

# Bulk Load Design Pattern for Majesco

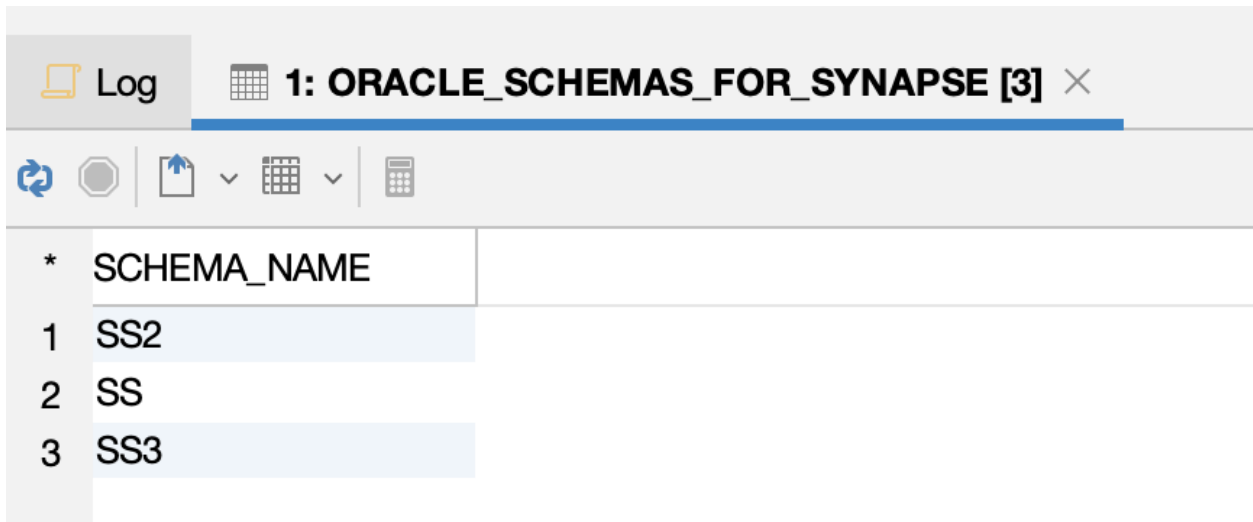
This document describes a three pipeline design pattern for Majesco's bulk load use case.





## Table of Contents

<b>Step #0: Create a table named ORACLE_SCHEMAS_FOR_SYNAPSE that holds a list of Schemas you want to replicate</b>	<b>2</b>
<b>Step #1: Create a Bulk Load Pipeline that is parameterized</b>	<b>3</b>
<b>Step #2: Create a Job Template for the bulk load pipeline</b>	<b>5</b>
<b>Step #3: Create a helper pipeline to create the table-pattern-configs</b>	<b>6</b>
<b>Step #4: Create a pipeline that creates and launches Job Template Instances</b>	<b>8</b>
<b>Step #5: Running everything</b>	<b>10</b>

Step #0: Create a table named ORACLE\_SCHEMAS\_FOR\_SYNAPSE that holds a list of Schemas you want to replicate

For example, in my environment I have this table with these three rows. See Step 4 below for details



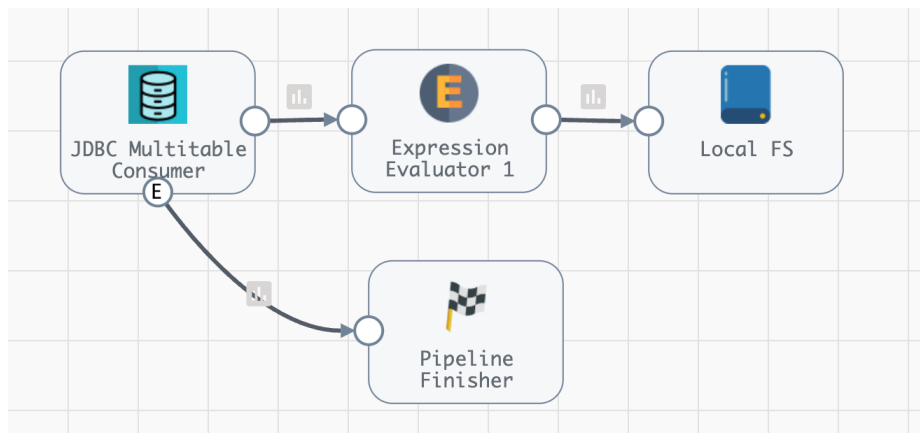
Log		1: ORACLE_SCHEMAS_FOR_SYNAPSE [3] X	
			
* SCHEMA_NAME			
1	SS2		
2	SS		
3	SS3		

# Step #1: Create a Bulk Load Pipeline that is parameterized

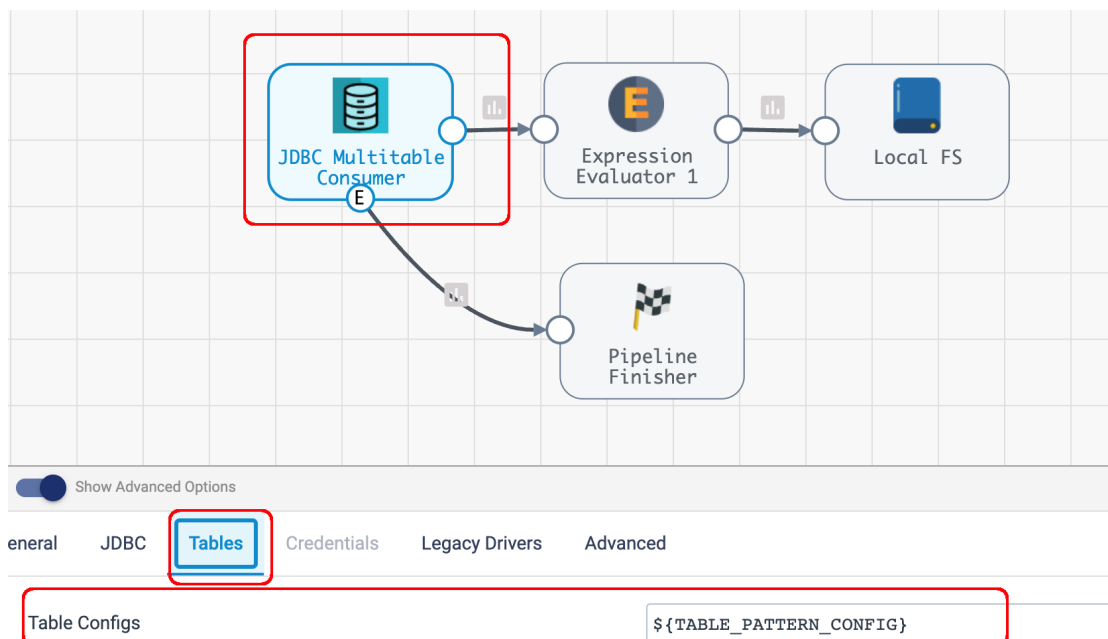
You can view an example of this pipeline [here](#)

Assume there is a Bulk Load Pipeline with a JDBC Multitable Origin that writes to Synapse (or as in my example, that writes to a Local FS):

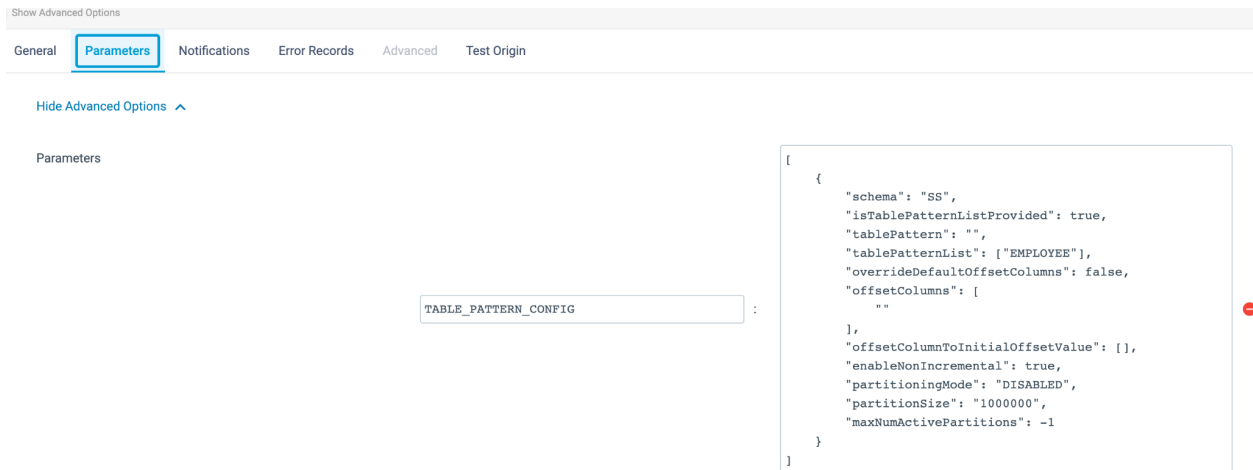
In my tests, my pipeline looks like this:



A key aspect of the JDBC Origin is that the Table Configs property is parameterized:



And that parameter is exposed at the pipeline level:



Hide Advanced Options ^

Parameters

TABLE\_PATTERN\_CONFIG :

```
[
  {
    "schema": "SS",
    "isTablePatternListProvided": true,
    "tablePattern": "",
    "tablePatternList": ["EMPLOYEE"],
    "overrideDefaultOffsetColumns": false,
    "offsetColumns": [
      ""
    ],
    "offsetColumnToInitialOffsetValue": [],
    "enableNonIncremental": true,
    "partitioningMode": "DISABLED",
    "partitionSize": "1000000",
    "maxNumActivePartitions": -1
  }
]
```

An example parameter value that would specify the origin should read three tables from a given schema would look like this:

```
[{"TABLE_PATTERN_CONFIG": "[{"schema":
\\"SS\\",\\"isTablePatternListProvided\\": true,\\"tablePattern\\":
\\"\\",\\"tablePatternList\\":
[\\"T_2PK\\",\\"T_KINESIS\\",\\"T_KINESIS_2\\"],\\"overrideDefaultOffsetColu
mns\\": false,\\"offsetColumns\\":
[\\"\\"],\\"offsetColumnToInitialOffsetValue\\":
[],\\"enableNonIncremental\\": true,\\"partitioningMode\\":
\\"DISABLED\\",\\"partitionSize\\":
\\"1000000\\",\\"maxNumActivePartitions\\": -1}]}"]}]
```

Don't worry about how ugly that parameter value looks; the subsequent pipelines will auto-generate those parameter values

## Step #2: Create a Job Template for the bulk load pipeline

You can view an example of the Job Template pipeline [here](#)

Note that the Job Template exposes the pipeline's parameter:

### Edit Job Template

#### 4 Set Parameter Defaults

Define the parameter values to start the pipeline with. Override the default values using simple or bulk edit mode. In bulk edit mode, configure parameter values in JSON format. [Learn more](#)

Parameter Name	Default Value	Static Parameter
TABLE_PATTERN_CONFIG :	<pre>[   {     "schema": "",     "isTablePatternListProvided": true,     "tablePattern": "",     "tablePatternList": [],     "overrideDefaultOffsetColumns": false,     "offsetColumns": [       ""     ],     "offsetColumnToInitialOffsetValue": [],     "enableNonIncremental": true,     "partitioningMode": "DISABLED",     "partitionSize": "1000000",     "maxNumActivePartitions": -1   } ]</pre>	<input type="checkbox"/>

 BULK EDIT MODE

Back

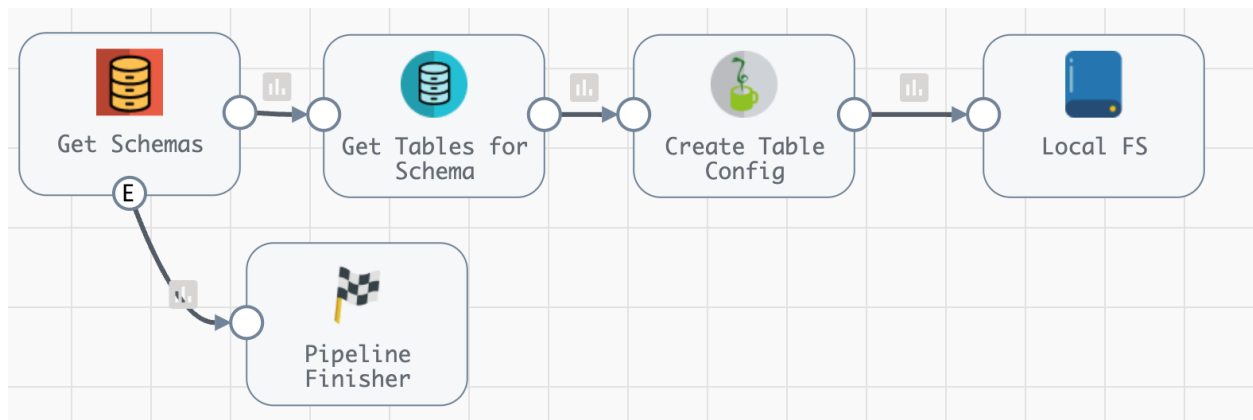
Save & Next

Save & Exit

## Step #3: Create a helper pipeline to create the table-pattern-configs

Create a helper pipeline to retrieve all tables for a set of schemas and to create the table-pattern-configs we'll pass to the Job Template instances

You can view an example of this pipeline [here](#)



This pipeline has two parameters:

OUTPUT_DIR	:	/Users/mark/data/table-pattern-configs	⊖
NUM_TABLES_PER_TEMPLATE_INSTANCE	:	3	⊖

The first parameter specifies where on the SDC machine it will write files to

The second parameter specifies how many tables will be included per Job Template instance.

So for example, if you want each Job Template instance to retrieve rows from 100 tables, set the parameter to 100. For my test I set it to 3.

The Get Schemas origin is a JDBC Query origin that retrieves a list of schemas from a table named ORACLE\_SCHEMAS\_FOR\_SYNAPSE that has just a single column named SCHEMA\_NAME. Create such a table if one does not exist, or use a Directory origin or a Dev Raw Data Generator, etc...

## IT IS IMPERATIVE THAT THIS ORIGIN HAS A BATCH SIZE OF 1

Max Batch Size (Records)

1

because the logic of the pipeline depends on that -- each record will correspond to one schema, and each schema will be processed one at a time.

The Get Tables for Schema stage is a JDBC Lookup that will append a LIST of Tables (and not Views) for each schema.

The Create Table Config stage is a Jython stage that takes each set of Schema and Table values and generates the parameter values a Job Template will use.

Here are two example rows of output from this pipeline:

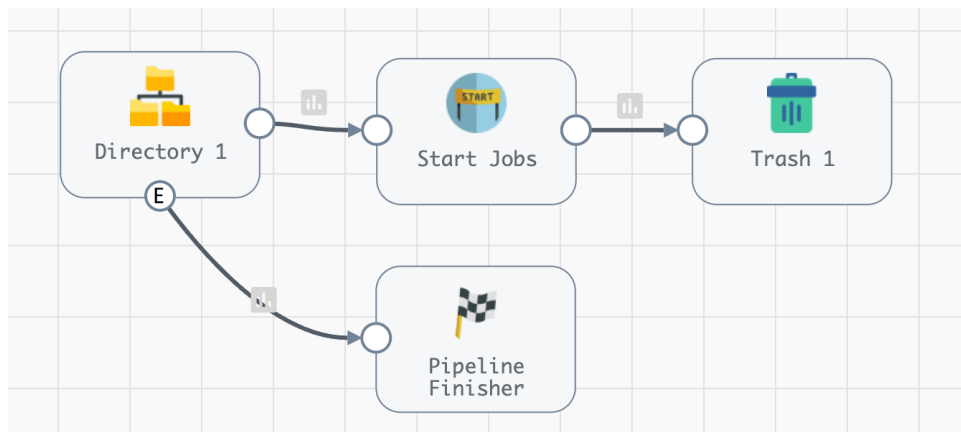
```
[{"TABLE_PATTERN_CONFIG": "[{"schema": "SS", "isTablePatternListProvided": true, "tablePattern": "", "tablePatternList": ["T_2PK", "T_KINESIS", "T_KINESIS_2"], "overrideDefaultOffsetColumns": false, "offsetColumns": [], "offsetColumnToInitialOffsetValue": [], "enableNonIncremental": true, "partitioningMode": "DISABLED", "partitionSize": "1000000", "maxNumActivePartitions": -1}]"]}]
```

```
[{"TABLE_PATTERN_CONFIG": [{"schema": "SS3", "isTablePatternListProvided": true, "tablePattern": "", "tablePatternList": ["T102"], "overrideDefaultOffsetColumns": false, "offsetColumns": [], "offsetColumnToInitialOffsetValue": [], "enableNonIncremental": true, "partitioningMode": "DISABLED", "partitionSize": "1000000", "maxNumActivePartitions": -1}]}]
```

## Step #4: Create a pipeline that creates and launches Job Template Instances

Create a pipeline that reads the table-pattern-configs from the file system and that creates and launches Job Template Instances

You can view an example of this pipeline [here](#)



Set these two pipeline parameters:

INPUT_DIR	:	/Users/mark/data/table-pattern-configs	⊖
NUM_CONCURRENT_TEMPLATE_INSTANCES	:	3	⊖

The first parameter should point to the same directory where the previous helper pipeline wrote the table-pattern-config records to local disk

The second parameter determines how many Job Template Instances will run concurrently.

Note that if you have multiple SDCs, the Job Template Instances can be placed on all of them for scale-out processing

For my test I'll set the concurrency to 3.

***Important point: this design is not fully optimized in that the three (or however many you specify) Job Template Instances will run concurrently, but all three (or however many instances) must all complete before the next set are launched.***



Set your Job Template ID, Control Hub URL and API creds in the Start Jobs Stage:

General **Job** Credentials HTTP Proxy TLS Logging

---

Task Name ⓘ

Control Hub URL ⓘ

Job Template ⓘ ☒

Job Template ID ⓘ

Instance Name Suffix ⓘ

Job Instance Runtime Parameters ⓘ

1	<code>\${record:value('/text')}</code>
---	--






Note the Job Instance Runtime Parameter is simply this:

```
${record:value('/text')}
```

## Step #5: Running everything

- Start by running the helper pipeline to query Oracle and to write a set of table-pattern-config records to local disk.
- Run the pipeline from step 4 with a small number of concurrent instances. For example, I tested with three. Once you start pipeline 4, you should see three Job Template Instances are running, like this:









Job Instances ( 3 with current filters )

<input type="checkbox"/>  Name	Pipeline	Version	 Last Modified On	Job Status	Pipeline Status
<input type="checkbox"/>  Oracle Bulk Load Template - 1	Oracle Bulk Load	v5	a few seconds ago	ACTIVE	STARTING
<input type="checkbox"/>  Oracle Bulk Load Template - 1	Oracle Bulk Load	v5	a few seconds ago	ACTIVE	STARTING
<input type="checkbox"/>  Oracle Bulk Load Template - 1	Oracle Bulk Load	v5	a few seconds ago	ACTIVE	STARTING

Items per page: 50 1 - 3 of 3

Once the first set completes, the next set of three should start:

Job Instances ( 51 with current filters )

<input type="checkbox"/>  Name	Pipeline	Version	 Last Modified On	Job Status	Pipeline Status
<input type="checkbox"/>  Oracle Bulk Load Te...	Oracle Bulk Load	v5	a few seconds ago	ACTIVE	RUNNING
<input type="checkbox"/>  Oracle Bulk Load Te...	Oracle Bulk Load	v5	a few seconds ago	ACTIVE	RUNNING
<input type="checkbox"/>  Oracle Bulk Load Te...	Oracle Bulk Load	v5	a few seconds ago	ACTIVE	RUNNING
<input type="checkbox"/>  Oracle Bulk Load Te...	Oracle Bulk Load	v5	a minute ago	INACTIVE	
<input type="checkbox"/>  Oracle Bulk Load Te...	Oracle Bulk Load	v5	2 minutes ago	INACTIVE	
<input type="checkbox"/>  Oracle Bulk Load Te...	Oracle Bulk Load	v5	2 minutes ago	INACTIVE	

You can inspect any running or completed Job to see the table-pattern-config it used:

[Job Instances](#) > Oracle Bulk Load Template - 1

Job Instance Details

Monitor Job


Start Job ▼

Delete

Share

Status	Start Time	Finish Time	Run Count
INACTIVE	Mar 14, 2023, 11:11:02 PM	Mar 14, 2023, 11:11:34 PM	1

Data Collectors

 10.10.10.169:18992

Job Instance Name

Oracle Bulk Load Template - 1

Pipeline

[Oracle Bulk Load \(v5\)](#)

Parent Job Template

[Oracle Bulk Load Template](#)

Enable Failover

true

Engine Labels

DEV

Runtime Parameters

TABLE\_PATTERN\_CONFIG={"schema": "SS","isTablePatternListProvided": true,"tablePattern": "", "tablePatternList": ["CONTACT","DEPT","EMPLOYEE"],"overrideDefaultOffsetColumns": false,"offsetColumns": [], "offsetColumnToInitialOffsetValue": [], "enableNonIncremental": true,"partitioningMode": "DISABLED","partitionSize": "1000000","maxNumActivePartitions": -1}}

Show Additional Info ▼