Extra Columns for Capture, Fail and History Tables

Each capture, fail or history table created by HVR contains columns from the replicated table it serves, plus extra columns. These columns contain information about what the captured operation was and where it must be sent.

The following extra columns can appear in capture, fail or history tables:

Column	Datatype	Description
hvr_seq	**float** or **byte10** on Ingres, **numeric** on Oracle and **timestamp** on SQL Server	Sequence in which capture triggers were fired. Operations are replayed on integrate databases in the same order, which is important for consistency.
hvr_tx_id	string	Transaction ID of captured change. This number is unique at any moment (each open transaction has a unique number) but may not be increasing.
hvr_tx_seq	string	Sequence number of transaction. For log–based capture this value can be mapped to the Ingres LSN or the Oracle SCN of the transaction's **commit** statement.
hvr_tx_countd own	number	Countdown of change within transaction, for example if a transaction contains three changes the first change would have countdown value 3, then 2, then 1. A value of zero indicates that commit information is missing for that change.
hvr_op	number	Operation type. Values are • **0** - Delete • **1** - Insert • **2** - After update • **3** - Before key update • **4** - Before non–key update • **5** - Truncate table • **8** - Delete affecting multiple rows • **22** - Key update with missing values • **32** - Resilient variant of **22** • **42** - Key update whose missing values have been augmented • **52** - Resilient variant of **42** A key–update sometimes appears as a before update followed by an after update, but is sometimes converted into a delete followed by an insert. A before non–key update row (**hvr_op=4**) can be removed by adding **Capture /NoBeforeUpdate**. During an on-line refresh, a delete, insert and delete can be 'in-doubt'; these are shown as **10**, **11** and **12** respectively. To ignore this 'in-doubt' information, use **mod(10)** to convert these back to **0**, **1** or **2**. Note that **Integrate /Burst** will convert a key update (**hvr_op 3** then **2**) into a delete+insert pair (**hvr_op 0** then **1**).
hvr_cap_loc	string	Name of location on which the change was captured.

hvr_cap_tstamp	normally date, sometimes integer (number of seconds since 1st Jan 1970 GMT)	Timestamp of captured change. For log-based capture this is the time that the change was committed (it did not logically "exist" before that moment). For trigger-based capture it when the DML statement itself was triggered. If HVR refresh fills this value, it uses the time that the refresh job started.
hvr_cap_user	string	Name of user who performed captured change. Supported only on few DBMSs, see capabilities Populates column **hvr_cap_user** for use in **ColumnProperties {hvr_cap_user}** substitutions.
hvr_address	string	Address of target location for change. Only set if action **Restrict /HorizColumn** is defined.
hvr_err_tstamp	date	Time at which integration error occurred. Written into fail table.
hvr_err_msg	long string	Integration error message written into fail table.
hvr_colval_mask	string	Mask showing which column values were missing or not updated. For example value **–––– m–** could mean that log–based capture of an update is missing a value for the second last column of a table.