Batch Commands Summary

Command Name	Description	Example
BATCH <LU>[@<DC>] FABRIC_COMMAND='<fabric command> ?' [WITH [AFFINITY='<affinity>'] [JOB_AFFINITY='<job affinity>'] [ASYNC=true/false] [GENERATE_ENTITIES_FIRST=true/false] [ALLOW_MULTIPLY=true/false] [MAX_WORKERS_PER_NODE=<number>]];/h6>	Start the Batch process and sync all LU instances: DC, specifies the DC name to force the Batch process in the specified DC. AFFINITY, list of nodes and DCs to be used during the Batch process execution. JOB_AFFINITY, affinity for the Batch process Job. ASYNC, defines whether the Batch process should run in a sync or async mode. Default is False. GENERATE_ENTITIES_FIRST, if set to True, generate all entities before processing them. FABRIC_COMMAND, Fabric command to be executed by the Batch process which can be any command that includes a '?' to represent a singular Entity ID. One of the following commands must be set: (for Migration, "sync_instance .?", for Broadway, "broadway LU.SampleFlow SampleIID=?", for CDC republish, "cdc_republish_instance Customer.?") ALLOW_MULTIPLY, when set to True, multiplies executions of the same Batch process command. Default is False. MAX_NODES, maximum (random) nodes participating in the Batch process. MAX_WORKERS_PER_NODE, enables setting a lower number of maximum workers to run on each node than the maximum number of workers defined in the config.ini file. (MAX_WORKERS_PER_NODE parameter).	BATCH CUSTOMER FABRIC_COMMAND="sync_instance Customer.?" with async='true'; This command migrates all customers from the source systems into the Fabric CUSTOMER keyspace in the Fabric database.
BATCH <LU>[@<DC>].<IG> fabric_command='<fabric command> ?' [WITH [AFFINITY='<affinity>'] [JOB_AFFINITY='<job affinity>'] [ASYNC=true/false] [GENERATE_ENTITIES_FIRST=true/false] [ALLOW_MULTIPLY=true/false] [MAX_NODES=<number>] [MAX_WORKERS_PER_NODE=<number>]];	Batch-processes a subset of the LUI based on the Instance Group specified by the <IG> parameter.	BATCH CUSTOMER.ig10CustomersList FABRIC_COMMAND="sync_instance CUST.?" with async='true'; This command migrates the customers defined in the 'ig10CustomersList' Instance Group into the CUSTOMER keyspace in the Fabric database.
BATCH <LU>[@<DC>] from <db_interface> using ('<SQL>') fabric_command='<fabric command> ?' [WITH [AFFINITY='<affinity>' [JOB_AFFINITY='<job affinity>'] [ASYNC=true/false] [GENERATE_ENTITIES_FIRST=true/false] [ALLOW_MULTIPLY=true/false] [MAX_NODES=<number>] [MAX_WORKERS_PER_NODE=<number>]];	Batch-processes a subset of the LUI based on a query to a source interface defined in the <db_interface> parameter.	BATCH CUSTOMER FROM CRM_DB USING ('select customer_id from CUSTOMER where customer_id <= 1000') FABRIC_COMMAND="sync_instance CUSTOMER.?" with async='true';
BATCH <LU>[@<DC>] from fabric fabric_command='<fabric command> ?' [WITH [AFFINITY='<affinity>'] [JOB_AFFINITY='<job affinity>'] [ASYNC=true/false] [GENERATE_ENTITIES_FIRST=true/false] [ALLOW_MULTIPLY=true/false] [MAX_NODES=<number>] [MAX_WORKERS_PER_NODE=<number>]];	Batch-processes a subset of the LUI based on existing instances in Fabric in the entity table.	BATCH Customer from fabric fabric_command='sync_instance Customer.?';
BATCH <LU>[@<DC>].(<instance 1,instance 2,...>) fabric_command='<fabric command> ?' [WITH [AFFINITY='<affinity>'] [JOB_AFFINITY='<job affinity>'] [ASYNC=true/false] [GENERATE_ENTITIES_FIRST=true/false] [ALLOW_MULTIPLY=true/false] [MAX_NODES=<number>] [MAX_WORKERS_PER_NODE=<number>]]	Batch-processes a subset of the LUI based on a list of instances defined in the <instance 1,instance 2,...> parameters list.	BATCH Customer.('100', '101', '102','103') FABRIC_COMMAND="sync_instance CUSTOMER.?" with async='true';
	Runs the Batch process using a function that will match the name if the instance from the external source with its ID as used internally by Fabric	1) BATCHF Customer.batchFtest4().ig20 FABRIC_COMMAND='sync_instance Customer.?';

BATCHF	The BATCHF command uses the exact same parameters as the BATCH command described above.	2) BATCHF Customer@DC1.batchFtest4() from HIS_DB using ('select customer_id from invoice where balance=12894') FABRIC_COMMAND='sync_instance Customer.?'; 3) BATCHF Customer.batchFtest4().('1','2','3') FABRIC_COMMAND='sync_instance Customer.?';
BATCH_RETRY '<batch_id>'	If the Batch process has not been completed, resumes a previous Batch process by reprocessing all failed or unhandled entities. Otherwise, it retries the failed entities only. If the Batch process is completed before the Retry command, Fabric gets the list of instances from the source DB. If the Batch process is completed before the Retry command, Fabric gets the list of failed entities from the batchprocess_entities_errors Cassandra table.	BATCH_RETRY '161f9717-bd93-4882-a3aa-7b58c1f61b27';
CANCEL BATCH ['<batch_id>']	Cancels the last started Batch process coordinated by the current node. The Cancel command must be executed from the node that started the operation. When adding the '<batch_id>' parameter, the Batch process with the defined batch_ID is cancelled. Note that in this case, the Cancel command does not need to be run from the node coordinating the specific Batch process.	CANCEL BATCH; CANCEL BATCH '568114fe-9ec8-4c9e-af11-6e3348eff6e9'; Note that the KILL command can be used as well, using the batch ID (*bid*): KILL '568114fe-9ec8-4c9e-af11-6e3348eff6e9';