HITACHI
Inspire the Next

Pentaho Data Integration (DI) Server in High Availability (HA) End to End

HITACHI
Inspire the Next

Change log (if you want to use it):

Date	Version	Author	Changes

Contents

This page intentionally left blank.

Overview

We have collected a set of best practice recommendations and information for you to use when you want to set up your Pentaho Data Integration (DI) servers with a clustered high availability (HA) solution. Our intended audience is Pentaho and database administrators, or anyone with a background in data source configuration who is interested in setting up data integration in a high availability environment.

Software	Version(s)
Pentaho	7.x, 8.x

The Components Reference in Pentaho Documentation has a complete list of supported software and hardware.

Before You Begin

Before beginning, use the following information to prepare for the procedures described in the main section of the document.

Terms You Should Know

Here are some terms you should be familiar with:

- **Cluster**: A collection of [application] servers (for example, Pentaho Servers or Carte Servers) that communicate with each other and that share and update a common data repository (usually a database) to make a set of services highly available to clients.
- **Quartz Scheduler**: The scheduling engine used by Pentaho for managing and executing schedules for jobs and reports used on data integration (DI) and business analytics (BA) applications.
- **Jackrabbit**: Pentaho's repository that holds all user, server, and content information.
- **Load Balancing**: The distribution of workloads across multiple computing resources.
- **HTTP traffic***:* A text-based client that can send requests and responses.

Table 1: Reasons to Use Load Balancing

Feature	Reason
High Availability (HA)	Client requests and application loads are distributed across many servers in the same data center, or many data centers, in either an active-active or active-passive mode. Requests and application loads automatically fail over, when using routing policies, in case of failure of primary servers.
Scalability	Load balancing can spread Pentaho application footprints across many servers that are either co-located or in different data centers. This serves growing application loads and reduces the CPU, memory, and I/O strains on a single server.

© Hitachi Vantara Corporation 2019. All Rights Reserved

Feature	Reason
Application Latency	Load balancing can solve application latency by routing client requests or traffic to servers that are geographically close to application clients, effectively solving old long network round trips.
Geolocation	Load balancing can provide a convenient way to achieve geolocation in circumstances where organizations require that client requests be restricted to data stored in certain geographic radii, based on point of origin.
Application Maintenance	Load balancing prevents the need for single server deployment of Pentaho or any other application to occasionally go offline for planned or emergency maintenance.
Disaster Recovery	Load balancing can be used to route traffic from a primary to a backup set of servers to prepare business continuity plans for medium to large customers running Pentaho, in the event of an emergency.

Other Prerequisites

In this guide, we will be using Apache HTTPD. All configurations will be shown in an Apache web server context.

1. This guide assumes that a load balancer is already installed and running.
2. You will need to make sure that each Pentaho server in your cluster is already configured to use the repository database of your choice. Make sure each server is pointing to the same database instance.
3. Initialize your database using the appropriate steps for your system. Pentaho documentation has instructions for PostgreSQL, MySQL, MS SQL Server, and Oracle databases.
4. After you have initialized and configured your repository, clean up temporary files by locating the `../pentaho-server/tomcat` directory and removing all files and folders from the `temp` and `work` folders.
5. Make sure that each server in your cluster is already configured to use the repository database of your choice, and that each server is pointing to the same database instance.

Use Case: Preparing a Server for High Traffic Flow

Fabiola is a database administrator who wants to set up her Pentaho DI server with a clustered HA solution to address increasing data processing and concurrent user connections. Pairing her server with this solution will place a load balancer in front of a cluster, sending traffic to only one DI server. She must first make sure that her servers are configured to use the repository database, and make sure that they are aimed at the same database instance.

© Hitachi Vantara Corporation 2019. All Rights Reserved

Pentaho DI Server High Availability

Most installations of Pentaho are single-server installations. This solution works well in small-sized and medium-sized organizations where users and developers are limited to a handful of people. However, in large scale deployments, you need a clustered high availability solution to address the increase in data processing and concurrent user connections.

In the Pentaho HA setup, the load balancer sits in front of a cluster and only one DI server gets traffic. The secondary server only gets traffic if the primary server goes down.

This model allows the servers to deal with failover: if one Pentaho server goes down, service is not interrupted but is instead taken over by a live server. In addition, distributed processing can be added using Carte.

A single point of failure with this model would be with the load balancer; therefore, further consideration to cluster the load balancer would need to be taken. In this best practice guide, we will only be covering the configuration of a single load balancer, so consider all the factors in your situation to determine if you should use only one load balancer.

You can find details on these topics in the following sections:

Pointing Each Server to the Same Database Instance

Initialize your database using the steps in the appropriate article for your system. Set Up a Cluster has sections for PostgreSQL, MySQL, MS SQL Server, and Oracle databases. After you have initialized and configured your repository, you should clean up these files by following these steps:

- Locate `...pentaho-server/tomcat` and remove all files and folders from the `temp` and `work` folders.
- Locate `...pentaho-server/pentaho-solutions/system/jackrabbit/repository` and remove all files and folders from the `workspaces` and `final repository` folders.

You now have a configured repository and are ready to move to the next step for clustering.

© Hitachi Vantara Corporation 2019. All Rights Reserved

Clustering the DI Server Nodes

Next, you need to configure the cluster nodes to use the same backend repository. The following instructions will walk you through this process.

Figure 1: Configure and Test Cluster Nodes

Configuring Jackrabbit Journal

Pentaho uses Apache Jackrabbit as its content repository. You will need to configure the Jackrabbit Journal for clustering by following the official instructions found in Pentaho Documentation: Configure Jackrabbit Journal.

Configuring Quartz

Pentaho uses Quartz for scheduling. As with the Jackrabbit Journal, Quartz also needs to be configured for a cluster. The official instructions can be found in Pentaho Documentation: Configure Quartz.

*Make sure to set all **Scheduled Jobs** on your passive node(s) to **PAUSE**. This will direct all Scheduled Jobs to your **active node** only and prevent round robin scheduling. Pentaho's Developer Center has information on how to Pause, Stop, or Start the scheduler.*

More information is available at the Quartz Configuration Reference site.

Starting and Testing the Cluster

Follow these instructions to start the cluster and verify that it is working properly.

1. Start the solution database.
2. Start the application server on each node.
3. Make sure that the load balancer can ping each node.
4. Access the login screen for each server in your cluster.
5. Browse directly to each node instead of going through the load balancer.

Web Application Load Balancing

Depending on where Pentaho is installed (on premises or in the cloud) and what version of Pentaho you have installed, you can configure load balancing of the web application tier (Pentaho Server or Business Analytics (BA)/DI Servers) using your choice of Apache web server, enterprise, or cloud load balancers.

© Hitachi Vantara Corporation 2019. All Rights Reserved

Apache: Configuring and Testing Load Balancer for HA

Now that each server in the Pentaho cluster can be accessed individually, it is time to configure the load balancer to manage traffic to each node.

The first section will cover configuring Apache as a load balancer with the DI server cluster nodes. We provide instructions for configuring Apache for both Windows and Linux.

Configuring Tomcat

To configure Tomcat:

1. Shut down the Tomcat server on each node in the cluster.
2. Open the `tomcat/conf/server.xml` file in a text editor.
3. Locate the following line: `<Engine name="Catalina" defaultHost="localhost">`.
4. Add the `jvmRoute` attribute like this:

```
<Engine name="Catalina" defaultHost="localhost" jvmRoute="server1">
```

The `jvmRoute` value will need to be unique for each node. This value will map to the `BalancerMember` setup in the following section when configuring Apache.

5. To close connections so they do not become orphaned and cause errors, locate the following line:

```
<Connector URIEncoding="UTF-8" port="8009" protocol="AJP/1.3"
redirectPort="8443" />
```

Change it to:

```
<Connector port="8009" protocol="AJP/1.3" redirectPort="8443"
           connectionTimeout="10000" keepAliveTimeout="10000" />
```

6. Edit the `tomcat/webapps/pentaho/WEB-INF/web.xml` and change the `fully-qualified-server-url param-value` to the load balancer's URL.
7. Start the Tomcat service after configuring this file.
8. Repeat this process throughout all nodes within the cluster.

Configuring Apache (Windows Environment)

To configure Apache for Windows:

1. Open the `C:\Program Files (x86)\Apache24\conf\httpd.conf` file in a text editor.
2. The following modules will need to be uncommented if they aren't already:
   - `LoadModule proxy_module modules/mod_proxy.so`
   - `LoadModule proxy_ajp_module modules/mod_proxy_ajp.so`
   - `LoadModule proxy_http_module modules/mod_proxy_http.so`
   - `LoadModule proxy_balancer_module modules/mod_proxy_balancer.so`

© Hitachi Vantara Corporation 2019. All Rights Reserved

3. At the bottom of the file, add the following code, making sure to reflect the `BalancerMember` hostname with your Pentaho cluster nodes:

```
ProxyPass /pentaho balancer://dicluster

<Proxy balancer://dicluster>

BalancerMember http://hostname1:8080/pentaho retry=30 BalancerMember
http://hostname2:8080/pentaho status=+H retry=0

</Proxy>
```

*In the above code snippet, notice that **sticky sessions** are not used or needed. That is because this configuration is used for HA failover only, not for true load balancing.*

4. After adding the above code, save the file and restart the Apache service.
5. You should now be able to browse to your load balancer and see the Pentaho User Console, for example:

```
http://loadbalancerhostname/pentahodi
```

Configuring Apache (Linux Environment)

To configure Apache for Linux:

1. Load the following modules:
    - `proxy_ajp`
    - `proxy_balancer`
    - `proxy_http`
    - `lbmethod_byrequests`

   You will need to use the proper OS specific commands to load these modules. For example, with Debian-based distributions, run the command to load the required modules:

```
sudo a2enmod proxy proxy_ajp proxy_balancer proxy_http lbmethod_byrequests
```

For RPM-based distributions, you will need to enable these modules in the `httpd.conf` file by making sure they are uncommented, similar to the Windows configuration above. These may already be uncommented by default.

2. Locate and edit the Apache configuration file.
    - For Debian-based distributions, this is located by default in `/etc/apache2/sites- available/000-default.conf`.
    - For RPM-based distributions, this is located by default in `/etc/httpd/conf/httpd.conf`.
3. Within this configuration file, you will need to edit the existing (or add a new) `VirtualHost` directive. It will need to be set up like this.

© Hitachi Vantara Corporation 2019. All Rights Reserved

4. Make sure to reflect the `BalancerMember` hostname with your Pentaho cluster nodes:

```
<VirtualHost *:80>
ServerAdmin webmaster@localhost DocumentRoot /var/www/html

ErrorLog ${APACHE_LOG_DIR}/error.log
CustomLog ${APACHE_LOG_DIR}/access.log combined


<Location "/pentaho">
ProxyPass balancer://dicluster ProxyPassReverseCookiePath / /pentaho
</Location>
</VirtualHost>


<Proxy balancer://dicluster>
BalancerMember http://linuxvm1:8080/pentaho retry=30 BalancerMember
http://linuxvm2:8080/pentaho status=+H retry=0
</Proxy>
```

*In the above code snippet, notice that **sticky sessions** are not used or needed. That is because this configuration is used for HA failover only, not for true load balancing.*

5. After adding the above code, save the file and restart the Apache service.
6. You should now be able to browse to your load balancer and see the Pentaho User Console, for example:

```
http://loadbalancerhostname/pentahodi
```

About Load Balancing and DI Server

The DI Servers use an **Active/Passive** approach for load balancing, which means that all connections will connect to the same node unless it goes down, in which case failover to the secondary node will happen.

*Make sure to set all **Scheduled Jobs** on your passive node(s) to **PAUSE**. This will direct all Scheduled Jobs to your **active node** only and prevent round robin scheduling. Pentaho's Developer Center has information on how to Pause, Stop, or Start the scheduler.*

In the code snippet in Apache Configuration (Linux Environment) above, you will see that `linuxvm1` is designated as primary node, while `linuxvm2` is designated as secondary, or Hot Standby. This is indicated with `linuxvm2` using the parameter `status=+H`. The primary node, `linuxvm1`, has the parameter of `retry=30`, which means if it goes down, Apache will try every 30 seconds to see if the node is back up. Once it is, traffic and scheduled jobs will then be routed back to the primary node again.

If you are looking for a true load balancing solution where you can configure multiple DI Servers or multiple Carte servers for parallel processing, then you will want to consider creating slave servers and cluster schemas. Pentaho documentation has information on how to set up Carte clusters.

© Hitachi Vantara Corporation 2019. All Rights Reserved

Configure a Load Balancer for DI Server High Availability

It is common for medium to large-sized organizations to standardize on an enterprise load balancer (for example, F5 or Citrix). If you install Pentaho in a cloud environment, you can easily use a cloud load balancer to route traffic directly to individual Tomcat containers running Pentaho Server or BA/DI Servers in dedicated or shared compute engines or virtual machines.

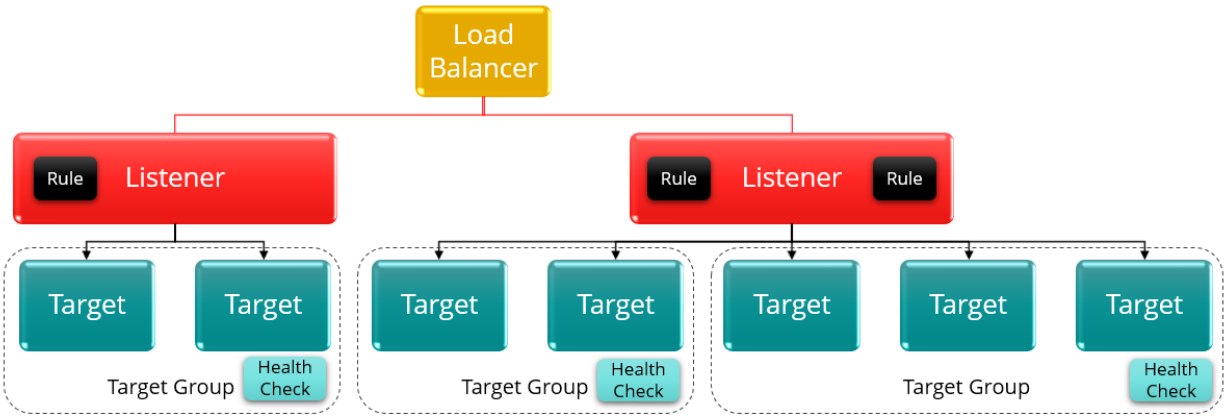


Figure 2: Using a Cloud Load Balancer

We recommend the following for load balancing setups:

Table 2: Load Balancing Recommendations

Recommendation	Details
Do not choose round robin	Round robin does not work for a DI-only environment.
Select load balancer for cloud provider	Select one of the following for your Pentaho web applications, according to your cloud provider: • HTTP(S) load balancer (GCP) • Application load balancer (AWS) • Application gateway (Microsoft Azure)
Set up automatic monitoring	Set up automatic monitoring and make sure health checks are reported or logged to your enterprise monitoring platform (Stackdriver for Google, Splunk, etc.), whether the load balancer is on-premises or in the cloud. See the figure before this table for an illustration.
Switch ports where the load balancer is exposed to internet traffic	In places your load balancer is exposed to internet traffic, switch ports from the defaults of `80` (`HTTP`) or `443` (`HTTPS`) to some other port, such as `8443`, both on the load balancer backend and on Tomcat. This provides an extra security measure.
Install Secure Sockets Layer (SSL) certificates directly on the load balancer	Installing SSL certificates directly on the load balancer, instead of on individual servers running Pentaho within Tomcat containers,

© Hitachi Vantara Corporation 2019. All Rights Reserved

Recommendation	Details
	makes it easier to add Tomcat nodes behind the cluster without having to pay for additional SSL certificates.
Mask the identity of servers	Translate IP addresses on outbound traffic, where possible to mask the identity of servers behind the load balancer.
Set up computing resources in different availability zones	Take advantage of cross-zone load balancing by setting up computing resources in different availability zones or regions, increasing application availability and resiliency, and simplifying your machine cycles.
Use routing policies for global load balancing	Using routing policies for global load balancing helps overcome the limitations of routing traffic across regions or zones. For example, in AWS, Route 53 policies allow global traffic to be routed across data centers around the globe.

You can find more information on these topics in the following sections:

- Configuring Enterprise/Cloud Load Balancers for Active/Passive
- Configuring Enterprise/Cloud Load Balancers for Active/Active
- Testing the DI Server Load Balancer
- High Availability for the Backend Repository

Configuring Enterprise/Cloud Load Balancers for Active/Passive

Clustered Pentaho DI Server installations should be run in active/passive mode. When using enterprise/cloud load balancers, you can configure failover manually or automatically:

- Configure manual failover by enabling both primary (active) and secondary (passive) nodes on the load balancers, and shutting down the Pentaho application on the secondary (passive) node.
- Configure automatic failover by running both primary (active) and secondary nodes, but disabling the secondary node from the load balancer console until the primary node becomes unavailable.

Refer to your enterprise load balancer's documentation for how to configure failover from one pool/node to another.

In this active/passive scenario, the enterprise load balancer URL will be used to recreate repository connections from PDI, and in the `repositories.xml` file for the Pentaho Server application.

© Hitachi Vantara Corporation 2019. All Rights Reserved

Configuring Enterprise/Cloud Load Balancers for Active/Active

While clustered Pentaho DI Server nodes must be run in an **active/passive** mode, you can use the **secondary node/server that hosts** the **passive** Pentaho Server application to provide additional computing capacity for jobs and transformations by running additional Carte servers.

Carte servers do *count against your number of licensed CPU cores. Confirm your license terms before installing additional Carte servers.*

Persistence, session affinity or stickiness **must be** enabled. In general, the most recommended way of enforcing affinity is using client browser-based cookies (source-IP based affinity has some restrictions based on client network setup).

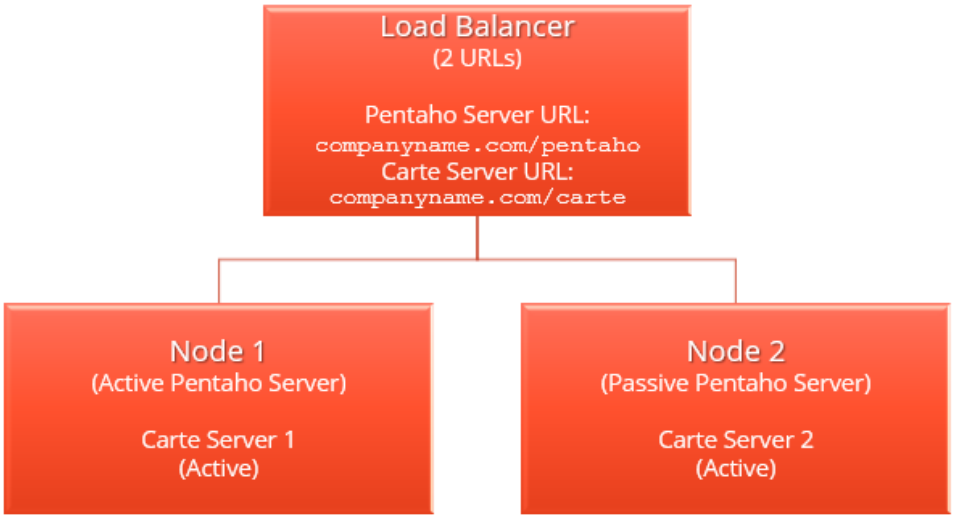


Figure 3: Carte Set Up

In this configuration, each Carte's `repositories.xml` file will reference the enterprise load balancer URL for Pentaho DI Servers, for example: `https://company.com/pentaho`.

The schedulers will use a different and dedicated URL that points exclusively to Carte servers as well as the active Pentaho DI Server, for example: `https://companyname.com/carte`.

© Hitachi Vantara Corporation 2019. All Rights Reserved

Testing the DI Server Load Balancer

Follow these steps to test your DI server load balancer:

1. When connecting to the DI server repository via the PDI Client, you will want to use the load balancer hostname followed by the web application name you configured in the Apache configuration file like this:

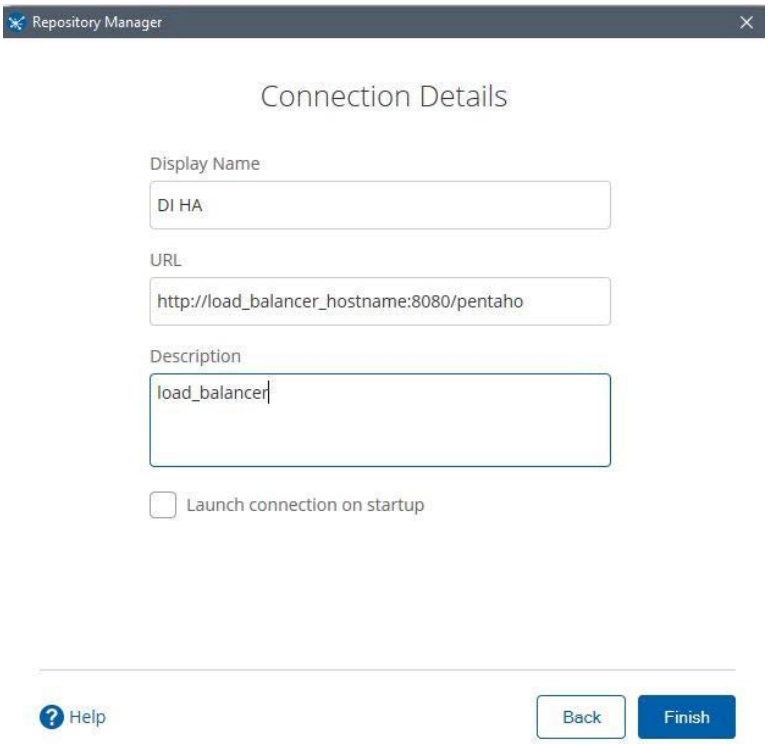


Figure 4: Connect to DI Server Repository

2. Once connected, create a slave connection using the same load balancer hostname and web application name like this:

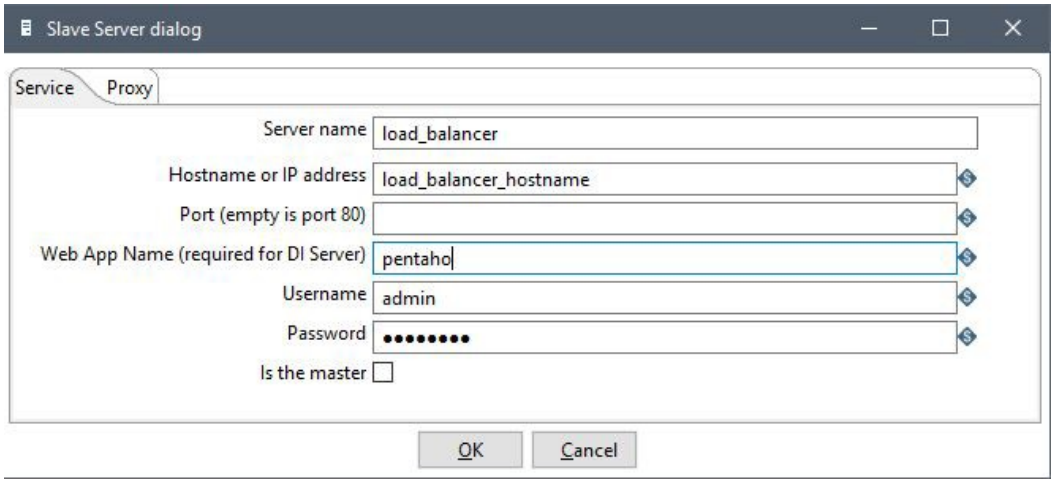


Figure 5: Create a Slave Connection Using Same Hostname

© Hitachi Vantara Corporation 2019. All Rights Reserved

3. Create a simple transformation; for example, one that just writes some text to the log and wraps it in a parent job. Configure the transformation to execute on a remote slave server like this:

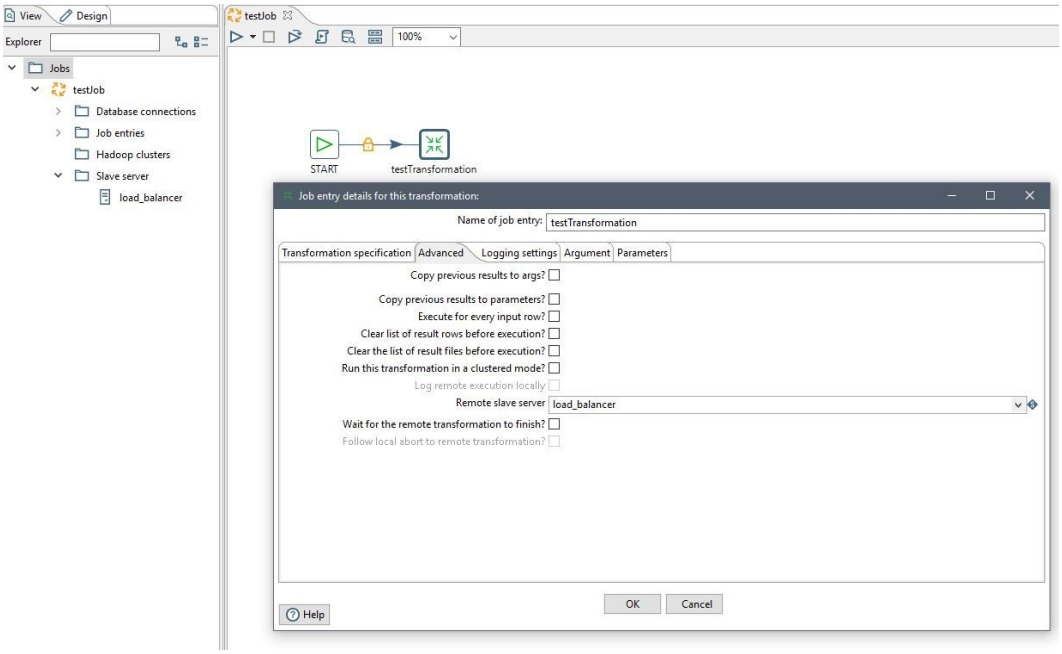
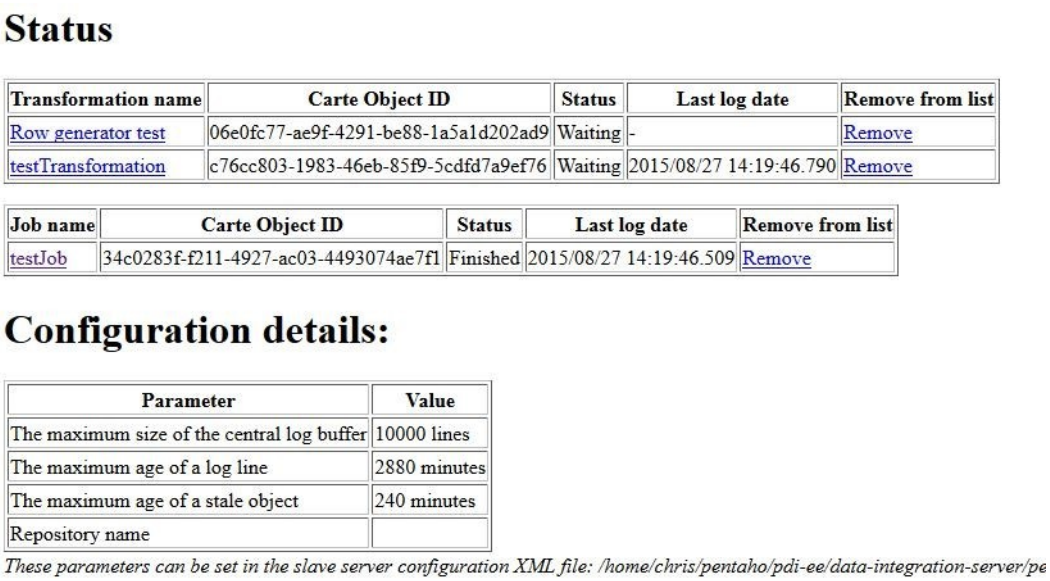


Figure 6: Create a Transformation to Execute on Slave Server

4. Schedule the job to run every minute and end within 10 minutes. This should give you enough time to see the job execute on the primary node, stop the primary node, and watch the schedule resume on the secondary node. The schedule can be created by going to **Action → Schedule**.
5. In a web browser, go directly to the node you have configured as `primary`. You should see both the job and transformation here, with a status of **Finished** and **Waiting**.



Status

Transformation name	Carte Object ID	Status	Last log date	Remove from list
Row generator test	06e0fc77-ae9f-4291-be88-1a5a1d202ad9	Waiting	-	Remove
testTransformation	c76cc803-1983-46eb-85f9-5cdfd7a9ef76	Waiting	2015/08/27 14:19:46.790	Remove

Job name	Carte Object ID	Status	Last log date	Remove from list
testJob	34c0283f-f211-4927-ac03-4493074ae7f1	Finished	2015/08/27 14:19:46.509	Remove

Configuration details:

Parameter	Value
The maximum size of the central log buffer	10000 lines
The maximum age of a log line	2880 minutes
The maximum age of a stale object	240 minutes
Repository name	

These parameters can be set in the slave server configuration XML file: /home/chris/pentaho/pdi-ee/data-integration-server/pe

Figure 7: Transformation and Job Status in Web Browser

© Hitachi Vantara Corporation 2019. All Rights Reserved

6. Shut down the primary node, and browse to the secondary node in the web browser. You should see the scheduled execution resume here now.
7. Start the primary node backup, and you should see the schedule resume on the primary node again.

High Availability for the Backend Repository

At this point, we have covered high availability for the DI server. Even if you have clustered the load balancers, you may still notice that a single point of failure still exists: the backend repository database.

We recommend that you cluster the database used to hold the repository. Doing this requires no special configuration on the Pentaho-side, since each of the servers will see a clustered database as a single entity. You will just need to follow the documentation from the proper vendor to cluster the database according to your environment and needs. Once you have done this, the same instructions for initializing your database in the prerequisites of this guide can be followed with no additional changes.

Related Information

Here are some links to information that you may find helpful while using this best practices document:

- Components Reference
- Configure Jackrabbit Journal
- Configure Quartz
- Pause the Scheduler
- Pentaho Developer Center
- Set Up a Cluster
- Shut Down the Scheduler
- Start the Scheduler
- Use Carte Clusters

The following vendor links can help you get started on clustering the supported databases:

- Oracle: Clustering
- MySQL: Cluster Reference Guides
- PostgreSQL: Creating a Database Cluster

© Hitachi Vantara Corporation 2019. All Rights Reserved