Requirements for SQL Server

Contents

SQL Server

Capture	Hub	Integrate
✓	✓	✓

This section describes the requirements, access privileges, and other features of HVR when using SQL Server for replication.

For the Capabilities supported by HVR on SQL Server, see Capabilities for SQL Server.

For information about the supported data types and mapping of data types in source DBMS to the corresponding data types in target DBMS or file format, see Data Type Mapping.

For instructions to quickly setup replication using SQL Server, see Quick Start for HVR - SQL Server.

Supported Editions

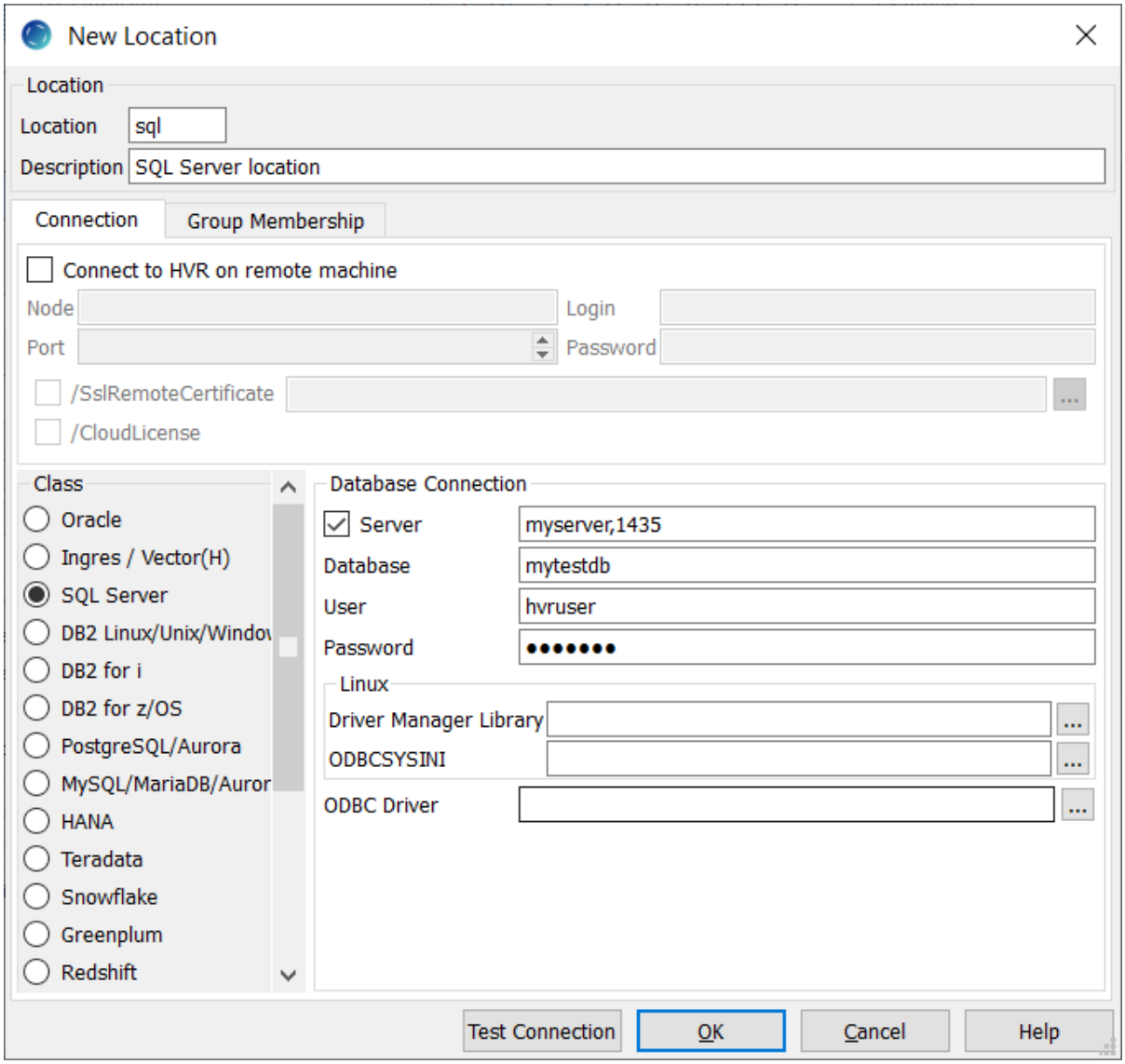
HVR supports the following SQL Server editions:

- SQL Server Developer Edition
- SQL Server Enterprise Edition
- SQL Server Standard Edition

For information about compatibility and supported versions of SQL Server with HVR platforms, see Platform Compatibility Matrix.

Location Connection

This section lists and describes the connection details required for creating SQL Server location in HVR.



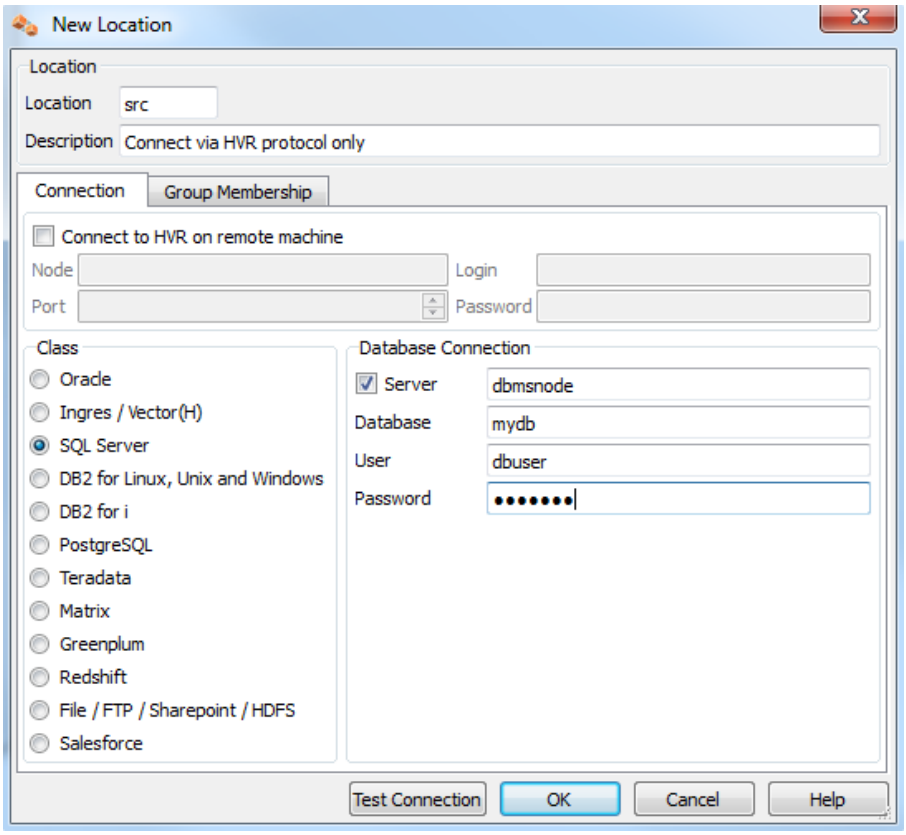
Field	Description

Server	The name of the machine on which SQL Server is running and the Port number or SQL Server instance name. The following formats are supported for this field: • server name : Specify only server name and HVR will automatically use the default port to connect to the server on which SQL Server is running. **Example:**myserver • *server name*,*port number* : Specify server name and port number separated by a comma (,) to connect to the server on which SQL Server is running. This format is required when using custom port for connection. **Example:** myserver,1435 • *server name**server instance name* : Specify server name and server instance name separated by a backslash (\) to connect to the server on which SQL Server is running. This format is not supported on Linux. Also, it is not supported when **Integrate /NoTriggerFiring** is to be defined for this location. **Example:** myserver\HVR5048 For more details on the connection methods, see Connecting HVR Hub to a Remote SQL Server Database.
Database	The name of the SQL Server database which is to be used for replication. **Example:** mytestdb
User	The username to connect HVR to SQL Server **Database**. **Example:** hvruser The login/user account used for connecting HVR to SQL Server database engine should be defined with SQL Server Authentication or Windows Authentication. If Windows Authentication is used, the **User** and **Password** field should be left blank (empty).
Password	The password of the **User** to connect HVR to SQL Server **Database**.
Linux	
Driver Manager Library	The optional directory path where the ODBC Driver Manager Library is installed. This field is applicable only for Linux/Unix operating system. For a default installation, the ODBC Driver Manager Library is available at **/usr /lib64** and does not need to be specified. However, when UnixODBC is installed in for example **/opt/unixodbc** the value for this field would be **/opt/unixodbc/lib**.
ODBCSYSINI	The optional directory path where **odbc.ini** and **odbcinst.ini** files are located. This field is applicable only for Linux/Unix operating system. For a default installation, these files are available at **/etc** and do not need to be specified. However, when UnixODBC is installed in for example **/opt/unixodbc** th e value for this field would be **/opt/unixodbc/etc**.
ODBC Driver	The user defined (installed) ODBC driver to connect HVR to the SQL Server **Data base**. It is recommended to leave this field empty, HVR will automatically load the correct driver for your current platform. Otherwise, select one of the available **SQL Server Native Client** options.

Connecting HVR Hub to a Remote SQL Server Database

For connecting HVR hub machine to a remote SQL Server database, the following three methods are available:

All of the following connection options can be used for both **Capture** and **Integrate**. Specifically: HVR's log-based capture can get changes from a database without HVR's executables being physically installed on the source machine.

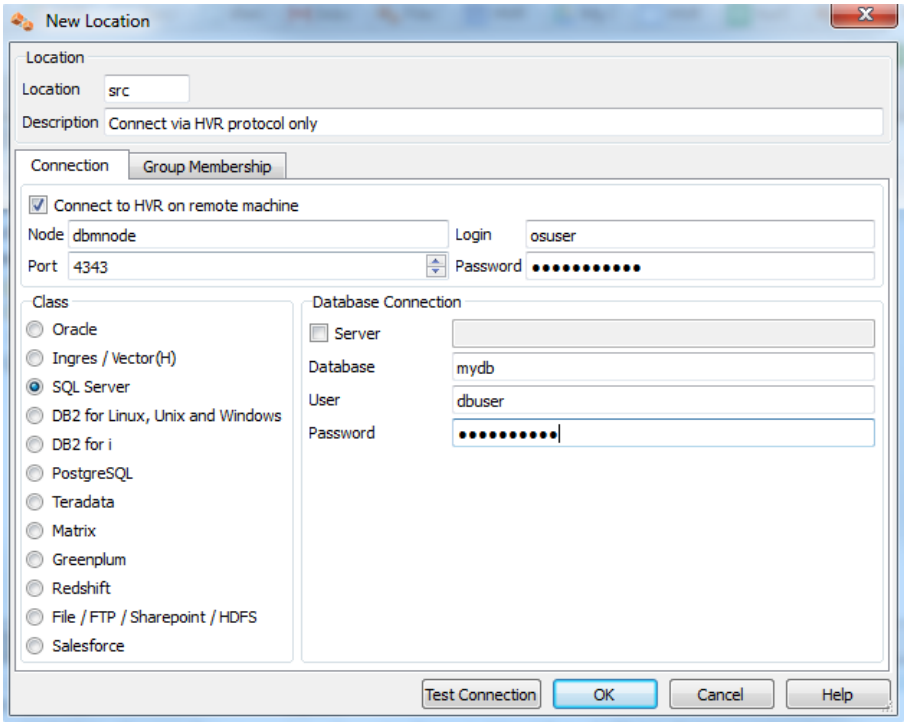


Method 1

Connect to a SQL Server database using the SQL Server protocol (equivalent to TNS).

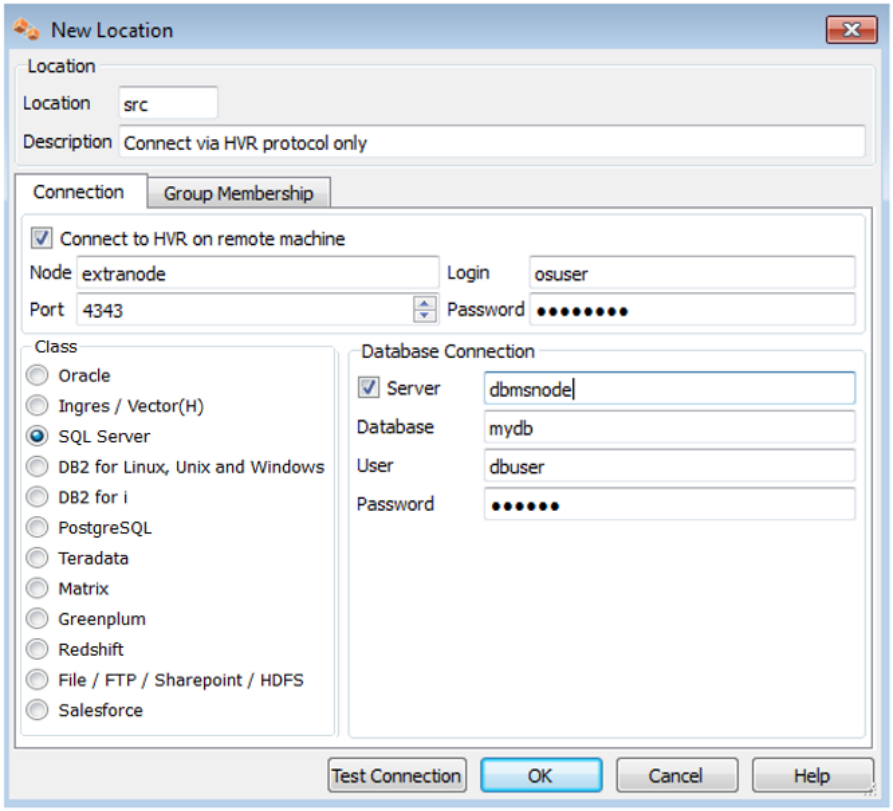
To use this method, Microsoft SQL Server Native Client should be installed on the machine from which HVR will connect to SQL Server database.

SQL Server Native Client can be downloaded from this Microsoft download page and the instructions for installation is available in Microsoft documentation - Installing SQL Server Native Client.



Method 2

Connect to a HVR installation running on the machine containing the SQL Server database using HVR's protocol on a special TCP/IP port number, e.g. 4343. On Windows this port is serviced by a Windows service called HVR Remote Listener. This option gives the best performance, but is the most intrusive.



Method 3

Connect first to a HVR installation on an extra machine using HVR's protocol (a sort of proxy) and then connect from there to the SQL Server database machine using the SQL Server protocol (equivalent to TNS). This option is useful when connecting from a Unix/Linux hub to avoid an (intrusive) installation of HVR's software on the machine containing the SQL Server database.

SQL Server on Linux

HVR supports **Capture** and **Integrate** for SQL Server running on Linux.

- HVR requires the Microsoft ODBC Driver (version 17.5 or higher) for SQL Server.
  1. Download and install the latest Microsoft ODBC Driver for SQL Server on Linux. For more information, refer to Installing the Microsoft ODBC Driver for SQL Server on Linux and macOS in Microsoft documentation.
  2. Create a symbolic link (symlink) for the ODBC driver. Following is an example for Microsoft ODBC Driver for SQL Server libmsodbcsql-17.5.so.1.1,

```
 ln -s  /opt/microsoft/msodbcsql17/lib64/libmsodbcsql-17.5.so.
1.1  $HVR_HOME/lib/libmsodbcsql-17.so
```

- The **User** (username used for connecting HVR to SQL Server) should have read access to the **.mdf** and **.ldf** files. For this, the **User** should typically be added to the Operating System user group **mssql**.
- After installing the Microsoft ODBC Driver for SQL Server, it is recommended to verify the dynamic dependencies. For example,

```
ldd $HVR_HOME/lib/hvr_ms17.so
```

Hub

HVR allows you to create hub database in SQL Server. The hub database is a small database present on the hub machine which HVR uses to control its replication activities. This database stores HVR catalog tables that hold all specifications of replication such as the names of the replicated databases, the list of replicated tables, and the replication direction.

Grants for Hub Database

To **Capture** changes from source database or to **Integrate** changes into target database, the HVR hub database(**User**) requires the privileges mentioned in this section.

It is recommended to create a new database (schema) for 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 hvrschema ;
grant create table to hvruser ;
grant create procedure to hvruser ;
grant select, insert, delete, update on schema::hvrschema to hvruser ;
grant control on schema::hvrschema to hvruser ;
alter user hvruser with default_schema=hvrschema ;
```

Capture

HVR allows you to **Capture** changes from an SQL Server database. HVR uses SQL Server ODBC driver to capture changes from an SQL Server location. This section describes the configuration requirements for **capturing** changes from SQL Server location. For the list of supported SQL Server versions, from which HVR can capture changes, see Capture changes from location in Capabilities.

Table Types

HVR supports capture from the following table types in SQL Server:

- clustered (row/page compressed and uncompressed)
- heap (row/page compressed and uncompressed)

HVR does not support capture from memory optimized tables and trigger-based capture from temporal tables.

Capture Methods

The following two methods are supported for capturing (**Capture**) changes from SQL Server:

- **LogReadMethod:** capture changes from transaction log files.
  - **DIRECT:** capture changes directly from SQL Server's logging files.
  - **SQL:** capture changes over an SQL connection.
- **Archive Log Only:** capture changes from transaction log backup files.

DIRECT Log Read Method

By default, HVR uses the **DIRECT** method to capture changes from the SQL Server's current and backup transaction log files, as well as compressed backup transaction log files. It is not required to define action **Capture** **/LogReadMethod**=**DIRECT**.

The benefits of the **DIRECT** method are:

- the **DIRECT** method is faster and less resource-intensive when capturing changes from database locations, especially for highly loaded databases. To ensure uninterrupted, low-latency CDC, capture must be running faster than the database is writing the logs. The **DIRECT** method and pipelined execution ensures optimum efficiency to keep up with the database log writers. As a result, when capture runs continuously, it will be capturing from the tail end of the log where the log writer(s) are writing.
- the **DIRECT** method supports capture from SQL Server Always On AG secondary database.

The **DIRECT** method requires the following:

- The database account must have **SysAdmin** (SA) privileges.
- The HVR remote agent must be installed on the SQL Server source database server.
- On Windows, the **HVR Remote Listener** service must run either:
  - as the same Windows user that the source SQL Server service runs, or
  - as other Windows user (e.g. HVR **User**), which must have **Debug programs** (**SeDebugPrivilege**) policy enabled.

    User account policies can be managed using the Windows **Local Security Policy** console (accessible from **Control Panel Administrative Tools**). The shortcut command to access this console is **secpol.msc**.

    1. In the **Local Security Policy** window, expand **Local Policies** and click **User Rights Assignment**.
    2. Double-click **Debug Programs** policy available in the list of policies.
    3. In the **Debug programs Properties** dialog, click **Add User or Group**; displays the **Select Users, Computers, Accounts, or Groups** dialog.
    4. In the **Enter the object names to select** field, enter the user account name to whom you want to enable this policy, and click **OK**.
    5. Click **OK**.

  By default, members of the Administrators group have this right.
- On Linux, the HVR **User** must have read access to the **.ldf** files. For this, the **User** should typically be added to the operating system user group **mssql**.

SQL Log Read Method

In the case of the **SQL** method, HVR captures changes over an SQL connection. This method uses stored database function calls to retrieve incremental log fragments. The benefits of the **SQL** method include:

- the ability to run with minimal OS and database privileges
- the ability to capture from a local/remote database without using an HVR agent

Note that remote capture is less efficient than capture using an HVR remote agent on the database server. The **SQL** method will impose more overhead on the transactional database than the **DIRECT** method.

The **SQL** method is slower than the **DIRECT** method, exposes additional load on a source database and, for certain column types, HVR may receive partial/incomplete values and may require HVR to perform additional steps to retrieve the full value from the source database, this is called augmenting (**/AugmentIncomplete**).

To capture changes using the **SQL** method, define action **Capture /LogReadMethod**=**SQL**.

Archive Log Only Method

The **Archive Log Only** method may be activated using options **/ArchiveLogPath**, **/ArchiveLogFormat**, and **/ArchiveLogOnly** in the **Capture** action. The **Archive Log Only** method will generally expose higher latency than non-**Archive Log Only** method because changes can only be captured when the transaction log backup file is created. The **Archive Log Only method** enables high performance log-based Change Data Capture (CDC) with minimal OS and minimal database privileges, at the cost of higher capture latency.

Grants for Log-Based Capture

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

1. 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.
2. 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.

3. 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.

4. A **sysadmin** user must grant the HVR user login a special read-only privilege in the **master** data base.

```
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.

5. 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.

6. 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;
```

7. 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.

Capturing from SQL Server Always On Availability Groups

HVR allows you to capture from SQL Server Always On Availability Groups (AG) - a technology which provides High-Availability (HA) and Disaster-Recovery (DR) solution in SQL Server.

When using **DIRECT** capture method, HVR can be configured to capture from either the primary or secondary node (active or passive).

However, when using **SQL** capture method, HVR can be configured to capture only from the primary node.

Configuring Failover for Connections to SQL Server Always On AG

If HVR is connecting using its own protocol to a database inside an SQL Server Always On AG, the following is required:

1. Create an HVR Remote Listener on each node.
2. Inside Failover Cluster Manager configure a Role with a static IP address that controls the HVR Remote Listener service on all nodes. Run **Configure Role Wizard, Next > Generic Service > HVR Remote Listener > Name > Next > Next > Next > Finish**. To configure a static IP address, click on **Resources > Name> IP address>** then change it to an available static address.
3. Configure a Group Listener inside the Availability Group. Start Management Studio on the primary node, click **AlwaysOn High Availability > Availability Groups > Name > Availability Group Listeners > Add Listener** ….

   For Always On AG inside Azure an extra step is required:
4. Configure an Internal or External Azure load balancer for the HVR Remote Listener using Powershell. Then attach each Azure node to this load balancer. When this is done, fill in the IP address of the Azure load balancer into the **Node** field in the **Connect to HVR on remote machine** section of the HVR location dialog.

HVR can now connect to Node as configured in step 1 or step 4 and Server as configured in Step 3.

Configuring Backup Mode and Transaction Archive Retention

HVR log-based capture requires that the source database is in **Full recovery** model and a full database backup has been done since this was enabled. Normally HVR reads changes from the 'online' transaction log file, but if HVR is interrupted (say for 2 hours) then it must be able to read from transaction log backup files to capture the older changes. HVR is not interested in full or incremental backups; it only reads transaction log backup files.

- HVR supports only native single-media-family backup(s).
- HVR does not support backup(s) on 'virtual devices'.
- For **DIRECT** method, the backup(s) must be accessible to HVR on the file system.
- For **SQL** method, the backup(s) must be accessible to SQL Server on the file system.
- If **SQL** method is defined with **Capture /ArchiveLogPath**, the backup(s) must be accessible to both SQL Server and HVR using the same path (UNC in case of network backup).

Transaction log (archive) retention: If a backup process has already moved these files to tape and deleted them, then HVR capture will give an error and an HVR Refresh will be needed before replication can be restarted. The amount of 'retention' needed (in hours or days) depends on organization factors (how real-time must it be?) and practical issues (does a refresh take 1 hour or 24 hours?).

HVR normally locates the transaction log backup files by querying the backup history tables in the **msdb** database. But if Always On AG is configured then this information source is not available on all nodes. So when HVR is used with Always On AG, the transaction log backups must be made on a directory which is both accessible from all Always On AG nodes and also from the machine where the HVR capture process is running (if this is different) via the same path name. HVR should be configured to find these files by defining action **Capture** with two additional parameters **/ArchiveLogPath** and **/ArchiveLogFormat**. The parameter **/ArchiveLogPath** should point to a file system directory which HVR will use for searching directly for the transaction log backup files, instead of querying **msdb**. The parameter **/ArchiveLogFormat** should specify a pattern for matching files in that directory. The pattern can contain these special characters:

Pattern	Description
*	Wildcard to match zero or more characters.
%d	Database name. This is not case sensitive.
%Y	Year (up to 4 digit decimal integer).
%M	Month (up to 2 digit decimal integer)
%D	Day (up to 2 digit decimal integer)
%h	Hours (up to 2 digit decimal integer)
%m	Minutes (up to 2 digit decimal integer)
%s	Seconds (up to 2 digit decimal integer)
%n	File sequence number (up to 64 bit decimal integer)
%%	Matches %

All other characters must match exactly. HVR uses the %Y, %M, %D, %h, %m, %s and %n values to order files.

Dropping the Source Database

Depending on the setting of action **Capture** **/SupplementalLogging** HVR will use some of SQL Server's own 'replication components' or it will use SQL Server's Change Data Capture (CDC) feature.

Based on this, HVR may enable the source database for publication, which will mean that attempts to drop the database will give an SQL Server error.

Alternatively HVR may enable Change Data Capture (CDC) on source databases, which will also mean attempts to drop the database may also give an SQL Server error because of the running CDC capture and cleanup jobs.

When command **HVR Initialize** is used with **Drop Objects** (option **-d**) then it will disable the 'publish' replication option if there are no other systems capturing from that database. It will also disable the CDC for the database, if there are no other CDC table instances exist in that database. The database can then be dropped.

To drop the database immediately (without running the **HVR Initialize** first) the **sysadmin** must perform the following SQL statement:

```
exec sp_replicationdboption 'capdb', 'publish', 'false'
```

```
use [capdb]
exec sp_cdc_disable_db
```

Grants for Trigger-Based Capture

HVR allows you to perform trigger-based capture (**Capture /TriggerBased**) from SQL Server. To enable triggerbased capture for SQL Server:

- HVR's user should be made a database owner (**db_owner** role).
- The extended stored procedure **hvrevent** should normally be installed on the capture machine. This is not needed if parameters **Capture /TriggerBased** is not defined or **/ToggleFrequency** or **Scheduling/CaptureStartTimes** or **/CaptureOnceOnStart** are defined. This step must be performed by a user that is a member of the system administrator role. For more information, see Installing HVR on Windows.

Limitations

- HVR does not support log-based capture from Amazon RDS for SQL Server.

Integrate and Refresh Target

HVR allows you to **Integrate** changes into SQL Server database. This section describes the configuration requirements for integrating changes (using **Integrate** and **Refresh**) into SQL Server location. For the list of supported SQL Server versions, into which HVR can integrate changes, see Integrate changes into location in Capabilities.

HVR uses the following interfaces to write data into an SQL Server location:

- SQL Server ODBC driver, used to perform continuous **Integrate** and row-wise **Refresh**
- SQL Server BCP interface, used for copying data into database tables during bulk **Refresh** and loading data into burst tables during **Integrate** with **/Burst**

Grants for HVR on Target Database

This section provides information about the user privileges required for replicating changes into SQL Server database using **HVR Refresh**.

When replicating changes into a target SQL Server database, HVR supports the following two permission models: **DbOwner**, and **Minimal**.

- **DbOwner**
  In this permission model, the HVR **User** must be made a database owner (**db_owner** role). Normally, the database objects which HVR sometimes creates will be part of the dbo schema as the replicated tables. Alternatively, these HVR database objects can be put in a special database schema so that they are not visible to other users. The following SQL is needed:

  ```
  create schema hvrschema;

  grant control on schema::hvrschema to hvruser;

  alter user hvruser with default_schema=hvrschema;
  ```

- **Minimal**
  In this permission model, the **User** does not need to be a database owner. This model cannot use parameter **/Schema** to change tables with a different owner. The following SQL is needed so that HVR can create its own tables:

  ```
  grant create table to hvruser;

  create schema hvrschema;

  grant control on schema::hvrschema to hvruser;

  alter user hvruser with default_schema=hvrschema;
  ```

  If action **Integrate /DbProc** is defined, then **create procedure** privilege is also needed.

Compare and Refresh Source

To perform **HVR Compare** or **HVR Refresh** (read from SQL Server), the HVR **User** requires the privileges mentioned in this section.

When HVR is reading rows from a database (no capture) it supports two permission models: **DbOwner**, and **Minimal**.

- **DbOwner**
  In this permission model, the HVR **User** must be made owner of the source database (**db_owner** role).

- **Minimal**
  In this permission model, the HVR **User** does not need to be a database owner.

If the HVR **User** needs to select from tables in another schema (for example if action **TableProperties /Schema** is defined), then **select** privilege should be granted.

```
grant select to hvruser; -- Let HVR read all tables

grant select on schema::dbo to hvruser; -- Let HVR only read DBO tables
```

Azure SQL Managed Instance

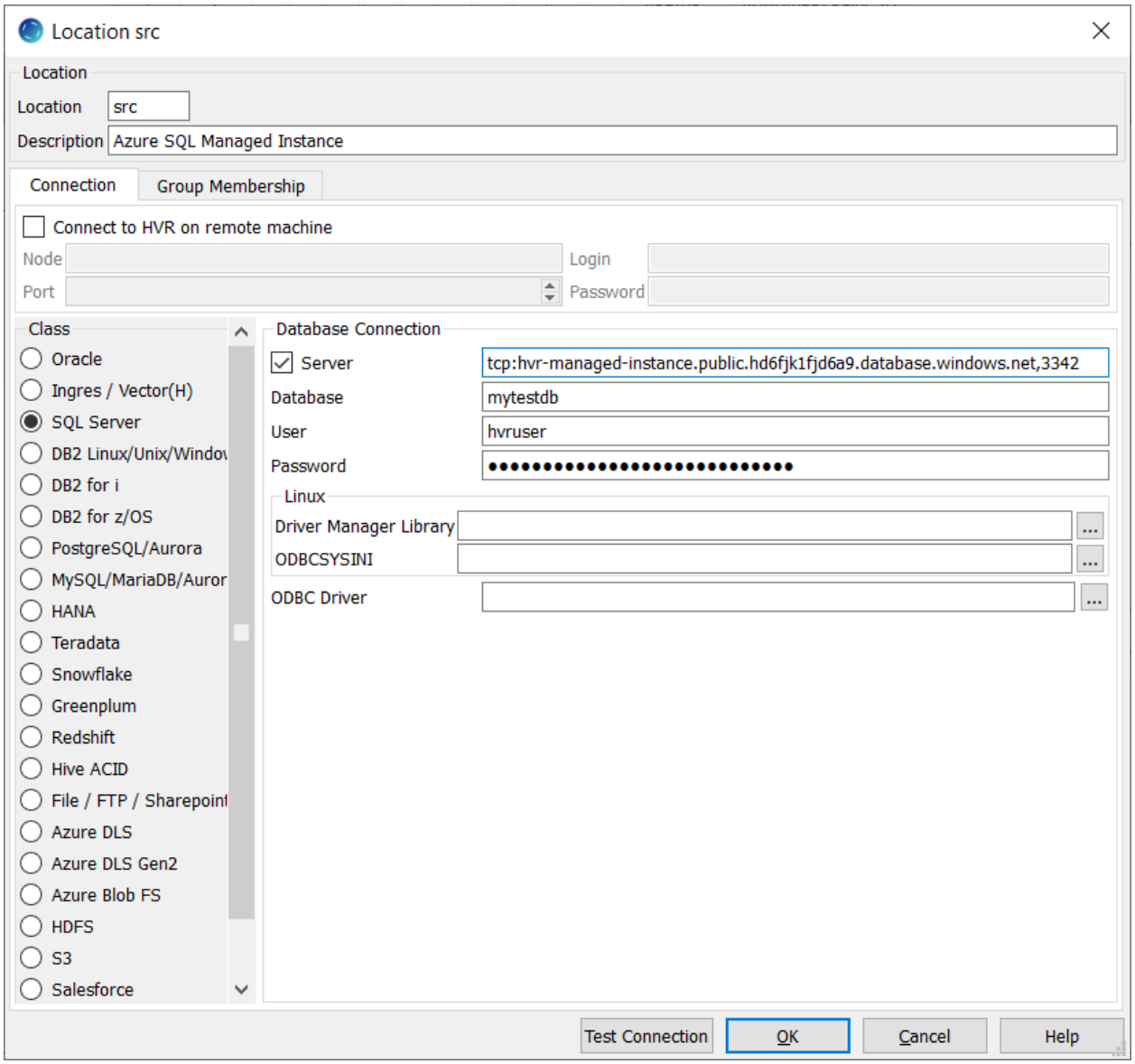
Since v5.7.5/3 and v5.7.0/8

HVR supports **Capture**, **Integrate**, **HVR Refresh** and **HVR Compare** on Azure SQL Managed Instance. HVR also supports Azure SQL Managed Instance as a hub database. For the Capabilities sup ported by HVR on Azure SQL Managed Instance, see Capabilities for Azure SQL Managed Instance.

- For the grants required for Azure SQL Managed Instance as a hub database, see section Grants for Hub Database.
- For the grants required to **capture** from Azure SQL Managed Instance, see section Grants for Log-Based Capture.
- For the grants required to **integrate** and **refresh** into Azure SQL Managed Instance, see section Grants for HVR on Target Database.
- For the grants required to perform **compare** in Azure SQL Managed Instance, see section Comp are and Refresh Source.

Location Connection

This section lists the connection details required for creating an Azure SQL Managed Instance location in HVR.



Field	Description
Server	The fully qualified host name for your Azure SQL Managed Instance. Specify the host name and port number separated by a comma (,) to connect to the server on which Azure SQL Managed Instance is running, according to the following format: **tcp:***server name*,*port number* **Example:** tcp:hvr-managed-instance.public.hd6fjk12b8a9.database.windows.net,3342
Database	The name of the database in the Azure SQL Managed Instance. **Example:** *mytestdb*
User	The username to connect HVR to the Azure SQL Managed Instance. The User must have all the required grants (see above).
Password	The password of the **User** to connect HVR to the Azure SQL Managed Instance.

Log-Based Capture

HVR supports only the **SQL** log read method for capturing changes from Azure SQL Managed Instance. This is the default capture method that will be automatically set when you use Azure SQL Managed Instance as a source location.

To capture changes from Azure SQL Managed Instance, the following action definition is required:

Group	Table	Action
Azure SQL Managed Instance	*	**Capture /LogReadMethod=SQL /LogTruncate=CAP_JOB_RETAIN**