DATASHEET

SQL Server Database Support

The SQL Server Database is a very commonly used database, often running very demanding and mission-critical workloads. SQL Server is a common data source for analytical environments and data lakes. This datasheet describes HVR's support for the SQL Server Database as a source for change data capture, and as a target for data delivery.

Version Compatibility

HVR supports SQL Server 2005 and above as a source and target for all editions, with log-based CDC only available on Standard, Developer, and Enterprise Editions. Connectivity to the SQL Server Database is established through the native SQL Server drivers.

Why HVR?

To support Change Data Capture from busy, mission-critical workloads running on the SQL Server Database, you need an efficient and robust, and flexible CDC solution. HVR is that solution, whether your need is for heterogeneous replication scenarios, or as a flexible, less intrusive and more performant alternative to Microsoft's native data replication capabilities. HVR provides not only CDC capabilities but also includes:

- Target table creation automatically mapping data to compatible data types
- Refresh i.e. one time data load, integrated with CDC to simplify initial load scenarios, granular down to the table level
- Compare and repair, also in heterogeneous environments
- Rich statistics visualization
- Automatic monitoring
- Graphical User Interface

One-time Load – Refresh – and Compare/Repair

One-time load, in HVR's terms "refresh", as well as compare/repair, starts with a select from the SQL Server Database. Parallelism can be specified across tables, and data is pulled out through a native SQL Server connection. For optimum performance and efficiency HVR recommends an architecture using an agent close to if not on the database server(s) to densely compress/decompress the data, and optionally encrypt/decrypt it. The native connection combined with optimized network utilization supports efficient and fast data retrieval.

On the target SQL Server Database HVR uses efficient direct path data loads for optimum load performance.

CDC from SQL Server

SQL Server's Transaction Log (TLog) records database changes to enable high availability and point in time recovery. The TLog is also the source for data replication products like HVR to know what operations were submitted to the database. The TLog is designed to be overwritten provided nothing is blocking overwrite from happening, and it will grow until so-called truncation takes place. A DBA can find out at any point in time in sys.databases what prevents log truncation. HVR uses its own parser for transaction log entries to support CDC.

Logical data replication, in which a target system is updated using an SQL statement or by otherwise recording the changes, requires knowledge of the row identifier. By default SQL Server is optimized to not record the row's logical (primary or unique) key. Currently the only two ways to instruct SQL Server to also write logical keys to the log is to enable:

1. Microsoft's CDC tables (not available on Standard Edition until SQL Server 2016 SP1), introducing CDC tables for every source table that must be replicated.
2. Articles as part of SQL Server's native replication. Articles require source tables to have a primary key, and prevent DDL operations on the table.

HVR supports both the use of CDC tables (default, if possible on the database edition) and Articles.

Both the creation of CDC tables and Articles activate internal SQL Server logic to prevent log truncation unless changes are pulled from the TLog to populate the CDC tables, or the distribution database, respectively. To limit impact on the system HVR will remove the agent jobs that perform SQL Server's log mining and take over releasing the truncation point for replication (unless non-HVR replication is in place).

The SQL Server source database should be running in full recovery mode (with an initial full backup taken) to ensure no data is lost. HVR supports CDC using both direct access to the (backup and online) transaction logs, and via remote capture through SQL and fn_dump_dblog, dynamically switching between backup transaction logs and the TLog as needed. Direct access to the logs requires the HVR executable on the SQL Server Database server which can either be the primary database, a standby node in an AlwaysOn cluster, or a separate system that only receives transaction log backups (in which case of course there will be latency until the log backup takes place). Simple recovery mode is also supported but only using a single source capture process.

HVR continuously watches DDL in the SQL Server database to ensure no data is lost when indexes are rebuilt, or to (optionally) replicate DDL changes to the target system.

Target SQL Server

HVR uses SQL Server's native connectivity protocol to connect to the database. Bulk data refreshes, including the initial data load, as well as loads into staging tables for burst integration, are done through efficient direct path append operations.

HVR is a Microsoft Partner Microsoft

ABOUT HVR

We accelerate data movement so that you can revolutionize your business. HVR is designed to move large volumes of data FAST and efficiently in modern environments for continuous real-time updates.

Our goal is to keep your data moving and in sync as you adopt new technologies for storing, streaming, and analyzing data. Our scalable solution gives you everything you need for efficient data replication from beginning to end so that you can readily revolutionize your business for the modern world.