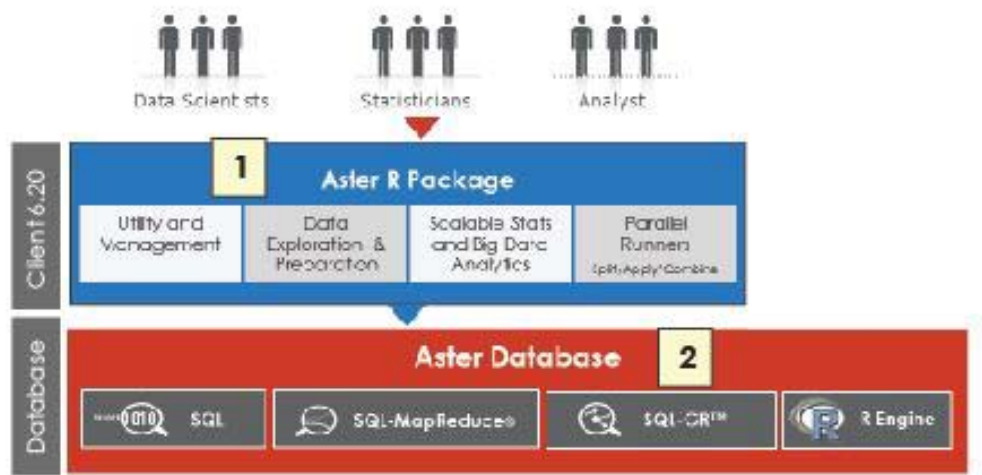
Teradata Aster R Product Overview

Teradata Aster R is a scalable R solution, which enables organizations to use R for business analytics including:

- Solving business problems that require large data volumes.
- Solving big data problems by using Aster analytics, which is accessible through R.

- Deploying R programs in an environment that is secure and manageable.
- Providing easy access to R models to run in a self-service manner.

The Teradata Aster R product solution is implemented using the R engine embedded in the Aster Database(2) and the Teradata Aster R package (1).



Features of the R engine in the Aster Database include:

- Open source R and packages downloaded from CRAN and installed directly in the Aster database.
- Supported in the database beginning with version 5.10 of the Aster Database and run using the SQL-MR Stream function.
- Use existing R code that you can run in the Aster Database.
- Built-in intelligence to handle data distribution and optimized processing in all or just one vWorker. Aster handles the data distribution so that the R user can focus on the analytics.
- Flexibility to run an R script and R function across all vWorkers. R users can easily deploy row or partition independent R code for parallel processing.
- Advanced capabilities for power users to build their own MapReduce functions by leveraging R packages.

For additional information, see [“Open Source R on Aster Cluster” on page 13](#).

The Aster R package is an R client that enables the push-down of R functions that consume data stored in the Aster Database for in-database execution through in-database R or a rewrite to SQL or SQL-MR, as appropriate. This environment:

- Provides Transparency.
- Avoids pulling data out to the client.
- Enables R computations that leverage more computing resources and data-parallelization.

The Aster R package includes R APIs, which enable R programmers to compute over a larger data set in the Aster Database than if they were to use a traditional R implementation. The ability to compute over a larger data set is accomplished by performing all computations

where the data resides (in the cluster nodes), instead of moving the data from the cluster to the client R environment.

For additional information, see [“Teradata Aster R Package” on page 14](#).

Terminology

- R — The set of standard packages that are automatically available in any R installation. These standard (core) packages include the basic functions that are required for R to operate as well as standard statistical and graphical functions.
- R Packages — All statistical functions beyond the base statistical packages that can be downloaded from CRAN mirrors.
- Aster Package Manager (APM) — An Aster Database cluster service that manages all third-party software installed on your Aster Database cluster. Currently, only the R programming language is supported.
- Auxiliary root area (/opt/aster/third-party) — A parent root directory for all third-party software installed on the Aster Database.
- Aster R Package — This package:
 - Provides an interface between R and the Aster Database.
 - Provides the implementation of the wrapper functions over the Analytical Foundation functions.
 - Includes the definition of ta.data.frame and related functions.
- Aster Analytical Foundation (AAF) — The Aster provided analytical functions that can be installed on the Aster Database and based on the SQL-MR framework.
- Queen — The queen is responsible for managing the system configuration and coordinating queries. Responsibilities of this machine include
 - accept, optimize and plan SQL or SQL/MR statements from clients
 - distribute work across the cluster
 - aggregate results from workers
 - send results back to clients
 - system management to ensure that replication levels of the data in the cluster are maintained.

For additional information on the architecture of the Aster Database, refer to the *Aster Database User Guide*.

- Workers — Worker machines are responsible for storage of and computation on distributed data. As your data volume grows, you can add more worker nodes to ensure query execution times scale linearly with data volume. Each worker machines hosts 1 or more (typically 4-6) vworkers (virtual workers) which are individual slices of data and computation that are performed on each worker. The vworkers receive phases of the execution plan that are relevant to the slice of the data they are responsible for. vworkers have the capability to perform inter-vworker communication in case the plan requires shuffling of data between worker machines. Once the computation is complete, the results are sent back to the queen.

For additional information on the architecture of the Aster Database, refer to the *Aster Database User Guide*.

Open Source R on Aster Cluster

R is designed to operate in a single server (single-threaded) on data that is entirely in the main memory of the system. Because of this design, R fails when the data becomes too large to fit in memory. The amount of memory depends on your specific system configuration and the actual memory available at a given point in time. This limitation is exacerbated by the call by value semantics of an R execution, which leads to many copies of data being created in memory as data flows from one function to another.

Data scientists and statisticians using R routinely analyze large data stored in relational databases. Currently, the only option available for data scientists to analyze data stored in a relational database is to read data out to the R environment. This leads to a number of problems, including time-consuming data extraction from (and export to) relational databases. This typically prohibits interactive data analysis, unnecessarily duplicates data storage in the organization, and requires a system with large amounts of memory and storage to run R and process large amounts of data.

The R integration in the Aster Database is aimed at addressing these challenges by enabling the in-database execution of R, both to avoid extraction of data from Aster Database and to scale R to large data sets through parallelized execution, which enables users to run multiple instances of their programs over partitioned data.

Specifically, the R integration enables:

- Simplified installation and administration of R and R packages.
- Installation of user R programs.
- Execution of user programs in such a way that each R instance directly accesses the data partition stored in each vWorker of the Aster Database cluster.

For information on installing R and common R packages, see [Chapter Chapter 2](#) , “Open Source R Installation”.

Supported R Functionality in Aster Database

This R functionality is supported on the Aster Database:

- Installation and uninstallation of R on the Aster Database cluster.
- Installation and uninstallation of R packages in the default location on queen node and all worker nodes.
- R installation from an online repository or a local repository containing RPMs of R and all their dependency RPMs.
- R package installation from an online repository or a local repository containing tar files of R packages and their dependency R packages.
- Automatic cleanup after failed or incomplete installs or upgrades, as well as reporting the failure to the log.

- Transparent new cluster node addition (node synchronization). When a new node is added to the cluster, R and all installed R packages are automatically deployed on the new node.
- Maintaining of consistency of R and its packages on all the worker nodes in the cluster.
- SQL-driven running of R programs over partitioned data on each virtual worker using the SQL-MR Streams module.

Unsupported R Functionality in Aster Database

This R Functionality is not supported on the Aster Database:

- The installer installs the latest version of R. Multi-version installations are not supported in this release.

Teradata Aster R Package

The Aster R package provides R users with:

- A number of functions organized into four functional areas:
 - Utility and Management Functions
 - File management functions for bulk load and extract of .csv files.
 - Database object management functions.
 - Connection functions.
 - File installation and management functions.
 - Data Access and Preparation Functions
 - Data structure functions.
 - Data exploration functions.
 - Data transformation functions.
 - Parallel Analytic Functions
 - Prebuilt parallel functions designed to run against all your data without the need for partitioning or parallel programming
 - Basic statistical functions
 - SQL pass-through functions providing access to all Aster big data analytic functions.
 - R Script Runners
 - Enable running R script in the R engine that is embedded in the Aster Discovery Platform
- R data objects - Virtual Data Frame, Virtual List, Virtual Factor and Virtual Vector.

A Virtual Data Frame is an extension of a regular R data frame and specifically designed to access table, view and query based data in the Aster Database. An instance of a Virtual Data Frame maps to a table or view in the database or a SQL query. Any access or manipulation on a Virtual Data Frame is translated into an equivalent SQL or SQL-MR query and is executed in the Aster Database using RODBC. The class attribute is set to

“ta.data.frame”. In the Aster R package, an instance of a Virtual Data Frame (TeradataAsterDataFrame) is typically named “tadf”. For additional information on Virtual Data Frames, see [“Using Aster Virtual Data Frames” on page 55](#).

For information on installing, using, and the contents of the Aster R package, see [Chapter 3, “Aster R Package”](#).

R Version Support

The Aster R package supports R 3.1.1 and all subsequent versions of R, until otherwise indicated in a replacement publication:



The version of R installed on the client and on the server must be the same in order to ensure proper operation of the Aster R software.

RConsole Support

Teradata Aster R supports using the RConsole command line tool with the Aster R package.

RStudio Support

Teradata Aster R supports using RStudio with the Aster R package, however RStudio is not included with the product. If you want to use RStudio, you must install and license it separately.

CHAPTER 2 Open Source R Installation

This section provides information on these topics:

- [R Installation Process Overview](#)
- [R Installation](#)
- [R Package Installation](#)

R Installation Process Overview

During R installation, a complete installation is first carried out on the queen and cluster-wide synchronization of the auxiliary root is used to install R on all the workers. Likewise, uninstallation is first performed on the queen and a cluster-wide synchronization of the auxiliary root is used to remove R from the cluster.

The R installation requires a number of dependency packages. The R package installer internally calls the Yum (for Red Hat) or Zypper (for SLES) package installer. These package installers install the latest version of a package or group of packages while ensuring that all dependencies are satisfied. You need to provide a valid repository (either official Red Hat or SLES mirrors or a suitable local repository mirror) to resolve the distribution package dependencies using the appropriate `ncli apm` command options. For more information about the `ncli apm` commands, see the *Aster Database User Guide*.

R Installation



Before you install R, back up these files because the R installer overwrites them: `/opt/aster`, `/opt/aster/third-party`, and `/opt/aster/third-party/R`.

The section provides information on these R installation topics:

- [Requirements](#)
- [Prerequisites](#)
- [Obtaining R Software](#)
- [Adding a Permanent Repository](#)

- [Installing R](#)
- [Managing R Packages](#)
- [Installing and Upgrading OS Packages](#)
- [Upgrading R](#)
- [Uninstalling R](#)

Requirements

The minimum installed software environment requirements for installing R are:

- Operating System:
 - Aster Appliance: a supported version of the Teradata SUSE Linux Enterprise Server (SLES) operating system.
 - Commodity Hardware: a supported version of the Red Hat Enterprise Linux operating system (RHEL) or the SUSE Linux Enterprise Server (SLES) operating system.
 - Aster Express: a supported version of the SUSE Linux Enterprise Server (SLES) operating system.
- Aster Database, version 6.10 (AD 6.10)
- Aster Client, version 6.20 (AC 6.20)
- Aster ODBC Driver (version AC 6.20 or higher)

Prerequisites

- The version of R installed on the server and client must be the same in order to ensure proper operation of the Aster R package software.
- The procedures in this section use `ncli` commands. If you need supplemental information on `ncli` commands, see the Aster Database User Guide.

Obtaining R Software

Refer to the specific type of installation you are performing for details on obtaining the R software for the installation (See [“Installing R” on page 18](#)).

Adding a Permanent Repository

A repository is required in order to use Yum (on RHEL) or zypper (on SLES) to install the R open source software on a supported version of the RHEL or the SLES operating systems.

Unless you add a permanent repository, the repository that is specified in the `ncli apm install R` command is added temporarily and is automatically removed when the command completes. If you do not want to specify a repository every time the `ncli apm install R` command is run, add a permanent repository. To add a permanent repository:

- 1 Log on to the Aster Database queen as root.
- 2 Set up a permanent repository by issuing this `ncli` command:

```
# ncli apm administer R --setuprepo=<repoName>,<repoURL>
```



The permanent repository added using the `ncli apm administer R` command is available only in the R sandbox area. If R is uninstalled, the repository is also removed

Removing a Permanent Repository

To remove a permanent repository, issue this `ncli` command:

```
# ncli apm administer R --removere repo=<repoName>
```

Installing R

To install R, select the procedure that aligns with the type of installation you want to perform:

- [Installing R on RHEL With Internet Access](#)
- [Installing R on RHEL Without Internet Access](#)
- [Installing R on SLES With Internet Access](#)
- [Installing R on SLES Without Internet Access](#)
- [Installing R on Aster Express VM With Internet Access](#)
- [Installing R on Aster Express VM Without Internet Access](#)

To validate your installation of R, which was performed using one of the above listed installation procedures, see [“Running a Sample R Script” on page 34](#).

Installing R on RHEL With Internet Access

This section describes the process to follow in order to install the latest version of R on an Aster Database cluster running a supported version of the RHEL OS in which the queen node can access external websites to download the required rpm packages.

Prerequisite

The `yum` command must work correctly on the queen node of the cluster where you intend to install R. To verify that the command is working correctly, confirm that an error is not returned after issuing this command on the queen node:

```
yum --version
```

Obtaining RHEL ISO Image File

Because the RHEL ISO image file is not freely available on the internet, contact your system or network administrator in order to obtain the RHEL ISO image file. Ensure that you request the same version of the RHEL ISO as that of your main RHEL operating system. Copy the RHEL ISO image file to the `/var/opt/teradata` directory on the queen node.

Mounting RHEL ISO Image File

To mount the RHEL ISO image file, perform these steps on the queen node:

- 1 Create the `/var/opt/teradata/rhel` directory:

```
mkdir -p /var/opt/teradata/rhel
```

- 2 Change to the `/var/opt/teradata` directory:

```
cd /var/opt/teradata
```

- 3 Mount the RHEL ISO image file:

- To mount the RHEL ISO image file on a system running version 6.4 of the RHEL OS, run this command:

```
mount -o loop rhel-server-6.4-x86_64-dvd.iso /var/opt/teradata/rhel
```

- To mount the RHEL ISO image file on a system running version 6.5 of the RHEL OS, run this command:

```
mount -o loop rhel-server-6.5-x86_64-dvd.iso /var/opt/teradata/rhel
```

Installing R

To install R, run this command on the queen:

```
ncli apm install R  
--repo=rhel,file:///var/opt/teradata/rhel
```

This command internally connects to http://ftp1.scientificlinux.org/linux/scientific/6.4/x86_64/os/Packages/ or http://ftp1.scientificlinux.org/linux/scientific/6.5/x86_64/os/Packages/ (depending on the version of the main operating system) and http://download.fedoraproject.org/pub/epel/6/x86_64/ in order to fetch the R rpms. If the R installation fails, verify that both urls are accessible. If the urls are not accessible, then install R using this alternative method, [Installing R on RHEL Without Internet Access](#).

To confirm that R successfully installed on the queen node, run these commands on the queen:

```
ncli apm show R --localconfig  
ncli apm show R --info
```

Syncing R Across All Workers

The previous command installs R only on the queen node. To sync R across all workers in the cluster and then activate the system, perform these ncli commands:

```
ncli system softrestart  
ncli system activate
```

To confirm that R successfully installed on all worker nodes in the cluster, run this command on the queen node:

```
ncli apm show R --localconfig
```

The R and R packages on Aster are installed in the `/opt/aster/third-party/R` directory on the queen and worker nodes in the Aster Database cluster. Refer to [“Running a Sample R Script” on page 34](#) for an example of running a sample R script.

Installing R on RHEL Without Internet Access

This section describes the process to follow in order to install the latest version of R on an Aster Database cluster running a supported version of the RHEL OS in which the queen node cannot access external websites to download the required rpm packages.

Prerequisites

- The `yum` and `createrepo` commands must work correctly on the queen node of the cluster where you intend to install R. To verify that the commands are working correctly, confirm that an error is not returned after issuing each of these commands on the queen node:

```
yum --version
createrepo --version
```

- The `wget` command must work correctly on the machine that you plan to use for downloading the R RPM files. To verify that the command is working correctly, confirm that an error is not returned after issuing this command on the machine that you plan to use for downloading the files:

```
wget --version
```

Obtaining RHEL ISO Image File

Because the RHEL ISO image file is not freely available on the internet, contact your system or network administrator in order to obtain the RHEL ISO image file. Ensure that you request the same version of the RHEL ISO as that of your main RHEL operating system. Copy the RHEL ISO image file to the `/var/opt/teradata` directory on the queen node.

Obtaining R RPM Files

Because the queen node on the Aster Database cluster doesn't have access to the internet, you must first download the R rpm files to a machine that has access to the internet, and then copy the rpm files from their download location to the queen node on the Aster Database cluster.

Follow these steps to download the required R and R dependency rpm files:

- 1 Perform these steps to download all required rpm files to the machine you have selected that has access to the internet.

- a Depending on the version of the installed RHEL OS, perform one of these actions:

- For a system running version 6.4 of the RHEL OS, run this command:

```
url="http://ftp1.scientificlinux.org/linux/scientific/6.4/x86_64/os/Packages/"
```

- For a system running version 6.5 of the RHEL OS, run this command:

```
url="http://ftp1.scientificlinux.org/linux/scientific/6.5/x86_64/os/Packages/"
```

- b Run these commands to download additional required rpm files:

```
pkgs="blas-|freetype-|lapack-|libc-|libjpeg|texinfo-tex"
```

```
for pkg in `wget -q $url -O - 2>&1 | grep -Po "(?<=>)(\$pkgs).*?(noarch|x86_64).rpm(?=<)"` \
;do wget -nc $url/$pkg; done
```

- c Run these commands to download additional required rpm files:

```
url="http://download.fedoraproject.org/pub/epel/6/x86_64/"
```

```
pkgs="R-|libRmath-"
```

```
for pkg in `wget -q $url -O - 2>&1 | grep -Po "(?<=>)(${pkgs}).*(noarch|x86_64).rpm(?=<)"`\  
;do wget -nc $url/$pkg; done
```



After running above commands, verify that the R, R-devel, R-core, R-core-devel, R-java, R-java-devel, texinfo-tex, libRmath, and libRmath packages are downloaded. If any of the packages are not downloaded, repeat the above steps.

- 2 If the `/var/opt/teradata/rhel-r` directory does not already exist on the queen node of the Aster Database cluster, create the directory by running this command:

```
mkdir -p /var/opt/teradata/rhel-r
```

- 3 Copy the rpm files from the machine on to which they were downloaded to the `/var/opt/teradata/rhel-r` directory on the queen node of the Aster Database cluster.
- 4 Create a yum repository from all files copied in the previous step. To create the repository, run these commands:

```
cd /var/opt/teradata/rhel-r  
createrepo .
```

Mounting RHEL ISO Image File

To mount the RHEL ISO image file, perform these steps on the queen node:

- 1 Create the `/var/opt/teradata/rhel` directory:

```
mkdir -p /var/opt/teradata/rhel
```

- 2 Change to the `/var/opt/teradata` directory:

```
cd /var/opt/teradata
```

- 3 Mount the RHEL ISO image file:

- To mount the RHEL ISO image file on a system running version 6.4 of the RHEL OS, run this command:

```
mount -o loop rhel-server-6.4-x86_64-dvd.iso /var/opt/teradata/rhel
```

- To mount the RHEL ISO image file on a system running version 6.5 of the RHEL OS, run this command:

```
mount -o loop rhel-server-6.5-x86_64-dvd.iso /var/opt/teradata/rhel
```

Installing R

To install R, run this command on the queen:

```
ncli apm install R --usedefaultrepo=False \  
--repo=rhel,file:///var/opt/teradata/rhel \  
--repo=rhel-r,file:///var/opt/teradata/rhel-r
```

To confirm that R successfully installed on the queen node, run these commands on the queen:

```
ncli apm show R --localconfig  
ncli apm show R --info
```

Syncing R Across All Workers

The previous command installs R only on the queen node. To sync R across all workers in the cluster and then activate the system, perform these ncli commands:

```
ncli system softrestart
ncli system activate
```

To confirm that R successfully installed on all worker nodes in the cluster, run this command on the queen node:

```
ncli apm show R --localconfig
```

The R and R packages on Aster are installed in the `/opt/aster/third-party/R` directory on the queen and worker nodes in the Aster Database cluster. Refer to “[Running a Sample R Script](#)” on page 34 for an example of running a sample R script.

Installing R on SLES With Internet Access

This section describes the process to follow in order to install the latest version of R on an Aster Database cluster running a supported version of the SLES OS in which the queen node can access external websites to download the required rpm packages.

Prerequisite

The `zypper` command must work correctly on the queen node of the cluster where you intend to install R. To verify that the command is working correctly, confirm that an error is not returned after issuing this command on the queen node:

```
zypper --version
```

Obtaining SLES ISO Image Files

Download the SUSE Linux Enterprise Server (SLES-11-SP2-DVD-x86_64-GM-DVD1.iso) and SUSE Linux Enterprise Software Development Kit (SLE-11-SP2-SDK-DVD-x86_64-GM-DVD1.iso) from the SUSE website, <https://download.suse.com>. To download the ISO image files, you will need to log in to the website using your Suse/Novell account (create an account if you don't already have one). Copy the downloaded DVD ISO files to the `/var/opt/teradata` directory on the queen node.

Mounting SLES ISO Image Files

To mount the SLES ISO image files, perform these steps on the queen node:

- 1 Create the `/var/opt/teradata/sles` and the `/var/opt/teradata/sles-sdk` directories:

```
mkdir -p /var/opt/teradata/sles
mkdir -p /var/opt/teradata/sles-sdk
```

- 2 Change to the `/var/opt/teradata` directory:

```
cd /var/opt/teradata
```

- 3 Mount the SLES ISO image files. To mount the SLES ISO image files, run these commands:

```
mount -o loop SLES-11-SP2-DVD-x86_64-GM-DVD1.iso /var/opt/teradata/sles
mount -o loop SLE-11-SP2-SDK-DVD-x86_64-GM-DVD1.iso /var/opt/teradata/sles-sdk
```

Installing R

To install R, run this command on the queen:

```
ncli apm install R \  
--repo=sles,file:///var/opt/teradata/sles \  
--repo=sles-sdk,file:///var/opt/teradata/sles-sdk
```

This command internally connects to http://download.opensuse.org/repositories/devel:/languages:/R:/released/SLE_11_SP2/ in order to fetch the R rpms. If the R installation fails, verify that http://download.opensuse.org/repositories/devel:/languages:/R:/released/SLE_11_SP2/ is accessible. If the url is not accessible, then install R using this alternative method, [Installing R on SLES Without Internet Access](#).

To confirm that R successfully installed on the queen node, run these commands on the queen:

```
ncli apm show R --localconfig  
ncli apm show R --info
```

Syncing R Across All Workers

The previous command installs R only on the queen node. To sync R across all workers in the cluster and then activate the system, perform these ncli commands:

```
ncli system softrestart  
ncli system activate
```

To confirm that R successfully installed on all worker nodes in the cluster, run this command on the queen node:

```
ncli apm show R --localconfig
```

The R and R packages on Aster are installed in the `/opt/aster/third-party/R` directory on the queen and worker nodes in the Aster Database cluster. Refer to [“Running a Sample R Script” on page 34](#) for an example of running a sample R script.

Installing R on SLES Without Internet Access

This section describes the process to follow in order to install the latest version of R on an Aster Database cluster running a supported version of the SLES OS in which the queen node cannot access external websites to download the required rpm packages.

Prerequisites

- The `zypper` command must work correctly on the queen node of the cluster where you intend to install R. To verify that the command is working correctly, confirm that an error is not returned after issuing this command on the queen node:

```
zypper --version
```

- The `wget` command must work correctly on the machine that you plan to use for downloading the SLES ISO image files from the SUSE website, and the R RPM files from the openSUSE website. To verify that the command is working correctly, confirm that an error is not returned after issuing this command on the machine that you plan to use for downloading the files:

```
wget --version
```

Obtaining SLES ISO Image Files

Because the queen node on the Aster Database cluster doesn't have access to the internet, you must first download the ISO image files to a machine that has access to the internet, and then copy the ISO image files from their download location to the queen node on the Aster Database cluster.

- 1 Download the SUSE Linux Enterprise Server (SLES-11-SP2-DVD-x86_64-GM-DVD1.iso) and SUSE Linux Enterprise Software Development Kit (SLE-11-SP2-SDK-DVD-x86_64-GM-DVD1.iso) from the SUSE website, <https://download.suse.com>, to the machine you have selected that has access to the internet. To download the ISO image files, you will need to log in to the website using your Suse/Novell account (create an account if you don't already have one).
- 2 Copy the DVD ISO files from the machine on to which they were downloaded to the /var/opt/teradata directory on the queen node of the Aster Database cluster.

Obtaining R Rpm Files

Because the queen node on the Aster Database cluster doesn't have access to the internet, you must first download the R rpm files to a machine that has access to the internet, and then copy the rpm files from their download location to the queen node on the Aster Database cluster.

Follow these steps to download the required R and R dependency rpm files:

- 1 Run these two commands to download all required rpm files to the machine you have selected that has access to the internet:

```
wget -r -nd --level=1 -e robots=off -H -A rpm \
http://download.opensuse.org/repositories/devel:/\
languages:/R:/released/SLE_11_SP2/x86_64/
```

```
wget -r -nd --level=1 -e robots=off -H -A rpm \
http://download.opensuse.org/repositories/devel:/\
languages:/R:/released/SLE_11_SP2/noarch/
```

- 2 If the /var/opt/teradata/sles-r directory does not already exist on the queen node of the Aster Database cluster, create the directory by running this command:

```
mkdir -p /var/opt/teradata/sles-r
```
- 3 Copy the rpm files from the machine on to which they were downloaded to the /var/opt/teradata/sles-r directory on the queen node of the Aster Database cluster.

Mounting SLES ISO Image Files

To mount the SLES ISO image files, perform these steps on the queen node:

- 1 Create the /var/opt/teradata/sles and the /var/opt/teradata/sles-sdk directories:

```
mkdir -p /var/opt/teradata/sles
mkdir -p /var/opt/teradata/sles-sdk
```
- 2 Change to the /var/opt/teradata directory:

```
cd /var/opt/teradata
```


- 3 Mount the SLES ISO image files. To mount the SLES ISO image files, run these commands:

```
mount -o loop SLES-11-SP2-DVD-x86_64-GM-DVD1.iso /var/opt/teradata/sles
mount -o loop SLE-11-SP2-SDK-DVD-x86_64-GM-DVD1.iso /var/opt/teradata/sles-sdk
```

Installing R

To install R, run this command on the queen:

```
ncli apm install R --usedefaultrepo=False \
--repo=sles,file:///var/opt/teradata/sles \
--repo=sles-sdk,file:///var/opt/teradata/sles-sdk \
--repo=sles-rl,file:///var/opt/teradata/sles-r
```

To confirm that R successfully installed on the queen node, run these commands on the queen:

```
ncli apm show R --localconfig
ncli apm show R --info
```

Syncing R Across All Workers

The previous command installs R only on the queen node. To sync R across all workers in the cluster and then activate the system, perform these ncli commands:

```
ncli system softrestart
ncli system activate
```

To confirm that R successfully installed on all nodes in the cluster, run this command on the queen node and each worker node:

```
ncli apm show R --localconfig
```

The R and R packages on Aster are installed in the `/opt/aster/third-party/R` directory on the queen and worker nodes in the Aster Database cluster. Refer to [“Running a Sample R Script” on page 34](#) for an example of running a sample R script.

Installing R on Aster Express VM With Internet Access

This section describes the process to follow in order to install the latest version of R on an Aster Express virtual machine (VM) running a supported version of the SLES OS in which the queen node can access external websites to download the required rpm packages.

Prerequisite

The `zypper` command must work correctly on the queen node of the cluster where you intend to install R. To verify that the command is working correctly, confirm that an error is not returned after issuing this command on the queen node:

```
zypper --version
```

Obtaining SLES ISO Image Files

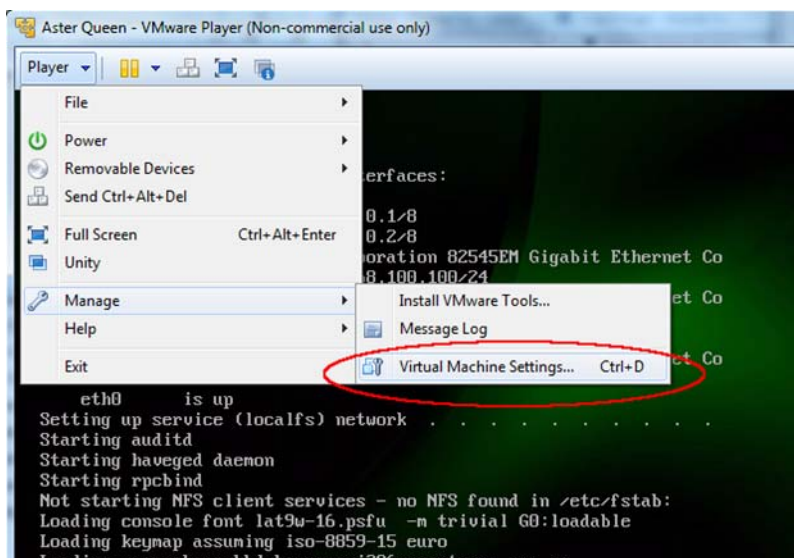
Download the SUSE Linux Enterprise Server (SLES-11-SP2-DVD-x86_64-GM-DVD1.iso) and SUSE Linux Enterprise Software Development Kit (SLE-11-SP2-SDK-DVD-x86_64-GM-DVD1.iso) from the SUSE website, <https://download.suse.com> to your host

machine. To download the ISO image files, you will need to log in to the website using your Suse/Novell account (create an account if you don't already have one).

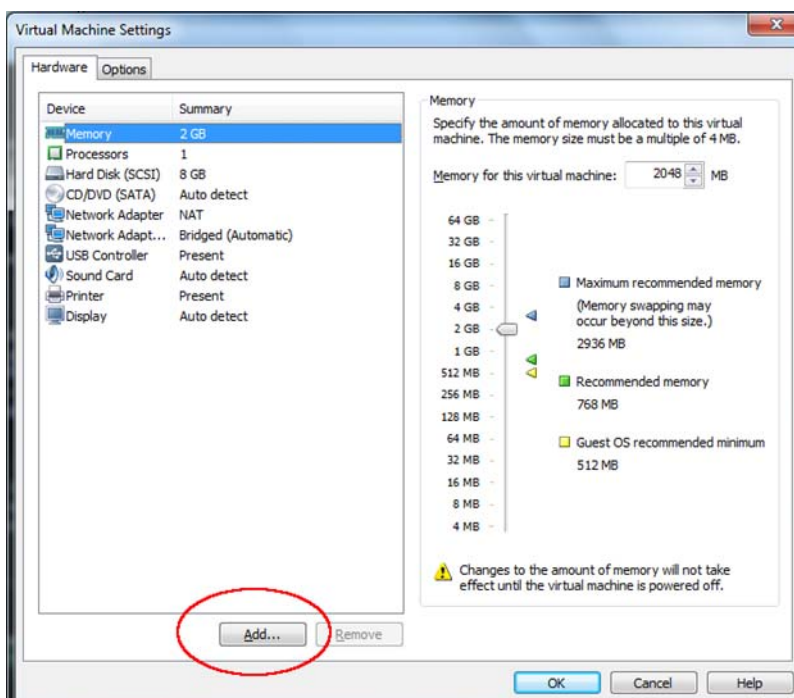
Creating VMWare CD/DVD Drive Connections

Create two CD/DVD drive connections on the queen node for the two ISO images that you downloaded in the previous step. To create the CD/DVD drive connections, perform these steps:

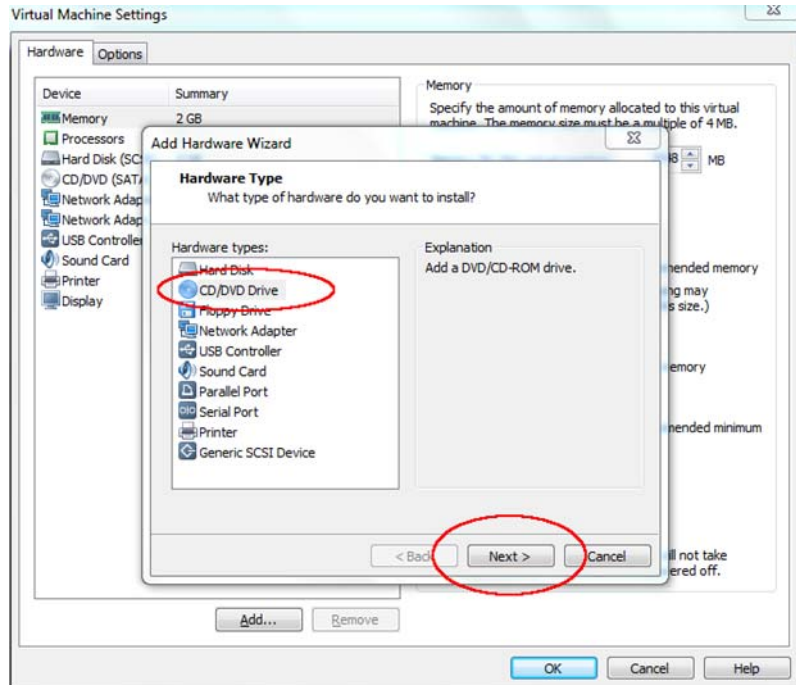
- 1 In the Aster Queen - VMware Player window, select **Player > Manage > Virtual Machine Settings**:



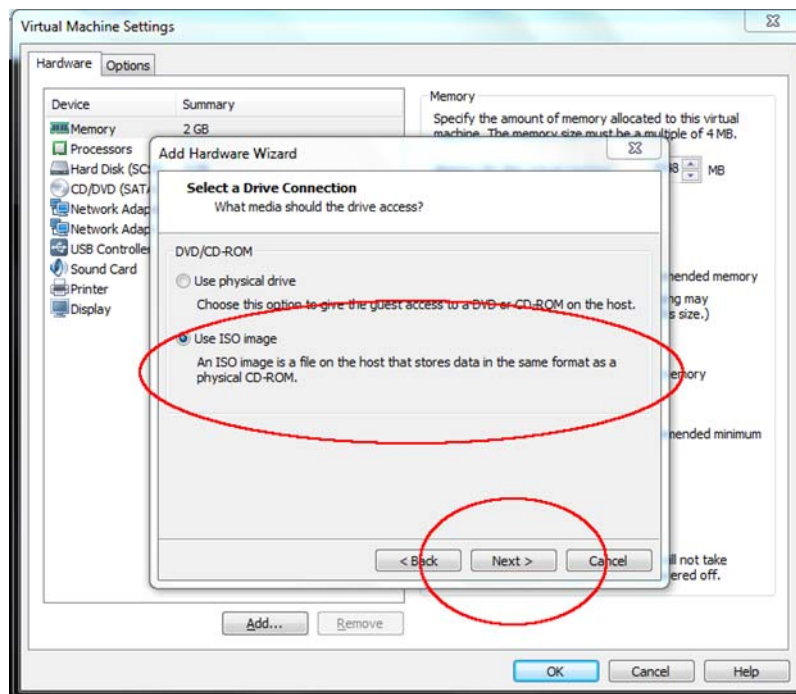
- 2 In the Virtual Machine Settings window, select **Add**:



- 3 In the Add Hardware Wizard window under Hardware types, select CD/DVD Drive, and then click Next:

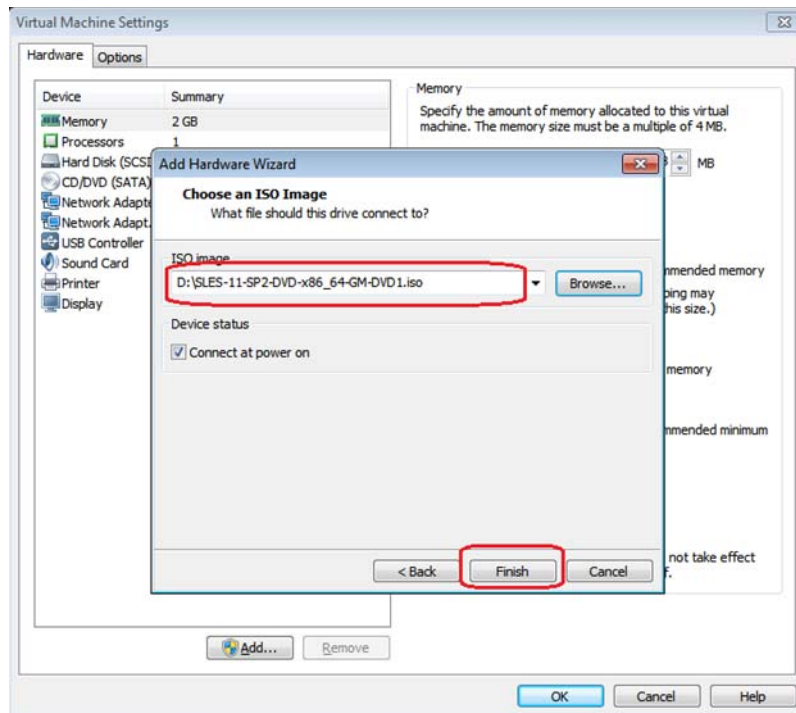


- 4 In the Add Hardware Wizard window under DVD/CD-ROM, select Use ISO image, and then click Next:



- 5 In the Add Hardware Wizard window:
 - a Under ISO image, browse on your file system for SLES-11-SP2-DVD-x86_64-GM-DVD1.iso.

- b Under Device status, select the **Connect at power on** option.
- c Click **Finish**:



- 6 Repeat the above steps in order to browse for SLE-11-SP2-SDK-DVD-x86_64-GM-DVD1.iso.

Mounting SLES ISO Image Files

To locate the CD ROM drives that you created in the previous step, run this command on the queen node.

```
ls /dev/cdrom*
```

This command will return entries named similar to `cdrom1` and `cdrom2`.

To mount the SLES ISO image files, perform these steps on the queen node:

- 1 Create the `/var/opt/teradata/sles` and the `/var/opt/teradata/sles-sdk` directories:

```
mkdir -p /var/opt/teradata/sles
mkdir -p /var/opt/teradata/sles-sdk
```

- 2 Mount the SLES ISO image files:

```
mount -o loop SLES-11-SP2-DVD-x86_64-GM-DVD1.iso /var/opt/teradata/sles
mount -o loop SLE-11-SP2-SDK-DVD-x86_64-GM-DVD1.iso /var/opt/teradata/sles-sdk
```

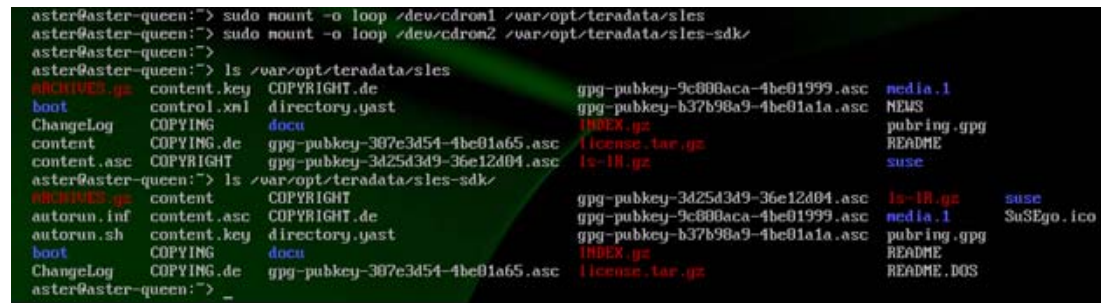
Confirming ISO Images Mounted

To confirm that the ISO image files are correctly mounted in the `/var/opt/teradata/sles` and `/var/opt/teradata/sles-sdk` directories, run these commands on the queen node:

```
ls /var/opt/teradata/sles
```

```
ls /var/opt/teradata/sles-sdk
```

If the ISO images are correctly mounted, the commands will return output similar to this output:



```
aster@aster-queen:~$ sudo mount -o loop /dev/cdrom1 /var/opt/teradata/sles
aster@aster-queen:~$ sudo mount -o loop /dev/cdrom2 /var/opt/teradata/sles-sdk
aster@aster-queen:~$ ls /var/opt/teradata/sles
ARCHIVES.gz  content.key  COPYRIGHT.de  gpg-pubkey-9c808aca-4be01999.asc  media.1
boot         control.xml  directory.gast  gpg-pubkey-b37b98a9-4be01a1a.asc  NEWS
ChangeLog    COPYING     docu           INDEX.gz  pubring.gpg
content      COPYING.de  gpg-pubkey-307c3d54-4be01a65.asc  license.tar.gz  README
content.asc  COPYRIGHT   gpg-pubkey-3d25d3d9-36e12d04.asc  is-11.gz       suse
aster@aster-queen:~$ ls /var/opt/teradata/sles-sdk
ARCHIVES.gz  content      COPYRIGHT     gpg-pubkey-3d25d3d9-36e12d04.asc  is-11.gz  suse
autorun.inf  content.asc  COPYRIGHT.de  gpg-pubkey-9c808aca-4be01999.asc  media.1   SuSEgo.ico
autorun.sh   content.key  directory.gast  gpg-pubkey-b37b98a9-4be01a1a.asc  pubring.gpg
boot         COPYING     docu           INDEX.gz  README
ChangeLog    COPYING.de  gpg-pubkey-307c3d54-4be01a65.asc  license.tar.gz  README.DOS
aster@aster-queen:~$
```

If the files are not correctly located in the directories, unmount the directories and then perform the “[Mounting SLES ISO Image Files](#)” on page 28 procedure again. To unmount the directories, run these commands on the queen node:

```
umount /var/opt/teradata/sles
umount /var/opt/teradata/sles-sdk
```

Installing R

To install R, run this command on the queen:

```
ncli apm install R \
--repo=sles,file:///var/opt/teradata/sles \
--repo=sles-sdk,file:///var/opt/teradata/sles-sdk
```

This command internally connects to http://download.opensuse.org/repositories/devel:/languages:/R:/released/SLE_11_SP2/ in order to fetch the R rpms. If the R installation fails, verify that http://download.opensuse.org/repositories/devel:/languages:/R:/released/SLE_11_SP2/ is accessible. If the url is not accessible, then install R using this alternative method, [Installing R on Aster Express VM Without Internet Access](#).

To confirm that R successfully installed on the queen node, run these commands on the queen:

```
ncli apm show R --localconfig
ncli apm show R --info
```

Syncing R Across All Workers

The previous command installs R only on the queen node. To sync R across all workers in the cluster and then activate the system, perform these ncli commands:

```
ncli system softrestart
ncli system activate
```

To confirm that R successfully installed on all worker nodes in the cluster, run this command on the queen node:

```
ncli apm show R --localconfig
```

The R and R packages on Aster are installed in the /opt/aster/third-party/R directory on the queen and worker nodes in the Aster Database cluster. Refer to “[Running a Sample R Script](#)” on page 34 for an example of running a sample R script.

Installing R on Aster Express VM Without Internet Access

This section describes the process to follow in order to install the latest version of R on an Aster Express VM running a supported version of the SLES OS in which the queen node cannot access external websites to download the required rpm packages.

Prerequisites

- The `zypper` command must work correctly on the queen node of the cluster where you intend to install R. To verify that the command is working correctly, confirm that an error is not returned after issuing this command on the queen node:

```
zypper --version
```

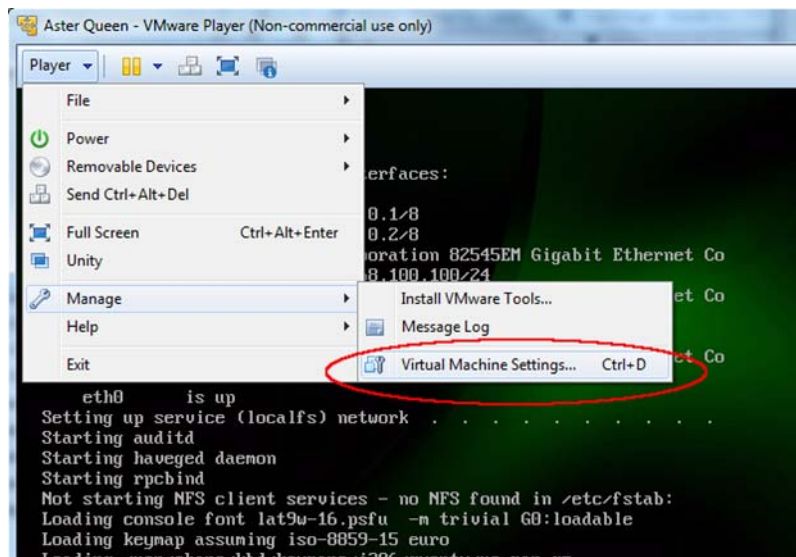
Obtaining SLES ISO Image Files

Download the SUSE Linux Enterprise Server (SLES-11-SP2-DVD-x86_64-GM-DVD1.iso) and SUSE Linux Enterprise Software Development Kit (SLE-11-SP2-SDK-DVD-x86_64-GM-DVD1.iso) from the SUSE website, <https://download.suse.com> to your host machine. To download the ISO image files, you will need to log in to the website using your Suse/Novell account (create an account if you don't already have one).

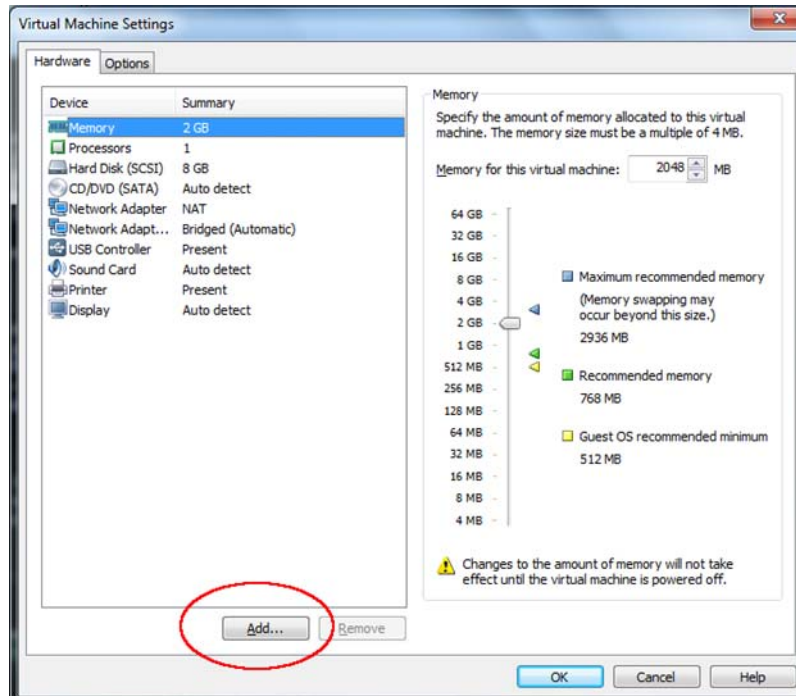
Creating VMWare CD/DVD Drive Connections

Create two CD/DVD drive connections on the queen node for the two ISO images that you downloaded in the previous step. To create the CD/DVD drive connections, perform these steps:

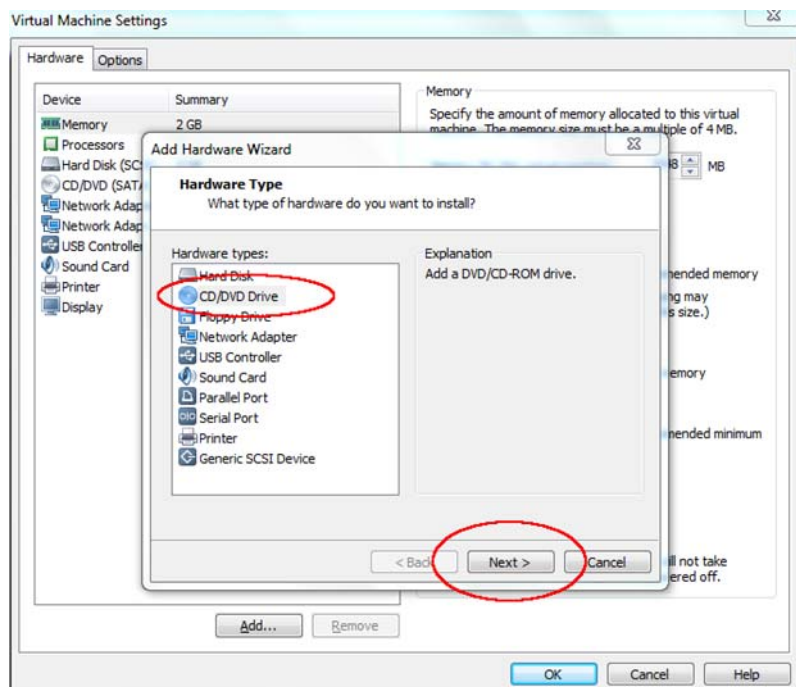
- In the Aster Queen - VMware Player window, select **Player > Manage > Virtual Machine Settings**:



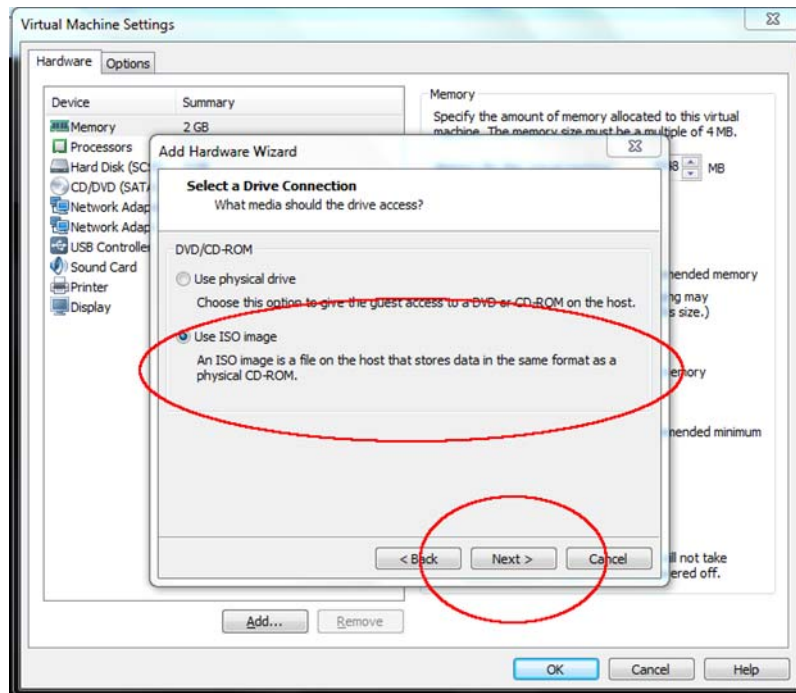
- In the Virtual Machine Settings window, select **Add**:



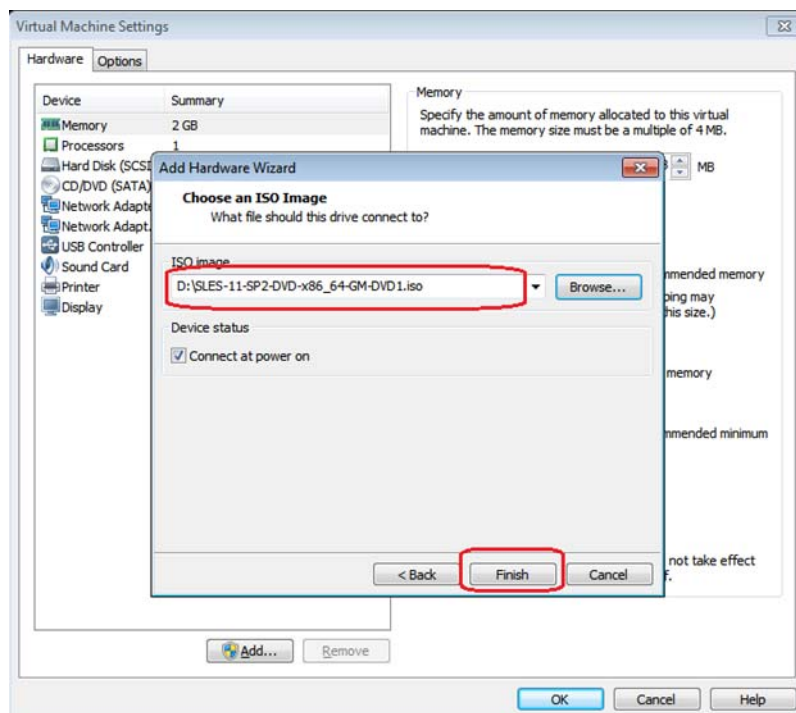
- 3 In the Add Hardware Wizard window under Hardware types, select CD/DVD Drive, and then click Next:



- 4 In the Add Hardware Wizard window under DVD/CD-ROM, select Use ISO image, and then click Next:



- 5 In the Add Hardware Wizard window:
 - a Under ISO image, browse on your file system for SLES-11-SP2-DVD-x86_64-GM-DVD1.iso.
 - b Under Device status, select the **Connect at power on** option.
 - c Click **Finish**:



- 6 Repeat the above steps in order to browse for SLE-11-SP2-SDK-DVD-x86_64-GM-DVD1.iso.

Mounting SLES ISO Image Files

To locate the CD ROM drives that you created in the previous step, run this command on the queen node.

```
ls /dev/cdrom*
```

This command will return entries named similar to cdrom1 and cdrom2.

To mount the SLES ISO image files, perform these steps on the queen node:

- 1 Create the /var/opt/teradata/sles and the /var/opt/teradata/sles-sdk directories:

```
mkdir -p /var/opt/teradata/sles
mkdir -p /var/opt/teradata/sles-sdk
```

- 2 Mount the SLES ISO image files:

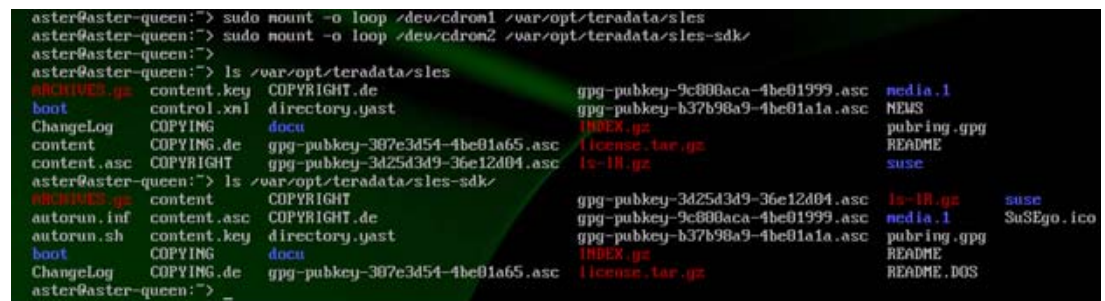
```
mount -o loop SLES-11-SP2-DVD-x86_64-GM-DVD1.iso /var/opt/teradata/sles
mount -o loop SLE-11-SP2-SDK-DVD-x86_64-GM-DVD1.iso /var/opt/teradata/sles-sdk
```

Confirming ISO Images Mounted

To confirm that the ISO image files are correctly mounted in the /var/opt/teradata/sles and /var/opt/teradata/sles-sdk directories, run these commands on the queen node:

```
ls /var/opt/teradata/sles
ls /var/opt/teradata/sles-sdk
```

If the ISO images are correctly mounted, the commands will return output similar to this output:



```
aster@aster-queen:~$ sudo mount -o loop /dev/cdrom1 /var/opt/teradata/sles
aster@aster-queen:~$ sudo mount -o loop /dev/cdrom2 /var/opt/teradata/sles-sdk
aster@aster-queen:~$ ls /var/opt/teradata/sles
ARCHIVES.gz  content.key  COPYRIGHT.de  gpg-pubkey-9c808aca-4be01999.asc  media.1
boot         control.xml  directory.gast  gpg-pubkey-b37b98a9-4be01a1a.asc  NEWS
ChangeLog    COPYING     docu           license.tar.gz  pubring.gpg
content      COPYING.de  gpg-pubkey-307e3d54-4be01a65.asc  README         suse
content.asc  COPYRIGHT   gpg-pubkey-3d25d3d9-36e12d04.asc  suse-10.gz
aster@aster-queen:~$ ls /var/opt/teradata/sles-sdk
ARCHIVES.gz  content      COPYRIGHT     gpg-pubkey-3d25d3d9-36e12d04.asc  is-10.gz  suse
autorun.inf  content.asc  COPYRIGHT.de  gpg-pubkey-9c808aca-4be01999.asc  media.1   SuSEgo.ico
autorun.sh   content.key  directory.gast  gpg-pubkey-b37b98a9-4be01a1a.asc  pubring.gpg
boot         COPYING     docu           license.tar.gz  README    README.DOS
ChangeLog    COPYING.de  gpg-pubkey-307e3d54-4be01a65.asc
```

If the files are not correctly located in the directories, unmount the directories and then perform the “[Mounting SLES ISO Image Files](#)” on page 33 procedure again. To unmount the directories, run these commands on the queen node:

```
umount /var/opt/teradata/sles
umount /var/opt/teradata/sles-sdk
```

Obtaining R Rpm Files

To obtain the R rpm files, Follow these steps to download the required R rpm files:

- 1 Download all R rpm files from these websites to a machine that can access external websites.
http://download.opensuse.org/repositories/devel:/languages:/R:/released/SLE_11_SP2/x86_64/
http://download.opensuse.org/repositories/devel:/languages:/R:/released/SLE_11_SP2/noarch/
- 2 If the `/var/opt/teradata/sles-r` directory does not already exist on the queen node, create the directory by running this command:

```
mkdir -p /var/opt/teradata/sles-r
```
- 3 Use the FTP client tool to move the rpm files from the machine on to which they were downloaded to the `/var/opt/teradata/sles-r` directory on the queen node.

Installing R

To install R, run this command on the queen:

```
ncli apm install R --usedefaultrepo=False \
--repo=sles,file:///var/opt/teradata/sles \
--repo=sles-sdk,file:///var/opt/teradata/sles-sdk \
--repo=sles-r1,file:///var/opt/teradata/sles-r
```

To confirm that R successfully installed on the queen node, run these commands on the queen:

```
ncli apm show R --localconfig
ncli apm show R --info
```

Syncing R Across All Workers

The previous command installs R only on the queen node. To sync R across all workers in the cluster and then activate the system, perform these ncli commands:

```
ncli system softrestart
ncli system activate
```

To confirm that R successfully installed on all worker nodes in the cluster, run this command on the queen node:

```
ncli apm show R --localconfig
```

The R and R packages on Aster are installed in the `/opt/aster/third-party/R` directory on the queen and worker nodes in the Aster Database cluster. Refer to [“Running a Sample R Script” on page 34](#) for an example of running a sample R script.

Running a Sample R Script

To ensure that you can run a sample R script, invoke an R program on the Aster Database cluster through SQL using the SQL-MR Stream module.

Perform these steps in order to run a sample R script:

- 1 Install a sample R script:
 - a On the queen node, create a file named `file.R` under the `/tmp` directory that contains this R script:

```
IN = file(description="stdin",open="r")
while(1)
{
```

```
# Create a frame to hold the input rows, without HEADER,
# and also to deal with end of stream
# read.table() is called inside the try block to detect when the
# program reaches the end of rows.

frame<-try(read.table(IN,header=FALSE,sep="\t",quote="",nrows=1,
  comment.char=""),silent=TRUE)
if(inherits(frame,"try-error"))
  break
write.table(frame,stdout(),col.names=FALSE,row.names=FALSE,quote=FALSE,sep="\t")
}
```

b Connect to the Aster Database, using the Aster Database Cluster Terminal (ACT) from the /tmp directory, and issue this command to install file.R:

```
beehive=>\install file.R
```

For information on using ACT, to connect to the Aster Database, refer to the Aster Client Guide.

2 Run the sample R script:

a Using ACT, connect to the Aster Database and issue these SQL statements:

```
create table mytable(a int) distribute by hash(a);
insert into mytable values (1000);
insert into mytable values (100);
SELECT * FROM stream(ON mytable SCRIPT('Rexec --vanilla file.R') OUTPUTS('*'));
```

b If the sample R script was executed successfully, the output from the above SELECT statement returns this result:

```
a
-----
1000
100
(2 rows)
```

Managing R Packages

This section provides information on managing R packages depending on whether or not the cluster has access to the internet. The topics include:

- [Managing R Package Operations With Internet Access](#)
- [Managing R Package Operations Without Internet Access](#)

Managing R Package Operations With Internet Access

This section provides information on managing R packages on a cluster that has access to the internet.

Listing Installed R Packages

To list the currently installed R packages, run this command:

```
ncli apm show R --packages
```

Checking R Packages Installation Status

To check the installation status of R packages, run this command:

```
ncli apm show R --packages=<RPackage1>,<RPackage2>,...
```

This example shows using the above command to check the installation status of a single package (tree):

```
ncli apm show R --packages=tree
```

This example shows using the above command to check the installation status of multiple packages (tree and lpSolve):

```
ncli apm show R --packages=tree,lpSolve
```

Installing R packages

To install R packages use this command.

```
ncli apm install R --packages=<RPackage1>,<RPackage2>,...
```

This example shows using the above command to install a single package (tree):

```
ncli apm install R --packages=tree
```

This example shows using the above command to install multiple packages (tree and lpSolve):

```
ncli apm install R --packages=tree,lpSolve
```

For a listing of all available R packages, visit the cran.r-project website, <http://cran.r-project.org>, and select Packages.



Some R packages have a dependency on OS packages. The installation of an R package will fail if any required OS packages that are required by the R package are not installed. To assist in determining the missing OS packages, check the log file, and then search on the internet for an error message that can identify the missing OS package. Once the missing OS packages are identified, install those before proceeding with installing the R package. For information on installing OS packages, see [“Installing and Upgrading OS Packages” on page 39](#).

Upgrading All R Packages

To upgrade all R packages use this command, which, although it will attempt to upgrade all installed R packages, will upgrade only the R packages that have a new version available.

```
ncli apm upgrade R --packages
```

Upgrading R Packages Selectively

No option is available to selectively upgrade a subset of the installed R packages. In order to update a subset of the installed R packages, use the R package install command (see [“Installing R packages” on page 36](#)), which, if new version is available, will update the specified R packages.

Removing R Packages

To remove R packages, use this command.

```
ncli apm remove R --packages=<RPackage1>,<RPackage2>,...
```

This example shows using the above command to remove a single package (tree):

```
ncli apm remove R --packages=tree
```

This example shows using the above command to remove multiple packages (tree and lpSolve):

```
ncli apm remove R --packages=tree,lpSolve
```

Managing R Package Operations Without Internet Access

This section provides information on managing R packages on a cluster that has no access to the internet.

Listing Installed R Packages

To list the currently installed R packages, run this command:

```
ncli apm show R --packages
```

Checking R Packages Installation Status

To check the installation status of R packages, run this command:

```
ncli apm show R --packages=<RPackage1>,<RPackage2>,...
```

This example shows using the above command to check the installation status of a single package (tree):

```
ncli apm show R --packages=tree
```

This example shows using the above command to check the installation status of multiple packages (tree and lpSolve):

```
ncli apm show R --packages=tree,lpSolve
```

Obtaining R Packages

Because the queen node on the cluster doesn't have access to the internet, you must first download the R packages to a machine that has access to the internet, and then copy the R packages from their download location to the queen node on the cluster. Follow these steps to download the R packages:

- 1 For a listing of all available R packages, visit the cran.r-project website, <http://cran.r-project.org>, and select Packages.
- 2 Download all R packages you want to install from the cran.r-project website, <http://cran.r-project.org>, to the machine you have selected that has access to the internet. Run this command to download the R packages:

```
wget http://cran.r-project.org/src/contrib/<R_package>
```

This example shows using the above command to download a single R package (lpSolve):

```
wget http://cran.r-project.org/src/contrib/lpSolve_5.6.10.tar.gz
```

- 3 If the `/var/opt/teradata/rpackages` directory does not already exist on the queen node, create the directory by running this command:

```
mkdir -p /var/opt/teradata/rpackages
```

- 4 Copy the R packages from the machine on to which they were downloaded to the `/var/opt/teradata/rpackages` directory on the queen node of the cluster.

Installing R packages

To install R packages use these commands.



If you are installing an R package (*x*) that has a dependency on other R packages (*y,z,...*), ensure that the R packages (*y,z,...*) are listed before R package (*x*) in the command syntax.



Some R packages have a dependency on OS packages. The installation of an R package will fail if any required OS packages that are required by the R package are not installed. To assist in determining the missing OS packages, check the log file, and then search on the internet for an error message that can identify the missing OS package. Once the missing OS packages are identified, install those before proceeding with installing the R package. For information on installing OS packages, see [“Installing and Upgrading OS Packages” on page 39](#).

```
cd /var/opt/teradata/rpackages
ncli apm install R --packages=<RPackage1TarFile>,<RPackage2TarFile>,...
```

This example shows using the install command to install a single package (lpSolve):

```
cd /var/opt/teradata/rpackages
ncli apm install R --packages=lpSolve_5.6.10.tar.gz
```

This example shows using the install command to install multiple packages (lpSolve and sampling):

```
cd /var/opt/teradata/rpackages
ncli apm install R --packages=lpSolve_5.6.10.tar.gz,sampling_2.6.tar.gz
```

Upgrading All R Packages

No option is available to upgrade the installed R packages from the local file. In order to update one or a list of the installed R packages, use the R package install command (see [“Installing R packages” on page 38](#)), which, if new version is available, will update the specified R packages.

Upgrading R Packages Selectively

No option is available to selectively upgrade a subset of the installed R packages. In order to update a subset of the installed R packages, use the R package install command (see [“Installing R packages” on page 38](#)), which, if new version is available, will update the specified R packages.

Removing R Packages

To remove R packages, use this command.

```
ncli apm remove R --packages=<RPackage1>,<RPackage2>,...
```

This example shows using the above command to remove a single package (tree):

```
ncli apm remove R --packages=tree
```

This example shows using the above command to remove multiple packages (tree and lpSolve):

```
ncli apm remove R --packages=tree,lpSolve
```

Installing and Upgrading OS Packages

This section provides information on installing and managing the OS packages inside of the R sandbox area. The topics include:

- [Installing OS Packages Inside R Sandbox Area](#)
- [Upgrading OS Packages Installed in R Sandbox Area](#)
- [Checking OS Package Status Installed in R Sandbox Area](#)

Installing OS Packages Inside R Sandbox Area

Some R packages have a dependency on OS packages. To resolve this dependency issue, the `ncli apm install R` command has an option for installing OS packages inside the R sandbox area.

To install OS packages inside the R sandbox area, use this command.

```
ncli apm install R --rpmpackages=<package1>,<package2>,... \
[--repo=<repoName1>,<repoUrl1>] [--repo=<repoName2>,<repoUrl2>] ...
```

This example shows using the above command to install a single OS package (tar) without specifying a repository:

```
ncli apm install R --rpmpackages=tar
```

This example shows using the above command to install multiple OS packages (less and zip) without specifying a repository:

```
ncli apm install R --rpmpackages=less,zip
```

There are two methods you can use to specify a repository; one method creates a temporary repository and the other method creates a permanent repository. To create a temporary repository, specify the `--repo` option in the above command. To create permanent repository, before running the above command, use the

`ncli apm administer R --setuprepo=<repoName>,<repoURL>` command. For information on creating a permanent repository, see [“Adding a Permanent Repository” on page 17](#).



The advantage to creating a permanent repository is that you will not need to specify the `--repo` option every time you run the command, whereas a temporary repository is removed when the command is exited.

This example shows using the `--repo` option in the above command to install a single OS package (tar):

```
ncli apm install R --rpmpackages=tar \
--repo=rhel,file:///var/opt/teradata/rhel
```

This example shows using the `--repo` option in the above command to install multiple OS packages (less and zip):

```
ncli apm install R --rpmpackages=less,zip \
--repo=rhel,file:///var/opt/teradata/rhel
```

This example shows creating a permanent repository, and then installing a single OS package (tar) followed by installing multiple OS packages (less and zip):

```
ncli apm administer R --setuprepo=rhel,file:///var/opt/teradata/rhel
ncli apm install R --rpmpackages=tar
ncli apm install R --rpmpackages=less,zip
```

Upgrading OS Packages Installed in R Sandbox Area

To upgrade OS packages installed in the R sandbox area, use this command.

```
ncli apm upgrade R --rpmpackages=<package1>,<package2>,... \
[--repo=<repoName1>,<repoUrl1>] [--repo=<repoName2>,<repoUrl2>] ...
```

This example shows using the `--repo` option in the above command to upgrade a single OS package (tar):

```
ncli apm upgrade R --rpmpackages=tar \
--repo=rhel,file:///var/opt/teradata/rhel
```

This example shows using the `--repo` option in the above command to upgrade multiple OS packages (less and zip):

```
ncli apm upgrade R --rpmpackages=less,zip \
--repo=rhel,file:///var/opt/teradata/rhel
```

This example shows creating a permanent repository, and then upgrading a single OS package (tar) followed by upgrading multiple OS packages (less and zip):

```
ncli apm administer R --setuprepo=rhel,file:///var/opt/teradata/rhel
ncli apm upgrade R --rpmpackages=tar
ncli apm upgrade R --rpmpackages=less,zip
```

Checking OS Package Status Installed in R Sandbox Area

To check the status of OS packages currently installed in the R sandbox area, use this command.

```
ncli apm show R --rpmpackage=<OS_package>
```

This example shows using the above command to check the status of an OS package (tar):

```
ncli apm show R --rpmpackage=tar
```

Upgrading R

To upgrade the version of R installed on your cluster, select the procedure that aligns with your operating system environment:

- [Upgrading R on RHEL or SLES With Internet Access](#)
- [Upgrading R on RHEL Without Internet Access](#)
- [Upgrading R on SLES Without Internet Access](#)

Upgrading R on RHEL or SLES With Internet Access

This section describes the process to follow in order to upgrade the installed version of R on an Aster Database cluster running a supported version of the RHEL OS or the SLES OS in which the queen node can access external websites to download the required rpm packages.

Upgrading R

To upgrade the installed version of R, run this command on the queen:


```
ncli apm upgrade R
```

This command internally connects to http://ftp1.scientificlinux.org/linux/scientific/6.4/x86_64/os/Packages/ or http://ftp1.scientificlinux.org/linux/scientific/6.5/x86_64/os/Packages/ (depending on the version of RHEL) and http://download.fedoraproject.org/pub/epel/6/x86_64/ on RHEL and http://download.opensuse.org/repositories/devel:/languages:/R:/released/SLE_11_SP2 on SLES in order to fetch the R rpms. R will be updated to the version available from those urls.

If the R upgrade fails, verify that urls are accessible. If the urls are not accessible, then upgrade R using one of these alternative methods, “[Upgrading R on RHEL Without Internet Access](#)” on page 41 or “[Upgrading R on SLES Without Internet Access](#)” on page 43.

To check the version of R installed on queen node, run this command on the queen node. If R was successfully upgraded on queen node, the version of R on the queen node is higher than that of the worker nodes.

```
ncli apm show R --localconfig
```

Syncing R Across All Workers

The previous command upgrades R only on the queen node. To sync R across all workers in the cluster and then activate the system, perform these ncli commands:

```
ncli system softrestart  
ncli system activate
```

To confirm that R successfully upgraded on all worker nodes in the cluster, run this command on the queen node:

```
ncli apm show R --localconfig
```

Upgrading R on RHEL Without Internet Access

This section describes the process to follow in order to upgrade the installed version of R on an Aster Database cluster running a supported version of the RHEL OS in which the queen node cannot access external websites to download the required rpm packages.

Obtaining R RPM Files

Because the queen node on the Aster Database cluster doesn't have access to the internet, you must first download the R rpm files to a machine that has access to the internet, and then copy the rpm files from their download location to the queen node on the Aster Database cluster.

Follow these steps to download the required R and R dependency rpm files:

- 1 Perform these steps to download all required rpm files to the machine you have selected that has access to the internet.

- a Depending on the version of the installed RHEL OS, perform one of these actions:

- For a system running version 6.4 of the RHEL OS, run this command:

```
url="http://ftp1.scientificlinux.org/linux/scientific/6.4/x86_64/os/Packages/"
```

- For a system running version 6.5 of the RHEL OS, run this command:

```
url="http://ftp1.scientificlinux.org/linux/scientific/6.5/x86_64/os/Packages/"
```

- b Run these commands to download additional required rpm files:

```
pkgs="blas-|freetype-|lapack-|libicu-|libjpeg|texinfo-tex"
```

```
for pkg in `wget -q $url -O - 2>&1 | grep -Po "(?<=>) ($pkgs) .* (noarch|x86_64) .rpm (?=<) "`; \
do wget -nc $url/$pkg; done
```

c Run these commands to download additional required rpm files:

```
url="http://download.fedoraproject.org/pub/epel/6/x86_64/"
```

```
pkgs="R-|libRmath-"
```

```
for pkg in `wget -q $url -O - 2>&1 | grep -Po "(?<=>) ($pkgs) .* (noarch|x86_64) .rpm (?=<) "`; \
do wget -nc $url/$pkg; done
```



After running above commands, verify that the R, R-devel, R-core, R-core-devel, R-java, R-java-devel, texinfo-tex, libRmath, and libRmath packages are downloaded. If any of the packages are not downloaded, repeat the above steps.

- 2 If the `/var/opt/teradata/rhel-r` directory does not already exist on the queen node of the Aster Database cluster, create the directory by running this command:

```
mkdir -p /var/opt/teradata/rhel-r
```

- 3 Copy the rpm files from the machine on to which they were downloaded to the `/var/opt/teradata/rhel-r` directory on the queen node of the Aster Database cluster.
- 4 Create a yum repository from all files copied in the previous step. To create the repository, run this command:

```
cd /var/opt/teradata/rhel-r
createrepo .
```

Upgrading R

To upgrade the installed version of R, run this command on the queen:

```
ncli apm upgrade R --usedefaultrepo=False \
--repo=rhel,file:///var/opt/teradata/rhel \
--repo=rhel-r,file:///var/opt/teradata/rhel-r
```

To check the version of R installed on queen node, run this command on the queen node. If R was successfully upgraded on queen node, the version of R on the queen node is higher than that of the worker nodes.

```
ncli apm show R --localconfig
```

Syncing R Across All Workers

The previous command upgrades R only on the queen node. To sync R across all workers in the cluster and then activate the system, perform these ncli commands:

```
ncli system softrestart
ncli system activate
```

To confirm that R successfully upgraded on all worker nodes in the cluster, run this command on the queen node:

```
ncli apm show R --localconfig
```

Upgrading R on SLES Without Internet Access

This section describes the process to follow in order to upgrade the latest version of R on an Aster Database cluster running a supported version of the SLES OS in which the queen node cannot access external websites to download the required rpm packages.

Obtaining R Rpm Files

Because the queen node on the Aster Database cluster doesn't have access to the internet, you must first download the R rpm files to a machine that has access to the internet, and then copy the rpm files from their download location to the queen node on the Aster Database cluster.

Follow these steps to download the required R and R dependency rpm files:

- 1 Run these two commands to download all required rpm files to the machine you have selected that has access to the internet:

```
wget -r -nd --level=1 -e robots=off -H -A rpm \  
http://download.opensuse.org/repositories/devel:/\  
languages:/R:/released/SLE_11_SP2/x86_64/
```

```
wget -r -nd --level=1 -e robots=off -H -A rpm \  
http://download.opensuse.org/repositories/devel:/\  
languages:/R:/released/SLE_11_SP2/noarch/
```

- 2 If the `/var/opt/teradata/sles-r` directory does not already exist on the queen node of the Aster Database cluster, create the directory by running this command:

```
mkdir -p /var/opt/teradata/sles-r
```
- 3 Copy the rpm files from the machine on to which they were downloaded to the `/var/opt/teradata/sles-r` directory on the queen node of the Aster Database cluster.

Upgrading R

To upgrade the installed version of R, run this command on the queen:

```
ncli apm upgrade R --usedefaultrepo=False \  
--repo=sles-r1,file:///var/opt/teradata/sles-r
```

To check the version of R installed on queen node, run this command on the queen node. If R was successfully upgraded on queen node, the version of R on the queen node is higher than that of the worker nodes.

```
ncli apm show R --localconfig
```

Syncing R Across All Workers

The previous command upgrades R only on the queen node. To sync R across all workers in the cluster and then activate the system, perform these ncli commands:

```
ncli system softrestart  
ncli system activate
```

To confirm that R successfully upgraded on all worker nodes in the cluster, run this command on the queen node:

```
ncli apm show R --localconfig
```

Uninstalling R

To uninstall R from the queen node on the Aster Database cluster, use this command.

```
ncli apm remove R
```

The previous command removes the `/opt/aster/third-party/R` directory from the queen node. To remove R from all workers in the cluster and then activate the system, perform these ncli commands:

```
ncli system softrestart
ncli system activate
```

R Package Installation

This section provides installation instructions for these commonly used R packages:

R Package	Installation Instruction Link
nloptr	Installing the R package 'nloptr' on RHEL or SLES
caret	Installing the R package 'caret' on RHEL or SLES
rJava	Installing the R package 'rJava' on SLES
rjags	Installing the R package 'rjags' on RHEL Installing the R package 'rjags' on SLES

Installing the R package 'nloptr' on RHEL or SLES

Perform these tasks to install the R package 'nloptr':

- 1 Install the system package tar. For example:


```
ncli apm install R --rpm-packages=tar \
--repo=sles,file:///var/opt/teradata/sles
```

Note: Change the repo path, if mounted on different path.
- 2 If your cluster is connected to the internet, proceed to Task 3. If your cluster is not connected to the internet, download the latest version of the system package 'NLOpt' tar.gz file (there is no 'r' at the end of the package name) from the website and then install it.


```
http://ab-initio.mit.edu/wiki/index.php/NLOpt
```
- 3 Install the R package 'nloptr'.
 - a For online installation, issue this command:


```
ncli apm install R --packages=nloptr
```
 - b For offline installation, issue this command:


```
ncli apm install R --packages=nloptr_1.0.4.tar.gz
```

Note: Replace `nloptr_1.0.4.tar.gz` with exact file name that you downloaded from the internet.

Installing the R package 'caret' on RHEL or SLES

Perform these tasks to install the R package 'caret':

- 1 Install the system package tar. For example:

```
ncli apm install R --rpmpackages=tar \
--repo=sles,file:///var/opt/teradata/sles
```

Note: Change the repo path, if mounted on different path.
- 2 If your cluster is connected to the internet, proceed to Task 3. If your cluster is not connected to the internet, download the latest version of the system package 'NLOpt' tar.gz file (there is no 'r' at the end of the package name) from the website and then install it.
<http://ab-initio.mit.edu/wiki/index.php/NLOpt>
- 3 Install the R package 'caret'.
 - a For online installation, issue this command:

```
ncli apm install R --packages=caret
```
 - b For offline installation, issue this command:

```
ncli apm install R --packages=caret_6.0-47.tar.gz
```

Note: Replace `caret_6.0-47.tar.gz` with exact file name that you downloaded from the internet.

Installing the R package 'rJava' on SLES

Perform these tasks to install the R package 'rJava' on the SLES OS:

- 1 Download the Oracle JDK (`jdk-7u75-linux-x64.rpm`) file from <http://www.oracle.com/technetwork/java/javase/downloads/jdk7-downloads-1880260.html>, and copy it to a directory of your choosing (for example, `/tmp/jdk`).
- 2 Install the JDK from the rpm file. For example:

```
ncli apm install R --rpmpackages=jdk --repo=jdk,file:///tmp/jdk/
```
- 3 Configure the java path with R by issuing this command:

```
chroot /opt/aster/third-party/R/ /bin/bash -c \
"export JAVA_HOME=/usr/java/jdk1.7.0_75/jre; R CMD javareconf"
```
- 4 Install the R package 'rJava'.
 - a For online installation, issue this command:

```
ncli apm install R --packages=rJava
```
 - b For offline installation, issue this command:

```
ncli apm install R --packages=rJava_0.9-6.tar.gz
```

Note: Replace `rJava_0.9-6.tar.gz` with exact file name that you downloaded from the internet.

Installing the R package 'rjags' on RHEL

Perform these tasks to install the R package 'rjags' on the RHEL OS:

- 1 The R package 'rjags' is dependent on system package 'JAGS'. However, the RPM file for JAGS is not available. As a result, you must install JAGS from source code. Download the JAGS source code from this location:

```
wget http://sourceforge.net/projects/mcmc-jags/files/JAGS/3.x/\
Source/JAGS-3.4.0.tar.gz
```
- 2 Install the system package 'JAGS' from the source code into the R sandbox environment by issuing these commands:
 - a

```
cp JAGS-3.4.0.tar.gz /opt/aster/third-party/R/
```
 - b

```
chroot /opt/aster/third-party/R/ /bin/bash -c \
"tar -xf JAGS-3.4.0.tar.gz"
```
 - c

```
chroot /opt/aster/third-party/R/ /bin/bash -c "cd JAGS-3.4.0; \
./configure --libdir=/usr/local/lib64 && make && make install"
```
- 3 Install the R package 'rjags'.
 - a For online installation, issue this command:

```
ncli apm install R --packages=rjags
```
 - b For offline installation, issue this command:

```
ncli apm install R --packages=rjags_3-15.tar.gz
```


Note: Replace `rjags_3-15.tar.gz` with exact file name that you downloaded from the internet.

Installing the R package 'rjags' on SLES

Perform these tasks to install the R package 'rjags' on the SLES OS:

- 1 The R package 'rjags' is dependent on system package 'JAGS'. However, the RPM file for JAGS is not available. As a result, you must install JAGS from source code. Download the JAGS source code from this location:

```
wget http://sourceforge.net/projects/mcmc-jags/files/JAGS/3.x/\
Source/JAGS-3.4.0.tar.gz
```
- 2 Determine if any system package 'blas' is already installed within the R sandbox environment by issuing this command, which may return two entries:

```
rpm --root=/opt/aster/third-party/R/ -qa | grep blas
```
- 3 If the above command indicates any system package 'blas' is already installed, remove all such packages by issuing this command, otherwise ignore this step.

```
rpm --root=/opt/aster/third-party/R/ -qa | grep blas | xargs rpm -e
```
- 4 The system package 'JAGS' is dependent on another system package, lapack. Install 'lapack' by issuing this command.

```
ncli apm install R --rpm-packages=lapack --repo=lapack,http://\
download.opensuse.org/repositories/home:/BSiposRKF/SLE_11_SP2/
```
- 5 Install the system package 'JAGS' from the source code into the R sandbox environment by issuing these commands:
 - a

```
cp JAGS-3.4.0.tar.gz /opt/aster/third-party/R/
```
 - b

```
chroot /opt/aster/third-party/R/ /bin/bash -c \
```

```
"tar -xf JAGS-3.4.0.tar.gz"
```

```
c chroot /opt/aster/third-party/R/ /bin/bash -c "cd JAGS-3.4.0; \  
./configure --libdir=/usr/local/lib64 && make && make install"
```

6 Install the R package 'rjags'.

a For online installation, issue this command:

```
ncli apm install R --packages=rjags
```

b For offline installation, issue this command:

```
ncli apm install R --packages=rjags_3-15.tar.gz
```

Note: Replace `rjags_3-15.tar.gz` with exact file name that you downloaded from the internet.

CHAPTER 3 Aster R Package

This section provides information on these topics:

- [Aster R Package Installation](#)
- [Aster R Package Overview](#)
- [Aster R Quick Start Reference](#)
- [Aster R Package Function Reference](#)
- [Aster R Package Limitations](#)
- [Aster R Package Considerations](#)
- [Data Type Mapping](#)

Aster R Package Installation

Requirements

The minimum installed software environment requirements for the Aster R package are:

- Aster Database, version 6.10 (AD 6.10)
- Aster Client, version 6.20 (AC 6.20)
- Aster ODBC Driver (version AC 6.20 or higher)

Prerequisites

- Verify that the version of R installed on the client and server is the same.



The version of R installed on the client and on the server must be the same in order to ensure proper operation of the Teradata Aster R client software.

- Ensure that ODBC is installed and configured on your system. For information on installing and configuring ODBC, refer to “Installing and Configuring ODBC” in the Aster Client Guide.
- To display bytea data correctly, Teradata recommends editing the `odbc.ini` file to include this setting:

ByteaAsVarchar=1

For information on editing the `odbc.ini` file, refer to “Installing and Configuring ODBC” in the Aster Client Guide.

Obtaining Aster R package

To obtain the Aster R package:

- 1 Find the most recent Aster R package posted on Teradata Developer Exchange:
<http://downloads.teradata.com/download/aster>
- 2 Click on the package name for your operating system.
- 3 Log in to Teradata Developer Exchange, if prompted.
- 4 Accept the End User License Agreement.
- 5 Copy the package to the computer you'll use to query your database.
- 6 Expand the package by unzipping or untarring the file, as appropriate.

Installation and Upgrade Note

After installing or upgrading to a newer version of the Aster R package, RStudio may not preview the inline help properly, and return an error similar to `In fetch(key) : internal error -3 in R_decompress1`. This issue is caused by a limitation in RStudio. The workaround for this limitation is to exit and then restart RStudio.

Installing or Upgrading Aster R package on Linux x86_64

To install or upgrade the Aster R package on Linux x86_64, execute this command from the R shell:

```
> install.packages("<<path to the package>>/  
TeradataAsterR_1.0.0_R_x86_64-unknown-linux-gnu.tar.gz")
```

The library path to the ODBC Driver Manager (for example, unixODBC), which is bounded to the RODB package, must be included in `LD_LIBRARY_PATH`.

Installing or Upgrading Aster R package on Linux i686

To install or upgrade the Aster R package on Linux i686, execute this command from the R shell:

```
> install.packages("<<path to the package>>/TeradataAsterR_1.0.0_R_i686-  
unknown-linux-gnu.tar.gz")
```

The library path to the ODBC Driver Manager (for example, unixODBC), which is bounded to the RODB package, must be included in `LD_LIBRARY_PATH`.

Installing or Upgrading Aster R package on Windows 32-bit and 64-bit

To install or upgrade the Aster R package on Windows 32-bit and 64-bit:

- 1 Execute this commands from the R shell:

```
> install.packages("<<path to the package>>/TeradataAsterR_1.0.0.zip")
```

Installing or Upgrading Aster R package Using RStudio

The RStudio menu-driven install process produces the `install.packages` command for users. Perform these steps to install or upgrade the Aster R package using RStudio:

- 1 From the RStudio menu, select **Tools > Install packages...**
- 2 In the **Install packages...** window:
 - a Set the "Install from" option to **Package Archive File (zip, tar.gz)**.
 - b Browse to set the "Package Archive" option to the path of the **TradataAsterR Client** package.
- 3 Click **Install**.

Uninstalling Aster R package

To uninstall the Aster R package, execute this command from the R shell:

```
> uninstall.packages("<<Aster_R_Package_Name>>")
```

Aster R Package Overview

The Aster R package primarily operates and computes on data sets present in the Aster Database. To expose the data sets to R users, an object type called a Virtual Data Frame is used. A Virtual Data Frame is an extension of a regular R data frame, and is specifically designed to access and manipulate data sets present in the Aster Database in order to make them look similar to an R data frame. The class attribute of a Virtual Data Frame is set to "ta.data.frame". An instance of a Virtual Data Frame maps to a table, view or query in the database, and any access or manipulation on a Virtual Data Frame is translated into an equivalent SQL query that is executed in the Aster Database. This SQL query based access and manipulation of the Virtual Data Frame enables easy scaling of R programming computation over Aster Database objects.

The Aster R package APIs can be broadly classified under these categories:

- **Table Access Functions:** This class of functions provides data access functions that make Aster Database tables accessible to R clients, and supports basic data exploration functions.
- **Aster Analytics Wrapper Functions:** This class of functions provides access to analytics functions present in database.
- **In-Database Runner Functions:** This class of functions provides a mechanism to execute user-defined R functions in the Aster Database.
- **Fast Import and Export Functions:** This class of functions is the set of APIs that provide fast importing and exporting of data to and from the Aster Database.

Accessing Aster R Inline Documentation

Complete documentation for all Aster R package functions is provided as inline documentation, which is accessible from the R console by using the `help(<function_name>)` command.

For example:

```
help(ta.data.frame)
```

Always refer to the inline documentation for detailed information on each function.

Accessing Aster R Vignette Documentation

In addition to inline documentation for individual functions, documentation in the form of vignettes is available for these Aster R topics:

- `introduction` - Provides an overview of the Aster R package.
- `installation` - Describes how to install the Aster R package.
- `connection` - Describes how to connect to the Teradata Aster Database.
- `virtualObjects` - Describes the Aster Virtual Data Frame.
- `fastImportExport` - Describes fast loading and exporting using the Aster Virtual Data Frame.
- `inDBRunner` - Describes how to use the In-Database R Function Runners.

To access the complete library of Aster R vignettes, from the R console, use this command:

```
vignette(package="TeradataAsterR")
```

To view a specific vignette for an Aster R topic listed above, from the R console, issue the command:

```
vignette("<vignette_name>", package="TeradataAsterR")
```

The specified vignette will open in your default browser.

For example:

```
vignette("virtualObjects", package="TeradataAsterR")
```

Aster R Quick Start Reference

This section provides information on these topics, which will enable you to quickly begin using the Aster R package:

- [Checking R Installation](#)
- [Attaching Aster R Package Library](#)
- [Creating Connection to Aster Database](#)
- [Exporting Data from Aster to R](#)
- [Loading Data from R to Aster](#)
- [Using Aster Virtual Data Frames](#)

- [Creating Tables in Aster Database](#)
- [Using In-Database R Function Runner](#)

Checking R Installation

Use the `ta.is.R.installed()` function to check if R is installed in the Aster Database.

For complete information, refer to the inline documentation for the `ta.is.R.installed()` function.

Example

```
ta.is.R.installed(silent = TRUE)
```

Attaching Aster R Package Library

Use the `require()` function to attach the Aster R package library (`TeradataAsterR`) to the current working session.

Example

```
require("TeradataAsterR")
```

Creating Connection to Aster Database

Use the `ta.connect()` function to establish a connection to the Aster Database. This call will return an RODB connection to the Aster Database. The same object is assigned to a global variable `taConnection`, which is implicitly used in other functions if you do not supply an explicit RODB connection object in the function call.



You must ensure that the Aster Database ODBC driver is properly installed and configured on the system where the `ta.connect()` function is invoked. Refer to the *Aster Client Guide* for information on installing and configuring the Aster Database ODBC driver.

For complete information, refer to the `connection` topic under the vignette documentation.

Consideration

If a user does not have permission to execute a stream function during `ta.connect()` or while using an in-database R function (which internally uses a stream function), an error is returned indicating that permission is denied. To work around the permission denied issue, issue this command:

```
GRANT EXECUTE ON FUNCTION nc_system.stream to <user>
```

Aster R creates a temporary schema for the connected user. If the user doesn't have permission to create a schema, the default schema is used. To work around using the default schema, issue this command to grant permission:

```
GRANT CREATE ON DATABASE <database> to <user>
```

Examples

```
...{r, eval = FALSE}  
ta.connect("AsterDSN")  
ta.connect("AsterDSN", uid = "ABC", pwd = "123", database = "DB1", dType  
= "odbc")  
...
```

Exporting Data from Aster to R

Use the `ta.pull()` function to export data from Aster to R. This function returns a data frame containing a representation of the data in the file. Character strings in the result (including factor levels) have a declared encoding (default "UTF-8").

Use the `ta.write.csv()` function to export data from Aster to a CSV file.

For complete information, refer to the `fastImportExport` topic under the vignette documentation.

Example

```
...{r, eval = FALSE}  
## create a data frame  
df <- data.frame(matrix(c(1 : 12),  
                        nrow = 4, ncol = 3,  
                        dimnames = list(c(1 : 4), c("col1", "col2", "col3"))))  
  
ta.create(df,  
          table = "tmp_table",  
          schemaName = "public",  
          tableType = "fact",  
          partitionKey = "col1",  
          row.names = TRUE)  
  
tadf <- ta.data.frame("tmp_table")  
  
## export data from an Aster data frame  
ta.pull(tadf)  
  
## export data from an Aster data frame to a csv file  
ta.write.csv(tadf, "data.csv")  
...
```

Loading Data from R to Aster

Use the `ta.push()` function to load data from R to Aster.

Use the `ta.read.csv()` function to load data from a CSV file to Aster.

For complete information, refer to the `fastImportExport` topic under the vignette documentation.

Example

```
...{r, eval = FALSE}  
## create an empty Aster data frame if you don't already have one  
col1 <- integer()  
col2 <- integer()
```

```
col3 <- integer()
mydata <- data.frame(col1, col2, col3)
ta.create(mydata, table="tmp_table", schemaName="public",
          colTypes=c(col1="integer", col2="integer", col3="integer"),
          tableType="fact", partitionKey="col1", row.names=FALSE)
tadf <- ta.data.frame("tmp_table")

ta.head(tadf)
## There is no data in tmp_table
##      [1] col1 col2 col3
##      <0 rows> (or 0-length row.names)

## create a data frame
df <- data.frame(matrix(c(1 : 12),
                        nrow = 4, ncol = 3,
                        dimnames = list(c(1 : 4), c("col1", "col2", "col3"))))

print(df)
## The data of the data frame df
##      col1 col2 col3
##      1    1    5    9
##      2    2    6   10
##      3    3    7   11
##      4    4    8   12

## load data from the data frame to the Aster data frame
ta.push(tadf, df)

ta.head(tadf)
## The data in tmp_table
##      col1 col2 col3
##      1    1    5    9
##      2    3    7   11
##      3    2    6   10
##      4    4    8   12

## load data from a csv file to the Aster data frame
ta.read.csv(tadf, "data.csv")

ta.head(tadf)
## The data in tmp_table
##      col1 col2 col3
##      1    1    5    9
##      2    1    5    9
##      3    2    6   10
##      4    2    6   10
##      5    3    7   11
##      6    3    7   11
##      7    4    8   12
##      8    4    8   12
## ...
```

Using Aster Virtual Data Frames

Aster Virtual Data Frame

Use the `ta.data.frame()` function to create a virtual data frame, which is based on an underlying select query on a table. Elements of the query are specified as input arguments to the function.

In addition to `ta.data.frame()`, the `ta.vector()` and `ta.factor()` functions are provided to represent a single column in the table. Note, a constructor for `ta.vector()` and `ta.factor()` is not included in the Aster R package, but `as.ta.vector()` and `as.ta.factor()` can be used to convert an existing virtual data frame having single column to `ta.vector()` or `ta.factor()`.

For complete information, refer to the `virtualObjects` topic under the vignette documentation.

Examples

```
...{r, eval = FALSE}
ta.data.frame("tmp_table")
ta.data.frame("tmp_table", schemaName = "public")
ta.data.frame("SELECT * FROM tmp_table", sourceType = "query")
...
```

Convert R Object

Use the `as.ta.data.frame()` function to convert an R object, for example, `data.frame` or Aster virtual data frame, `ta.data.frame`.

Example

```
...{r, eval = FALSE}
df <- data.frame(matrix(c(1:12), nrow = 4, ncol = 3,
                        dimnames = list(c(1:4), c("col1", "col2", "col3"))))
tadf <- as.ta.data.frame(df,
                        table = "tmp_table", schemaName = "public",
                        tableType = "fact", partitionKey = "col1",
                        row.names = TRUE)
...
```

Convert R Object to Aster Virtual Vector

Use the `as.ta.vector()` function to convert an R object, for example, `data.frame` or an Aster virtual object, for example, `ta.data.frame`, to a `ta.vector`.

Consideration

The object to be converted must have only one column.

Example

```
...{r, eval = FALSE}
vec <- c(1, 2, 3)
tadf <- as.ta.vector(vec,
                    table = "tmp_table", schemaName = "public", tableType =
"dimension")
...
```

Convert R Object to Aster Virtual Factor

Use the `as.ta.factor()` function to convert an R object, for example, `data.frame` or an Aster virtual object, for example, `ta.data.frame`, to a `ta.factor`.

Consideration

The object to be converted must have only one column and the elements type must be character.

Example

```
...{r, eval = FALSE}
fac <- as.factor(c("a", "b", "c"))
tadf <- as.ta.factor(fac,
  table = "tmp_table", schemaName = "public", tableType =
  "dimension")
...
```

Creating Tables in Aster Database

Use the `ta.create()` function to create tables in the Aster Database by using an R data frame and optional settings. The parameters are converted into a regular SQL query that creates a table in the Aster Database.

For complete information, refer to the `fastImportExport` topic under the vignette documentation.

Examples

```
...{r, eval = FALSE}
df <- data.frame(matrix(c(1:12),
  nrow = 4, ncol = 3,
  dimnames = list(c(1:4), c("col1", "col2", "col3"))))

ta.create(df,
  table = "tmp_table",
  schemaName = "public",
  tableType = "fact",
  partitionKey = "col1",
  row.names = TRUE)
...
```

Using In-Database R Function Runner

These functions (and wrapper functions) enable users to translate an R function into an R script or an R expression:

- [ta.eval\(\) Function](#)
- [ta.apply\(\)](#)
- [ta.tapply\(\)](#)

The Aster in-database R function runners parallelize R programs in a partial-final manner under the Map Reduce framework in Aster. The key capability offered by this construct is the ability to push a block of R code inside the Aster database as opposed to the execution of a single R function.

For complete information, refer to the `inDBRunner` topic under the vignette documentation.

ta.eval() Function

Use the `ta.eval()` function to calculate an R function without any input from an Aster data frame.

Examples

```
...{r, eval = FALSE}  
ta.eval(R.Version)  
ta.eval(function(a, b, c) return(a + b + c), 1, 2, 3, FUN.upload =  
"script")  
ta.eval("sum", 1, 2, 3)
```

ta.apply() Function

Use the `ta.apply()` function to apply an R function on a given Aster data frame.

ta.aggregateApply() Function

The `ta.aggregateApply()` function applies a given R function on tables of an Aster data frame.

ta.colApply() Function

The `ta.colApply()` function applies a given R function on columns of an Aster data frame.

ta.rowApply() Function

The `ta.rowApply()` function applies a given R function on rows of an Aster data frame.

Considerations

- Character values with "\n" are not supported.
- The R apply function operates only on matrices and not data frames. If the input to the R apply function is a data frame, the function tries to convert the data frame to a matrix using the `as.matrix` function.

Because matrices cannot contain mixed data types, the `as.matrix` function converts numeric columns to strings and creates a matrix of strings. As a result, R functions must not be used on mixed type data frames.

- You must project out the columns in the `tadf` argument of the `ta.apply` function so that all input columns are either numeric or character.

Examples

```
...{r, eval = FALSE}  
tadf <- ta.data.frame("tablename")  
ta.apply(tadf, 1, FUN = sum)  
ta.apply(tadf, 1, FUN = function(x, b) return(x + b), b = 1, FUN.upload =  
"script")
```

ta.tapply() Function

Use the `ta.tapply()` function to apply an R function on a partitioned Aster data frame.

ta.partitionApply() Function and ta.by() Function

Use the `ta.partitionApply()` and `ta.by()` functions as a partitioned-oriented shortcut for the `ta.tapply()` function.

Considerations

- Character values with "\n" are not supported.
- The R apply function operates only on matrices and not data frames. If the input to the R apply function is a data frame, the function tries to convert the data frame to a matrix using the `as.matrix` function.

Because matrices cannot contain mixed data types, the `as.matrix` function converts numeric columns to strings and creates a matrix of strings. As a result, R functions must not be used on mixed type data frames.

- You must project out the columns in the `tadf` argument of the `ta.apply` function so that all input columns are either numeric or character.

Example

```
...{r, eval = FALSE}
tadf <- ta.data.frame("tablename")
ta.tapply(tadf, FUN = sum, INDEX = tadf[, c("val_3")])
```

Aster R Package Function Reference

This section provides a categorized complete listing of the functions included in the Aster R package, and includes these topics:

- [Utility and Management Functions](#)
- [Data Structure Management Functions](#)
- [Data Exploration Functions](#)
- [Data Transformation Functions](#)
- [Basic Statistical Functions](#)
- [R Map Reduce Runner Functions](#)
- [Parallel Stats and Big Data Analytics Functions](#)

Utility and Management Functions

This section provides descriptions for these utility and management functions. For complete usage information on each function, refer to the inline documentation for the function.

File Management	Schema Management	R Package Management
• ta.write.csv()	• ta.refreshSchema()	• ta.install.packages()
• ta.read.csv()	• ta.attachSchema()	• ta.installed.packages()
Table Management	• ta.detachSchema()	• ta.remove.packages()
• ta.dropTable()	Virtual Dataframe Management	R Files Management

• ta.create()	• ta.get()	• ta.install.files()
• ta.pull()	Temp Object Management	• ta.installed.files()
• ta.push()	• ta.findTempObject()	• ta.remove.files()
• ta.reify()	Connection Management	R Script Management
Object Management	• ta.connect()	• ta.install.scripts()
• ta.dbObjects.ls()	• ta.disconnect()	• ta.installed.scripts()
• ta.ls()	Utility Management	• ta.remove.scripts()
	• ta.is.R.Installed()	
	• ta.R.Version()	

File Management

ta.write.csv()

Use the `ta.write.csv()` function to export data from the Aster Database to a CSV file.

ta.read.csv()

Use the `ta.read.csv()` function to load data from a CSV file to the Aster Database.

Table Management

ta.dropTable()

Use the `ta.dropTable()` function to drop one or more tables from the Aster Database.

ta.create()

Use the `ta.create()` function to create tables in the Aster Database by using an R data frame with several optional settings. The parameters are converted into a regular SQL query that will create a table in the Aster Database.

ta.pull()

Use the `ta.pull()` function to export data from the Aster Database to R. This function returns a data frame containing a representation of the data in the file. Character strings in the result (including factor levels) have a declared encoding (default "UTF-8").

ta.push()

Use the `ta.push()` function to load data from R to the Aster Database.

ta.reify()

Use the `ta.reify()` function to take a virtual data frame as input and materialize it as an analytic table in temporary schema.

Object Management

ta.dbObjects.ls()

Use the `ta.dbObjects.ls()` function to list all table and views in a given schema.

ta.ls()

Use the `ta.ls()` function to list all the Aster R objects.

Schema Management

ta.refreshSchema()

Use the `ta.refreshSchema()` function to refresh the list of tables and views in the search path.

ta.attachSchema()

Use the `ta.attachSchema()` function to set the database objects in the search path.



If the number of tables and views under a schema is quite large, Teradata recommends using the `ta.data.frame()` function to get a virtual data frame. Otherwise, it can take a long time to obtain an object under the schema.

ta.detachSchema()

Use the `ta.detachSchema()` function to remove the objects from the search path.

Virtual Dataframe Management

ta.get()

Use the `ta.get()` function to search for a virtual data frame by name.

Temp Object Management

ta.findTempObject()

Use the `ta.findTempObject()` function to scan through an accessible schema to look for temporary objects created by the database user, and return a data frame that identifies the schema name, object name and object type.

Connection Management

ta.connect()

Use the `ta.connect()` function to establish a connection to the Aster Database. See also, [“Creating Connection to Aster Database” on page 52](#).

ta.disconnect()

Use the `ta.disconnect()` function to disconnect the RODB connection to the Aster Database.

Utility Management

ta.is.R.installed()

Use the `ta.is.R.installed()` function to check if R is installed in the Aster Database.

ta.R.Version()

Use the `ta.R.version()` function to obtain the version of R in the Aster Database.

R Package Management

ta.install.packages()

Use the `ta.install.packages()` function to install R library packages into R inside the Aster Database.

ta.installed.packages()

Use the `ta.installed.packages()` function to install files used in the SQL-MR platform into the Aster Database.

ta.remove.packages()

Use the `ta.remove.packages()` function to remove R library packages from R inside the Aster Database.

R Files Management

ta.install.files()

Use the `ta.install.files()` function to install files used in the SQL-MR platform into the Aster Database.

ta.installed.files()

Use the `ta.installed.files()` function to list the installed files used in the SQL-MR platform.

ta.remove.files()

Use the `ta.remove.files()` function to remove files used in the SQL-MR platform from the Aster Database.

R Script Management

ta.install.scripts()

Use the `ta.install.scripts()` function to install scripts used in STREAM Rexec into the Aster Database.

ta.installed.scripts()

Use the `ta.installed.scripts()` function to show the installed R packages inside the Aster Database.

ta.remove.scripts()

Use the `ta.remove.scripts()` function to remove scripts used in STREAM Rexec from the Aster Database.

Data Structure Management Functions

This section provides descriptions for these functions, which enable users to manage the structure of the data, along with the data structure conversions supported by the Aster R package. For usage information on each function, refer to the inline documentation for the function.

Create Aster R Object	Conversion to Aster R Objects	Aster R List Creation
• ta.data.frame()	• as.ta.data.frame()	• ta.split()
Conversion to R Objects	• as.vector()	• ta.list()
• as.data.frame()	• as.vector()	Check if Aster R Object
• as.matrix()		• is.ta.data.frame()
• as.vector()		• is.ta.factor()
		• is.ta.vector()

Supported Data Structure Conversions

This table shows the data structure conversions that are supported by the Aster R package.

Table 3 - 1: Supported Data Structure Conversions

From -> To	as.ta.data.frame	as.ta.vector	as.ta.factor	as.data.frame	as.vector	factor (as.vector(.))
data.frame	X	X (only with a single column)	X (only with a single column)			
factor	X	X	X			
vector	X	X	X			
ta.data.frame		X (only with a single column)	X (only with a single column)	X	X	X

Table 3 - 1: Supported Data Structure Conversions

From -> To	as.ta.data.frame	as.ta.vector	as.ta.factor	as.data.frame	as.vector	factor (as.vector(.))
ta.factor	X	X		X	X	X
ta.vector	X		X	X	X	X

Create Aster R Object

ta.data.frame()

Use the `ta.data.frame()` function to create a virtual data frame, which is based on an underlying select query on a table. Elements of the query are specified as input arguments to the function. Properties such as "order by" can be set directly as an object attribute.

Conversion to R Objects

as.data.frame()

Use the `as.data.frame()` function to return an R data frame.

as.matrix()

Use the `as.matrix()` function to return a matrix over the supplied virtual data frame.

as.vector()

Use the `as.vector()` function to return a vector over the supplied virtual data frame.

Conversion to Aster R Objects

as.ta.data.frame()

Use the `as.ta.data.frame()` function to convert an R object (for example, `data.frame`) or a virtual Aster object (for example, `ta.data.frame`) to a `ta.data.frame`.

as.ta.factor()

Use the `as.ta.factor()` function to convert an R object (for example, `data.frame`) or a virtual Aster object (for example, `ta.data.frame`) to a `ta.factor`.

as.ta.vector()

Use the `as.ta.vector()` function to convert an R object (for example, `data.frame`) or a virtual Aster object (for example, `ta.data.frame`) to a `ta.vector`.

Aster R List Creation

ta.split()

Use the `ta.split()` function to split virtual data frames into multiple virtual data frames using `colName` as the factor data type. This function will return a `ta.list` containing all virtual data frames.

ta.list()

Use the `ta.list()` function to return a `ta.list` containing Aster virtual objects.

Check if Aster R Object

is.ta.data.frame()

Use the `is.ta.data.frame()` function to return whether a given object is a virtual data frame.

is.ta.factor()

Use the `is.ta.factor()` function to return whether a given object is a virtual factor.

is.ta.vector()

Use the `is.ta.vector()` function to return whether a given object is a virtual vector.

Data Exploration Functions

This section provides descriptions for these functions, which enable users to explore the data. For usage information on each function, refer to the inline documentation for the function.

Print	Print Rows	Explore Structure
• print()	• ta.show()	• ta.dim()
• print.ta.data.frame()	• ta.head()	• ta.length()
• print.ta.data.frame.ordered()	• ta.tail()	• ta.names()
• print.ta.factor()		• ta.ncol()
• print.ta.list()		• ta.nrow()
• print.ta.vector()		• ta.colnames()
		• ta.dimnames()

Print

print()

The `print()` function is the default print function for Aster Database objects.

print.ta.data.frame()

The `print.ta.data.frame()` function is the default print function for a virtual data frame.

print.ta.data.frame.ordered()

The `print.ta.data.frame.ordered()` function is the default print function for an ordered virtual data frame.

print.ta.factor()

The `print.ta.factor()` function is the default print function for a virtual factor.

print.ta.list()

The `print.ta.list()` function is the default print function for a virtual list.

print.ta.vector()

The `print.ta.vector()` function is the default print function for a virtual vector.

Print Rows

ta.show()

Use the `ta.show()` function to display the data for a virtual frame.

ta.head()

Use the `ta.head()` function to return the top six rows of a given virtual data frame.

ta.tail()

Use the `ta.tail()` function to return the bottom rows of a given virtual data frame.

Explore Structure

ta.dim()

Use the `ta.dim()` function to return the `dim` attribute as a numeric vector for the given virtual data frame.

ta.length()

Use the `ta.length()` function to return the length of given Aster virtual objects. For `ta.vector` and `ta.factor`, the length returned is the number of rows. For `ta.data.frame`, the length returned is the number of columns. For `ta.list`, the length returned is the number of elements (for example, `ta.data.frame`).

ta.names()

Use the `ta.names()` function to return the names attribute as a character vector for the given virtual data frame.

ta.ncol()

Use the `ta.ncol()` function to return the number of columns present in a virtual data frame.

ta.nrow()

Use the `ta.nrow()` function to return the number of rows present in virtual data frame.

ta.colnames()

Use the `ta.colnames()` function to return the column names of the virtual data frame.

ta.dimnames()

Use the `ta.dimnames()` function to return a character vector having dimension names.

Data Transformation Functions

This section provides descriptions for these functions, which enable users to transform the data. For usage information on each function, refer to the inline documentation for the function.

Combine Data	Subsetting	Data Selection
• ta.merge()	• ta.subset()	• [
• ta.join()	Recoding	• [[
• ta.cbind()	• ta.recode()	• \$\$
• ta.rbind()	Transform Columns	
Sort and Order	• ta.transform()	
• ta.order()	• ta.with()	
	• ta.within()	

Combine Data

ta.merge()

Use the `ta.merge()` function to return a query based virtual data frame that is constructed using a join on two input virtual frames.

ta.join()

Use the `ta.join()` function to join two virtual Aster objects with join keys.

ta.cbind()

Use the `ta.cbind()` function to return a virtual frame joining the underlying table in the first two arguments. An error vector is returned in case of an error.



"int8" is not supported by R. "int8" is cast as "double precision".

ta.rbind()

Use the `ta.rbind()` function to return a union of two virtual data frames. An error vector is returned in case of an error.

Sort and Order

ta.order()

Use the `ta.order()` function to return a virtual data frame in the order specified in the call.

Subsetting

ta.subset()

Use the `ta.subset()` function to return a virtual data frame after applying subset criteria.

Recoding

ta.recode()

Use the `ta.recode()` function to recode a numeric vector or character vector-based virtual frame using the simple recode specifications supplied in the second argument. This function returns a virtual frame.

Transform Columns

ta.transform()

Use the `ta.transform()` function to update the underlying table with the value supplied and a new virtual frame or character vector of error messages.



- Using the `ta.transform()` function to transform the distribution key column is not supported.
- Simple SQL expressions using numbers, strings and a column name are supported. However, the results when using other types of expressions are not predictable.

ta.with()

Use the `ta.with()` function to evaluate the expression using the supplied virtual data frame, and return a value for the evaluated expression.

ta.within()

Use the `ta.within()` function to evaluate the expression using the supplied virtual data frame, and return a modified virtual data frame.

Data Selection

[

The `[` indexing operator is applied on a virtual data frame, virtual vector or virtual factor, and returns a virtual data frame, virtual vector or virtual factor. For example, `tadf["col1"]` returns a virtual vector of `col1`.

[[]

The `[[]` indexing operator is applied on a virtual data frame, virtual vector or virtual factor, and returns a virtual data frame, virtual vector or virtual factor. For example, `tadf["col1"]` returns a virtual vector of `col1`.

\$

The `$` operator operates on a virtual data frame, and returns a virtual vector having a single column. For example, `tadf$col1` returns a virtual vector of `col1`.

Basic Statistical Functions

This section provides descriptions for these basic statistical functions. For usage information on each function, refer to the inline documentation for the function.

Sample	Count	Min
• ta.sample()	• ta.approxDCount()	• ta.min()
Frequency Table	Five Number Summary	• ta.colMins()
• ta.table()	• ta.summary()	Percentile
• ta.hist()	Mean	• ta.approxPercentile()
Unique Values	• ta.mean()	• ta.percentile()
• ta.unique()	• ta.colMeans()	Membership
Standard Deviation	Sum	• ta.all()
• ta.sd()	• ta.sum()	• ta.any()
Correlation	• ta.colSums()	Averages
• ta.cor()	Range	• ta.emavg()
Covariance	• ta.range()	• ta.smavg()
• ta.cov()	Max	• ta.wmavg()

Variance	• ta.max()	• ta.vwap()
• ta.var()	• ta.colMaxs()	
• ta.colVars()		

Consideration

NaN Values are not Supported for Numeric Data Type

The NaN value is not supported for the numeric data type in Aster ODBC, and, as a result, will return in this error message:

```
Error in taQuery(queryStr) :
  HY000 40321 [AsterData][Support] (40321) String value 'NaN' resulted in
  an invalid numeric.
```

If you encounter this error message, you can use one of these two methods to resolve the issue:

- Add "as.double = TRUE" to the function call for the SQL based statistics functions (ta.max, ta.colMaxs, ta.min, ta.colMins, ta.sum, ta.colSums, ta.mean, ta.colMeans, ta.rowSums, ta.rowMeans, ta.colVars, ta.cov and ta.sd). For example:

```
> ta.sd(tadf, as.double = TRUE)
      id      val
8.225975    NaN
```

- Add "NumericAndDecimalAsDouble=1" under DSN in the odbc.ini file. For example:

```
[AsterDSN]
Driver=/home/beehive/work/multibranch/build/lib/libAsterDriver.so
SERVER=153.65.179.60
PORT=2406
DATABASE=beehive
UID=beehive
PWD=beehive
ByteaAsVarchar=1
NumericAndDecimalAsDouble=1
```

Sample

ta.sample()

Use the `ta.sample()` function to draw rows randomly from the input table. The function offers two sampling schemes:

- A simple Bernoulli (Binomial) sampling on a row-by-row basis with given sample rates.
- Sampling without replacement that selects a given number of rows.

The sampling can be applied to the entire relation, which is called unconditional sampling, or be refined with conditions, which is called conditional sampling.

Frequency Table

ta.table()

Use the `ta.table()` function to return a virtual data frame with a frequency count. The column used for computing the frequency must be of type `integer`. The R `table()` function returns a table, but the `ta.table` function returns a virtual data frame because the Aster implementation doesn't have an object of class "ta.table". This implementation is the same as when the `as.data.frame()` function is applied on a table object.

Histogram

ta.hist()

Use the `ta.hist()` function to map each input row to one or more bins based on criteria you specify and will return the row count for each bin.

Unique Values

ta.unique()

Use the `ta.unique()` function to return a unique virtual data frame after all duplicates are removed.

Standard Deviation

ta.sd()

Use the `ta.sd()` function to compute the sample or population standard deviation for each column of an Aster data frame.

Correlation

ta.cor()

Use the `ta.cor()` function to compute a global correlation between any pair of columns (COLUMNPAIRS) from a table.

Covariance

ta.cov()

Use the `ta.cov()` function to calculate the covariance of an Aster data frame.

Variance

ta.var()

Use the `ta.var()` function to compute the variance of an Aster data frame.

ta.colVars()

Use the `ta.colVars()` function to compute the sample or population variance for each column of an Aster data frame.

Count

ta.approxDCount()

Use the `ta.approxDCount()` function to quickly estimate the number of distinct values in a column or combination of columns, while scanning the table only once.

Five Number Summary

ta.summary()

Use the `ta.summary()` function to print a summary of numeric columns by listing out the min, max, median, first quartile, and third quartile.



The `ta.summary()` function does not support "bigint" columns due to a limitation of RODB that recognizes "bigint" and "bool" as an "unknown" data type. Only these data types are supported: "int", "smallint", "double", "real", "float", "numeric", "decimal".

Mean

ta.mean()

Use the `ta.mean()` function to compute the average (arithmetic mean) of all values in an Aster data frame.

ta.colMeans()

Use the `ta.colMeans()` function to compute the average (arithmetic mean) of all values for each column in an Aster data frame.

Sum

ta.sum()

Use the `ta.sum()` function to compute the sum of all values in an Aster data frame.

ta.colSums()

Use the `ta.colSums()` function to compute the sum of all values for each column in an Aster data frame.

Range

ta.range()

Use the `ta.range()` function to return a numeric vector for a virtual data frame. The first number in the result is the minimum value of the virtual data frame. The second number in the result is the maximum value of the virtual data frame. It returns The range of the virtual data frame is returned. This function only operates on numeric values.

Max

ta.max()

Use the `ta.max()` function to compute the maximum value in an Aster data frame.

ta.colMaxs()

Use the `ta.colMaxs()` function to compute the maximum value for each column in an Aster data frame.

Min

ta.min()

Use the `ta.min()` function to compute the minimum value in an Aster data frame.

ta.colMins()

Use the `ta.colMins()` function to compute the minimum value for each column in an Aster data frame.

Percentile

ta.approxPercentile()

Use the `ta.approxPercentile()` function to give e-approximate quantile summaries of a set of N elements, where e is the value you specify as the function's `ERROR` parameter. Given any rank r , an e-approximate summary returns a value whose rank r' is guaranteed to be within the interval $[r - eN, r + eN]$. The algorithm has a worst case space requirement of $O((1/e) * \log(eN))$.

ta.percentile()

Use the `ta.percentile()` function to generate the exact percentiles for a group of numbers.

Membership

ta.all()

Use the `ta.all()` function for an aster virtual data frame to return a logical value when applied to a virtual data frame. It returns TRUE when all the elements of virtual data frame are not zero. It only works for numeric values.

ta.any()

Use the `ta.any()` function for an aster virtual data frame to return a logical value when applied to a virtual data frame. It returns TRUE when one element of virtual data frame is not zero. It only works for numeric values.

Averages

ta.emavg()

Use the `ta.emavg()` function to compute the average over a number of points in a time series while applying a damping (weighting) factor to older values. The weighting for the older values decreases exponentially without entirely discarding the older values.

ta.smavg()

Use the `ta.smavg()` function to compute the average over a number of points in a series.

ta.wmavg()

Use the `ta.wmavg()` function to compute the average over a number of points in a time series while applying a damping (weighting) factor to older values. The weighting for the older values decreases arithmetically.

ta.vwap()

Use the `ta.vwap()` function to compute, for each in a series of equal-length intervals, the volume-weighted average price of a traded item.

R Map Reduce Runner Functions

This section provides descriptions for these wrapper functions. The wrapper functions enable users to translate an R function into an R script or an R expression and push in a block of R code inside the Aster Database for execution as opposed to the execution of a single R function. For usage information on each function, refer to the inline documentation for the function.

• ta.eval()	• ta.colApply()	• ta.by()
• ta.apply()	• ta.rowApply()	• ta.partitionApply()
• ta.aggregateApply()	• ta.tapply()	

ta.eval()

Use the `ta.eval()` function to calculate an R function without any input from an Aster data frame.

ta.apply()

Use the `ta.apply()` function to apply an R function on a given Aster data frame.

Considerations

- Character values with "\n" are not supported.
- The R apply function operates only on matrices and not data frames. If the input to the R apply function is a data frame, the function tries to convert the data frame to a matrix using the `as.matrix` function.

Because matrices cannot contain mixed data types, the `as.matrix` function converts numeric columns to strings and creates a matrix of strings. As a result, R functions must not be used on mixed type data frames.

You must project out the columns in the `tadf` argument of the `ta.apply` function so that all input columns are either numeric or character.

Aggregate-Oriented Apply Wrapper Function

ta.aggregateApply()

Use the `ta.aggregateApply()` function to apply a given R function on tables of an Aster data frame.

Column-Oriented Apply Wrapper Function

ta.colApply()

Use the `ta.colApply()` function to apply a given R function on columns of an Aster data frame.



The `ta.colApply()` function returns the same results as this function: `ta.apply(tadf, MARGIN = 2, FUN, COMBINER.FUN, ...)`.

Row-Oriented Apply Wrapper Function

ta.rowApply()

Use the `ta.rowApply()` function to apply a given R function on rows of an Aster data frame.



The `ta.rowApply()` function returns the same results as this function: `ta.apply(tadf, MARGIN = 1, FUN, ...)`.

Partition-Oriented Apply Wrapper Function

ta.tapply()

Use the `ta.tapply()` function to apply an R function on a partitioned Aster data frame.

Considerations

- Character values with "\n" are not supported.
- The R apply function operates only on matrices and not data frames. If the input to the R apply function is a data frame, the function tries to convert the data frame to a matrix using the `as.matrix` function.

Because matrices cannot contain mixed data types, the `as.matrix` function converts numeric columns to strings and creates a matrix of strings. As a result, R functions must not be used on mixed type data frames.

You must project out the columns in the `tadf` argument of the `ta.apply` function so that all input columns are either numeric or character.

ta.by()

The `ta.by()` function is a partition-oriented shortcut for the `ta.tapply()` function



The `ta.by()` function returns the same results as this function: `ta.tapply(tadf, INDEX, FUN, ...)`.

ta.partitionApply()

The `ta.partitionApply()` function is a partition-oriented shortcut for the `ta.tapply()` function



The `ta.partitionApply()` function returns the same results as this function: `ta.tapply(tadf, INDEX, FUN, ...)`.

Parallel Stats and Big Data Analytics Functions

This section provides descriptions for these functions, which enable users to analyze the data in an Aster Database. For usage information on each function, refer to the inline documentation for the function.

Statistics Analysis	Cluster Analysis	Naive Bays
• ta.confusionMatrix()	• ta.kmeans()	• ta.naiveBayes()
• ta.glm()	• ta.minhash()	• ta.naiveBayes.predict()
• ta.glm.predict()	Time Series, Path and Attribution Analysis	Text Analysis
• ta.knn()	• ta.attribution()	• ta.ldaTopicPrinter()

• ta.lars()	• ta.frequentPaths()	• ta.ldaInference()
• ta.lars.predict()	• ta.nPath()	• ta.lda()
• ta.pca()	• ta.sax()	• ta.sentiment.evaluate()
Association Analysis	• ta.sessionize()	• ta.sentiment.extract()
• ta.basket()	Decision Tree	• ta.sentiment()
• ta.cFilter()	• ta.forest()	• ta.textClassifier.predict()
• ta.recommender()	• ta.forest.evaluate()	• ta.textClassifier.evaluate()
	• ta.forest.predict()	• ta.textClassifier()

Statistics Analysis

ta.confusionMatrix()

Use the `ta.confusionMatrix()` function to generate a confusion matrix, which is a specific table layout that allows visualization of the performance of an algorithm.

ta.glm()

Use the `ta.glm()` function to perform linear regression analysis for any of a number of distribution functions using a user-specified distribution family and link function. The link function is chosen based upon the distribution family used and the assumed nonlinear distribution of expected outcomes.

Supported link models in the Aster Database are ordinary linear regression, logistic regression (logit model), and Poisson log-linear model.

ta.glm.predict()

Use the `ta.glm.predict()` function to perform generalized linear model prediction. This function enables you use the model generated by the `ta.glm` function to predict new input data.

ta.knn()

Use the `ta.knn()` function to classify data objects based on proximity to other data objects with known classification. The object with known classification or labels serve as training data.

ta.lars()

Use the `ta.lars()` function as a variant of linear regression that selects the most important variables, one by one, and fits the coefficients dynamically.

ta.lars.predict()

Use the `ta.lars.predict()` function to take in the new data and the model generated by the `lars` function, and output the predictions.

ta.pca()

Use the `ta.pca()` function to perform principal component analysis (PCA) on an Aster data frame. PCA is a common unsupervised learning technique, which is useful for both exploratory data analysis and dimensionality reduction.

Association Analysis

ta.basket()

Use the `ta.basket()` function to generate sets (baskets) of items by permuting rows in a `ta.data.frame`.

ta.cFilter()

Use the `ta.cFilter()` function to perform collaborative filtering via a series of SQL commands and SQL-MapReduce functions.

ta.recommender()

Use the `ta.recommender()` function to make recommendations (for example, items or products that users should consider purchasing) by using an item-based, collaborative filtering function that uses a weighted-sum algorithm.

Cluster Analysis

ta.kmeans()

Use the `ta.kmeans()` function to classify a given data set through a certain number of clusters (assume *k* clusters) fixed a priori. The main idea is to define *k* centroids, one for each cluster.

ta.minhash()

Use the `ta.minhash()` function as a probabilistic clustering method that assigns a pair of users to the same cluster with probability proportional to the overlap between the set of items that these users have bought (this relationship between users and items mimics various other transactional models).

Time Series, Path and Attribution Analysis

ta.attribution()

Use the `ta.attribution()` function to calculate attributions by using a wide range of distribution models

ta.frequentPaths()

Use the `ta.frequentPaths()` function for mining frequent subsequences (such as patterns) in a sequence database.

ta.nPath()

Use the `ta.nPath()` function to perform regular pattern matching over a sequence of rows from one or more inputs.

ta.sax()

Use the `ta.sax()` function to transform original time series data into symbolic strings. Once this transformation is complete, the data is more suitable for many additional types of manipulation, both because of its smaller size and the relative ease with which patterns can be identified and compared.

ta.sessionize()

Use the `ta.sessionize()` function to map each click in a clickstream to a unique session identifier.

Decision Tree

ta.forest()

Use the `ta.forest()` function to take a set of training data as input and use it to generate a predictive model, which can later be used as input to the `ta.forest.predict` function. The `ta.forest.predict` function uses the model to make predictions.

ta.forest.evaluate()

Use the `ta.forest.evaluate()` function to analyze the model generated by the `ta.forest` function and give weight to the variables used in the model.

ta.forest.predict()

Use the `ta.forest.predict()` function to generate predictions on a response variable for a test set of data by using the model generated by the `ta.forest` function.

Naive Bayes

ta.naiveBayes()

Use the `ta.naiveBayes()` function to train a naive bayes model using some input data.

ta.naiveBayes.predict()

Use the `ta.naiveBayes.predict()` function to predict the outcome for a test set of data by using the model generated by the `ta.naiveBayes` function.

Text Analysis

ta.LdaTopicPrinter()

Use the `ta.LdaTopicPrinter()` function to extract readable information from the model table generated by `ta.Lda`.

ta.LdaInference()

Use the `ta.LdaInference()` function to estimate the topic weight in new documents based on a pretrained model. You can then use the distribution, which is one of the features of the document, for other tasks such as classification and clustering.

ta.Lda()

Use the `ta.Lda()` function to estimate the correlation between the topics and words according to the topic number and other parameters. In addition, optionally, `ta.Lda` can generate the topic distributions for each document.

ta.sentiment.evaluate()

Use the `ta.sentiment.evaluate()` function to evaluate the precision and recall of the `ta.extract.sentiment` function after training a new model or uploading a new sentiment word dictionary.

ta.sentiment.extract()

Use the `ta.sentiment.extract()` function to extract the opinion or sentiment from input text. This function helps to extract the polarity of the content as positive, negative or neutral.

ta.sentiment()

Use the `ta.sentiment()` function to train a maximum entropy classifier for sentiment analysis. The training result is saved as a file and automatically installed on the Aster Database.

ta.textClassifier.predict()

Use the `ta.textClassifier.predict()` function to predict the category for input text by using the specified model.

ta.textClassifier.evaluate()

Use the `ta.textClassifier.evaluate()` function to evaluate the precision, recall and f-measure for a newly trained model.

ta.textClassifier()

Use the `ta.textClassifier()` function to train a machine learning classifier for text classification. The training result is saved as a file.

Aster R Package Limitations

This section provides information on limitations of the Aster R package.

Statistics Functions Behavior for bool, char, bit, and varbit Data Types

The behavior of the bool, char, bit, and varbit data types for statistics functions is:

- For character columns, the values can contain "", "NA", or an all digit string. For example, "123".
- For boolean columns, the Aster boolean is:
 - converted to 1 or 0 (numeric) through RODBC
 - converted to "t" or "f" through ta.pull
 - converted to "t" or "f" in the STREAM Rexec call
- For bit and varbit columns, the values only contain "1" and "0".

In each case, RODBC will recognize the data type value as a numeric (double or integer) values. As a result, statistics functions do not detect non-numeric columns before executing the SQL.

Large NUMERIC Values not Displayed From a Virtual Data Frame

A virtual data frame cannot display large NUMERIC values, and instead NA will be displayed. The maximum number of digits that can be displayed is 1.797693135e+308.

In-Database R Function Runners Support Maximum of 1599 Columns

The Aster Database can support tables up to a maximum of 1600 columns in a table. However when data is redistributed, a column is added as a partition key. As a result, the in-database R function runners (for example, ta.apply() and ta.tapply()) support a maximum of 1599 columns.

Aster R Results Differ from Native R for - and ! Operators

The ! and - operators do not work for non-numeric columns (for example, varchar type).

However, if the column contains numeric data, it will work for the ta.pull() function because the values in the varchar column are numeric and the underlying RODBC protocol changes the data types when imported to the R environment.

Aster R functions are not Supported in Expressions

The use of Aster R functions in expressions is not supported and will return an error. For example:

```
> tadf[int_col == local(var1) / ta.min(int_col), "int_col2"] <- 4
Error in .ta.describe.columns(query) :
  42000 34 [AsterData] [nCluster] (34) ERROR: syntax error at or near "(".
```



```
[TeradataAsterR] ERROR: Failed to call SQLPrepare 'select * from (SELECT
"row_names","bigint_col","int_col","smallint_col","char_col",CASE WHEN
"int_col" = 100.0 / TA.MIN("int_col") THEN (4.0)::int ELSE
"int_col2"::int END AS "int_col2","boolean_col" FROM
"replacement_expr_fact") as src_obj LIMIT 0'
Error in ta.data.frame(queryStr, sourceType = "query", schemaName =
attr(tadf, :
  Exiting since failed to create virtual data frame.
```

Aster R Functions With External C Calls are not Uploaded

Aster R input functions that contain external C calls (for example, runif), are not uploaded when using the `deparse` or the `serialize` method directly and will return an error. For example:

```
> ta.eval(runif, 1, FUN.upload = "deparse")
Error in taQuery(queryStr, stringsAsFactors = FALSE, stopOnError = TRUE)
:
  HY000 34 [AsterData][nCluster] (34) ERROR: SQL-MR function STREAM
failed: Stream process exited with non-zero exit value (1). Last few
lines of output were:
Error in tempFUN1422354862988274097443(c(1)) : object 'C_runif' not
found
Calls: asterWriteFun ... tempFUN1422354862988274097443 <- serialize <-
sstrtoi
Execution halted
.
> runif
function (n, min = 0, max = 1)
.External(C_runif, n, min, max)
<bytecode: 0x154e5a0>
<environment: namespace:stats>
```

A workaround is to use the `script` or the `name` method, or wrap the function into another function. For example:

```
ta.eval(function(x) runif(x), 1, FUN.upload = "deparse")
```

ta.write.csv() Encoding Limitation When Exporting to Windows

When exporting data to CSV files using the `ta.write.csv()` function, some multi-byte characters will be converted to octal format in Windows, for example latin characters in CHS locale are converted from "Â" to "<U+00C2>". This behavior is the similar to the behavior of the native R `write.csv()` function.

To work around this limitation when we using `ta.pull()` to export data in Windows with ANSI locale/encoding, add the `encoding` argument to specify which encoding will be output, otherwise, UTF-8 is used as the default, which will not be supported in the Windows CMD shell. For example:

```
# export Chinese characters in CHS locale
ta.pull(tadf, encoding="GB2312")
```

Changing Column Type of Virtual Data Frame in Aster R Limitation

Aster R does not support changing the column type in the same manner as it is done in native R. For example, this column type change is not supported in Aster R:

```
df$col = as.numeric(df$col)
```

To work around this issue, use one of these methods to change the column type in Aster R:

- Use the `ta.transform` function:

```
ta.transform(df, col = as.numeric(col))
```

- Use the `ta.reify` function:

```
res <- ta.reify(df, tableName = "new_tablename", colTypes = c(col =  
"numeric"))
```

- Use the `ta.create` function:

```
res <- ta.create(df, tableName = "new_tablename", colTypes = c(col =  
"numeric"))
```

ta.cbind Reordering Inputs Limitation

The `ta.cbind` function concatenates data sets, but also reorders the inputs. To work around this issue, `rownames` must be specified in order to return ordered output.

ta.subset Limitation When Using an R Object or an Aster R Object

The `ta.subset` function works when using a constant, but not when using an R object or an Aster R object. This example shows how to work around this limitation:

```
ta.dbx3 <- ta.dbx[ta.dbx$hdl > 0,]  
ta.dbx3 <- ta.dbx[ta.dbx$hdl > ta.dbx$ldl,]  
ta.dbx4 <- ta.dbx[ta.dbx$hdl == ta.dbx$ldl,]
```

ta.create Limitation for UTF-8 Characters

UTF-8 characters are not supported in the `ta.create` function. For example:

```
ta.create(finaldata, table = "twitter_raw1", schemaName = "public",  
tableType = "dimension", row.names=FALSE)
```

To work around this limitation, convert UTF8 characters to ASCII and then push the data into the data frame.

bytea Data Type not Supported for Aster R Functions

The Aster R data type `bytea` is not supported for Aster R functions. However, displaying `bytea` data that is retrieved from the Aster Database to R is supported.

Aster R Package Considerations

This section provides information on topics that impact the use of the Aster R package.

ta.head and ta.tail Functions can Generate Random Order of Results

The `ta.head ()` and `ta.tail ()` functions can generate a random order of the results from the Aster virtual data frame.

Using the DataDirect Driver Manager

In Linux, the unixODBC 2.3.1 and DataDirect 7.1 driver managers are supported the same in Teradata Aster R as they are in the Aster ODBC driver.

In R, however, RODB will always bind to one driver manager. If you want to switch to a driver manager other than unixODBC 2.3.1, you can perform this procedure to rebuild RODB with a specified driver manager:

```
ODBC_INCLUDE=<<<path to driver manager>>>/include \  
ODBC_LIBS=<<<path to driver manager>>>/lib \  
<<<path ot R>>>/bin/R  
> install.packages("RODBC")
```

SSL/SSO Authentication is not Supported

At this time, SSL/SSO authentication for secure communication is not supported

Saving Your Workspace is not Supported

When you exit the R console, you are prompted as to whether or not you want to save you workspace. At this time, saving your workspace upon exiting the R console is not supported.

Data Access from Native R Data Frames and Aster Virtual Data Frames

The `$` operator for a native R data frame and an Aster virtual data frame return a vector which allows you to use the `[]` operator to fetch elements from the vector.

Maximum and Minimum Value Ranges for Double Data Types

The maximum and minimum value ranges for Double data types are:

- -1.79769313486232e+308 is out of range for the data type double precision.
- 1.79769313486232e+308 is out of range for the data type double precision.

For example, The maximum value of double 1.7976931348623157E+308 contains 16 digits after the floating point, which after it is rounded to 14 digits, 1.79769313486232E+308, is out of range.

Timestamp and Timestamptz

Timestamp and Timestamptz are not precise data types in Aster and postgresQL.

Mapping Between integer64 in R and bigint in the Aster Database

R doesn't support 64-bit integers without the installation of an external library. Because of this limitation, 9223372036854775807 is stored as a numeric data type, which changes the original value to 9223372036854775808.

You can use one of these two methods to resolve the issue:

- Install the bit64 R package. Aster R supports the mapping between "integer64" in R and "bigint" in the Aster database.
- Store 9223372036854775807 as a character. For example:

```
> bi <- c("-9223372036854775808", "9223372036854775807", NULL,
"2147483647", "-2147483648")
> mydata<-data.frame(bi)
> ta.push("test_bigint", mydata)
```

"BC" in Date or Timestamp Type Value is not Recognized by RODBC

RODBC does not recognize "BC" in a date or timestamp type value. If the value of a date or timestamp type (or sub-type) contains "BC", "BC" is trimmed from the output.

Numeric Type Double Precision for Empty Table not Consistent with R

RODBC displays NA for blank values and tables containing no data. R on the other hand, will return a message. This issue is caused when data is returned from the Aster database to R, numeric columns in an empty table are recognized as "character" data types, which is a limitation of RODBC.

The NA value reported by RODBC is the result of the Aster database returning NULL for a query with a function against an empty table that RODBC then converts to NA.

Type Conversion Behavior for Numeric Values in Quotes in an Expression

For some operators in Aster (for example, "int_col > '1000'") the optimizer will add a type conversion automatically (for example, "int_col > cast('1000' as integer)"A).

This conversion is not available in R.

Because Aster does not detect the potential conversions, which could indicate the available conversions, the translated SQL is put into Aster, and an error message at run-time is returned if the conversion is not available.

Include methods Library to Print Virtual Data Frame to Console from R Script

To print a virtual data frame to the console from an R script executed using the Rscript command, ensure that the script includes the methods library. For example:

```
library(methods)
```

General SQL Wrapper Workaround When Executing a SQL Query

The workaround in order to use the general SQL wrapper, sqlQuery(), to execute a SQL query is shown in this example:

```
ac <- ta.connect('aster2100')
sqlQuery(ac, 'select count from r.tst;')
```

Data Type Mapping

This section lists the data type mapping between the Aster Database, ODBC, RODB, Teradata Aster R (ta.data.frame), R and STREAM (ta.apply and ta.tapply).

Table 3 - 2: Data Type Mapping

Aster	ODBC	RODBC	Teradata Aster R ta.data.frame	R	STREAM ta.apply ta.tapply	Notes
smallint	SQL_SMALLINT / 5	"smallint"	"smallint"	"integer"	"integer"	
						In R, -2147483648 encoding stores the integer as NA. <ul style="list-style-type: none"> -2147483648 is converted to NA (integer) through RODB.
integer	SQL_INTEGER / 4	"int"	"int"	"integer"	"integer"	<ul style="list-style-type: none"> -2147483648 is converted to -2147483648 (numeric) through ta.pull. -2147483648 is converted to -2147483648 (numeric) through STREAM Rexec.
bigint	SQL_BIGINT / -5	"unknown"	"bigint"	"numeric"	"numeric"	Converted to numeric through RODB.
real	SQL_REAL / 7	"real"	"real"	"double"	"double"	
double precision	SQL_FLOAT / 6	"float"	"float"	"double"	"double"	There are differences in precision between RODB, ta.pull and STREAM Rexec: <ul style="list-style-type: none"> Fifteen digits of precision in RODB and ta.pull. Sixteen digits of precision in STREAM Rexec.
	SQL_NUMERIC / 2					
numeric [p[, s]]	SQL_DOUBLE / 8 (NumericAndDecimalAs Double = 1)	"numeric" "double"	"numeric" "double"	"numeric" "double"	"numeric"	The "numeric" in R is not a precise data type.

Table 3 - 2: Data Type Mapping

Aster	ODBC	RODBC	Teradata Aster R ta.data.frame	R	STREAM ta.apply ta.tapply	Notes
boolean	SQL_BIT / -7	"unknown"	"boolean"	"integer"	"logical" (for AD 6.10 and later releases) "character"	<p>1The Aster boolean values are:</p> <ul style="list-style-type: none"> converted to 1 or 0 (numeric) through RODBC. converted to 1 or 0 in R. converted to "t" or "f" through ta.pull. converted to "t" or "f" in a STREAM Rexec call. <p>2RODBC recognizes a boolean value as a numeric data type in R (1 for TRUE, and 0 for FALSE).</p>
bit [(n)]	SQL_CHAR / 1	"char"	"char"	"numeric"	"character"	<p>1RODBC always converts a "bit" data type into a numeric data type in R. For example, "0010" is converted to 10 (ten).</p> <p>2When using values for comparison, Aster R package functions require the values to be in single quotes in order to produce valid results.</p>
bit varying [(n)]	SQL_VARCHAR / 12	"varchar"	"varchar"	"numeric"	"character"	<p>1RODBC always converts a "bit" data type into a numeric data type in R. For example, "0010" is converted to 10 (ten).</p> <p>2When using values for comparison, Aster R package functions require the values to be in single quotes in order to produce valid results.</p>
bytea	SQL_VARBINARY / -3	"varchar"	"varchar"	"character"	"character"	<p>When ByteaAsVarchar=1 in the odbc.ini file, values will all digits, "NaN", "NA", and so on are recognized as numeric values in R by RODBC.</p>

Table 3 - 2: Data Type Mapping

Aster	ODBC	RODBC	Teradata Aster R ta.data.frame	R	STREAM ta.apply ta.tapply	Notes
character [(n)]	SQL_CHAR / 1	"char"	"char"	"character"	"character"	1RODBC will truncate character values by 65535 bytes. 2When ByteAsVarchar=1 in the odbc.ini file, values will all digits, "NaN", "NA", and so on are recognized as numeric values in R by RODBC.
character varying [(n)]	SQL_VARCHAR / 12	"varchar"	"varchar"	"character"	"character"	1RODBC will truncate character values by 65535 bytes. 2When ByteAsVarchar=1 in the odbc.ini file, values will all digits, "NaN", "NA", and so on are recognized as numeric values in R by RODBC.
date	SQL_TYPE_DATE / 91	"date"	"date"	"Date"	"Date"	1The range of the "Date" data type in R is 0001-01-1 through 9999-12-31. 2Out-of-range values can be converted to a "character" data type by ta.pull().
time [(p)] without time zone	SQL_TYPE_TIME / 92	"time"	"time"	"character"	"character"	
time [(p)] with time zone	SQL_VARCHAR / 12 (9)	"varchar"	"varchar"	"character"	"character"	

Table 3 - 2: Data Type Mapping

Aster	ODBC	RODBC	Teradata Aster R ta.data.frame	R	STREAM ta.apply ta.tapply	Notes
timestamp [(p)] without time zone	SQL_TYPE_TIMESTAMP / 93	"timestamp"	"timestamp"	c("POSIXct", "POSIXt")	c("POSIXct", "POSIXt")	1The range of the "POSIXct" data type in R is 0001-01-1 through 9999-12-31. 2Out-of-range values can be converted to the "character" data type by ta.pull().
timestamp [(p)] with time zone	SQL_VARCHAR / 12 (9)	"varchar"	"varchar"	"character"	"character"	
interval	SQL_VARCHAR / 12	"varchar"	"varchar"	"character"	"character"	
serial { local global }	SQL_INTEGER / 4	"int"	"int"	"integer"	"integer"	
bigserial { local global }	SQL_BIGINT / -5	"unknown"	"bigint"	"numeric"	"numeric"	
ip4	SQL_VARCHAR / 12	"varchar"	"varchar"	"character"	"character"	
ip4range	SQL_VARCHAR / 12	"varchar"	"varchar"	"character"	"character"	
uuid	SQL_VARCHAR / 12	"varchar"	"varchar"	"character"	"character"	

CHAPTER 4 **Invoke R Using SQL-MR Stream Module**

This section provides information about an alternative method that you can use to invoke an R program by using the SQL-MR Stream module, and includes these topics:

- [Execution Model For R](#)
- [Writing an R Program to Run Inside the Aster Database](#)
- [Data Type Mapping Between R and Aster Database](#)
- [Troubleshooting](#)

Execution Model For R

R is installed in the sandbox area on each cluster node at `/opt/aster/third-party/R`, which requires R to be executed using the `chroot` command:

```
$ chroot /opt/aster/third-party/R <R/Rscript>
```

For simplicity, a wrapper script (Rexec) is provided and is located in `/home/beehive/bin/`
`utils/exec/Rexec`, which allows R execution either from Act or as a root user executing Rscript files in `/home/extensibility`.

This approach has numerous advantages:

- It makes it relatively easy to synchronize R across the Aster Database cluster or deploy R on a newly added cluster node.
- It minimizes security risks from R programs because the sandbox limits the portion of the file system the R program can see to the auxiliary root of the sandbox. In other words, the R program does not have access to the entire file system.
- It eliminates the possibility of conflicts between the dependent libraries of R and the other libraries installed on the cluster node because all the dependent packages installed by R and R packages are isolated from the rest of the software installed on the node.

R Program Invocation

During the invocation of a query that runs a user-installed R program using the Stream SQL-MR function, the Stream function calls a runner function called Rexec, which runs as the

extensibility system user, to set up the R environment and invoke the user-installed R program.

As a part of this setup, the extensibility user needs to be added to the sandbox area. Also, the SQL-MR working directory is mounted on the auxiliary root to allow Rscripts to read any ancillary installed files they need to access. Then, Rexec changes to the directory in the auxiliary root area containing the user-installed R program, as well as the ancillary installed files, and executes the R program.

To invoke a query that runs a user-defined Rscript using the Stream SQL-MR, use this command:

```
Rexec [<options>] [-e <expression>] <Rscript_file>
```

This command assumes that the specified Rscript file, the included model files, and other included Rscript files are already installed using the `act \install` command.

For example:

```
Rexec --vanilla naivebayes.R
```

Writing an R Program to Run Inside the Aster Database

The main supported data structure to exchange data between the Aster database and the R environment is the R data frame, which has a similar data representation to a table in the Aster Database.

A typical R program may use `read.table` to read in input data and use `write.table` to write a result out, although you can use any other mechanism as long as a delimited list of values and rows are read and written by the program.

This example shows a simple R program, called `echo_input.R`, that receives a row from an Aster Database vWorker and writes out the same row back.

```
IN = file(description="stdin",open="r")
while(1)
{
  # Create a frame to hold the input rows, without HEADER,
  # and also to deal with end of stream
  # read.table() is called inside the try block to detect when the
  # program reaches the end of rows.

  frame<-try(read.table(IN,header=FALSE,sep="\t",quote="",nrows=1,
    comment.char=""),silent=TRUE)
  if(inherits(frame,"try-error"))
    break
  write.table(frame,stdout(),col.names=FALSE,row.names=FALSE,quote=FALSE,sep="\t")
}
```

The Aster Database vWorker and the R program exchange rows using standard input and output. Hence, the R program should be written to read in a row from the standard input and

write a row to the standard input. The program does not need to generate an output for each row read. For example, it can generate one row after reading the entire set of input rows.

You can invoke R programs using the SQL-MR Stream module. For example, you can invoke `echo_input.R` using the following SQL query:

```
SELECT *
FROM stream(
    ON mytable
    SCRIPT('Rexec --vanilla echo_input.R')
    OUTPUTS(' * '))
);
```

The R program can also take any input parameter during its invocation, just like the way parameters are passed to RScript invocations, and parse the parameters using an appropriate package like `getopt`.

Also, if the R program needs to operate on a subset of columns of the input table, the `ON` clause can be used to write a subquery that projects out the required columns. Also, if the R program returns rows with a different schema from the input rows or adds more columns to the input rows, then you can use the `OUTPUTS` clause to specify these columns.

For more information about the Stream module, refer to “Stream API for Python, Perl, and Other Scripts” in the Aster Developer Guide.

Note that returning rows from an R program involves two distinct steps:

- The first step is to specify the names and types of rows returned by the R program in the `OUTPUTS` clause. The column names used should comply with Aster database’s identifier naming conventions.
- The second step is to write the R program to return rows without any headers. Also note that the correct delimiters need to be used.

If you need to transfer lists, matrixes, or other R data structures from R to the Aster Database, you have to convert them to data frames using the `as.data.frame()` function. Also, the R program must have only one result that is in the form of a data frame.



You can run an R program that does not read in any input data but produces an output table. In this case, you can either use some dummy table in the `ON` clause or just use a subquery like “`SELECT 1`” in the `ON` clause and ignore the input in the R program.

Writing an Output File from an R Program to the File System

It is possible to have your R program consume an input table and write the output or some other data to the file system as long as R is running as a user that has permission to write in the target directory. If you do not specify any directory, your file is written in the default working directory, which gets cleaned up at the end of the query’s execution. Hence, you need to specify some other directory under the auxiliary root (for example, `/opt/aster/third-party`) to which you have a write permission.

When writing files out, you need to keep in mind that multiple instances of your R program are running on each node of the cluster. Hence, make sure that the name of your output file from each instance contains some unique identifier.

This example shows a simple R program, called `echo_input2.R`, that receives a row from an Aster Database vWorker and writes out the same row back to a file.

```
IN = file(description="stdin",open="r")
while(1)
{
  # Create a frame to hold the input rows, without HEADER,
  # and also to deal with end of stream
  # read.table() is called inside the try block to detect when the
  # program reaches the end of rows.

  frame<-try(read.table(IN,header=FALSE,sep="\t",quote="",nrows=1),silent=TRUE)
  if(inherits(frame,"try-error"))
    break
  outFile=file('/tmp/output.txt','w')
  write.table(frame, outFile, row.names = FALSE,append = FALSE,
              col.names = FALSE, sep = "\t")
}
```

Using the PARTITION BY Clause in Queries that Invoke R Programs

The PARTITION BY clause ensures that the rows that contain the same value for the partitioning (splitting) expression will occur on the same vWorker for processing. Because they occur on the same vWorker, they will be processed by a single R program task.

The R program is invoked once per vWorker, not once per partition. The PARTITION BY ensures that each partition is processed by one vWorker. But when multiple partitions are processed by a single vWorker (e.g. when there are three or more partitions in a system with two vWorkers), then more than one partition may end up on a single vWorker. In such a case, the script detects the partition boundary (i.e. passing the partitioning column as a script argument) and ensures that partition-wise results are returned for each partition. Hence, the semantics of PARTITION BY when used with R are not exactly the same as that of a Reduce.

So essentially, you can be assured that all rows of a partition will be processed by a single vWorker task, but that does not mean that the vWorker will be processing just one partition.

You may choose to use the statistics returned from each vWorker as your end result, or you may choose to aggregate the vWorker results into a combined result using a client-side R program. In the latter case, make sure that your computation fits the split-apply-combine (data parallel) paradigm.

The following example R program shows the use of a PARTITION BY clause. This program, called `partition_sum.R`, computes the sum of all the values in each row and then adds that sum to a partition-wise sum:

```
IN = file(description="stdin",open="r")
partition_sum=as.integer(0)
while(1)
{
  row_frame<-try(read.table(IN,header=FALSE,sep="\t",quote="",nrows=1),silent=TRUE)
  if(inherits(row_frame,"try-error"))
```

```

        break
    row_sum<- apply(row_frame[,3:ncol(row_frame)],1,sum)
    partition_sum <- row_sum + partition_sum
    last_row<-row_frame
}
write.table(t(c(last_row[,2],partition_sum)),stdout(),col.names=FALSE,row.names=FALSE,qu
ote=FALSE,sep="\t")

```

Notice that the write-table function is called after all input rows are consumed.

The following is the SQL-MR query for running the above program in the Aster Database:

```

SELECT *
FROM stream (
    ON (select a, b, c, d from inputtable)
    PARTITION BY b
    SCRIPT ('Rexec --vanilla partition_sum.R')
    OUTPUTS ('b int', 'sum int')
);

```

This query produces a result for each vWorker. Aster Database redistributes the rows in the input table based on their values for the column “b” in such a way that all rows that share the same value are processed on one vWorker, by one instance of the R program.



Tip: The Aster Database provides a special partitioning construct that can be used to execute just one instance of the R program thereby processing the entire input table by one R program instance. To use this construct, simply use “PARTITION BY 1”. This construct enables you to push any R program that cannot be parallelized (for example, model development) into the Aster Database and avoid having to extract the data out and instead exploit the larger computing power of the Aster Database.

Data Type Mapping Between R and Aster Database

The exchange of R data and Aster Database data is done by mapping an R data frame to an Aster Database table. The R program needs to be able to accept and map the input data to the right types, and the output from the R program should match what is specified in the OUTPUTS clause of the Stream function (except in the case of a “*” expression in the OUTPUTS clause, which expects data with the same types as the input data).

[Table 4 - 3](#) shows the recommended mapping of data types from R to Aster Database.

Table 4 - 3: Data Type Mapping

R Type	SQL Type
integer	int2, int4
numeric (double)	int8, float4, float8, float(p), numeric [(p,s)]
boolean	logical
bytea	object
everything else	character

However, there are a few mapping issues that you need to be aware of, which are described in the following topics.

Character

Character types can be used to exchange a wide variety of data between Aster Database and R, but note that the exchange of characters can be very time-consuming. So, it is recommended that you consider whether it is necessary for a given character column to be transferred to the R environment or whether it can be substituted with the integer or double data type.

Bit Data Type

The R environment cannot automatically determine the type of bit data and properly handle it. Hence, the Rscript needs to explicitly accept the bit string as a character string and convert it to a raw data type. For example, when using the `read.table()` function, you can achieve this conversion as follows:

```
frame<-try(read.table(IN,header=FALSE,sep="\t",quote="",  
nrows=1,colClasses=c("character")), silent=TRUE)
```



The Bit Varying data type is properly handled by the `read.table()` function.

Bytea Data Type

Bytea data values are handled as R raw types by the R environment. However, note that the `read.table()` function uses the number symbol (“#”) as a comment character and hence data after “#” is truncated. One way to avoid the truncation when reading bytea values using `read.table()` is to turn off the interpretation of comment characters as shown in the this example:

```
frame<-try(read.table(IN,header=FALSE,sep="\t",  
quote="",nrows=1,comment.char=""), silent=TRUE)
```

BIGINT Data Type

The base R environment uses finite precision arithmetic and numbers are accurately represented only up to 15 or 16 decimal places. Hence, you need to make sure you take the proper care when working with bigint values that are larger than the values that can be accurately represented. To transfer such numbers between Aster Database and the R environment, you can use a character representation as shown in this example:

```
frame<-try(read.table(IN,header=FALSE,sep="\t",quote="",  
nrows=1,colClasses=c("character")), silent=TRUE)
```

You can then use packages like `int64` and `gmp` to convert the numbers back to 64-bit integers in your R program.

For example, using the `int64` package (<http://cran.r-project.org/web/packages/int64/>), you can convert a bigint represented as a character back to a 64-bit integer value as follows:

```
y<-as.int64(c("-3940427841425010000", "-4236711481380030000"))
```

Also, here is an example that shows how you can perform arithmetic with a BIGINT value:

```
# head -20 *
==> bigints.csv <==
1,4699533205482374612
2,-3526377826377322350
3,1
4,-1
5,0

==> bigints.R <==
#!/usr/bin/Rscript
library(int64)
IN = file(description="stdin",open="r")
while(1)
{
  # Create a frame to hold input Rows ,
  # without HEADER and also to deal with end of stream
  # read.table() should be inside try block
  frame<-try(read.table(IN,header=FALSE,sep="\t",quote="",
                        nrows=1,colClasses=c("character")),silent=TRUE)
  if(inherits(frame,"try-error"))
    break

  # instantiate a int64 object based on BIGINT in table and add 1 to it
  value<-as.int64(frame$V1[1])
  value = value + 1

  # create a new frame with the original BIGINT value
  frame2<-cbind(frame, value)
  write.table(frame2,stdout(),col.names=FALSE,
              row.names=FALSE,quote=FALSE,sep="\t")
}

==> bigints.sql <==
DROP TABLE IF EXISTS bigints;
CREATE TABLE bigints(seq INT, num BIGINT) DISTRIBUTE BY HASH(seq);

# ncluster_loader -c -w beehive -B bigints.sql bigints bigints.csv
Trying to connect to the loader '192.80.170.44'.
Loading tuples using node '192.80.170.44'.
5 tuples were successfully loaded into table 'bigints'.

# act -w beehive
Welcome to act 05.10.00.00, the Aster nCluster Terminal.

Type: \copyright for distribution terms
      \h for help with SQL commands
      \? for help with act commands
      \g or terminate with semicolon to execute query
      \q to quit

beehive=> \install bigints.R
beehive=> SELECT * FROM STREAM( ON (SELECT num FROM bigints)
SCRIPT('Rexec --vanilla bigints.R') OUTPUTS ('*', 'sum BIGINT'));
      num | sum
-----+-----
          1 |
4699533205482374612 | 4699533205482374613
```


	-1		0
-3526377826377322350			-3526377826377322349
0			1
(5 rows)			

Null (Missing) Value Handling

Null values in the Aster Database are typically mapped to an “NA” value in the R environment. However, the Aster Database environment passes null values as either the value “NULL” or as empty strings.

To instruct the Aster Database to replace null values with “NA” before passing them to the R environment, you can use the NULLSTRING argument of the Stream SQL-MR function, which allows you to use any value as a replacement for null values before they are passed to the R environment. Likewise, the Aster Database recognizes values given in the NULLSTRING argument as NULL values in data returned from the R environment.

For more information about the handling of null values, refer to “Stream Function Usage” in the Aster Client Guide.

Hash(#) Character Handling

The read.table() function uses the number symbol (“#”) as a comment character and hence data after “#” is truncated. One way to avoid truncation when using read.table(), which applies to all types, is to turn off the interpretation of comment characters as shown in the this example:

```
frame<-try(read.table(IN,header=FALSE,sep="\t",
quote=" ",nrows=1,comment.char=""),silent=TRUE)
```

Troubleshooting

Possible reasons for the failure in Rscript execution using the Stream are:

- Improper R installation on the Aster Database cluster.
- Improper installation of R packages, required by a user-defined Rscript, on an Aster Database cluster.
- Temporary command execution failure due to node failure.
- Stream execution failure.

To detect any of these possible failures, execute R from the command line on the queen:

- `Rexec R <command_line_argument>`
This command invokes the R shell.
- `Rexec Rscript [<options>] [-e <expression>] <Rscript_file arguments>`
This command invokes the specified Rscript file, assuming that the Rscript file and ancillary installed files are located in /home/extensibility area or any sub-directory.
If Rscript execution is not successful on the queen, the R installation files in queen are corrupted and R needs to be reinstalled on the cluster.

- Check for R installation on all the nodes using this ncli option:

```
ncli apm install R --localconfig
```

- If R is not installed on a worker, execute this command to ensure seamless R installation on all the reachable worker nodes of the cluster:

```
ncli apm administer R --synchronize
```

- Check for installation status of all the R packages by running this command:

```
ncli apm show R -packages=<package name>
```

Then, check for the existence of the directories in the output of this command on the worker:

```
ncli node runonanother ls /opt/aster/third-party/R/usr/lib64/R/library
```

If some packages are missing, run this command on the queen:

```
ncli apm administer R --synchronize
```

- If the stream execution still throws an error, this means that the problem could be in Stream execution and needs to be investigated.

A

Aster R Package

- considerations 82
- data type mapping 85
- function reference 58
- inline documentation 51
- installation 48
- limitations 80
- overview 14, 50
- quick start reference 51
- supported data structure conversions 62
- vignette documentation 51

Aster Virtual Data Frame 14

D

Data Type Mapping

- between Aster, ODBC, RODB, TeradataAsterR, R and STREAM 85
- between R and Aster Database 94

R

R

- inline documentation 51
- installation 16, 44
- installation process overview 16
- invoking R using SQL-MR Stream module 90
- package installation 44
- programming language overview 10
- supported functionality in Aster Database 13
- supported versions 15
- unsupported functionality in Aster Database 14

RConsole

- support for 15

RStudio

- support for 15

T

Teradata Aster R

- Aster R package 11, 14
- open source R on Aster cluster 13
- product overview 10
- R engine 11

Terminology 12

