Quick Start for HVR - Azure Data Lake Storage Gen2

Contents

This quick start guide helps you to get started with HVR for replicating data into Azure Data Lake Storage (DLS) Gen2.

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

The example here demonstrates how to replicate tables from one local SQL Server database (source location) to Azure DLS Gen2 (target location) residing in the Azure cloud. In real-life scenarios, the source location(s) and the target location(s) typically reside on different machines, and the HVR hub can reside on a separate machine or on the source or target machine. However, in this demonstration, for simplicity, we have the source and hub database on the same machine, and the target on Azure cloud.

For Linux (x64) and Windows (x64), prior to HVR 5.7.0/8 and 5.7.5/4, it is required to install the Hadoop client on the machine from which HVR will access Azure DLS Gen2. For more information, see section Hadoop Client in Requirements for Azure Data Lake Storage Gen2.

Before proceeding with this demonstration, ensure that the requirements for using HVR with Azure DLS Gen2 are met.

For information about access privileges and advanced configuration required for performing replication using SQL Server and Azure DLS Gen2, see:

- Requirements for SQL Server
- Requirements for Azure Data Lake Storage Gen2

Create Demo Databases and Tables

The initial step of this demonstration is to create:

- a database in the source location
- tables in the source database (and insert values into these tables)
- an Azure DLS Gen2 storage account and a storage container in the target location

Source Location

Skip this section if you already have a database with tables which you plan to use for this replication.

For this demonstration, in the source location, create a database (e.g. **sourcedb**) with two tables (e.g. **dm01_product** and **dm01_order**), and insert values into these tables.

Sample SQL statements to create schema and tables in source location, and also to insert values into these tables,

- Create Source Database

```
create database sourcedb
```

- Create Tables in Source Database

```
create table dm01_product (
prod_id int not null,
prod_price decimal(10,2) not null,
prod_descrip varchar(100) not null
)
```

```
create table dm01_order (
prod_id int not null,
ord_id int not null,
cust_name varchar(100) not null,
cust_addr varchar(100) null
)
```

- Insert Values in Source Tables

```
insert into dm01_product
values (100, 90, 'Book')
```

```
insert into dm01_order
values (100, 123, 'Customer1', 'P.O. Box 122, Anytown, Anycountry')
```

Target Location

For this demonstration, create the following in the target location (on Azure portal):

- Azure DLS Gen2 storage account (e.g. **adlsstorageaccount**).
- Azure storage container (e.g. **adlscontainer**).
- If the authentication type to be used is **OAuth**, then perform App Registration (in Azure Active Directory) to get the location connection parameters - **OAuth2 Endpoint**, **Client ID**, **Client Secret**.

For information about creating a storage account, container, see Azure Documentation.

Create Hub Database

This section describes how to create a hub database. 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 SQL Server.

Create the hub database using the SQL Server Management Studio. Alternatively, use the following SQL statement:

```
create database hvrhub
```

It is recommended to create a new database schema (e.g. **hvrhubschema**) for the HVR hub. If an existing database is to be used for HVR Hub, then the HVR's catalog tables can be separated by creating a new database schema and associating it with HVR's user as follows:

```
create schema hvrhubschema

alter user hvrhub with default_schema=hvrhubschema
```

Grants/Access Privileges

This section describes the grants/access privileges required for the source and hub databases.

1. Configure the privileges for the source database (**sourcedb**).

   HVR's log-based capture using the **SQL** log read method supports three permission models: **SysAdmin**, **DbOwner**, and **Minimal**. However, the **DIRECT** log read method supports only the **SysAdmin** model.

   The **DbOwner** and **Minimal** models are only available for SQL Server 2012 and above. For the older versions of SQL Server, the **SysAdmin** model should be used.

   - **SysAdmin**
     The **User** should be granted a **sysadmin** role. There is no need for operators to perform special SQL statements manually. For this permission model, perform only installation steps 1 and 2 below; the others are unnecessary.

     This permission model is required when using **Capture /LogReadMethod=DIRECT**, except when using the **DIRECT** log read method combined with **Capture /ArchiveLogOnly**.
   - **DbOwner**
     The **User** should be granted a **db_owner** role for the source database, but not a **sysadmin** role. An operator must perform several special SQL statements manually only when setting up a new database for capture. For this permission model, perform only installation steps 1-4 and 7 below; numbers 5-6 are unnecessary.
   - **Minimal**
     The **User** does not require **sysadmin** or **db_owner** roles at runtime. But whenever an **HVR Initialize** command is run (for example to add a new table to a channel) then a user with a **db_owner** privilege must perform SQL statements manually. For this permission model, perform all the installation steps below: numbers 1-7.

   Installation steps depend on the setting of action **Capture /SupplementalLogging**. It can use the elements of SQL Server's Change Data Capture feature or the elements of SQL Server's own replication components (called 'articles'), or both. Articles can only be created on tables with a primary key. CDC tables are used by default on the SQL Server Enterprise and Developer editions. Articles are always used for the SQL Server Standard edition (prior to SQL Server 2016 Service Pack 1) to implement supplemental logging.

   Installation Steps

   a. For log-based capture from the SQL Server Enterprise Edition or Developer Edition, if articles are used (see above), HVR requires that the SQL Server Replication Components option is installed. This step is needed once when HVR is installed for an SQL Server instance.
   b. If articles are used (see above), a user with a **sysadmin** privilege must create a distribution database for the SQL Server instance, unless one already exists. To do this, a user with a **sysadmin** privilege should run SQL Server wizard **Configure Distribution Wizard**, which can be run by clicking **Replication > Configure Distribution...** Any database name can be supplied (click **Next > Next > Next**). This step is needed once when HVR is installed for an SQL Server instance.

      If Always On AG is installed and articles are used, then only one distribution database should

be configured. Either this can be set up inside the first node and the other nodes get a distributor that points to it. Or the distribution database can be located outside the Always On AG cluster and each node gets a distributor that points to it there.

c. For this step and subsequent steps, the HVR binaries must already be installed. For log read method **SQL**, a user with a **sysadmin** privilege must create a special 'wrapper' SQL procedures called **sp_hvr_dblog** and **sp_hvr_dbcc** or **sp_hvr_dbtable** so that the HVR can call the SQL Server's read-only function **fn_dump_dblog**. This must be done inside the SQL Server database's special database **msdb**, not the actual capture database. The SQL query to create these procedures is available in the file called **hvrcapsysadmin.sql** in directory **%HVR_HOME%\sql\sqlserver**. The HVR user must then be allowed to execute this procedure. For this, the HVR User (e.g. **hvruser**) must be added to the special **msdb** database and the following grants must be provided:

```
use msdb;

create user hvruser for login hvruser;

grant execute on sp_hvr_dblog to hvruser;

grant execute on sp_hvr_dbcc to hvruser;      -- only for HVR
versions upto 5.3.1/4

grant execute on sp_hvr_dbtable to hvruser;   -- only for HVR
versions since 5.3.1/5
```

This step is needed once when HVR is installed for an SQL Server instance. But if Always On AG is installed, then this step is needed on each Always On AG node.

d. A **sysadmin** user must grant the HVR user login a special read-only privilege in the **master** database.

```
use master;

grant view server state to hvruser;
```

This step is needed once when HVR is installed for an SQL Server instance. But if Always On AG is installed, then this step is needed on each Always On AG node.

e. A user with **db_owner** (or **sysadmin**) privilege must create 'wrapper' SQL procedures in each capture database so that HVR can call the SQL Server's read-only procedures **sp_helppublication**, **sp_helparticle** and **fn_dblog**. The SQL query to create these three read-only procedures is available in the file called **hvrcapdbowner.sql** in directory **%HVR_HOME%\sql\sqlserver**. The **User** must then be allowed to execute these procedures.

The following grants must be provided inside each capture database:

```
use capdb;

grant execute on sp_hvr_check_publication to hvruser;

grant execute on sp_hvr_check_article to hvruser;

grant execute on sp_hvr_dblog to hvruser;

grant execute on sp_hvr_repldone to hvruser;

grant execute on sp_hvr_repltrans to hvruser;
```

This step is needed once when each new source database is being set up.

f. A user with **db_owner** (or **sysadmin**) privilege must grant the HVR user a read-only privilege.

This step is needed once when each new source database is being set up.

```
use capdb;

alter role db_datareader add member hvruser;
```

g. When the **HVR Initialize** command is performed, it may need to perform SQL statements that would require **sysadmin** or **db_owner** privilege. One example is that it may need to create an Article on a replicated table to track its changes. In that case, **HVR Initialize** will write a script containing necessary SQL statements, and then show a popup asking for this file to be executed. The file will be written in directory **%HVR_CONFIG%\files** on the capture machine; its exact filename and the necessary permission level is shown in the error message. The first time **HVR Initialize** gives this message, a user with a **sysadmin** privilege must perform these SQL statements. Subsequently, these SQL statements can be performed by a user that just has a **db_owner** privilege.

2. Configure the privileges for the hub database (**hvrhub**).

```
grant create table to hvrhub
grant create procedure to hvrhub

grant select, insert, delete, update on schema::hubschema to hvrhub
grant control on schema::hvrhubschema to hvrhub
```

Download and Install HVR

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 licensing 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.

Launch HVR GUI

This section describes how to launch **HVR GUI** on various operating systems.

- On Windows and macOS, double-click the HVR shortcut icon available on the desktop or execute command **hvrgui** in the CLI.

- On Linux, double-click the hvrgui file available in the HVR_extracted_path/bin directory or execute command **hvrgui** in the CLI.

  Linux requires applications like X server or VNC viewer to execute **HVR GUI**.
- On Unix, **HVR GUI** is not supported. So, **HVR GUI** should be run on a remote PC (with Windows, Linux, or macOS) to control HVR installed on the Unix machine.

Register Hub

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

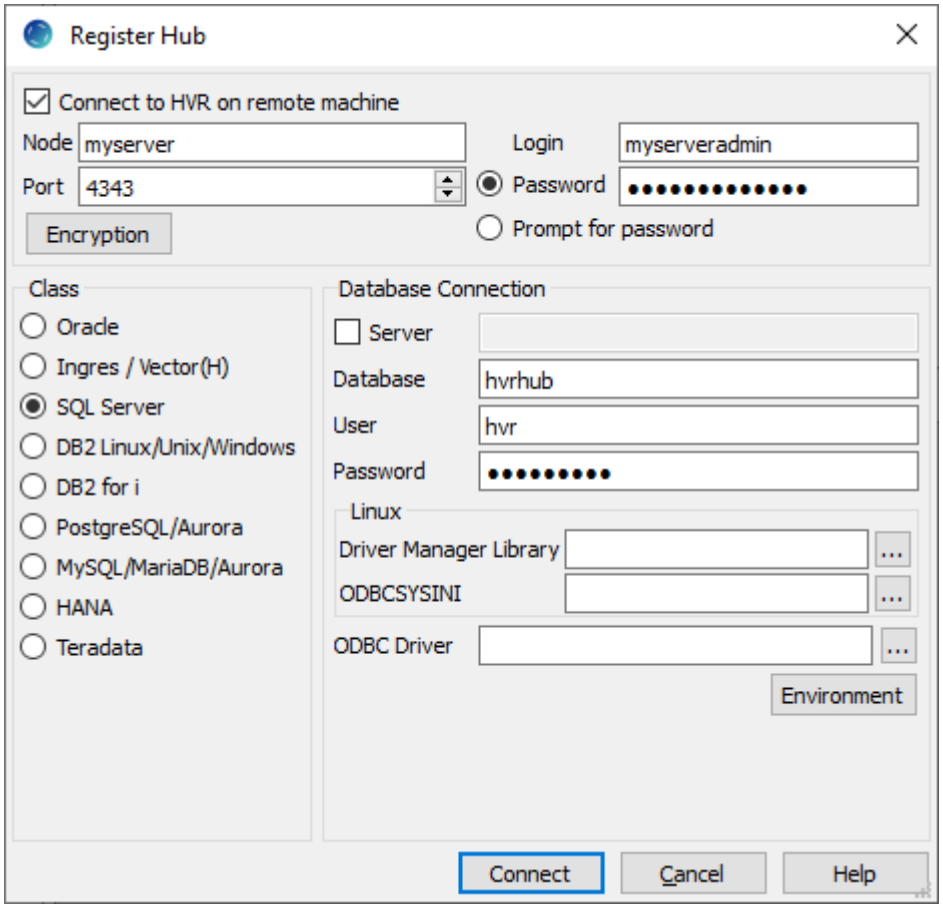
When **HVR GUI** is launched for the first time, the **Register Hub** dialog is displayed automatically. The **Register Hub** dialog can also be accessed from the main menu **File Register Hub**.

Skip steps 1 to 4, if **HVR GUI** is executed directly on the hub machine or if HVR hub is connected to a remote SQL Server database without using the SQL Server protocol. For more information about connecting HVR hub with remote SQL Server database, see section Connecting HVR Hub to a Remote SQL Server Database in Requirements for SQL Server.

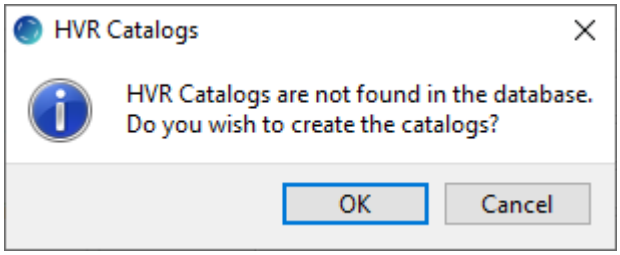
1. Click **Connect to HVR on remote machine**.

   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 **SQL Server** in the **Class** pane.
6. Specify **Database Connection** details. For more information about the **Database Connection** fields, see section Location Connection.
   a. Enter the HVR hub database name in the **Database** field. For example, **hvrhub**.
   b. Enter the SQL Server user name in the **User** field. This username is used to connect HVR to the SQL Server database. For example, **hvr**.
   c. Enter the password for the SQL Server user in the **Password** field.

7. Click **Connect**.



8. Click **OK** 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.



Upon successful connection to the hub database, the navigation tree pane displays the hub machine and the hub database. **Location Configuration**, **Channel Definitions**, and **Scheduler** are displayed under the hub database.

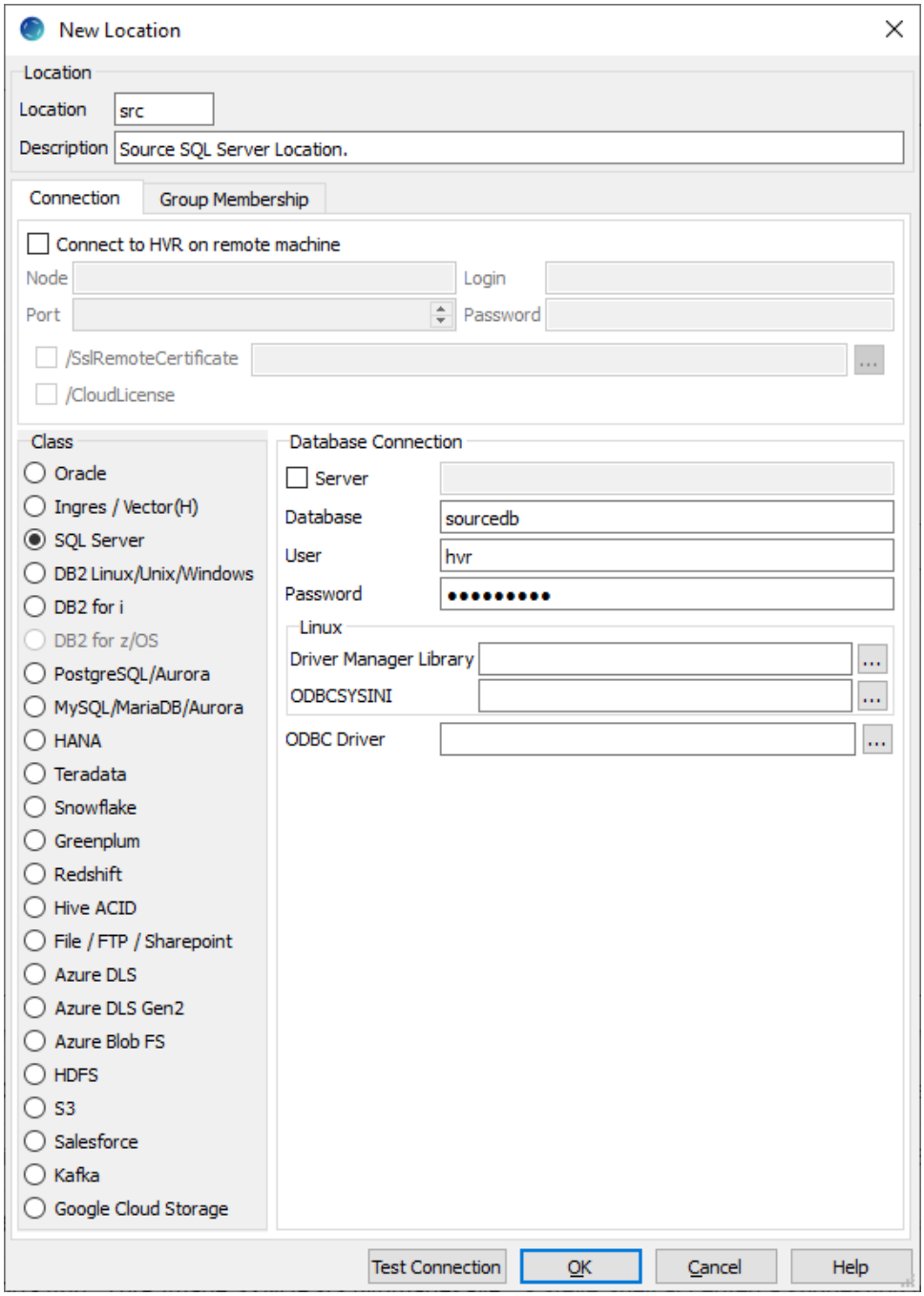
Create Locations

This section describes how to create locations in HVR GUI. A location is a storage place (for example, database or file storage) from where HVR captures (source location) or integrates (target location) changes.

- Create a source location (**src**) connected to the source database (**sourcedb**):
  1. In the navigation tree pane, right-click **Location Configuration** **New Location**.
  2. Enter **Location** name and **Description** for the location.
  3. For this demonstration, it is not required to select **Connect to HVR on remote machine** beca use the source database (**sourcedb**) and the hub are located on the same machine.
  4. Select **SQL Server** in **Class**.
  5. Specify **Database Connection** details. For more information about the **Database Connection** fields, see section Location Connection.
     a. Enter the source database name in **Database**. For example, **sourcedb**.

b. Enter the SQL Server user name in **User**. This username is used for connecting HVR to SQL Server database. For example, **hvr**.
c. Enter the password for the SQL Server user in **Password**.
d. Click **Test Connection** to verify the connection to the source database.

6. Click **OK**.

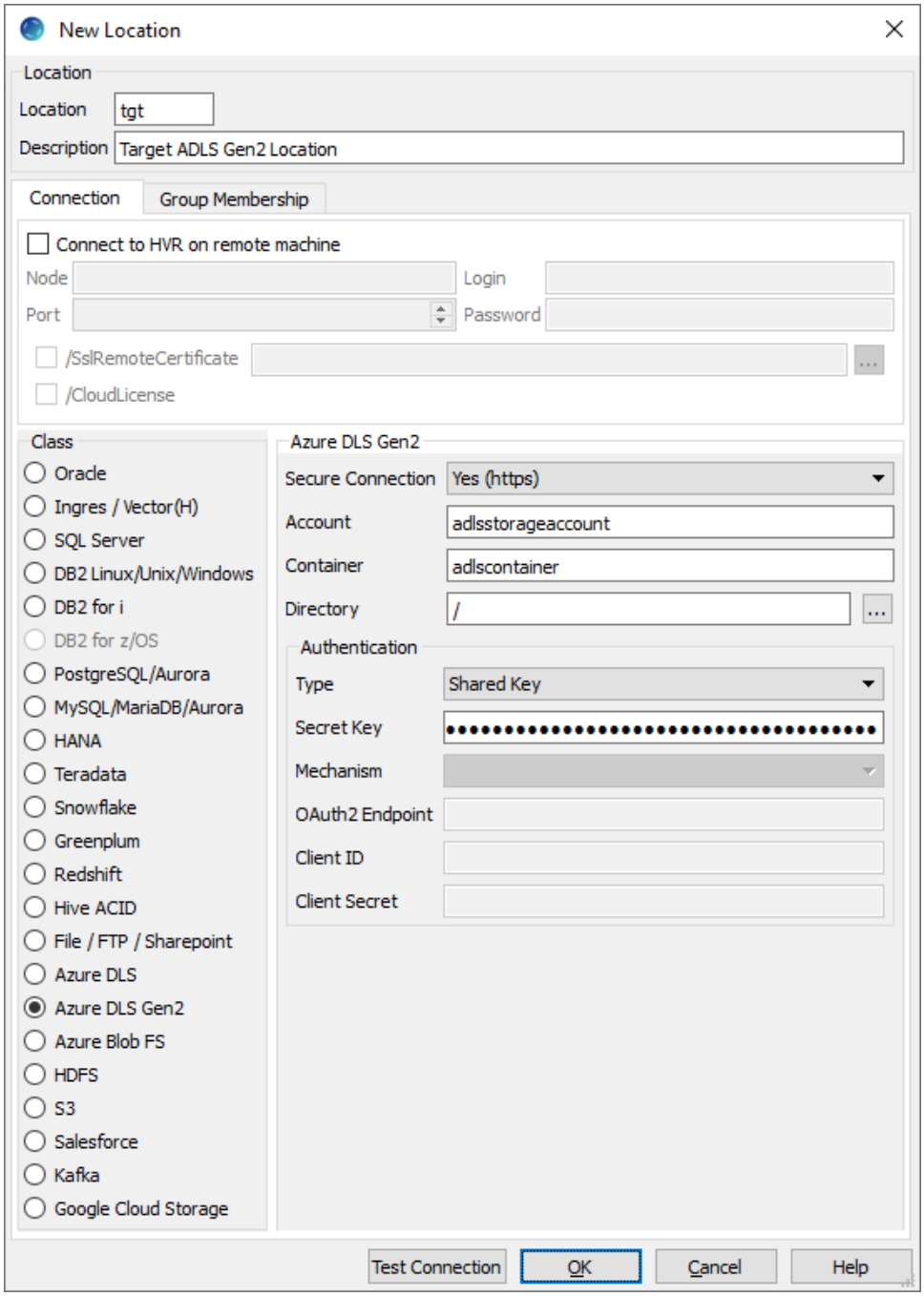


- Create a target location (**tgt**) connected to the target Azure DLS Gen2 storage account (**adlsstorage account**):
  1. In the navigation tree pane, right-click **Location Configuration New Location**.
  2. Enter **Location** name and **Description** for the location.
  3. Select **Azure DLS Gen2** in **Class**.
  4. Specify **Azure DLS Gen2** connection details. For more information about the **Azure DLS Gen2** connection fields, see section Location Connection.
     a. Select the security type for connection in **Secure Connection**. For example, **Yes (https)**.
     b. Enter the Azure DLS Gen2 storage account name in **Account**. For example, **adlsstora geaccount**.
     c. Enter the Azure storage container name in **Container**. For example, **adlscontainer**.

d. Enter the Azure storage directory name in **Directory**. For example, to use the root directory, enter **/**.
e. Specify **Authentication** details:
    i. Select the type of authentication for connecting HVR to the Azure DLS in **Type**. For example, **Shared Key**.
    ii. Enter the access key of the storage account in **Secret Key**.

    If the **Authentication** type selected is **OAuth**, enter the credentials in **OAuth2 Endpoint**, **Client ID**, **Client Secret**.
f. Enter the SQL Server user name in **User**. This username is used for connecting HVR to SQL Server database. For example, **hvr**.
g. Enter the password for the SQL Server user in **Password**.
h. Click **Test Connection** to verify the connection to location database.

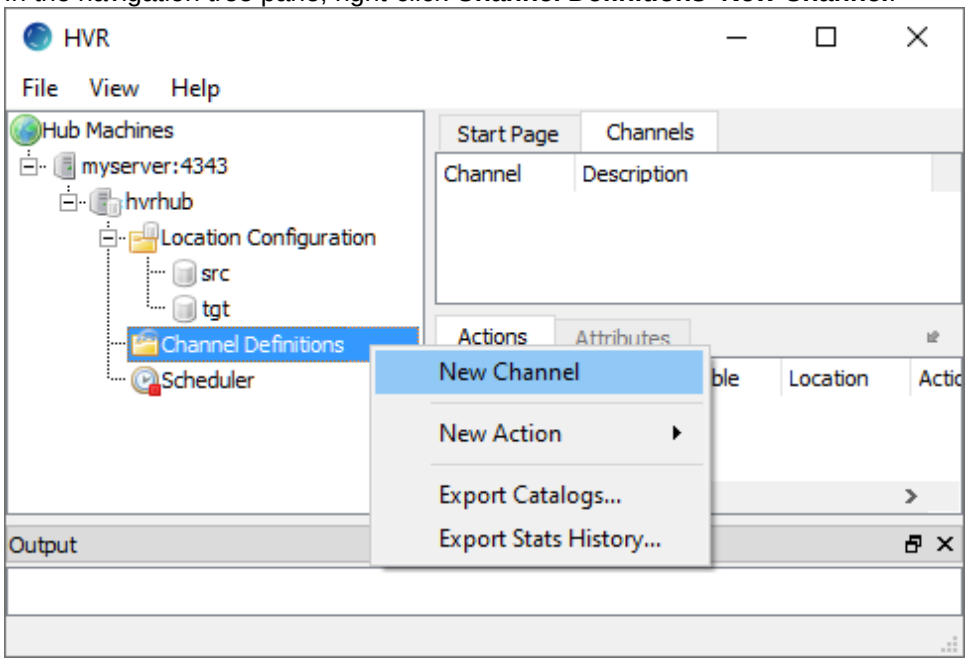
5. Click **OK**.



Create Channel

This section describes how to create a channel (**hvrdemo**) in HVR. A channel groups together the locations and tables that are involved in replication. It also contain actions that control the replication.

1. In the navigation tree pane, right-click **Channel Definitions New Channel**.



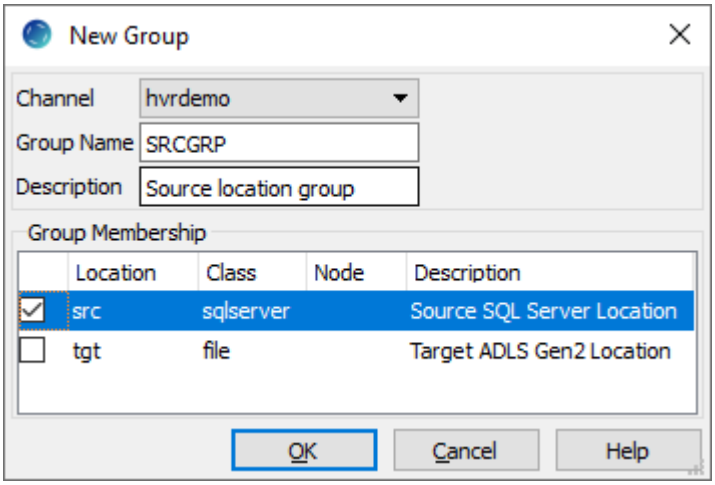
2. In the **New Channel** dialog, enter **Channel** name and **Description** for the channel.
3. Click **OK**.

Create Location Groups

This section describes how to create location groups in a channel. The location groups are used for defining actions 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.

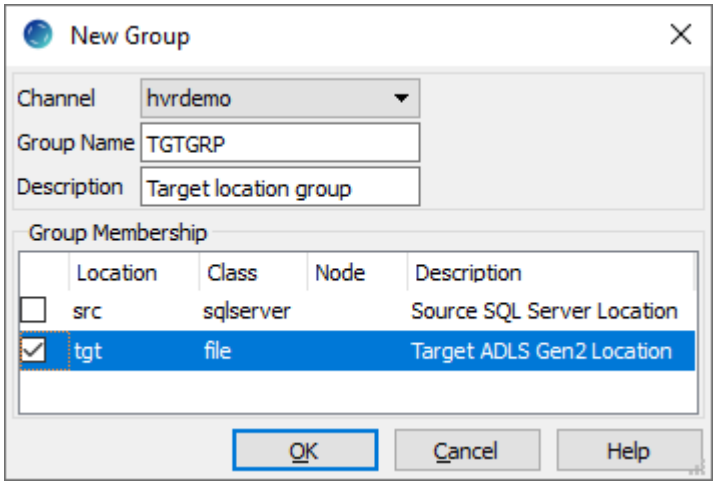
For this demonstration, create one source location group (**SRCGRP**) and one target location group (**TGTGRP**).

1. In the navigation tree pane, click **+** next to the channel (**hvrdemo**).
2. Create source location group (**SRCGRP**):
   a. Right-click **Location Groups New Group**.
   b. Enter **Group Name** and **Description** for the location group.
   c. Select source location (**src**) in **Group Membership**.



d. Click **OK**.

3. Create target location group (**TGTGRP**):
   a. Right-click **Location Groups New Group**.
   b. Enter **Group Name** and **Description** for the location group.
   c. Select target location (**tgt**) in **Group Membership**.

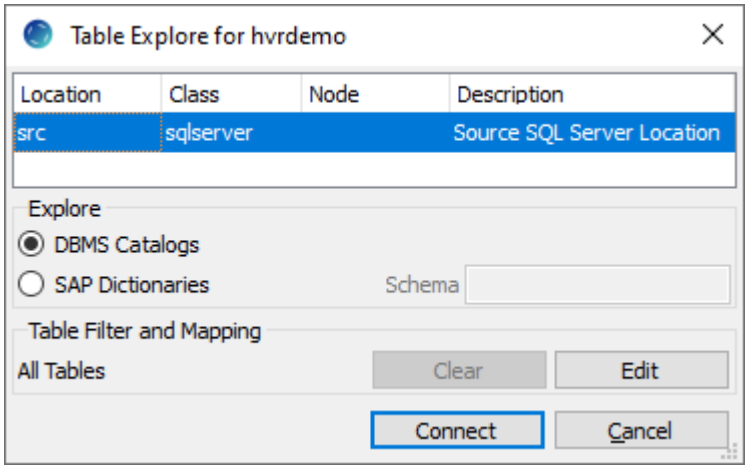


d. Click **OK**.

Select Table(s)

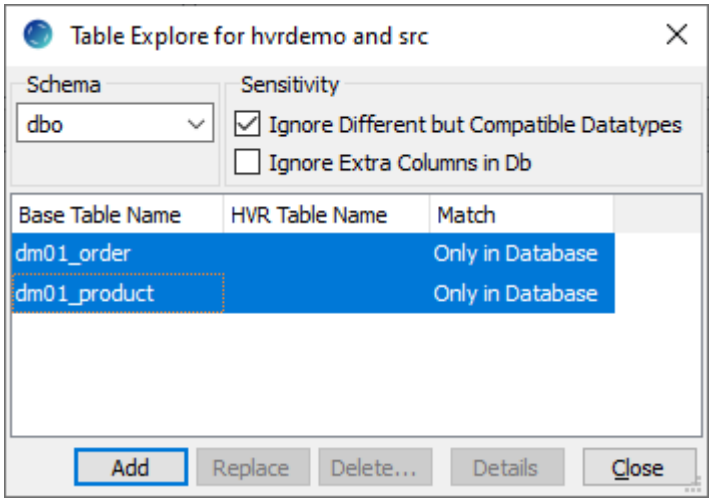
This section describes how to select the tables (**dm01_product** and **dm01_order**) from the source location for replication. The **Table Explore** dialog allows you to select schema(s) and/or table(s) for replication.

1. Right-click **Tables Table Explore**.
2. Select source location (**src**) from the list.



3. Click **Connect**.
4. Select tables from the **Table Explore** dialog. Use the **Shift** or **Ctrl** key to select multiple tables or **Ctr l+A** to select all tables.

5. Click **Add** to add the selected tables to the channel.

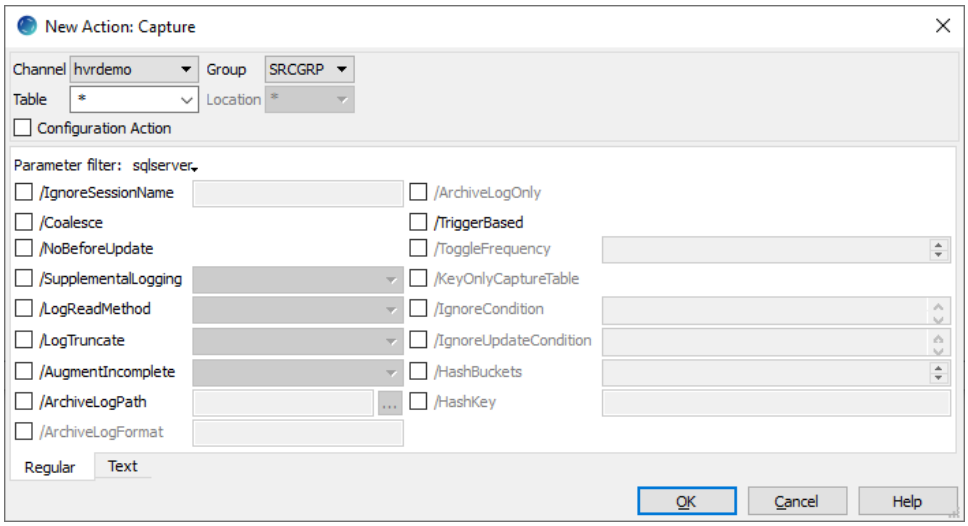


6. Click **OK** in **HVR Table Name** dialog.
7. Click **Close** in **Table Explore** dialog.

Define Actions

This section describes how to define actions on the location groups (**SRCGRP** and **TGTGRP**). Actions define the behavior of a replication activity.

1. Define action **Capture** to capture changes from all tables in the source location group.
   a. Right-click source location group **SRCGRP** **New Action** **Capture**.

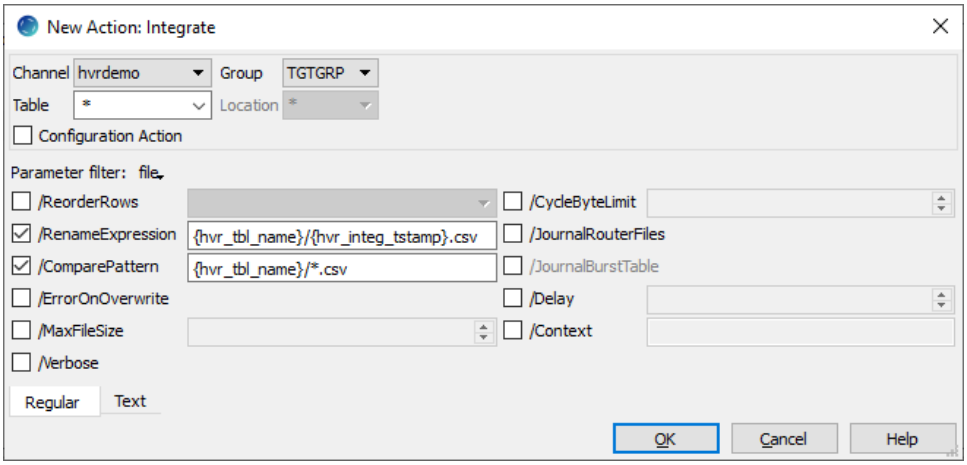


   b. Click **OK**.
2. Define action **Integrate** to integrate changes into all tables in the target location group.
   a. Right-click target location group **TGTGRP** **New Action** **Integrate**.
   b. Select parameter **/RenameExpression**. This parameter definition is required for segregation and naming of files integrated into target location.
      i. Enter **{hvr_tbl_name}/{hvr_integ_tstamp}.csv**.

      For each table in the source, a separate folder (with the same name as the table name) is created in the target location, and the files replicated for each table are saved into these folders. The files are named with a timestamp of the moment when the file was integrated into the target location. For other expressions that can be used with this parameter, see **Integrate /RenameExpression**.
   c. Select parameter **/CompareExpression**. This parameter definition is required for performing **HVR Compare** (to verify the replication) against a file location.

      Direct file compare does not support Avro, Parquet or JSON file formats.
      i. Enter **{hvr_tbl_name}/*.csv**.

d. Click **OK**.

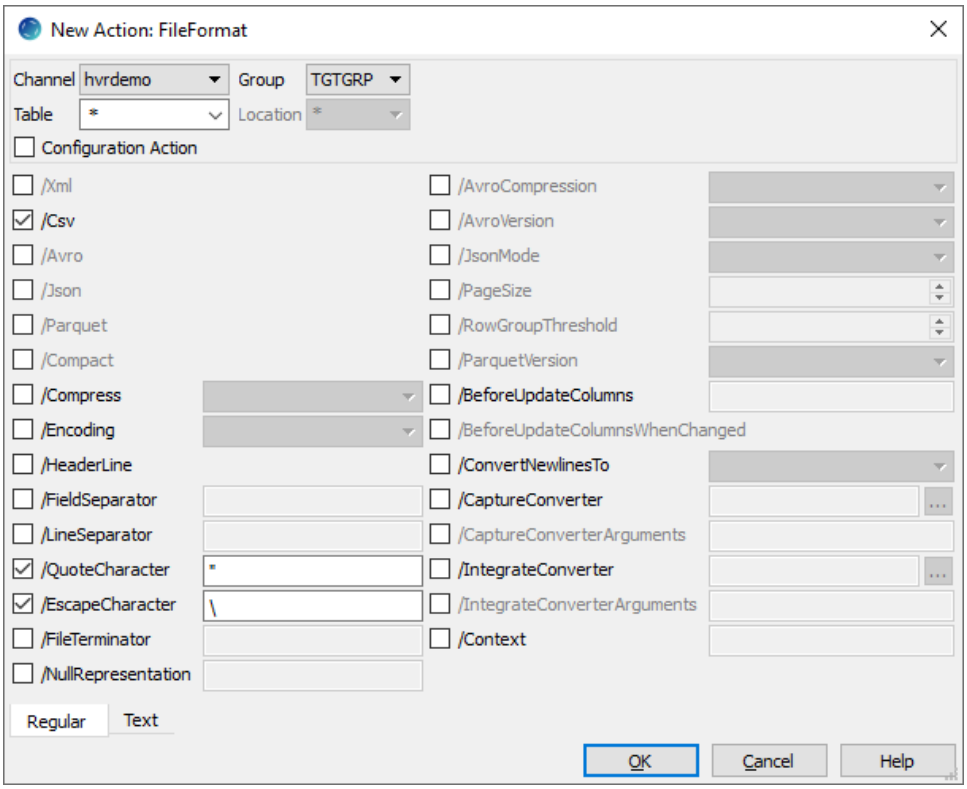


3. Define action **FileFormat** to specify the target file format such as XML, CSV, Avro, or JSON. For this demonstration, the target file format is CSV.

   a. Right-click target location group **TGTGRP** **New Action** **FileFormat**.
   b. Select parameter **/Csv** to set the target file format as CSV.
   c. Select parameter **/QuoteCharacter** to escape any delimiters (e.g. comma) present in a column. If a comma is present in the source location's table, then the **QuoteCharacter** defined here is added to the beginning and end of the column value in the target location CSV file to indicate that the content enclosed within the **QuoteCharacter** belongs to one column.
      i. Enter **"** (double quotes).

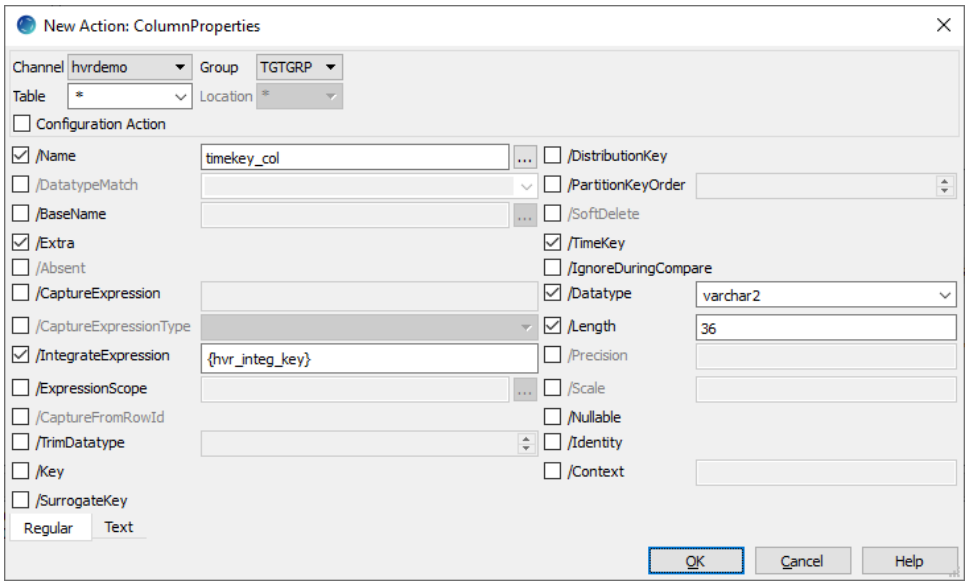
         This will escape any delimiters (e.g. comma) present in any column. For example, if there is an address column containing value **mystreet, mycity, mycountry**, then in the target CSV file it is replicated as **"mystreet, mycity, mycountry"**.
   d. Select parameter **/EscapeCharacter** to escape the **QuoteCharacter** defined. If the value in the column for which **/QuoteCharacter** is defined already contains a character matching the **QuoteCharacter**, then that character is escaped using the **EscapeCharacter**.
      i. Enter **** (backslash).

         For example, with **/QuoteCharacter** and **/EscapeCharacter** defined, if there is an address column containing value **mystreet, "mycity", mycountry**, then in the target CSV file it is replicated as **"mystreet, \"mycity\", mycountry"**.



e. Click **OK**.

4. Define action **ColumnProperties** to define properties for a column being replicated. By default, for file-based target locations, HVR does not replicate the **delete** operation performed at the source location. To integrate the **delete** operation, an extra column for **TimeKey** needs to be added in the target location. For this demonstration, the **TimeKey** value is generated based on the **/IntegrateExpression** value **{hvr_integ_key}** (a 16-byte string value (hex characters) which is unique and continuously incremented for all rows integrated into the target location).
   a. Right-click target location group **TGTGRP New Action ColumnProperties**.
   b. Select parameter **/Name**. This parameter defines the name for the extra column in the target location.
      i. Enter **timekey_col**.
   c. Select parameter **/Extra**. This parameter defines that this is an extra column in the target location (a column which is not present in the source location).
   d. Select parameter **/IntegrateExpression**. This parameter defines the expression to be used for generating the **TimeKey** value.
      i. Enter **{hvr_integ_key}**.
   e. Select parameter **/TimeKey**. This parameter defines that this is a **TimeKey** column.
   f. Select parameter **/Datatype**. This parameter defines the data type for the **TimeKey** column.
      i. Select **varchar2.**
   g. Select parameter **/Length**. This parameter defines the data type length for the **TimeKey** column.
      i. Enter **36**.

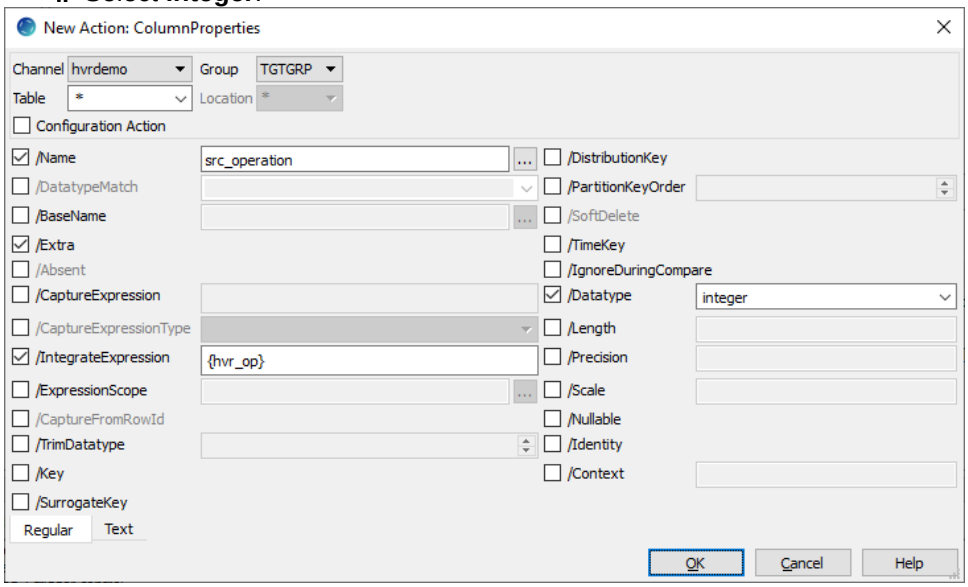


h. Click **OK**.

5. Define another action **ColumnProperties** to define properties for a column being replicated. This action definition adds the source operation type (using **hvr_op**) information in the target location.

   This action definition is required for performing **HVR Compare** if **ColumnProperties** **/TimeKey** colu mn is defined on a target file location.

   a. Right-click target location group **TGTGRP** **New Action** **ColumnProperties**.
   b. Select parameter **/Name**. This parameter defines the name for the extra column in the target location.
      i. Enter **src_operation**.
   c. Select parameter **/Extra**. This parameter defines that this is an extra column in the target location (a column which is not present in the source location).
   d. Select parameter **/IntegrateExpression**. This parameter defines the expression to be used for generating the information about source operation type.
      i. Enter **{hvr_op}**.
   e. Select parameter **/DataType**. This parameter defines the data type for this extra column in the target location.
      i. Select **integer**.



   f. Click **OK.**

The **Actions** pane only displays actions related to the object selected in the navigation tree pane. Click the channel name (**hvrdemo**) to view all the actions defined for the selected channel.

Optional Actions

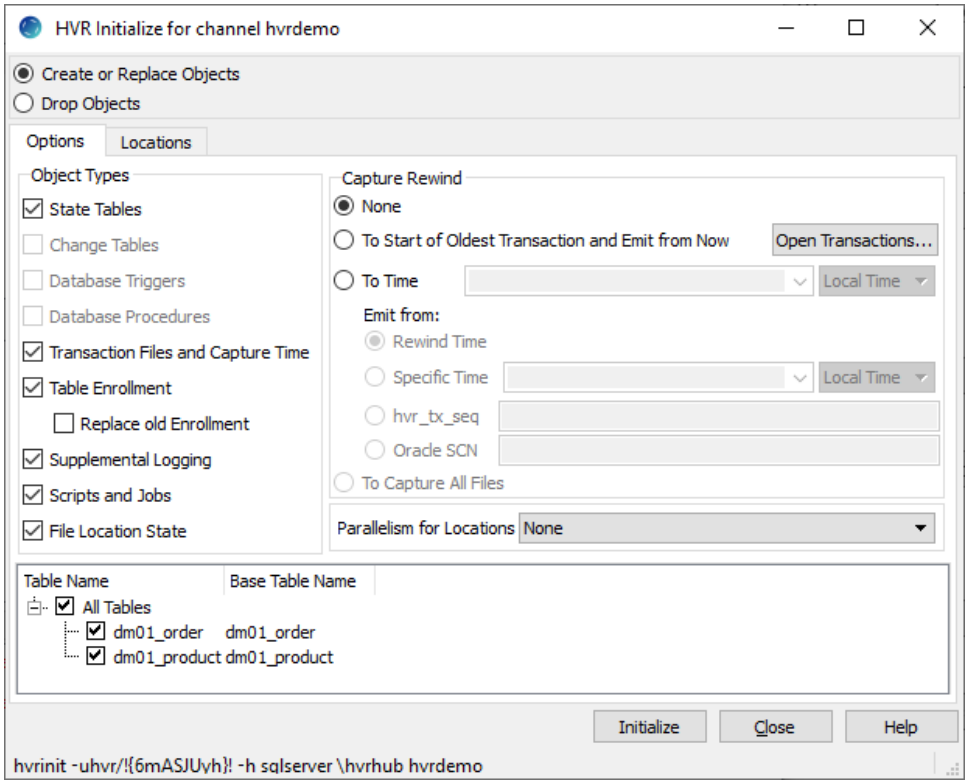
The following actions can be defined optionally for this replication:

Group	Table	Action	Annotation
TGTGRP	*	**Integrate /ReorderRows= BATCH_BY_TABLE**	Controls the order in which changes are written to files. This action definition is required only if **Integrate /RenameExpression** does not contain a substitution which depends on the table name. For more information, see **Integrate /ReorderRows**.
TGTGRP	*	**FileFormat /NullRepresentation**	This additional parameter can be defined in **FileFormat** to represent a NULL column.
TGTGRP	*	**ColumnProperties /Name="src_timestamp", /Extra, /IntegrateExpression={hvr_cap_tstamp}, /DataType=timestamp /Precision=3**	This action definition is optional because HVR does not require this for performing **Integrate** or **Refresh** but for further processing, this timestamp may be used to know what has been processed. This action definition can be used to add an extra column in a target file which can be a source commit timestamp (**{hvr_cap_tstamp}**) or integrate timestamp (**{hvr_integ_tstamp}**). For other expressions, see **ColumnProperties /IntegrateExpression**.

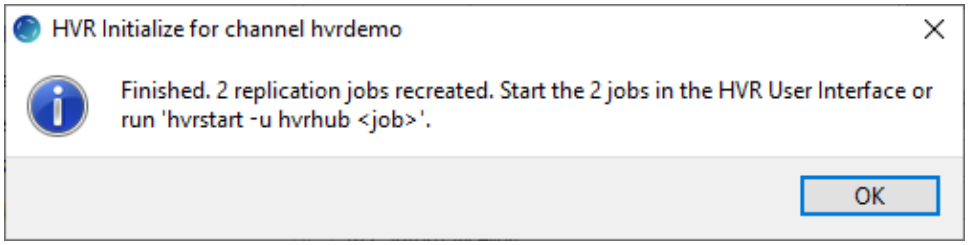
Initialize

This section describes how to initialize the replication. **HVR Initialize** first checks the channel and creates the replication jobs in the **HVR Scheduler**.

1. Right-click channel **hvr_demo HVR Initialize**.
2. Select **Create or Replace Objects** in the **HVR Initialize** dialog.
3. Click **Initialize**.



4. Click **OK**.



5. Click **Close** in the **HVR Initialize** dialog.

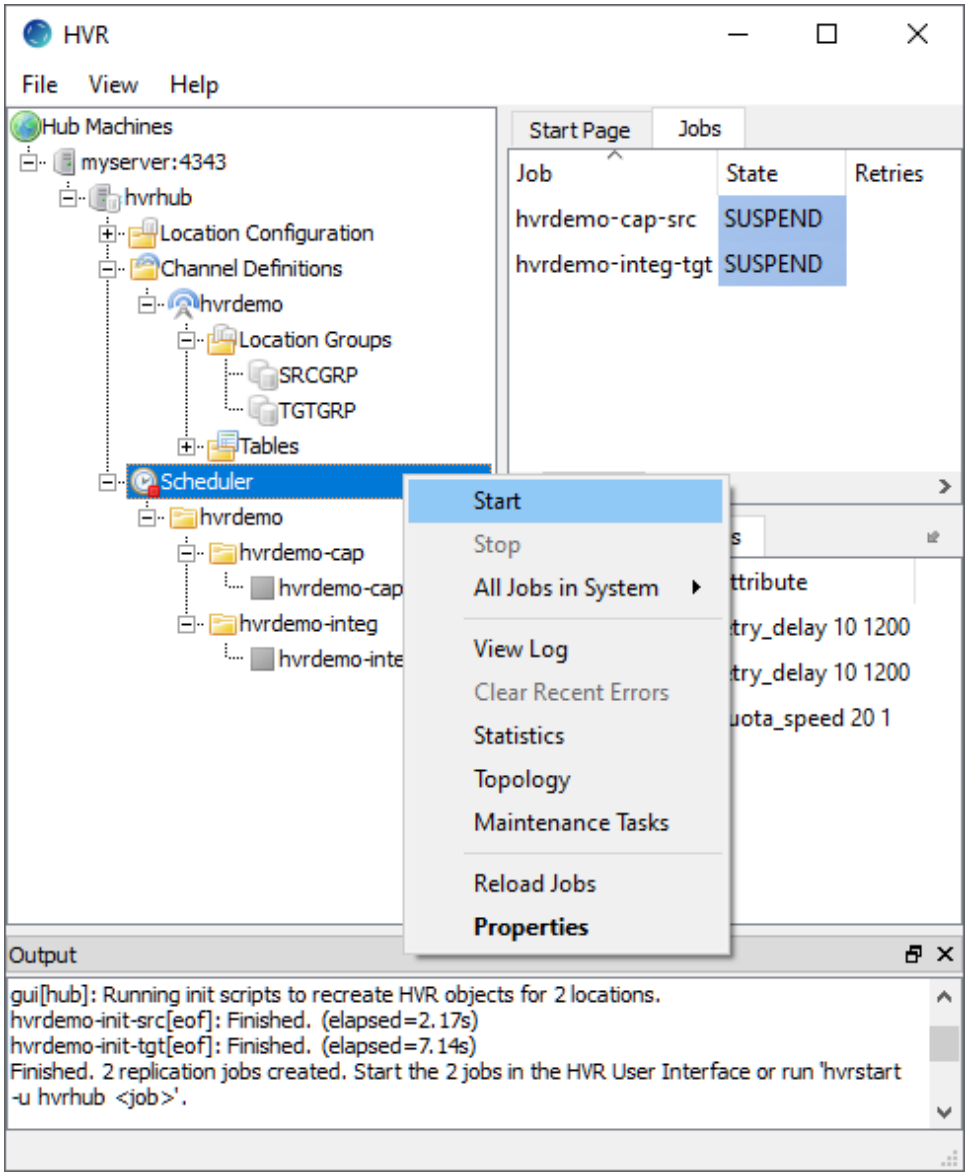
Click the **Scheduler** node in the navigation tree pane to view the capture and integrate jobs in the **Jobs** pa ne.

For more information about initiating replication in HVR, see section Replication Overview.

Start Scheduler

This section describes how to start the **HVR Scheduler**. The **HVR Scheduler** is a process which runs jobs defined in the catalog table **HVR_JOB**. This catalog table can be found in the hub database. On Unix or Linux, the **HVR Scheduler** runs as a daemon. On Windows, the **HVR Scheduler** runs as a system service.

1. In the navigation tree pane, right-click **Scheduler Start**.



2. On Windows, the following steps are required to create the **HVR Scheduler** system service.

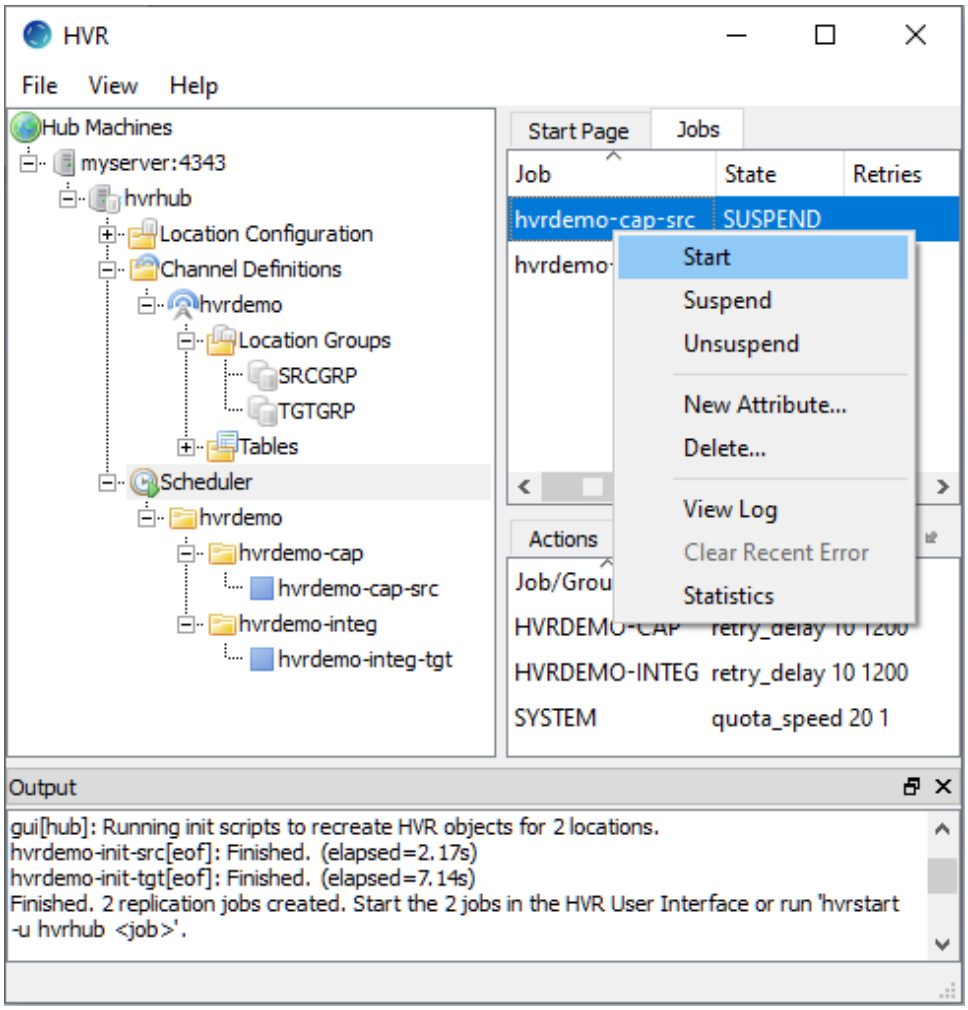
    a. Click **Create...** in the prompt asking to create the service **hvrscheduler_hvrhub**.

    SC-Hvr-QSG-ADLSG2_SchedulerWinService
    b. Select **Local System Account ('SYSTEM')** in the **Create Windows Service** dialog.
    c. Click **Create**.

Start Capture Job

This section describes how to start the job for capturing changes from the source location (**src**). By starting the **Capture** job in **HVR Scheduler**, HVR begins capturing all changes since the moment **HVR Initialize** was executed. This 'capture begin moment' can be modified using option **Capture Rewind** available in the **Advanced Options** tab of the **HVR Initialize** dialog.

1. In the navigation tree pane, click **Scheduler**.
2. Start the capture job. In the **Jobs** pane, right-click capture job **hvrdemo-cap-src** **Start**.



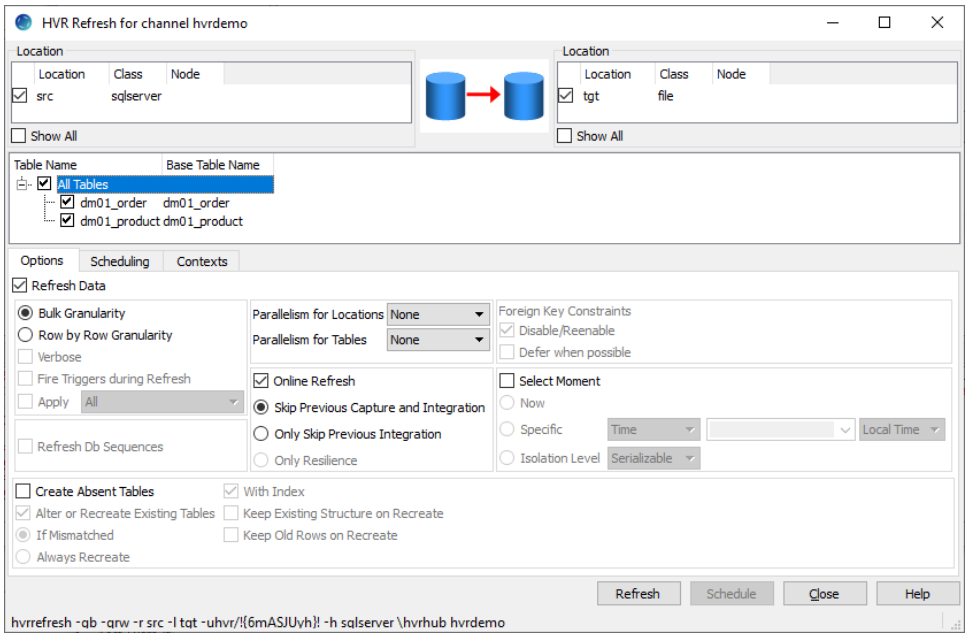
3. Click **Yes** in the **Start** dialog.

On starting the capture job (**hvrdemo-cap-src**) successfully, the status of the job changes from **SUSPEND** to **RUNNING**.

Refresh

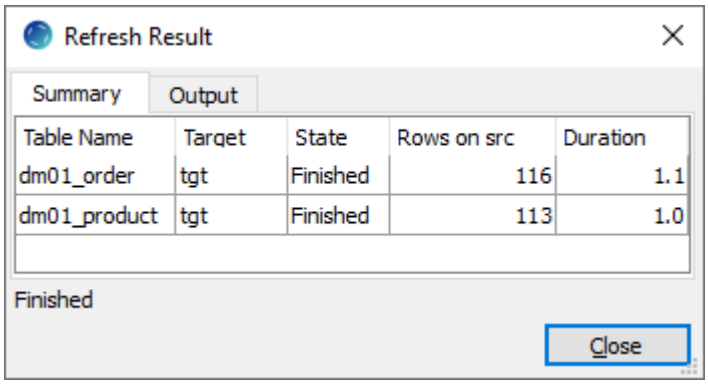
This section describes how to perform initial load into the target database. **HVR Refresh** copies all existing data from the source location (**src**) to the target location (**tgt**) and optionally creates new tables and keys in the target location.

1. In the navigation tree pane, right-click channel **hvrdemo HVR Refresh**.
2. Select the table(s) that needs to be copied from the source location to the target location.
3. Click **Refresh**.



4. Click **Yes** to begin **HVR Refresh**.

   When the refresh is completed, the **Refresh Result** dialog displays the total number of rows replicated from the selected tables.



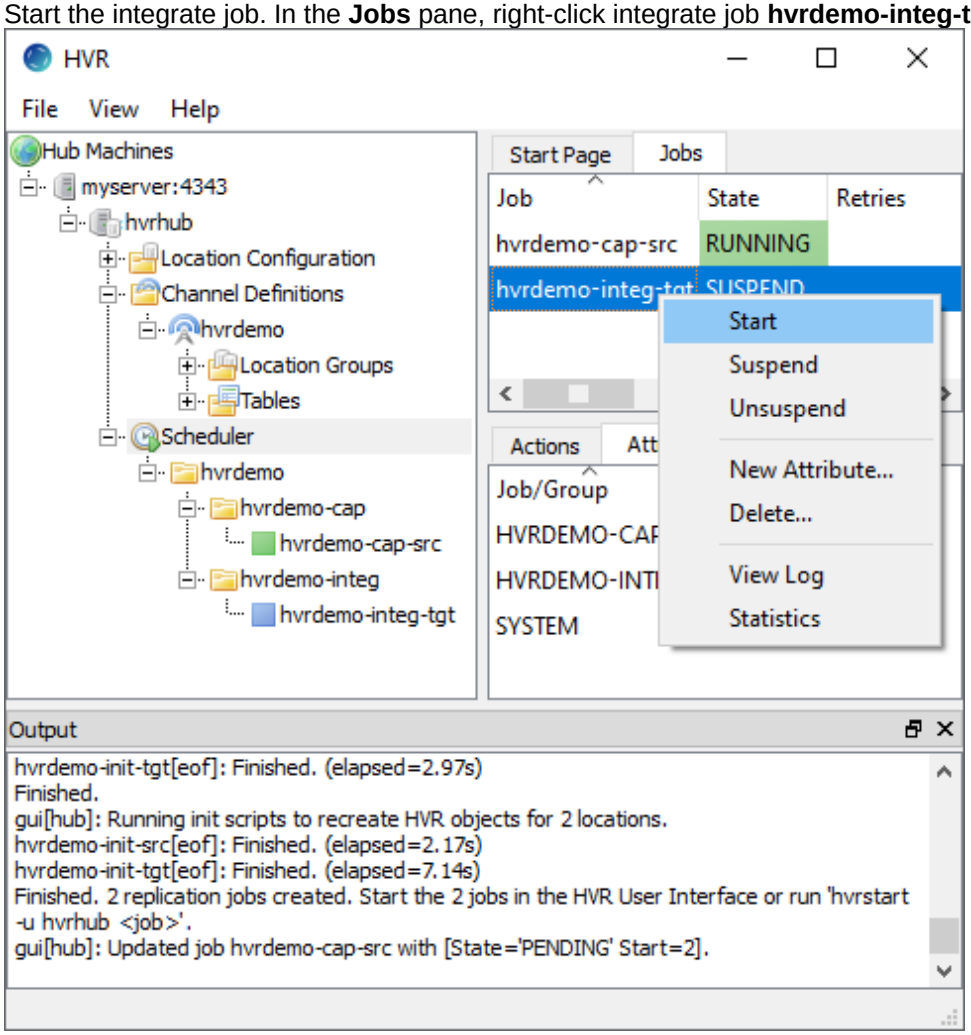
5. Click **Close** in the **Refresh Result** dialog and then in the **HVR Refresh** dialog.

Start Integrate Job

This section describes how to start the job to integrate changes into the target location (**tgt**).

1. In the navigation tree pane, click **Scheduler**.

2. Start the integrate job. In the **Jobs** pane, right-click integrate job **hvrdemo-integ-tgt** **Start**.



3. Click **Yes** in the **Start** dialog.

On starting the integrate job (**hvr_demo-integ-tgt**) successfully, the status of the job changes from **SUSPEND** to **RUNNING**.

Verify Replication

This section describes two methods for verifying the HVR's replication activity.

- Viewing Log File
- Using HVR Compare

Viewing Log File

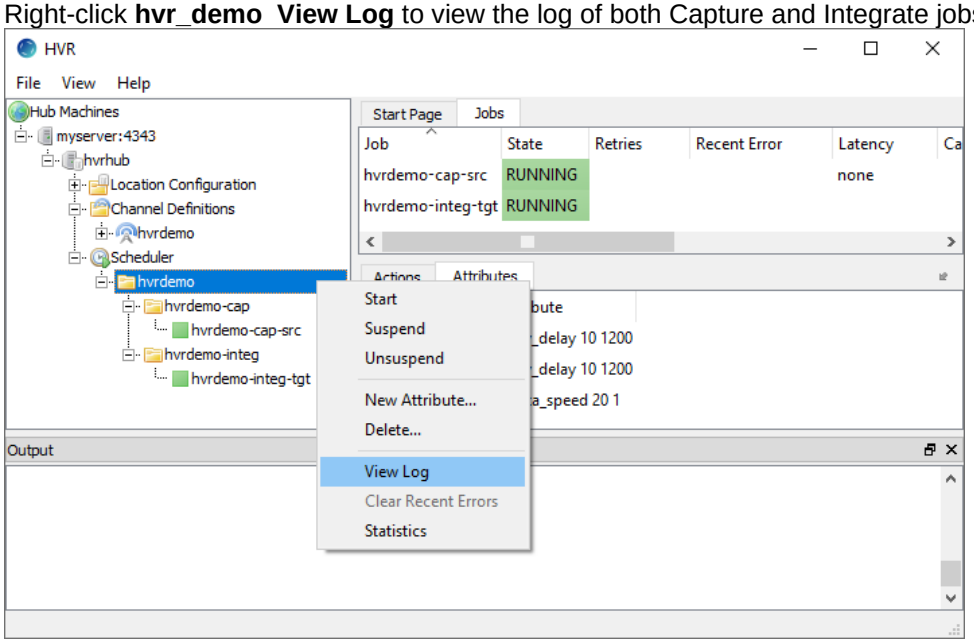
HVR creates separate log files for the hub, channel (**hvrdemo**), and for each replication jobs (**hvrdemo-cap-src** and **hvrdemo-integ-tgt**). These log files contain the details of the changes captured and integrated.

Replication can be verified by inspecting the channel log file.

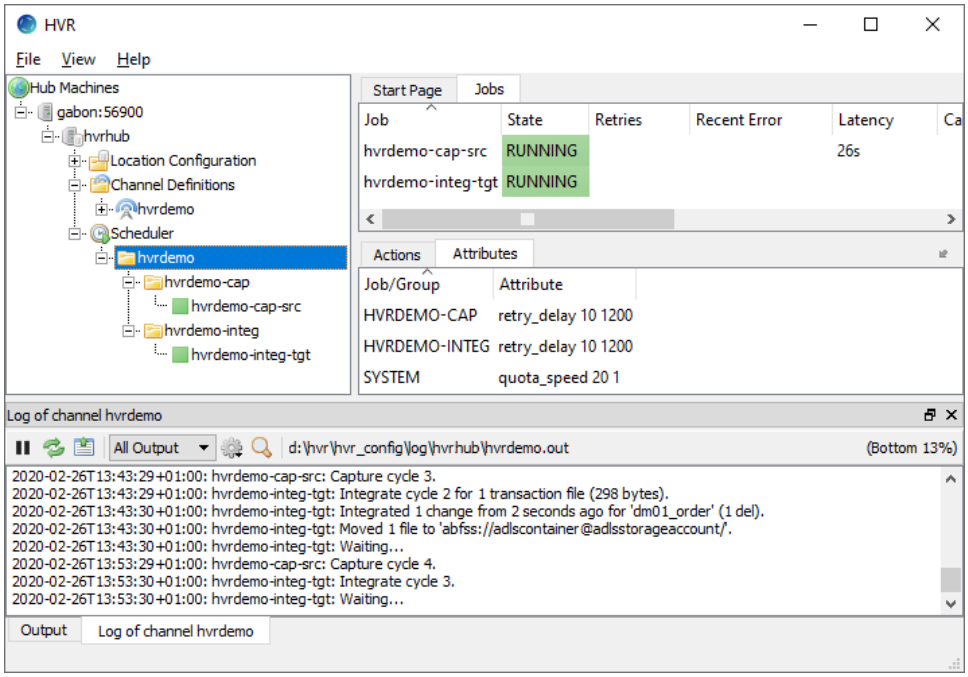
To view the replication activity log:

1. In the navigation tree pane, click **+** next to the **Scheduler**.

2. Right-click **hvr_demo View Log** to view the log of both Capture and Integrate jobs.



3. The logs for both Capture and Integrate jobs is displayed in the logs pane (Log of channel hvrdemo) at the bottom of the screen.



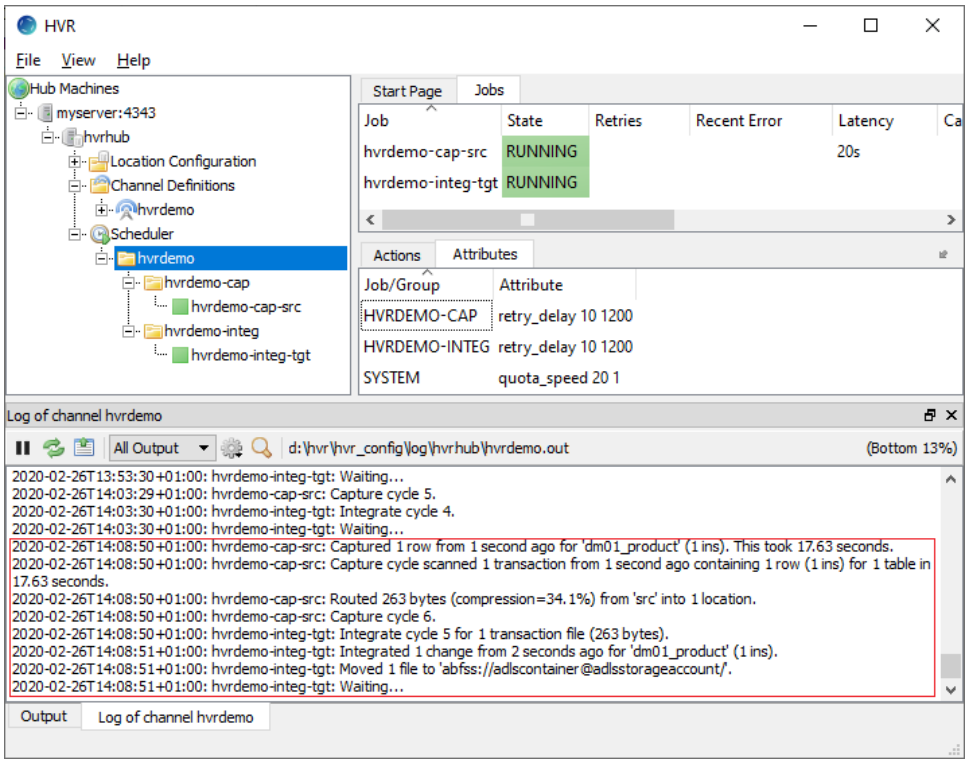
The directory path for the HVR log files is displayed in the log tab.

Right-clicking on a particular job and selecting **View Log** displays logs related to that job alone.

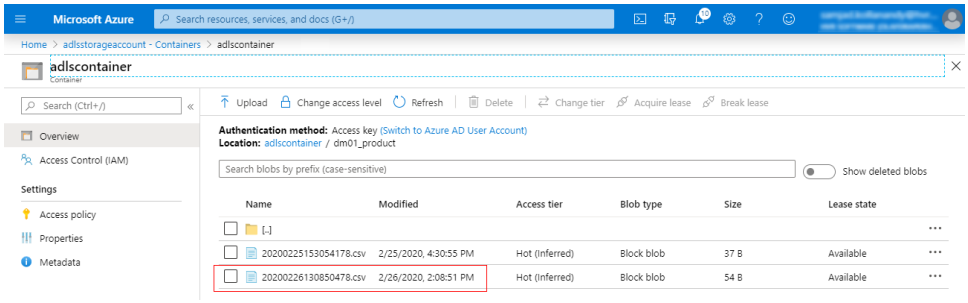
4. Insert, update, or delete value(s) in the source location database. For example:

```
insert into dm01_product
values (101, 91, 'Pencil')
```

5. The output log is updated and indicates that the change is captured from the source location and integrated into the target location:



6. This replication can also be verified by manually checking the ADLS storage directory (**dm01_product**).



Using HVR Compare

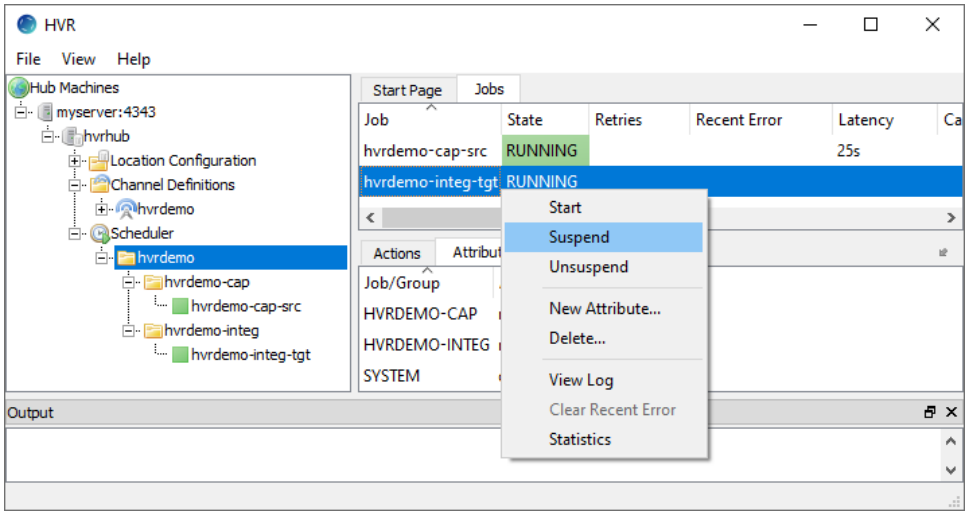
HVR Compare allows you to verify the replication activity by comparing the data in source and target locations. For file locations without Hive External Tables, HVR supports only Direct File Compare.

To run **HVR Compare** against a file location, ensure that the parameter **Integrate /ComparePattern** is defined.

To compare the source and target locations:

1. Suspend the integrate job (**hvrdemo-integ-tgt**),
    a. In the navigation tree pane, click **Scheduler**.

b. In the **Jobs** pane, right-click integrate job **hvrdemo-integ-tgt Suspend**.



c. Click **Yes** in the **Suspend** dialog.

2. Insert, update, or delete value(s) in the source location database. For example:

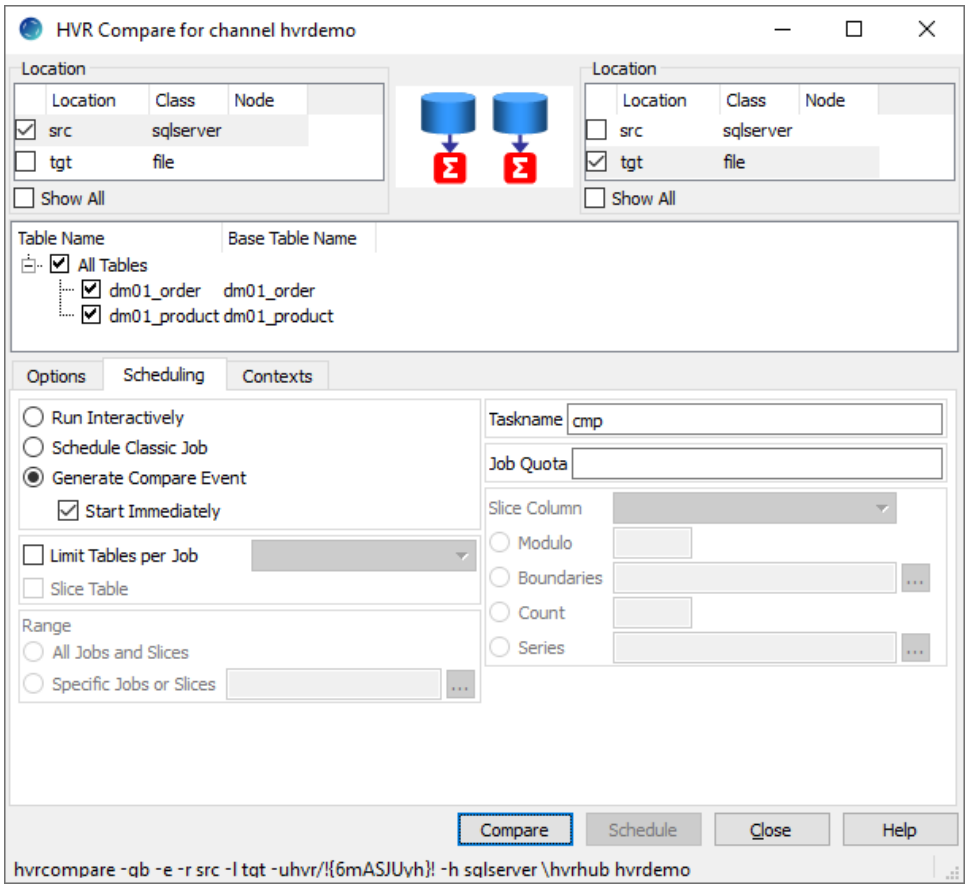
```
insert into dm01_product
values (101, 91, 'Pencil')
```

3. Execute **HVR Compare**:

a. In the navigation tree pane, right-click channel **hvrdemo HVR Compare**.

b. Select the source location (**src**) on the left side and the target location (**tgt**) on the right side.

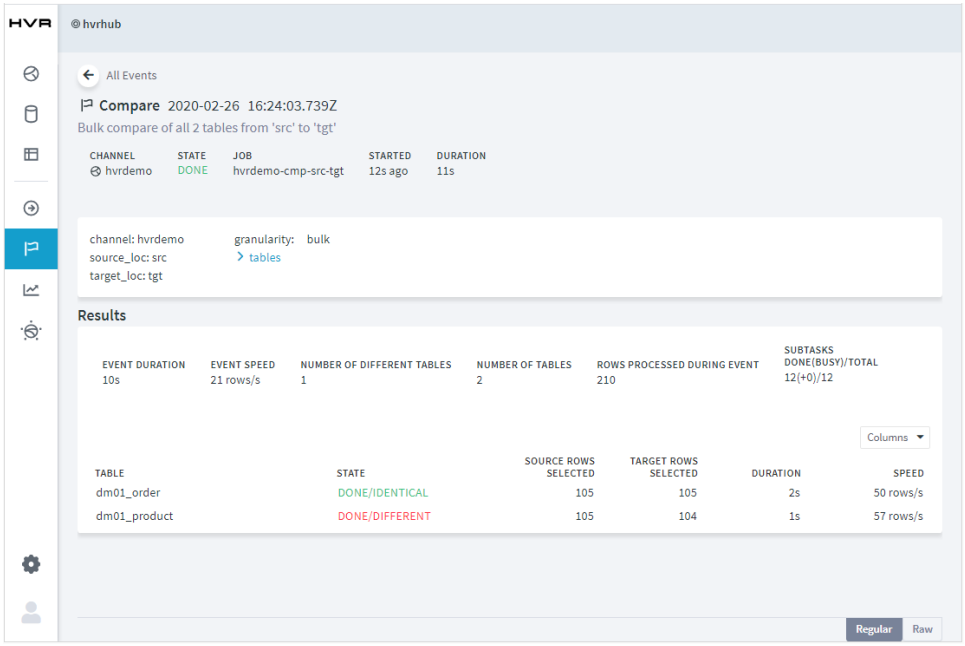
c. Select **Generate Compare Event** in the **Scheduling** tab



d. Click **Compare**.

e. On completion, the compare result is displayed in the web browser. If the **State** column displays **DONE/DIFFERENT**, it indicates the data in the source and target locations are not

identical.



4. Start Integrate Job (**hvrdemo-integ-tgt**), the changes made in source location (in step 2) will be integrated to the target location now.
5. Execute **HVR Compare** again (step 3). In the compare result screen, if the **State** column displays **D ONE/IDENTICAL**, it indicates the changes are replicated successfully.