Quick Start for HVR - HDFS

Contents

This quick start guide helps you to get started with HVR for replicating data into Hadoop Distributed File System (HDFS).

To proceed with this replication you must have basic understanding about HVR's architecture and terminologies like Hub, Location, Channel, Location Groups, Actions etc.

The example here demonstrates how to replicate tables from one Oracle schema (source location) to HDFS (target location).

Before proceeding with this example ensure that the requirements for using HVR with Oracle and Greenplum are met.

For information about access privileges and advanced configuration changes required for performing replication using Oracle and HDFS, see:

- Requirements for Oracle
- Requirements for HDFS

Create Test Schema and Tables

Create Source Schema

```
create user sourcedb identified by hvr default tablespace users temporary
tablespace temp quota unlimited on users;
```

Create Tables in Source Schema

```
create table sourcedb.dm51_product
(
 prod_id number(10) not null,
 prod_price number(10,2) not null,
 prod_descrip varchar2(100) not null,
 primary key (prod_id)
);
```

```
create table sourcedb.dm51_order
(
 prod_id number(10) not null,
 ord_id number(10) not null,
 cust_name varchar2(100) not null,
 cust_addr varchar2(100),
 primary key (prod_id, ord_id)
);
```

Insert Values in Source Tables

```
insert into sourcedb.dm51_product values (100, 90, 'Book');
```

```
insert into sourcedb.dm51_order values (100, 123, 'Customer1', 'P.O. Box
122, Anytown, Anycountry');
```

Install HVR on the hub

An HVR distribution is available for download at https://www.hvr-software.com/account/. To request a trial version, visit https://www.hvr-software.com/free-trial/.

Install HVR on a hub machine. For details on installing HVR, see the respective operating system sections:

- Installing HVR on UNIX or Linux
- Installing HVR on Windows
- Installing HVR on macOS

The HVR distribution requires a license key in order for the software to operate. Please see the HVR licensi ng page for more details on how to install the HVR license.

After the installation, you can control HVR using the HVR graphical user interface (**HVR GUI**).

- If the hub machine is Windows, then **HVR GUI** can be executed directly on the hub machine.
    - To control HVR remotely from your PC, connect to the hub machine using Windows Remote Desktop Connection and launch **HVR GUI** on the hub machine.
- If the hub machine is Linux, then **HVR GUI** can be executed directly on the hub machine. However, an application like X Server or VNC viewer must be installed to run **HVR GUI** directly on Linux.
    - To control HVR remotely from your PC, install HVR on the PC (with Windows or macOS) and configure the **HVR Remote Listener** on the hub machine.
- If the hub machine is Unix, then **HVR GUI** should typically be run remotely from a PC to control HVR installed on the hub machine. To do this, install HVR on the PC (with Windows or macOS) and configure the **HVR Remote Listener** on the hub machine.

The **HVR Remote Listener** allows you to connect **HVR GUI** available on your PC to the remote HVR hub machine. For more information about connecting to remote HVR installation, see Configuring Remote Installation of HVR on Unix or Linux and Configuring Remote Installation of HVR on Windows.

Install HVR to connect to Hadoop

First read section Architecture Overview which explains the HVR's terminology and architecture. In particular this explains the importance of a hub database.

In order to connect to Hadoop as a target HVR must be running on a Linux environment. This can be the HVR hub running on Linux, or HVR can be running as an agent on a Linux server to connect into HDFS, optionally on the Hadoop namenode. The HDFS interface requires a Java 7 Runtime Environment and Hadoop connectivity libraries.

Install the HVR software on the Linux machine by following the installation steps in section Installing HVR on Unix or Linux.
Follow the steps in Requirements for HDFS
for the Java Runtime environment and the Hadoop connectivity libraries.

Create the Hub Database

This section describes how to create a hub database (schema). The hub database is a repository database that HVR uses to control its replication activities. It contains HVR catalog tables that hold all specifications of replication such as the names of the replicated databases, the replication direction and the list of tables to be replicated. For more information about HVR hub server and database, see section Hub Server in System Requirements.

HVR supports the creation of a hub database on certain databases (location classes) only. For the list of supported location classes, see section Hub Database in Capabilities.

For this demonstration, the hub database (e.g. **hvrhub**) is created in Oracle.

- Create the hub database (**hvrhub**) with password (**hvr**).

  ```
  create user hvrhub
  identified by hvr
  default tablespace users
  temporary tablespace temp
  quota unlimited on users;
  ```

Connect To Hub Database

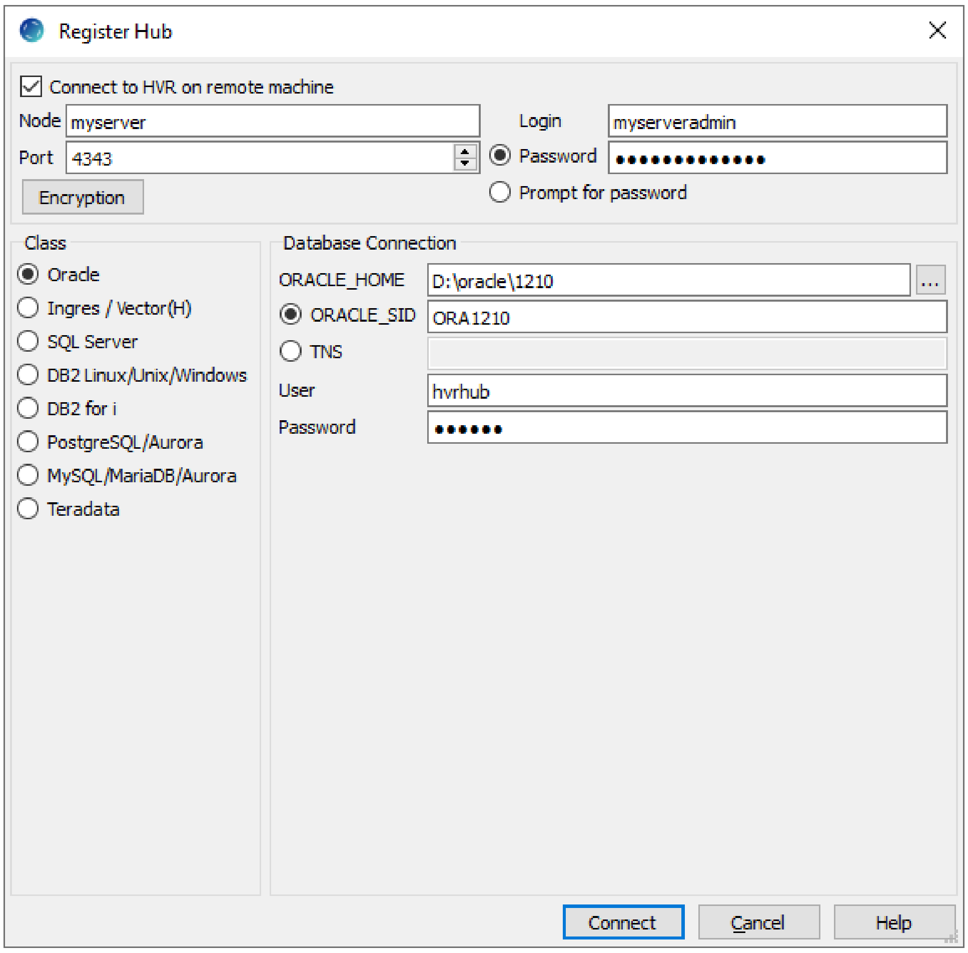
This section describes how to connect **HVR GUI** to the hub database.

When you launch HVR GUI for the first time, the **Register Hub** dialog is displayed automatically. The **Register Hub** dialog can also be accessed from menu **File** by selecting **Register Hub**. Skip steps 1 to 4 if you want to run **HVR GUI** directly on the hub machine.

1. Click **Connect to HVR on remote machine**.

   To connect **HVR GUI** on a PC to a remote HVR hub machine, the **HVR Remote Listener** must be configured and running on the HVR hub machine.
2. Enter the name or IP address of the hub machine in the **Node** field (e.g. **myserver**).
3. Enter the port number (defined in the **HVR Remote Listener** of the hub machine) in the **Port** field (e.g. **4343**).
4. Enter the **Login** (e.g. **myserveradmin**) and **Password** for the hub machine. By default, this is the operating system login credentials of the hub machine.
5. Select **Oracle** in the **Class** pane.
6. Specify **Database Connection** details.
   a. Enter the directory path in **ORACLE_HOME**. You can also click the browse button to select the directory path.
   b. Enter the Oracle System ID in **ORACLE_SID** or **TNS** credentials.
   c. Enter the user name of the hub database in **User** (e.g. **hvrhub**).
   d. Enter the password for the hub database in **Password** (e.g. **hvr**).

7. Click **Connect**.



Register Hub

[✓] Connect to HVR on remote machine

Node: myserver
Port: 4343
Login: myserveradmin
(●) Password: ●●●●●●●●●●●●●
( ) Prompt for password

Encryption

Class
(●) Oracle
( ) Ingres / Vector(H)
( ) SQL Server
( ) DB2 Linux/Unix/Windows
( ) DB2 for i
( ) PostgreSQL/Aurora
( ) MySQL/MariaDB/Aurora
( ) Teradata

Database Connection
ORACLE_HOME: D:\oracle\1210
(●) ORACLE_SID: ORA1210
( ) TNS
User: hvrhub
Password: ●●●●●●

Connect | Cancel | Help

8. Click **Yes** in the prompt dialog asking to create catalog tables in the hub database.

   HVR displays this prompt when connecting to a hub database for the first time.

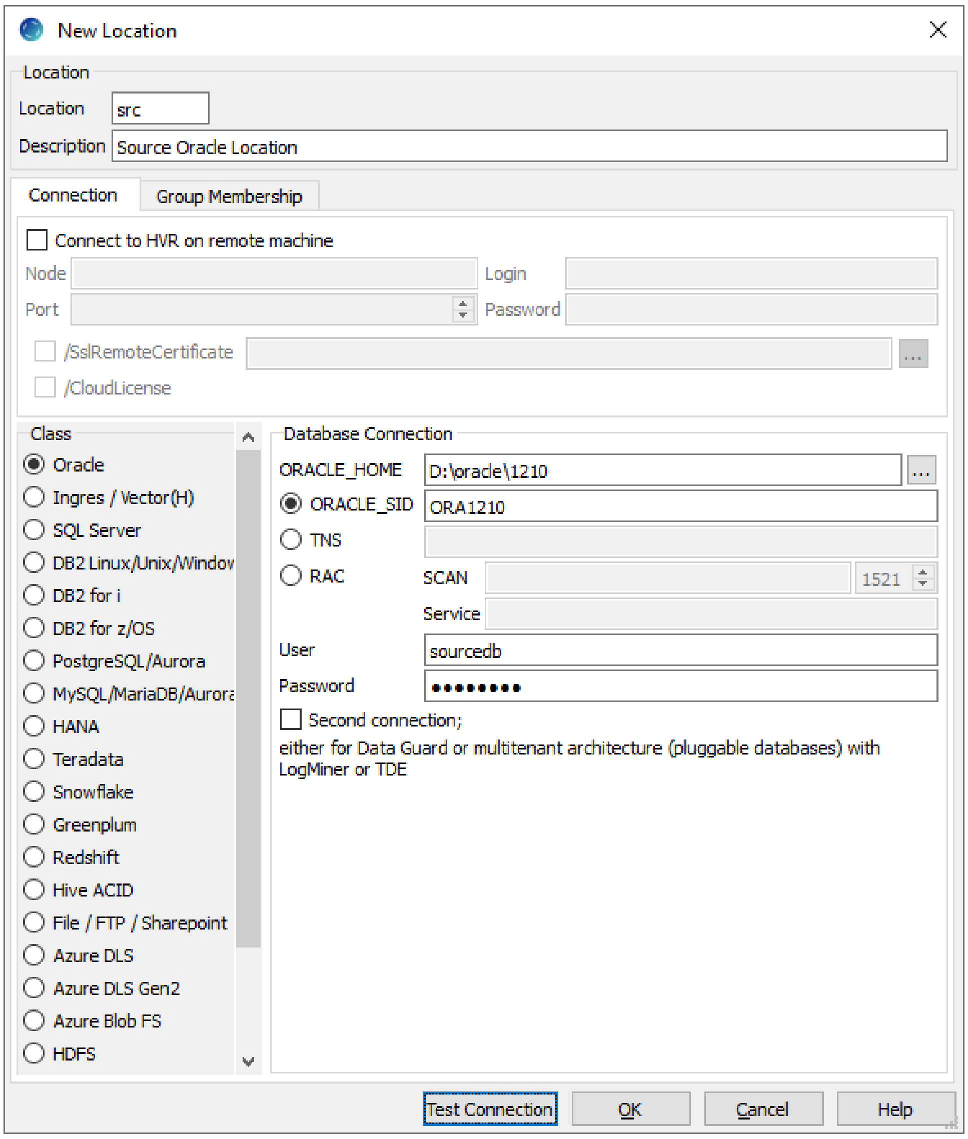
   On connecting successfully to the hub database, the navigation tree pane displays the hub machine and the hub database. **Location Configuration**, **Channel Definitions**, and **Scheduler** are displaye d under the hub database.

Create Oracle Location

1. In navigation tree pane, right-click **Location Configuration** **New Location**.
2. Enter **Location** name and **Description** for the location.
3. Select **Oracle** in **Class**.
4. Provide **Database Connection** details. For more information on **Database Connection** fields, see section Location Connection.
   a. Enter directory path for **ORACLE_HOME**. You can also click browse to select directory path.
   b. Enter Oracle System ID in **ORACLE_SID** or **TNS** credential or **RAC** credential.

      For **RAC** connectivity, ensure to provide remote machine connection details under **Connection** tab.
   c. Enter user name of schema in **User**. For example, **sourcedb**.
   d. Enter password for schema in **Password**. For example, **hvr**.
5. Click **Test Connection** to verify the connection to location database.

6. Click **OK**.



New Location

Location

Location: src

Description: Source Oracle Location

Connection | Group Membership

[ ] Connect to HVR on remote machine

Node | Login

Port | Password

[ ] /SslRemoteCertificate

[ ] /CloudLicense

Class

(●) Oracle
( ) Ingres / Vector(H)
( ) SQL Server
( ) DB2 Linux/Unix/Window
( ) DB2 for i
( ) DB2 for z/OS
( ) PostgreSQL/Aurora
( ) MySQL/MariaDB/Aurora
( ) HANA
( ) Teradata
( ) Snowflake
( ) Greenplum
( ) Redshift
( ) Hive ACID
( ) File / FTP / Sharepoint
( ) Azure DLS
( ) Azure DLS Gen2
( ) Azure Blob FS
( ) HDFS

Database Connection

ORACLE_HOME: D:\oracle\1210

(●) ORACLE_SID: ORA1210

( ) TNS

( ) RAC — SCAN: 1521 — Service:

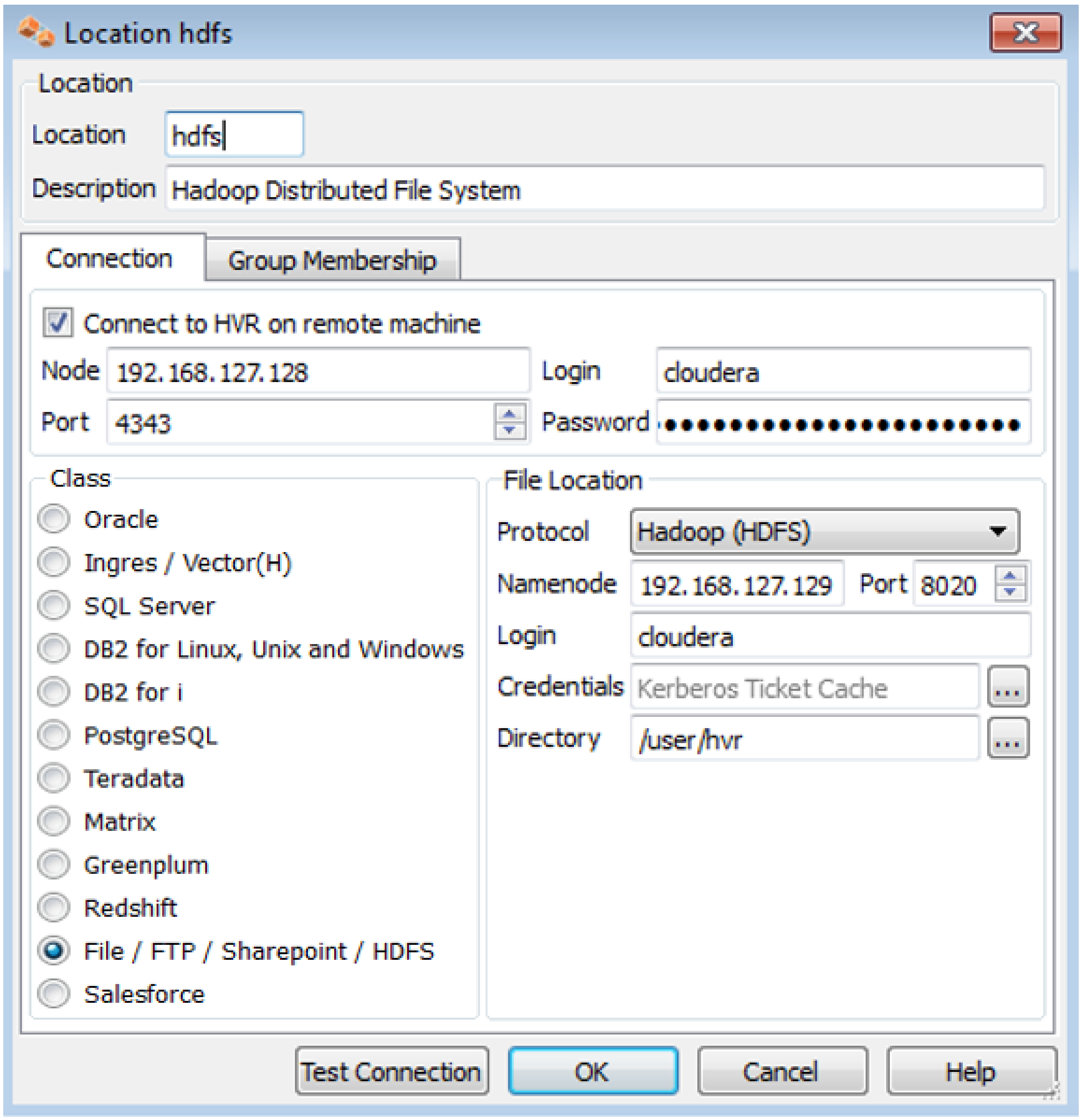
User: sourcedb

Password: ●●●●●●●●

[ ] Second connection;
either for Data Guard or multitenant architecture (pluggable databases) with LogMiner or TDE

Create HDFS Location

Create a location for HDFS using right-click on **Location Configuration New Location**.


Location hdfs

Location

Location: hdfs

Description: Hadoop Distributed File System

Connection | Group Membership

[✓] Connect to HVR on remote machine

Node: 192.168.127.128 Login: cloudera

Port: 4343 Password: ●●●●●●●●●●●●●●●●●●●●●●

Class
- Oracle
- Ingres / Vector(H)
- SQL Server
- DB2 for Linux, Unix and Windows
- DB2 for i
- PostgreSQL
- Teradata
- Matrix
- Greenplum
- Redshift
- [✓ File / FTP / Sharepoint / HDFS]
- Salesforce

File Location

Protocol: Hadoop (HDFS)

Namenode: 192.168.127.129 Port: 8020

Login: cloudera

Credentials: Kerberos Ticket Cache

Directory: /user/hvr

In this example **Connect to HVR on remote machine** is checked assuming this Linux environment is not the hub machine. If it is then it must not be checked.

Ignore the **Group Membership** tab for now.

In order to connect to a Hadoop cluster with Kerberos authentication, you can consult HDFS Authentication and Kerberos.

Create a Channel

This section describes how to create a channel in HVR. A channel groups together the locations and tables that are required for replication. It also contains actions that control how replication should be done.

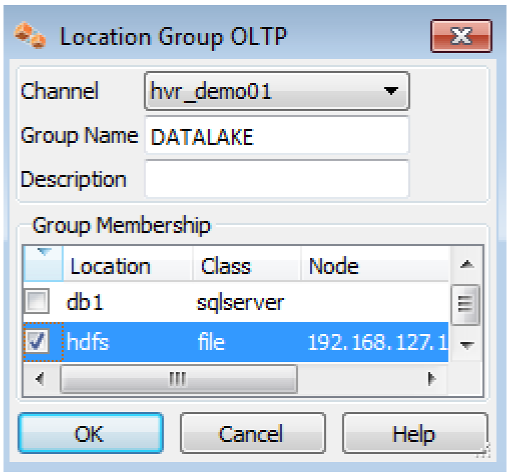
1. In navigation tree pane, right-click **Channel Definitions** **New Channel**.
2. Enter **Channel** name and **Description** for the channel in **New Channel** dialog.
3. Click **OK**.

Create Location Groups

This section describes how to create location groups in a channel. The location groups are used for defining action on the location. Typically a channel contains two location groups - one for the source location and one for the target location. Each location group can contain multiple locations.

In this example, create one source location group (**OLTP**) and one target location group (**DATALAKE**).

1. In navigation tree pane, click **+** next to the channel (**hvrdemo**).
2. Create source location group (**OLTP**):
    a. Right-click **Location Groups New Group**.
    b. Enter **Group Name** and **Description** for the location group.
    c. Select source location (**db1**) from **Group Membership**.
    d. Click **OK**.
3. Create target location group (**DATALAKE**):
    a. Right-click **Location Groups New Group**.
    b. Enter **Group Name** and **Description** for the location group.
    c. Select target location (**hdfs**) from **Group Membership**.
    d. Click **OK**.


Location Group OLTP

Channel: hvr_demo01
Group Name: DATALAKE
Description:

Group Membership

	Location	Class	Node
☐	db1	sqlserver	
✓	hdfs	file	192.168.127.1

Select Table(s)

This section describes how to select the tables from source location for replication. **Table Explore** allows you to select schema(s) and/or table(s) for replication.

1. Right-click **Tables Table Explore**.
2. Select source location (**db1**) from the list.
3. Click **Connect**.
4. Select tables from **Table Explore** dialog. Press **Shift** key to select multiple tables or **Ctrl+A** to select all tables.
5. Click **Add** to add the selected tables.
6. Click **OK** in **HVR Table Name** dialog.
7. Click **Close** in **Table Explore** dialog.

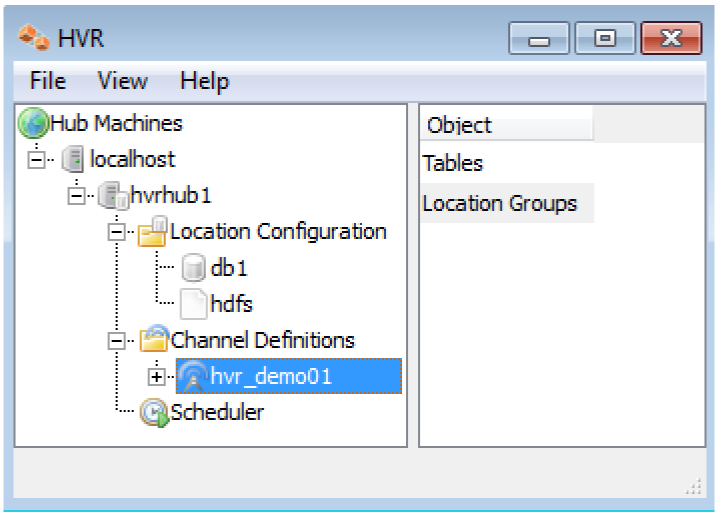
Define Actions

The new channel needs actions to define the replication.

- Right-click on group **OLTP New Action Capture**.

- Right-click on **Group DATALAKE New Action FileFormat**. Select parameter **/Csv**, and optionally specify arguments to change the format of the output file.

- Right-click on **Group DATALAKE New Action Integrate**. Check **/RenameExpression**, and put in **/{hvr_tbl_name}/{hvr_integ_tstamp}.csv** to create new files for every integrate cycle. Also add **ColumnProperties /Name=hvr_op_val /Extra /IntegrateExpression={hvr_op} /TimeKey** to include the operation type per record (0 for delete, 1 for insert, 2 for update). For a very busy system check the option **/Burst** to combine multiple transactions into one (larger) file per table.
- Right–click on **Group DATALAKE New Action Scheduling**. Check **/IntegrateStartTimes**, and select from the calendar. For example for a 10 minute refresh interval starting at the top of the hour check multiples of 10 for resulting *RefreshStartTimes /0,10,20,30,40,50 * * * **.

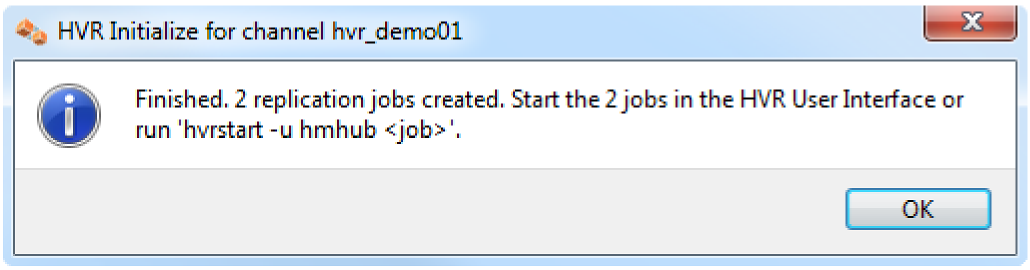


The Actions pane only displays actions related to the objects selected in the left–hand pane. So click on channel **hvr_demo01** to see both actions.

Enable Replication with HVR Initialize

Now that the channel definition is complete, create the runtime replication system.

Right–click on channel **hvr_demo01 HVR Initialize**. Choose **Create or Replace Objects** and click **HVR Initialize**.


HVR Initialize for channel hvr_demo01

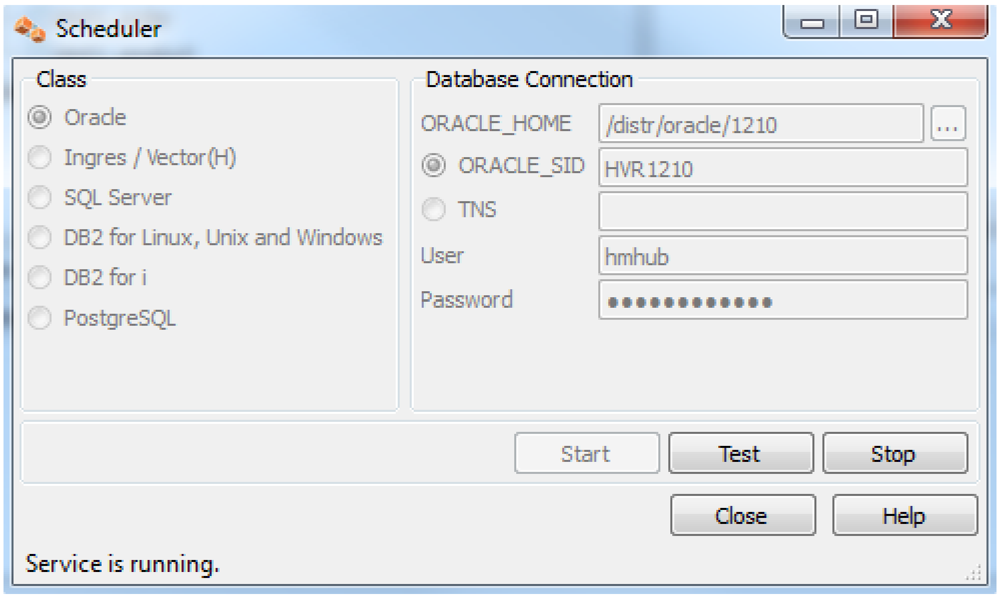
Finished. 2 replication jobs created. Start the 2 jobs in the HVR User Interface or run 'hvrstart -u hmhub <job>'.

From the moment that HVR Initialize is done, all new transactions that start on the database **testdb1** will be captured by HVR when its capture job looks inside the transaction logs.

HVR Initialize also creates two replication jobs, which can be seen under the Scheduler node in the GUI.

Start Scheduling of Capture Job

Start the Scheduler on the hub machine by clicking in the HVR GUI on the **Scheduler** node of the hub database.


Scheduler

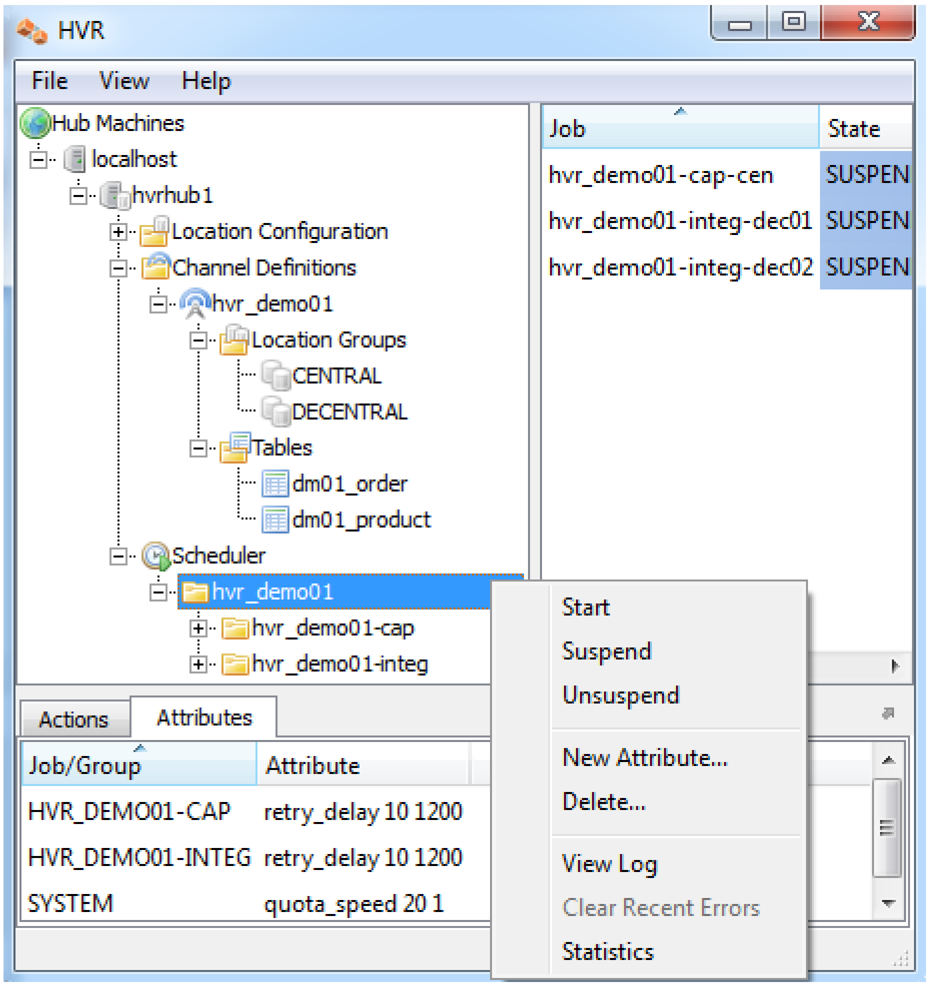
Class
- Oracle [✓]
- Ingres / Vector(H)
- SQL Server
- DB2 for Linux, Unix and Windows
- DB2 for i
- PostgreSQL

Database Connection

ORACLE_HOME	/distr/oracle/1210
ORACLE_SID [✓]	HVR1210
TNS	
User	hmhub
Password	●●●●●●●●●●●●

Service is running.

Next, instruct the HVR Scheduler to trigger the capture job to keep the capture job current with transaction log generation.



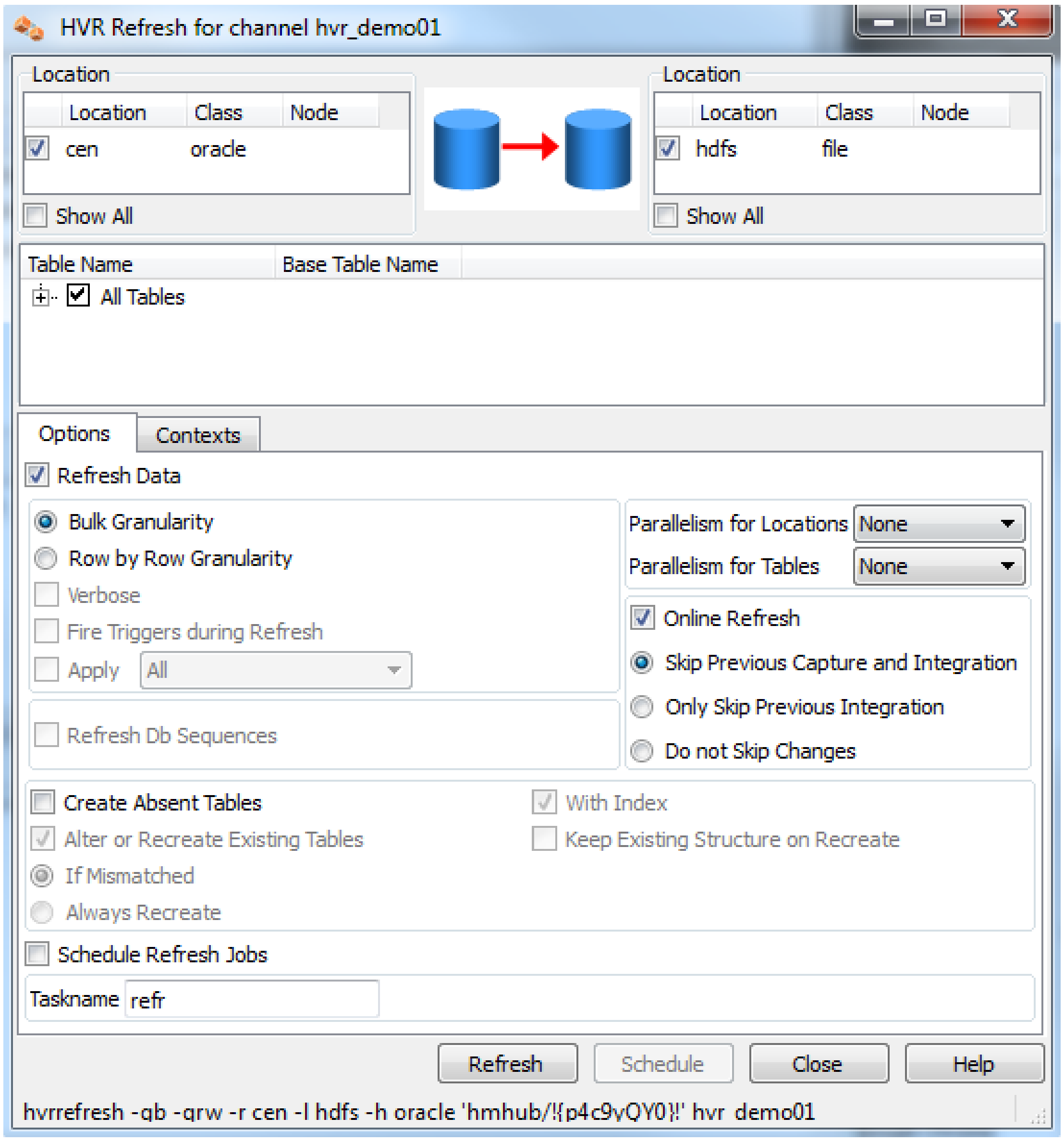
Job	State
hvr_demo01-cap-cen	SUSPEN
hvr_demo01-integ-dec01	SUSPEN
hvr_demo01-integ-dec02	SUSPEN

Job/Group	Attribute
HVR_DEMO01-CAP	retry_delay 10 1200
HVR_DEMO01-INTEG	retry_delay 10 1200
SYSTEM	quota_speed 20 1

The capture job inside the Scheduler executes a script under **$HVR_CONFIG/job/hvrhub/hvr_demo01** that has the same name as the job. So job **hvr_demo01–cap–db1** detects changes on database **testdb1** a nd stores these as transactions files on the hub machine.

Perform Initial Load

Perform the initial load from the OLTP database into HDFS using **HVR Refresh**. Right-click on the channel **hvr_demo01 HVR Refresh**.


HVR Refresh for channel hvr_demo01

Location

	Location	Class	Node
✓	cen	oracle	

☐ Show All

Location

	Location	Class	Node
✓	hdfs	file	

☐ Show All

Table Name	Base Table Name
✓ All Tables	

Options | Contexts

- ✓ Refresh Data
- ● Bulk Granularity
- ○ Row by Row Granularity
- ☐ Verbose
- ☐ Fire Triggers during Refresh
- ☐ Apply: All
- ☐ Refresh Db Sequences
- Parallelism for Locations: None
- Parallelism for Tables: None
- ✓ Online Refresh
- ● Skip Previous Capture and Integration
- ○ Only Skip Previous Integration
- ○ Do not Skip Changes
- ☐ Create Absent Tables
- ✓ With Index
- ✓ Alter or Recreate Existing Tables
- ☐ Keep Existing Structure on Recreate
- ● If Mismatched
- ○ Always Recreate
- ☐ Schedule Refresh Jobs
- Taskname: refr

Refresh | Schedule | Close | Help

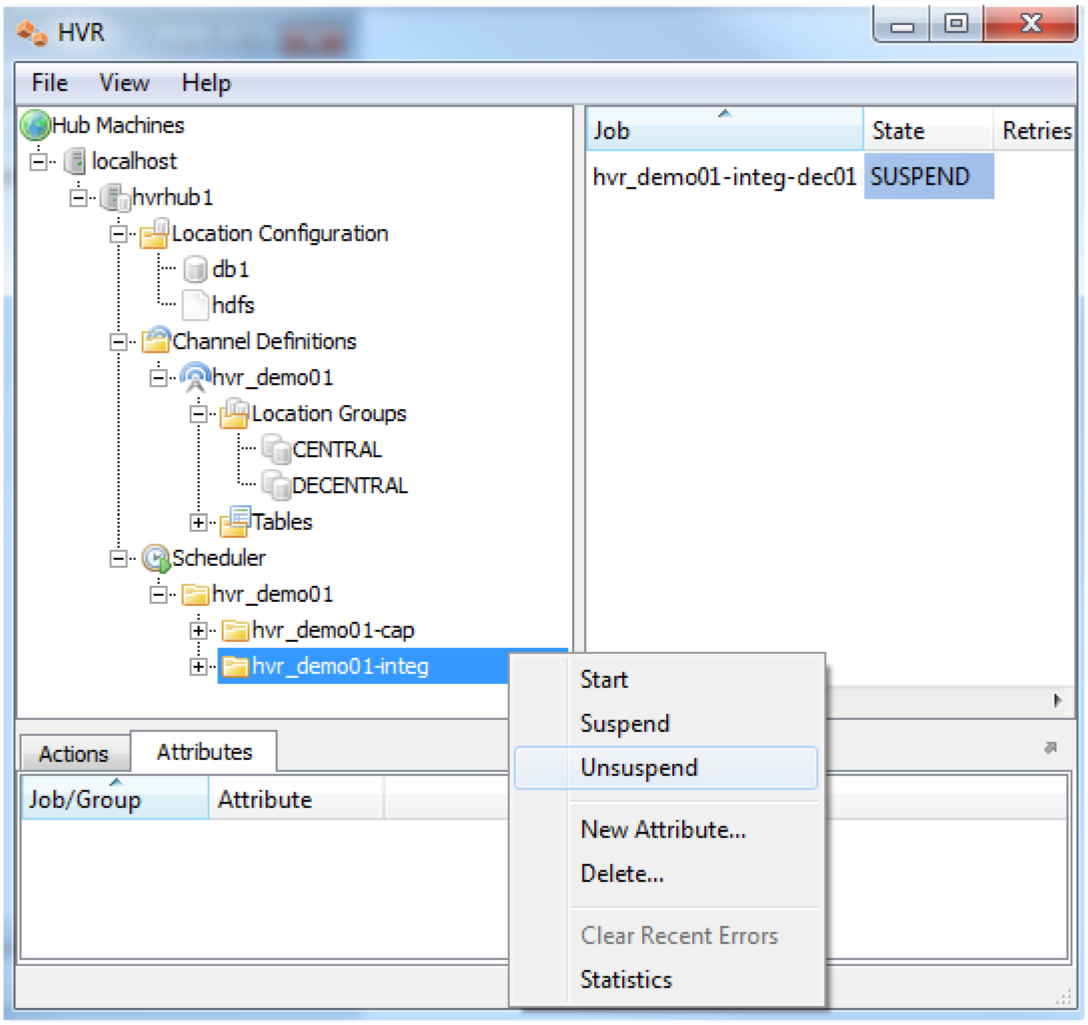
```
hvrrefresh -gb -qrw -r cen -l hdfs -h oracle 'hmhub/!{p4c9vQY0}!' hvr_demo01
```

Make sure the option for **Online Refresh** is checked and select **Skip Previous Capture and Integration**. Optionally set **Parallelism for Tables**.

Run the Refresh.

Start Scheduling of Capture Job

Instruct the HVR Scheduler to Unsuspend the integrate job to push changes into HDFS according to the defined schedule.



The integrate job inside the Scheduler executes a script under **$HVR_CONFIG/job/hvrhub/hvr_demo01** that has the same name as the job. So job **hvr_demo01–integ-hdfs** picks up transaction files on the hub on a defined schedule and creates files on HDFS containing these changes.

A scheduled job that is not running is in a **PENDING** state.

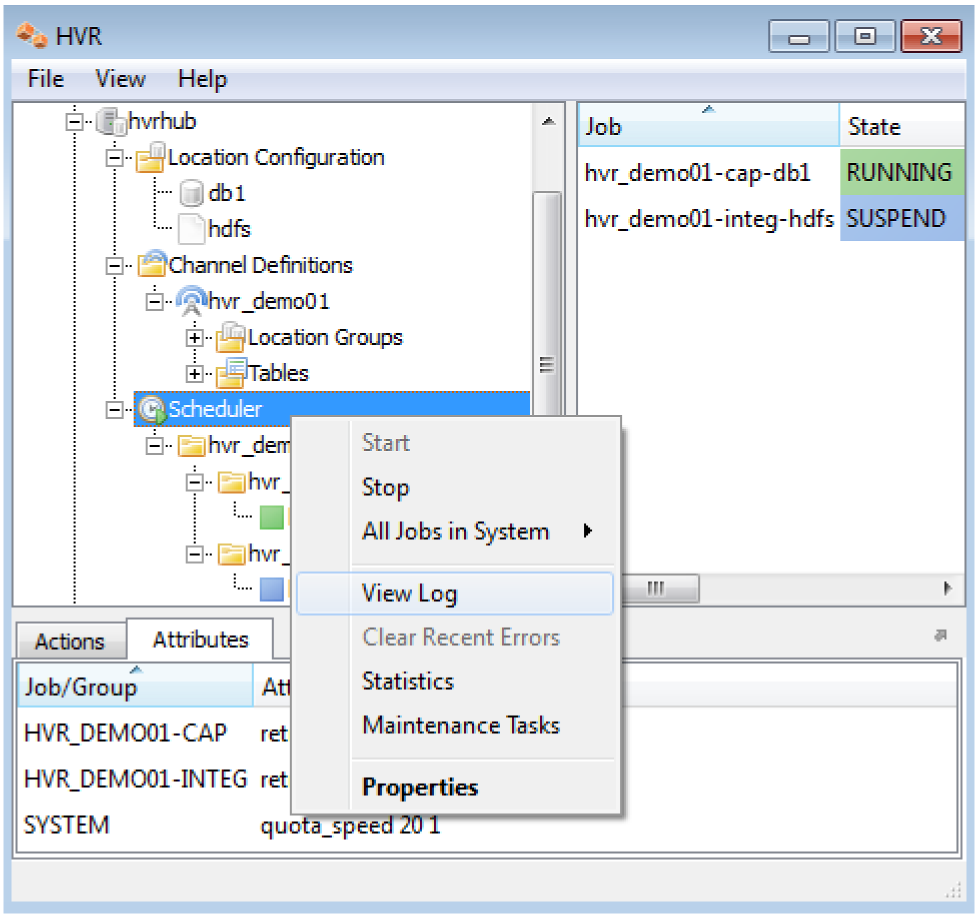
Test Replication

To test replication, make a change in **testdb1**:

```
testdb1/hvr
SQL> insert into dm01_product values (1, 19.99, 'DVD');
SQL> commit;
```

Next, instruct the HVR Scheduler to Trigger the integrate job rather than wait for the next scheduled run.

In the HVR log file you can see the output of the jobs by clicking on **View Log**. This log file can be found in **$HVR_CONFIG/log/***hubdb***/hvr.out**.



Job	State
hvr_demo01-cap-db1	RUNNING
hvr_demo01-integ-hdfs	SUSPEND

Job/Group	Att
HVR_DEMO01-CAP	ret
HVR_DEMO01-INTEG	ret
SYSTEM	quota_speed 20 1

The job output looks like this:

```
 hvr_demo01-cap-db1: Scanned 1 transaction containing 1 row (1 ins) for 1
table.
 hvr_demo01-cap-db1: Routed 215 bytes (compression=40.6%) from 'db1' into 2
locations.
 hvr_demo01-cap-db1: Capture cycle 3.
 hvr_demo01-integ-hdfs: Integrate cycle 1 for 1 transaction file (215 bytes).
 hvr_demo01-integ-hdfs: Moved 1 file to 'hdfs://cloudera@192.168.127.128/user
/hvr'.
 hvr_demo01-integ-hdfs: Processed 1 expired 'trigger' control file.
 hvr_demo01-integ-hdfs: Finished. (elapsed=4.04s)
```

This indicates that the jobs replicated the original change to **HDFS**. A query on **HDFS** confirms this:

```
 [cloudera@quickstart ~]$ hadoop fs -ls /user/hvr
 Found 2 items
 drwxr-xr-x   - cloudera supergroup          0 2015-02-06 10:38 /user/hvr
/_hvr_state
 -rw-r--r--   3 cloudera supergroup         12 2015-02-06 10:38 /user/hvr
/dm01_product_20150206183858632.csv
 [cloudera@quickstart ~]$ hadoop fs -cat /user/hvr
/dm01_product_20150206183858632.csv
```

```
1,19.99,DVD
[cloudera@quickstart ~]$
```