FAQ Sqoop2 Integration tests

- How To run integration tests?
- How to investigate failure of the integration tests?
- I'm running tests on Mac computer and my focus is stolen several times during the test execution
- How to run the integration tests on LocalJobRunner instead of MiniCluster
- How does the integration test suite work?
- How does debug the integration tests?
- What DB does integration tests use today for storing the Sqoop entities?
  - By default it is embedded Derby
- What are the datasets we use in some of the integration tests?
- Where to look for MR Job related logs in the integration tests?
- What happens when integration tests are abruptly terminated due to CTRL + C or failures?
- Unusual Tomcat failed to start issue found?

Some of the well known quirks for the Sqoop2 integration test suite is documented here so that developers can be aware of what to expect when running

How To run integration tests?

We recommend not running integration tests from your IDE - there can be some strange and unexpected errors there.

You can run the entire integration test suite with from the root directory:

```
mvn clean integration-test
```

This will however also run the unit tests and hence it will take some time. If you want to iteratively run only the integration tests (all or just subset), you need to install Sqoop artifacts to your local maven cache:

```
mvn clean install -DskipTests
```

Then you can run just the integration tests with: (This will skip the unit tests)

```
mvn clean integration-test -pl test
```

Assuming that you've installed the Sqoop artifacts into local maven cache, you can run one simple test using: (notice that for one test we're using target test rather then integration-test)

```
mvn clean test -pl test -Dtest=org.apache.sqoop.integration.connector.kafka.FromRDBMSToKafkaTest
```

There are different profiles as well: slow and fast. The fast integration tests run by default, while the slow integration tests need to be explicitly ran:

```
mvn clean integration-test -Dslow
mvn clean integration-test -Dfast
```

How to investigate failure of the integration tests?

Sqoop integration tests are truly integration - we run separate tomcat process with Sqoop 2 server and Hadoop MiniClusters to emulate real cluster as much as possible while still running on one single node. This is however making troubleshooting a bit difficult as various logs are on different places (as would be the case for real cluster). List of various important logs:

<table>
<thead>
<tr>
<th>Location</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>test/target/surefire-reports/${testClass}-output.txt</td>
<td>Main testing log. It will contain any code executed directly from the test Java Class - helper methods creating or inserting data to databases will log here. In addition this log will contain logs from MiniClusters - e.g. logs from HDFS /Mapreduce/YARN daemons.</td>
</tr>
<tr>
<td>test/target/sqoop-cargo-tests/${testClass}/${testName}/sqoop-mini-cluster/log/tomcat.log</td>
<td>Tomcat log's, on production system this might be called catalina.log</td>
</tr>
</tbody>
</table>

Some of the well known quirks for the Sqoop2 integration test suite is documented here so that developers can be aware of what to expect when running

How To run integration tests?

We recommend not running integration tests from your IDE - there can be some strange and unexpected errors there.

You can run the entire integration test suite with from the root directory:

```
mvn clean integration-test
```

This will however also run the unit tests and hence it will take some time. If you want to iteratively run only the integration tests (all or just subset), you need to install Sqoop artifacts to your local maven cache:

```
mvn clean install -DskipTests
```

Then you can run just the integration tests with: (This will skip the unit tests)

```
mvn clean integration-test -pl test
```

Assuming that you've installed the Sqoop artifacts into local maven cache, you can run one simple test using: (notice that for one test we're using target test rather then integration-test)

```
mvn clean test -pl test -Dtest=org.apache.sqoop.integration.connector.kafka.FromRDBMSToKafkaTest
```

There are different profiles as well: slow and fast. The fast integration tests run by default, while the slow integration tests need to be explicitly ran:

```
mvn clean integration-test -Dslow
mvn clean integration-test -Dfast
```

How to investigate failure of the integration tests?

Sqoop integration tests are truly integration - we run separate tomcat process with Sqoop 2 server and Hadoop MiniClusters to emulate real cluster as much as possible while still running on one single node. This is however making troubleshooting a bit difficult as various logs are on different places (as would be the case for real cluster). List of various important logs:

<table>
<thead>
<tr>
<th>Location</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>test/target/surefire-reports/${testClass}-output.txt</td>
<td>Main testing log. It will contain any code executed directly from the test Java Class - helper methods creating or inserting data to databases will log here. In addition this log will contain logs from MiniClusters - e.g. logs from HDFS /Mapreduce/YARN daemons.</td>
</tr>
<tr>
<td>test/target/sqoop-cargo-tests/${testClass}/${testName}/sqoop-mini-cluster/log/tomcat.log</td>
<td>Tomcat log's, on production system this might be called catalina.log</td>
</tr>
</tbody>
</table>
I'm running tests on Mac computer and my focus is stolen several times during the test execution

If you see new Java processes created UI application and stealing the focus, then you should export this property to avoid that:

```bash
export _JAVA_OPTIONS=-Djava.awt.headless=true
```

How to run the integration tests on LocalJobRunner instead of MiniCluster

To run with local mapreduce (faster and theoretically you should be able to attach a debugger):

```bash
mvn clean integration-test -pl test -Dsqoop.hadoop.runner.class=org.apache.sqoop.test.hadoop.HadoopLocalRunner
```

How does the integration test suite work?

Minicluster (pseudo distributed mode)

- They use the Hadoop Minicluster** behind the scenes to simulate the MR execution engine environment.
- Read more about Minicluster [here](http://gdfm.me/2010/08/03/how-to-run-a-minicluster-based-junit-test-with-eclipse/)
- The integration tests are tightly tied to the MR Execution engine at this point. Some rework will be needed to get this working in a Spark execution engine context.

LocalMode (localRunner mode)

- When using this option `-Dsqoop.hadoop.runner.class=org.apache.sqoop.test.hadoop.HadoopLocalRunner`, it does not use the minicluster and much faster.
- [http://wiki.apache.org/hadoop/HowToDebugMapReducePrograms](http://wiki.apache.org/hadoop/HowToDebugMapReducePrograms)

In our code, this is how we detect that it is using localRunner

```java
/**
 * Detect MapReduce local mode.
 */
private boolean isLocal() {
    // If framework is set to YARN, then we can't be running in local mode
    if("yarn".equals(globalConfiguration.get("mapreduce.framework.name"))) {
        return false;
    }
    // If job tracker address is "local" then we're running in local mode
    return "local".equals(globalConfiguration.get("mapreduce.jobtracker.address")) || "local".equals(globalConfiguration.get("mapred.job.tracker"));
}
```

How does debug the integration tests?

//todo:VB

What DB does integration tests use today for storing the Sqoop entities?

By default it is embedded Derby

```java
public class DerbyProvider extends DatabaseProvider {
  @Override
  public void start() {
    // Start embedded server
    try {
      port = NetworkUtils.findAvailablePort();
      LOG.info("Will bind to port " + port);
      server = new NetworkServerControl(InetAddress.getByName("localhost"), port);
      server.start(new LoggerWriter(LOG, Level.INFO));
      // Start won't throw an exception in case that it fails to start, one
      // have to explicitly call ping() in order to verify if the server is
      // up. Check DERBY-1465 for more details.
      server.ping();
    } catch (Exception e) {
      LOG.error("Can't start Derby network server", e);
      throw new RuntimeException("Can't derby server", e);
    }
    super.start();
  }
}

NOTE: Even though there are other providers such as MySQLProvider and PostgreSQLProvider, they are not used in any of the tests.

What are the datasets we use in some of the integration tests?

Anything that extends the following base class

```java
public abstract class DataSet { ..}
```

Where to look for MR Job related logs in the integration tests?

Look under

`/path/to/sqoop2/test/target` under your source folder. Inside each of the MiniMRCluster_XXXX folders there will sub folders and logs.

and

`/path/to/sqoop2/test/target/sqoop-cargo-tests`

For a specific test:

sqoop2/test/target/sqoop-cargo-tests/org.apache.sqoop.integration.connector.jdbc.generic.FromRDBMSToHDFSTest
/testColumns/log/sqoop.log

sqoop2/test/target/sqoop-cargo-tests/org.apache.sqoop.integration.connector.jdbc.generic.FromRDBMSToHDFSTest
/testColumns/log/tomcat.log
What happens when integration tests are abruptly terminated due to CTRL + C or failures?

Please look for zombie java processes and kill them all before running the integration tests. Currently the cluster does not cleanly shutdown.
If zombie java exists, please kill them before starting integration tests next time, because these zombie java processes will hold the ports (such as 12000, 8080, 1527) and the following integration test could not obtain these ports. Then the following integration test will be stuck.

```
ps -ef | grep java
killall -9 java
```

or more advanced....
```
for p in `ps aux | grep java | grep YarnChild| sed -re "s/<username> ([0-9]+) ./\1/"`; do echo $p; kill -9 $p; done
```

Unusual Tomcat failed to start issue found?

First check the tomcat.log under /path/to/sqoop/test/target/sqoop-cargo-tests/ org.apache.sqoop.integration.connector.jdbc.generic.FromRDBMSToHDFSTest/testBasic/log/tomcat.log


Solution: Nuke the directory /var/folders/l8/hyl1hnqj3vq57gdf8f9nb0740000gp/T/cargo