



Sqoop Incremental Import

...

Delta Load

IMPORTANT

Copyright Infringement and Illegal Content Sharing Notice

All course content designs, video, audio, text, graphics, logos, images are Copyright© and are protected by India and international copyright laws. All rights reserved.

Permission to download the contents (wherever applicable) for the sole purpose of individual reading and preparing yourself to crack the interview only. Any other use of study materials – including reproduction, modification, distribution, republishing, transmission, display – without the prior written permission of Author is strictly prohibited.

Trendytech Insights legal team, along with thousands of our students, actively searches the Internet for copyright infringements. Violators subject to prosecution.



Sqoop Incremental Import

Sqoop provides an incremental import mode which can be used to retrieve only rows newer than some previously-imported set of rows.

--check-column (col) char not supported
--incremental (mode) append , lastmodified
--last-value (value)

Sqoop supports two types of incremental imports: append and lastmodified. You can use the --incremental argument to specify the type of incremental import to perform.



Sqoop Incremental Import

You should specify **append mode** when importing a table where new rows are continually being added with increasing row id values. You specify the column containing the row's id with `--check-column`. Sqoop imports rows where the check column has a value greater than the one specified with `--last-value`.

An alternate table update strategy supported by Sqoop is called **lastmodified mode**. You should use this when rows of the source table may be updated, and each such update will set the value of a last-modified column to the current timestamp. Rows where the check column holds a timestamp more recent than the timestamp specified with `--last-value` are imported.



Visualize orders data in mysql

connect to mysql

use retail_db;

select * from orders;

We can see the
last row of
orders table
has order_id
68883

68864	2014-06-18 00:00:00	9634	ON_HOLD
68865	2014-06-19 00:00:00	4567	SUSPECTED_FRAUD
68866	2014-06-20 00:00:00	3890	PENDING_PAYMENT
68867	2014-06-23 00:00:00	869	CANCELED
68868	2014-06-24 00:00:00	10184	PENDING
68869	2014-06-25 00:00:00	7456	PROCESSING
68870	2014-06-26 00:00:00	3343	COMPLETE
68871	2014-06-28 00:00:00	4960	PENDING
68872	2014-06-29 00:00:00	3354	COMPLETE
68873	2014-06-30 00:00:00	4545	PENDING
68874	2014-07-03 00:00:00	1601	COMPLETE
68875	2014-07-04 00:00:00	10637	ON_HOLD
68876	2014-07-06 00:00:00	4124	COMPLETE
68877	2014-07-07 00:00:00	9692	ON_HOLD
68878	2014-07-08 00:00:00	6753	COMPLETE
68879	2014-07-09 00:00:00	778	COMPLETE
68880	2014-07-13 00:00:00	1117	COMPLETE
68881	2014-07-19 00:00:00	2518	PENDING_PAYMENT
68882	2014-07-22 00:00:00	10000	ON_HOLD
68883	2014-07-23 00:00:00	5533	COMPLETE

68883 rows in set (0.04 sec)



Sqoop import - append mode

```
sqoop-import \  
--connect jdbc:mysql://quickstart.cloudera:3306/retail_db \  
--username root \  
--password cloudera \  
--table orders \  
--warehouse-dir /data \  
--incremental append \  
--check-column order_id \  
--last-value 0
```

Incremental mode is **append**

Check column is **order_id**

Last value is **0**



When job is running we can see the below

```
20/04/14 16:27:22 INFO tool.ImportTool: Incremental Import based on column order_id`  
20/04/14 16:27:22 INFO tool.ImportTool: Lower bound value: 0  
20/04/14 16:27:22 INFO tool.ImportTool: Upper bound value: 68883  
20/04/14 16:27:22 WARN manager.MySQLManager: It looks like you are importing from mysql.
```

```
20/04/14 16:27:48 INFO mapreduce.ImportJobBase: Transferred 2.861 MB in 25.6311 seconds (114.3001 KB/sec)  
20/04/14 16:27:48 INFO mapreduce.ImportJobBase: Retrieved 68883 records.  
20/04/14 16:27:48 INFO util.AppendUtils: Creating missing output directory - orders  
20/04/14 16:27:48 INFO tool.ImportTool: Incremental import complete! To run another incremental import of all data following this import, supply the following arguments:  
20/04/14 16:27:48 INFO tool.ImportTool: --incremental append  
20/04/14 16:27:48 INFO tool.ImportTool: --check-column order_id → Check these  
20/04/14 16:27:48 INFO tool.ImportTool: --last-value 68883  
20/04/14 16:27:48 INFO tool.ImportTool: (Consider saving this with 'sqoop job --create')
```



Insert new records in orders table

```
insert into orders values(68884,'2014-07-23 00:00:00',5522,'COMPLETE');  
insert into orders values(68885,'2014-07-23 00:00:00',5522,'COMPLETE');  
insert into orders values(68886,'2014-07-23 00:00:00',5522,'COMPLETE');  
insert into orders values(68887,'2014-07-23 00:00:00',5522,'COMPLETE');  
insert into orders values(68888,'2014-07-23 00:00:00',5522,'COMPLETE');  
insert into orders values(68889,'2014-07-23 00:00:00',5522,'COMPLETE');
```

```
commit;
```




New records inserted successfully

```
mysql> insert into orders values(68884,'2014-07-23 00:00:00',5522,'COMPLETE');  
Query OK, 1 row affected (0.01 sec)  
  
mysql> insert into orders values(68885,'2014-07-23 00:00:00',5522,'COMPLETE');  
Query OK, 1 row affected (0.00 sec)  
  
mysql> insert into orders values(68886,'2014-07-23 00:00:00',5522,'COMPLETE');  
Query OK, 1 row affected (0.00 sec)  
  
mysql> insert into orders values(68887,'2014-07-23 00:00:00',5522,'COMPLETE');  
Query OK, 1 row affected (0.01 sec)  
  
mysql> insert into orders values(68888,'2014-07-23 00:00:00',5522,'COMPLETE');  
Query OK, 1 row affected (0.00 sec)  
  
mysql> insert into orders values(68889,'2014-07-23 00:00:00',5522,'COMPLETE');  
Query OK, 1 row affected (0.01 sec)  
  
mysql> commit;  
Query OK, 0 rows affected (0.00 sec)
```



Run sqoop import to get the delta rows

```
sqoop-import \  
--connect jdbc:mysql://quickstart.cloudera:3306/retail_db \  
--username root \  
--password cloudera \  
--table orders \  
--warehouse-dir /data \  
--incremental append \  
--check-column order_id \  
--last-value 68883
```

Incremental mode is append

Check column is order_id

Last value is 68883



When import is running we can see boundary query and split size.

```
d`  
20/04/14 16:30:18 INFO tool.ImportTool: Lower bound value: 68883  
20/04/14 16:30:18 INFO tool.ImportTool: Upper bound value: 68889  
20/04/14 16:30:18 WARN manager.MySQLManager: It looks like you are importing  
mysql.
```

```
20/04/14 16:30:20 INFO db.DBInputFormat: Using read committed transaction isolation  
20/04/14 16:30:20 INFO db.DataDrivenDBInputFormat: BoundingValsQuery: SELECT MIN(`o  
rder_id`), MAX(`order_id`) FROM `orders` WHERE ( `order_id` > 68883 AND `order_id`  
<= 68889 )  
20/04/14 16:30:20 INFO db.IntegerSplitter: Split size: 1; Num splits: 4 from: 68884  
to: 68889  
20/04/14 16:30:21 INFO mapreduce.JobSubmitter: number of splits:4  
20/04/14 16:30:21 INFO mapreduce.JobSubmitter: Submitting tokens for job: job_15863  
88936914 0022
```



When import is running we can see the last value fetched

Bytes Written=252

```
20/04/14 16:30:41 INFO mapreduce.ImportJobBase: Transferred 252 bytes in 23.1813 seconds (10.8708 bytes/sec)
20/04/14 16:30:41 INFO mapreduce.ImportJobBase: Retrieved 6 records.
20/04/14 16:30:41 INFO util.AppendUtils: Appending to directory orders
20/04/14 16:30:41 INFO util.AppendUtils: Using found partition 4
20/04/14 16:30:41 INFO tool.ImportTool: Incremental import complete! To run another incremental import of all data following this import, supply the following arguments:
20/04/14 16:30:41 INFO tool.ImportTool: --incremental append
20/04/14 16:30:41 INFO tool.ImportTool: --check-column order_id
20/04/14 16:30:41 INFO tool.ImportTool: --last-value 68889
20/04/14 16:30:41 INFO tool.ImportTool: (Consider saving this with 'sqoop job --create')
```



What are counters?

```
20/04/14 16:30:41 INFO mapreduce.Job: Job job_1586388936914_0022 completed successfully
20/04/14 16:30:41 INFO mapreduce.Job: Counters: 30
  File System Counters
    FILE: Number of bytes read=0
    FILE: Number of bytes written=688492
    FILE: Number of read operations=0
    FILE: Number of large read operations=0
    FILE: Number of write operations=0
    HDFS: Number of bytes read=473
    HDFS: Number of bytes written=252
    HDFS: Number of read operations=16
    HDFS: Number of large read operations=0
    HDFS: Number of write operations=8
  Job Counters
    Launched map tasks=4
    Other local map tasks=4
    Total time spent by all maps in occupied slots (ms)=29883
    Total time spent by all reduces in occupied slots (ms)=0
    Total time spent by all map tasks (ms)=29883
    Total vcore-milliseconds taken by all map tasks=29883
    Total megabyte-milliseconds taken by all map tasks=30600192
  Map-Reduce Framework
    Map input records=6
```

Counters are stats related to the job. For example bytes read, bytes written etc.



Sqoop import - lastmodified mode

Before proceeding delete the orders directory inside data.
hadoop fs -rm -R /data/orders

```
sqoop-import \  
--connect jdbc:mysql://quickstart.cloudera:3306/retail_db \  
--username root \  
--password cloudera \  
--table orders \  
--warehouse-dir /data \  
--incremental lastmodified \  
--check-column order_date \  
--last-value 0 \  
--append
```

Incremental mode is lastmodified

Check column is order_date

Last value is 0



When import is running we can see boundary query and split size.

```
20/04/14 16:41:02 INFO tool.ImportTool: Lower bound value: '0'
20/04/14 16:41:02 INFO tool.ImportTool: Upper bound value: '2020-04-14 16:41:02.0'
20/04/14 16:41:03 INFO db.DBInputFormat: Using read committed transaction isolation
20/04/14 16:41:03 INFO db.DataDrivenDBInputFormat: BoundingValsQuery: SELECT MIN(`o
rder_id`), MAX(`order_id`) FROM `orders` WHERE ( `order_date` >= '0' AND `order_dat
e` < '2020-04-14 16:41:02.0' )
20/04/14 16:41:03 INFO db.IntegerSplitter: Split size: 17222; Num splits: 4 from: 1
to: 68889
20/04/14 16:41:03 INFO mapreduce.JobSubmitter: number of splits:4

20/04/14 16:41:27 INFO mapreduce.ImportJobBase: Retrieved 68889 records.
20/04/14 16:41:27 INFO util.AppendUtils: Creating missing output directory - orders
20/04/14 16:41:27 INFO tool.ImportTool: Incremental import complete! To run another
incremental import of all data following this import, supply the following argumen
ts:
20/04/14 16:41:27 INFO tool.ImportTool: --incremental lastmodified
20/04/14 16:41:27 INFO tool.ImportTool: --check-column order_date
20/04/14 16:41:27 INFO tool.ImportTool: --last-value 2020-04-14 16:41:02.0
20/04/14 16:41:27 INFO tool.ImportTool: (Consider saving this with 'sqoop job --cre
ate')
```



Add a few new records & update few records in orders table

```
insert into orders values(68890,current_timestamp,5523,'COMPLETE');  
insert into orders values(68891,current_timestamp,5523,'COMPLETE');  
insert into orders values(68892,current_timestamp,5523,'COMPLETE');  
insert into orders values(68893,current_timestamp,5523,'COMPLETE');  
insert into orders values(68894,current_timestamp,5523,'COMPLETE');
```

```
update orders set order_status='COMPLETE' and order_date =  
current_timestamp WHERE ORDER_ID = 68862;
```

```
commit;
```




Sqoop import - lastmodified mode

Run sqoop import again

```
sqoop-import \  
--connect jdbc:mysql://quickstart.cloudera:3306/retail_db \  
--username root \  
--password cloudera \  
--table orders \  
--warehouse-dir /data \  
--incremental lastmodified \  
--check-column order_date \  
--last-value '2020-04-14 16:41:02' \  
--append
```

Incremental mode is lastmodified

Check column is order_date

Last value is 2020-04-14 16:41:02



When import is running we can see boundary query and split size.

```
20/04/14 16:56:17 INFO tool.ImportTool: Incremental import based on column `order_date`  
20/04/14 16:56:17 INFO tool.ImportTool: Lower bound value: '2020-04-14 16:41:02'  
20/04/14 16:56:17 INFO tool.ImportTool: Upper bound value: '2020-04-14 16:56:17.0'  
20/04/14 16:56:17 WARN manager.MySQLManager: It looks like you are importing from mysql.
```

```
20/04/14 16:56:19 INFO db.DBInputFormat: Using read committed transaction isolation  
20/04/14 16:56:19 INFO db.DataDrivenDBInputFormat: BoundingValsQuery: SELECT MIN(`order_id`), MAX(`order_id`) FROM `orders` WHERE ( `order_date` >= '2020-04-14 16:41:02' AND `order_date` < '2020-04-14 16:56:17.0' )  
20/04/14 16:56:19 INFO db.IntegerSplitter: Split size: 1; Num splits: 4 from: 68890 to: 68894
```



When import is running we can see the last value

Bytes Written=210

```
20/04/14 16:56:41 INFO mapreduce.ImportJobBase: Transferred 210 bytes in 23.7956 seconds (8.8252 bytes/sec)
20/04/14 16:56:41 INFO mapreduce.ImportJobBase: Retrieved 5 records.
20/04/14 16:56:41 INFO util.AppendUtils: Appending to directory orders
20/04/14 16:56:41 INFO util.AppendUtils: Using found partition 4
20/04/14 16:56:41 INFO tool.ImportTool: Incremental import complete! To run another incremental import of all data following this import, supply the following arguments:
20/04/14 16:56:41 INFO tool.ImportTool: --incremental lastmodified
20/04/14 16:56:41 INFO tool.ImportTool: --check-column order_date
20/04/14 16:56:41 INFO tool.ImportTool: --last-value 2020-04-14 16:56:17.0
20/04/14 16:56:41 INFO tool.ImportTool: (Consider saving this with 'sqoop job --create')
```



Sqoop import - lastmodified mode

Sqoop import with last modified, supplied with

--append will result in duplicate records in HDFS directory.

What if we just want the latest records in HDFS?

Then in that case we need to use

----merge-key <merge-column>



Incremental merge-key

The merge tool allows you to combine two datasets where entries in one dataset should overwrite entries of an older dataset. For example, an incremental import run in last-modified mode will generate multiple datasets in HDFS where successively newer data appears in each dataset. The merge tool will "flatten" two datasets into one, taking the newest available records for each primary key.

The merge tool is typically run after an incremental import with the date-last-modified mode (`sqoop import --incremental lastmodified ...`).



Incremental lastmodified merge-key

Run sqoop import again

```
sqoop-import \  
--connect jdbc:mysql://quickstart.cloudera:3306/retail_db \  
--username root \  
--password cloudera \  
--table orders \  
--warehouse-dir /data \  
--incremental lastmodified \  
--check-column order_date \  
--last-value '2020-04-14 16:41:02' \  
--merge-key order_id
```

Incremental mode is lastmodified

Check column is order_date

Last value is 2020-04-14 16:41:02



Incremental lastmodified merge-key

```
20/04/14 17:10:59 INFO mapreduce.ImportJobBase: Transferred 210 bytes in 24.4553 seconds (8.5871 bytes/sec)
20/04/14 17:10:59 INFO mapreduce.ImportJobBase: Retrieved 5 records.
20/04/14 17:10:59 INFO tool.ImportTool: Final destination exists, will run merge job.
20/04/14 17:10:59 INFO Configuration.deprecation: mapred.job.tracker is deprecated. Instead, use mapreduce.jobtracker.address
20/04/14 17:10:59 INFO Configuration.deprecation: mapred.output.key.class is deprecated. Instead, use mapreduce.job.output.key.class
20/04/14 17:10:59 INFO client.RMProxy: Connecting to ResourceManager at /0.0.0.0:8032
20/04/14 17:11:00 INFO input.FileInputFormat
20/04/14 17:11:01 INFO mapreduce.JobSubmitter
20/04/14 17:11:01 INFO mapreduce.JobSubmitter
38936914_0028
20/04/14 17:11:01 INFO impl.YarnClientImpl
38936914_0028
20/04/14 17:11:01 INFO mapreduce.Job: The
Cloudera:8088/proxy/application_15863889369
20/04/14 17:11:01 INFO mapreduce.Job: Runn
20/04/14 17:11:08 INFO mapreduce.Job: Job
de : false
20/04/14 17:11:08 INFO mapreduce.Job: map 0% reduce 0%
```

Once the records are imported then a mapreduce job will run to merge new records with old ones.



Incremental lastmodified merge-key

```
[cloudera@quickstart ~]$ hadoop fs -ls /data/orders
Found 2 items
-rw-r--r--    1 cloudera cloudera           0 2020-04-14 17:12 /data/orders/_SUCCESS
-rw-r--r--    1 cloudera cloudera 3000406 2020-04-14 17:12 /data/orders/part-r-000000
```

```
[cloudera@quickstart ~]$ █
```

We can see that all the files in output folder are now merged in just one single file.



We have learnt Sqoop incremental import

Happy Learning!!!



5 Star Google Rated
Big Data Course

LEARN FROM THE EXPERT



9108179578

Call for more details



Follow US

Trainer Mr. Sumit Mittal

Phone 9108179578

Email trendytech.sumit@gmail.com

Website <https://trendytech.in/courses/big-data-online-training/>

LinkedIn <https://www.linkedin.com/in/bigdatabysumit/>

Twitter @BigdataBySumit

Instagram bigdatabysumit

Facebook <https://www.facebook.com/trendytech.in/>

Youtube TrendyTech