cloudera®

CDH 6.0 Changes for Apache Hive

Important Notice

The information in this document is subject to change without notice. Cloudera shall not be liable for any damages resulting from technical errors or omissions which may be present in this document, or from use of this document.

Cloudera, Inc.
395 Page Mill Road
Palo Alto, CA 94306
info@cloudera.com
US: 1-888-789-1488
Intl: 1-650-362-0488
www.cloudera.com

Release Information

Version: Cloudera Enterprise 6.0.x
Date: February 22, 2019

Table of Contents

Apache Hive Changes in CDH 6.0

In CDH 6.0, Apache Hive has been upgraded to version 2.1, resulting in many added new features and changes that are described in the following topics:

- Apache Hive Components Changes in CDH 6.0 on page 4
- Hive on Spark Changes in CDH 6.0 on page 16

> **Note:** For more information about the release, see the CDH 6 Release Notes. For information about features, components, or functionality that have been deprecated or removed from the CDH 6 release, see Deprecated Items.

Apache Hive Components Changes in CDH 6.0

The following sections cover the changes in CDH 6.0 HiveServer2, Hive metastore, which includes changes to HiveQL syntax, and any API changes:

- Apache Hive Components New Features in CDH 6.0 on page 4
- Apache Hive Components Incompatible Changes in CDH 6.0 on page 7

> **Note:** For more information about the release, see the CDH 6 Release Notes. For information about features, components, or functionality that have been deprecated or removed from the CDH 6 release, see Deprecated Items.

Apache Hive Components New Features in CDH 6.0

The following features have been added to Hive in CDH 6.0:

> **Note:** For more information about the release, see the CDH 6 Release Notes. For information about features, components, or functionality that have been deprecated or removed from the CDH 6 release, see Deprecated Items.

- Query Vectorization Support for Parquet Files on page 4
- Support for UNION DISTINCT on page 5
- Support for NULLS FIRST/NULLS LAST on page 5
- Added Support for Windowing and Analytics Functions on page 5
- Table or Partition Statistics Editing on page 6
- SHOW CREATE DATABASE Support on page 6
- Support for Multiple-Column IN Clause on page 6
- Support for More Hive Functions on page 6

Query Vectorization Support for Parquet Files

By default, the Hive query execution engine processes one row of a table at a time. The single row of data goes through all the operators in the query before the next row is processed, resulting in very inefficient CPU usage. In vectorized query execution, data rows are batched together and represented as a set of column vectors. The query engine then processes these vectors of columns, which greatly reduces CPU usage for typical query operations like scans, filters, aggregates, and joins.

Hive query vectorization is enabled by setting the `hive.vectorized.execution.enabled` property to `true`. In both CDH 5 and CDH 6, this property is set to true by default. But in CDH 5, vectorized query execution in Hive is only possible on ORC-formatted tables, which Cloudera recommends that you do not use for overall compatibility with the

CDH platform. Instead, Cloudera recommends that you use tables in the Parquet format because all CDH components support this format and can be consumed by all CDH components. In CDH 6, query vectorization is supported for Parquet tables in Hive.

For more information, see Enabling Query Vectorization and Apache Parquet Tables with Hive in CDH.

Support for UNION DISTINCT

Support has been added for the `UNION DISTINCT` clause in HiveQL. See HIVE-9039 and the Apache wiki for more details. This feature introduces the following incompatible changes to HiveQL:

- **Behavior in CDH 5:**
  - `SORT BY`, `CLUSTER BY`, `ORDER BY`, `LIMIT`, and `DISTRIBUTE BY` can be specified without delineating parentheses either before a `UNION ALL` clause or at the end of the query, resulting in the following behaviors:
    - When specified before, these clauses are applied to the query before `UNION ALL` is applied.
    - When specified at the end of the query, these clauses are applied to the query after `UNION ALL` is applied.
  - The `UNION` clause is equivalent to `UNION ALL`, in which no duplicates are removed.
- **Behavior in CDH 6:**
  - `SORT BY`, `CLUSTER BY`, `ORDER BY`, `LIMIT`, and `DISTRIBUTE BY` can be specified without delineating parentheses *only* at the end of the query, resulting in the following behaviors:
    - These clauses are applied to the entire query.
    - Specifying these clauses before the `UNION ALL` clause results in a parsing error.
  - The `UNION` clause is equivalent to `UNION DISTINCT`, in which all duplicates are removed.

Support for NULLS FIRST/NULLS LAST

Support has been added for `NULLS FIRST` and `NULLS LAST` options. These options can be used to determine whether null values appear before or after non-null data values when the `ORDER BY` clause is used. Hive follows the SQL:2003 standard for this feature, but the SQL standard does not specify the behavior by default. By default in Hive, null values are sorted as if lower than non-null values. This means that `NULLS FIRST` is the default behavior for ASC order, and `NULLS LAST` is the default behavior for DESC order. See Syntax of Order By on the Apache Hive wiki and HIVE-12994 for further details.

Here are some usage examples:

```
SELECT x.* FROM table1 x ORDER BY a ASC NULLS FIRST;
SELECT x.* FROM table1 x ORDER BY a ASC NULLS LAST;
```

Added Support for Windowing and Analytics Functions

Support for the following has been added to CDH 6.0:

- Using `DISTINCT` with windowing functions. See HIVE-9534 for details.
- Support for `ORDER BY` and a windowing clause when `DISTINCT` is used in a partitioning clause. See HIVE-13453 for details.
- Support to reference aggregate functions within the `OVER` clause. See HIVE-13475 for details.

For further details, see the Apache Language Manual on Windowing and Analytics.

Table or Partition Statistics Editing

Support has been added for editing the statistics information that is stored for a table or a partition. For example, you can run the following statement to set the number of rows for a table to `1000`:

```
ALTER TABLE table1 UPDATE STATISTICS SET ('numRows'='1000');
```

For more information, see HIVE-12730 and the Apache wiki.

SHOW CREATE DATABASE Support

Support has been added for the `SHOW CREATE DATABASE` command, which prints out the DDL statement that was used to create a database:

```
SHOW CREATE DATABASE database1;
```

For more information, see HIVE-11706

Support for Multiple-Column IN Clause

Support has been added so that the `IN` clause in queries operates over multiple column references. The new syntax is boldfaced in the following example:

```
CREATE TABLE test (col1 int, col2 int);
INSERT INTO TABLE test VALUES (1, 1), (1, 2), (2, 1), (2, 2);
SELECT * FROM test;
+------------+------------+
| test.col1  | test.col2  |
+------------+------------+
| 1          | 1          |
| 1          | 2          |
| 2          | 1          |
| 2          | 2          |
+------------+------------+
SELECT * FROM test WHERE (col1, col2) IN ((1, 1));
+------------+------------+
| test.col1  | test.col2  |
+------------+------------+
| 1          | 1          |
+------------+------------+
```

For more information, see HIVE-11600

Support for More Hive Functions

Support has been added for the following Hive UDFs:

`bround`	`chr`	`factorial`
`floor_day`	`floor_hours`	`floor_minute`
`floor_month`	`floor_quarter`	`floor_second`
`floor_week`	`floor_year`	`grouping`
`mask`	`mask_first_n`	`mask_hash`
`mask_last_n`	`mask_show_first_n`	`mask_show_last_n`
`quarter`	`replace`	`sha1`
`sha`	`shiftleft`	`shiftright`

shiftrightunsigned	substring_index	

All built-in Hive functions can be listed with the command SHOW FUNCTIONS; and a short description that explains what a function does is returned with the command DESCRIBE *<function_name>*; For more information about Hive functions, see the Apache wiki and Managing UDFs in the Cloudera enterprise documentation.

Apache Hive Components Incompatible Changes in CDH 6.0

See below for Hive changes that are not backwards compatible in CDH 6.0.

Note: For more information about the release, see the CDH 6 Release Notes. For information about features, components, or functionality that have been deprecated or removed from the CDH 6 release, see Deprecated Items.

UNION ALL Statements Involving Data Types from Different Type Groups No Longer Use Implicit Type Casting

Prior to this change, Hive performed implicit casts when data types from different type groups were specified in queries that use UNION ALL. For example, before CDH 6.0, if you had the two following tables:

Table "one"

```
+------------+------------+------------+--+
| one.col_1  | one.col_2  | one.col_3  |
+------------+------------+------------+--+
| 21         | hello_all  | b          |
+------------+------------+------------+--+
```

Where col_1 datatype is int, col_2 datatype is string, and col_3 datatype is char(1).

Table "two"

```
+------------+------------+------------+--+
| two.col_4  | two.col_5  | two.col_6  |
+------------+------------+------------+--+
| 75.0       | abcde      | 45         |
+------------+------------+------------+--+
```

Where col_4 datatype is double, col_5 datatype is varchar(5), and col_6 datatype is int.

And you ran the following UNION ALL query against these two tables:

```
SELECT * FROM one UNION ALL SELECT col_4 AS col_1, col_5 AS col_2, col_6 AS 
col_3 FROM two;
```

You received the following result set:

```
+------------+------------+------------+--+
| _u1.col_1  | _u1.col_2  | _u1.col_3  |
+------------+------------+------------+--+
| 75.0       | abcde      | 4          |
| 21.0       | hello      | b          |
+------------+------------+------------+--+
```

Note that this statement implicitly casts the values from table one with the following errors resulting in data loss:

- one.col_1 is cast to a double datatype
- one.col_2 is cast to a varchar(5) datatype, which truncates the original value from hello_all to hello
- one.col_3 is cast to a char(1) datatype, which truncates the original value from 45 to 4

In CDH 6.0, no implicit cast is performed across different type groups. For example, STRING, CHAR, and VARCHAR are in one type group, and INT, BIGINT, and DECIMAL are in another type group, and so on. So, in CDH 6.0 and later, the above query that uses UNION ALL, returns an exception for the columns that contain datatypes that are not part of a type group. In CDH 6.0 and later, Hive performs the implicit cast only *within* type groups and not *across* different type groups. For more information, see HIVE-14251.

OFFLINE and NO_DROP Options Removed from Table and Partition DDL

Support for Hive table and partition protection options have been removed in CDH 6.0, which includes removal of the following functionality:

- Support has been removed for:
  - ENABLE | DISABLE NO_DROP [CASCADE]
  - ENABLE | DISABLE OFFLINE
  - ALTER TABLE … IGNORE PROTECTION
- The following support has also been removed from the HiveMetastoreClient class:

  The ignoreProtection parameter has been removed from the dropPartitions methods in the IMetaStoreClient interface.

For more information, see HIVE-11145.

Cloudera recommends that you use Apache Sentry to replace most of this functionality. Although Sentry governs permissions on ALTER TABLE, it does not include permissions that are specific to a partition. See Authorization Privilege Model for Hive and Impala and Configuring the Sentry Service.

DESCRIBE Query Syntax Change

In CDH 6.0 syntax has changed for DESCRIBE queries as follows:

- DESCRIBE queries where the column name is separated by the table name using a period is no longer supported:

```
DESCRIBE testTable.testColumn;
```

  Instead, the table name and column name must be separated with a space:

```
DESCRIBE testTable testColumn;
```

- The partition_spec must appear *after* the table name, but *before* the optional column name:

```
DESCRIBE default.testTable PARTITION (part_col = 100) testColumn;
```

For more details, see the Apache wiki and HIVE-12184.

CREATE TABLE Change: Periods and Colons No Longer Allowed in Column Names

In CDH 6.0, CREATE TABLE statements fail if any of the specified column names contain a period or a colon. For more information, see HIVE-10120 and the Apache wiki.

Reserved and Non-Reserved Keyword Changes in HiveQL

Hive reserved and non-reserved keywords have changed in CDH 6.0. *Reserved keywords* cannot be used as table or column names unless they are enclosed with back ticks (for example, `data`). *Non-reserved keywords* can be used as table or column names without enclosing them with back ticks. Non-reserved keywords have proscribed meanings in HiveQL, but can still be used as table or column names. For more information about the changes to reserved and non-reserved words listed below, see HIVE-6617 and HIVE-14872.

In CDH 6.0, the following changes have occurred with Hive reserved and non-reserved keywords:

- Hive New Reserved Keywords Added in CDH 6.0 on page 9
- Hive Non-Reserved Keywords Converted to Reserved Keywords in CDH 6.0 on page 9
- Hive Reserved Keywords Converted to Non-Reserved Keywords in CDH 6.0 on page 10
- Hive New Non-Reserved Keywords Added in CDH 6.0 on page 10
- Hive Non-Reserved Keyword Removed in CDH 6.0 on page 10

Hive New Reserved Keywords Added in CDH 6.0

The following table contains new reserved keywords that have been added:

COMMIT	CONSTRAINT	DEC	EXCEPT
FOREIGN	INTERVAL	MERGE	NUMERIC
ONLY	PRIMARY	REFERENCES	ROLLBACK
START			

Hive Non-Reserved Keywords Converted to Reserved Keywords in CDH 6.0

The following table contains non-reserved keywords that have been converted to be reserved keywords:

ALL	ALTER	ARRAY	AS
AUTHORIZATION	BETWEEN	BIGINT	BINARY
BOOLEAN	BOTH	BY	CREATE
CUBE	CURSOR	DATE	DECIMAL

DOUBLE	DELETE	DESCRIBE	DROP
EXISTS	EXTERNAL	FALSE	FETCH
FLOAT	FOR	FULL	GRANT
GROUP	GROUPING	IMPORT	IN
INT	INNER	INSERT	INTERSECT
INTO	IS	LATERAL	LEFT
LIKE	LOCAL	NONE	NULL
OF	ORDER	OUT	OUTER
PARTITION	PERCENT	PROCEDURE	RANGE
READS	REGEXP	REVOKE	RIGHT
RLIKE	ROLLUP	ROW	ROWS
SET	SMALLINT	TABLE	TIMESTAMP
TO	TRIGGER	TRUNCATE	UNION
UPDATE	USER	USING	VALUES
WITH	TRUE		

Hive Reserved Keywords Converted to Non-Reserved Keywords in CDH 6.0

The following table contains reserved keywords that have been converted to be non-reserved keywords:

CURRENT_DATE	CURRENT_TIMESTAMP	HOLD_DDLTIME	IGNORE
NO_DROP	OFFLINE	PROTECTION	READONLY

Hive New Non-Reserved Keywords Added in CDH 6.0

The following table contains new non-reserved keywords that have been added:

ABORT	AUTOCOMMIT	CACHE	DAY
DAYOFWEEK	DAYS	DETAIL	DUMP
EXPRESSION	HOUR	HOURS	ISOLATION
KEY	LAST	LEVEL	MATCHED
MINUTE	MINUTES	MONTH	MONTHS
NORELY	NOVALIDATE	NULLS	OFFSET
OPERATOR	RELY	SECOND	SECONDS
SNAPSHOT	STATUS	SUMMARY	TRANSACTION
VALIDATE	VECTORIZATION	VIEWS	WAIT
WORK	WRITE	YEAR	YEARS

Hive Non-Reserved Keyword Removed in CDH 6.0

The following non-reserved keyword has been removed:

DEFAULT

Apache Hive API Changes in CDH 6.0

The following backwards incompatible changes have been made to the Hive API in CDH 6.0:

- AddPartitionMessage.getPartitions() Can Return NULL on page 11
- DropPartitionEvent and PreDropPartitionEvent Class Changes on page 11
- GenericUDF.getTimestampValue Method Now Returns Timestamp Instead of Date on page 11
- GenericUDF.getConstantLongValue Has Been Removed on page 11
- Increased Width of Hive Metastore Configuration Columns on page 11

AddPartitionMessage.getPartitions() Can Return NULL

The `getPartitions()` method has been removed from the `AddPartitionEvent` class in the `org.apache.hadoop.hive.metastore.events` interface. It was removed to prevent out-of-memory errors when the list of partitions is too large.

Instead use the `getPartitionIterator()` method. For more information, see HIVE-9609 and the AddPartitionEvent documentation.

DropPartitionEvent and PreDropPartitionEvent Class Changes

The `getPartitions()` method has been removed and replaced by the `getPartitionIterator()` method in the `DropPartitionEvent` class and the `PreDropPartitionEvent` class.

In addition, the `(Partition partition, boolean deleteData, HiveMetastore.HMSHandler handler)` constructors have been deleted from the `PreDropPartitionEvent` class. For more information, see HIVE-9674 and the PreDropPartitionEvent documentation.

GenericUDF.getTimestampValue Method Now Returns Timestamp Instead of Date

The `getTimestampValue` method in the `GenericUDF` class now returns a `TIMESTAMP` value instead of a `DATE` value. For more information, see HIVE-10275 and the GenericUDF documentation.

GenericUDF.getConstantLongValue Has Been Removed

The `getConstantLongValue` method has been removed from the `GenericUDF` class. It has been noted by the community that this method is not used in Hive. For more information, see HIVE-10710 and the GenericUDF documentation.

Increased Width of Hive Metastore Configuration Columns

The columns used for configuration values in the Hive metastore have been increased in width, resulting in the following incompatible changes in the `org.apache.hadoop.hive.metastore.api` interface.

This change introduced an incompatible change to the `get_table_names_by_filter` method of the `ThriftHiveMetastore` class. Before this change, this method accepts a `string` filter, which allows clients to filter a table by its `TABLEPROPERTIES` value. For example:

```
org.apache.hadoop.hive.metastore.api.hive_metastoreConstants.HIVE_FILTER_FIELD_
       PARAMS + "test_param_1 <> \"yellow\"";

org.apache.hadoop.hive.metastore.api.hive_metastoreConstants.HIVE_FILTER_FIELD_
       PARAMS + "test_param_1 = \"yellow\"";
```

After this change, the `TABLE_PARAMS.PARAM_VALUE` column is now a `CLOB` data type. Depending on the type of database that you use (for example, MySQL, Oracle, or PostgresSQL), the semantics may have changed and operators like "=", "<>", and "!=" might not be supported. Refer to the documentation for your database for more information. You must use operators that are compatible with `CLOB` data types. There is no equivalent "<>" operator that is compatible with `CLOB`. So there is no equivalent operator for the above example that uses the "<>" inequality operator. The equivalent for "=" is the `LIKE` operator so you would rewrite the second example above as:

```
org.apache.hadoop.hive.metastore.api.hive_metastoreConstants.HIVE_FILTER_FIELD_
```

```
        PARAMS + "test_param_1 LIKE \"yellow"";
```

For more information, see HIVE-12274.

Apache Hive Configuration Changes in CDH 6.0

The following backwards incompatible changes have been made to Hive configuration properties in CDH 6.0:

- Bucketing and Sorting Enforced by Default When Inserting Data into Hive Tables on page 12
- Hive Throws an Exception When Processing HDFS Directories Containing Unsupported Characters on page 12
- Hive Strict Checks Have Been Re-factored To Be More Granular on page 13
- Java XML Serialization Has Been Removed on page 13
- Configuration Property Enabling Column Position Usage with GROUP BY and ORDER BY Separated into Two Properties on page 13
- HiveServer2 Impersonation Property (hive.server2.enable.impersonation) Removed on page 14
- Changed Default File Format for Storing Intermediate Query Results on page 14

Bucketing and Sorting Enforced by Default When Inserting Data into Hive Tables

The configuration properties `hive.enforce.sorting` and `hive.enforce.bucketing` have been removed. When set to false, these configurations disabled enforcement of sorted and bucketed tables when data was inserted into a table. Removing these configuration properties effectively sets these properties to `true`. In CDH 6.0, bucketing and sorting are enforced on Hive tables during insertions and cannot be turned off. For more information, see the Apache wiki topic on hive.enforce.bucketing and the topic on hive.enforce.sorting.

Hive Throws an Exception When Processing HDFS Directories Containing Unsupported Characters

Directories in HDFS can contain unprintable or unsupported characters that are not visible even when you run the `hadoop fs -ls` command on the directories. When external tables are created with the `MSCK REPAIR TABLE` command, the partitions using these HDFS directories that contain unsupported characters are unusable for Hive. To avoid this, the configuration parameter `hive.msck.path.validation` has been added. This configuration property controls the behavior of the `MSCK REPAIR TABLE` command, enabling you to set whether validation checks are run on the HDFS directories when `MSCK REPAIR TABLE` is run.

The property `hive.msck.path.validation` can be set to one of the following values:

Value Name	Description
throw	Causes Hive to throw an exception when it tries to process an HDFS directory that contains unsupported characters with the `MSCK REPAIR TABLE` command. This is the default setting for `hive.msck.path.validation`.
skip	Causes Hive to skip the skip the directories that contain unsupported characters, but still repairs the others.
ignore	Causes Hive to completely skip any validation of HDFS directories when the `MSCK REPAIR TABLE` command is run. This setting can cause bugs because unusable partitions are created.

By default, the `hive.msck.path.validation` property is set to `throw`, which causes Hive to throw an exception when `MSCK REPAIR TABLE` is run and HDFS directories containing unsupported characters are encountered. To work around this, set this property to `skip` until you can repair the HDFS directories that contain unsupported characters.

To set this property in Cloudera Manager:

1. In the Admin Console, select the Hive service.
2. Click the **Configuration** tab.
3. Search for the **Hive Service Advanced Configuration Snippet (Safety Valve) for hive-site.xml** setting.

4. In the **Hive Service Advanced Configuration Snippet (Safety Valve) for hive-site.xml** setting, add the **Name** of the property, the **Value** (`throw`, `skip`, or `ignore`), and a **Description** of the setting.
5. Click **Save Changes** and restart the service.

For more information, see HIVE-10722.

Hive Strict Checks Have Been Re-factored To Be More Granular

Originally, the configuration property `hive.mapred.mode` was added to restrict certain types of queries from running. Now it has been broken down into more fine-grained configurations, one for each type of restricted query pattern. The configuration property `hive.mapred.mode` has been removed and replaced with the following configuration properties, which provide more granular control of Hive strict checks:

Configuration Property	Description	Default Value
`hive.strict.checks.bucketing`	When set to `true`, running `LOAD DATA` queries against bucketed tables is not allowed.	`true`. This is a backwards incompatible change.
`hive.strict.checks.type.safety`	When set to `true`, comparing `bigint` to `string` data types or `bigint` to `double` data types is not allowed.	`true`. This is a backwards incompatible change.
`hive.strict.checks.orderby.no.limit`	When set to `true`, prevents queries from being run that contain an `ORDER BY` clause with no `LIMIT` clause.	`false`
`hive.strict.checks.no.partition.filter`	When set to `true`, prevents queries from being run that scan a partitioned table but do not filter on the partition column.	`false`
`hive.strict.checks.cartesian.product`	When set to `true`, prevents queries from being run that contain a Cartesian product (also known as a cross join).	`false`

All of these properties can be set with Cloudera Manager in the following configuration settings for the Hive service:

- **Restrict LOAD Queries Against Bucketed Tables** (`hive.strict.checks.bucketing`)
- **Restrict Unsafe Data Type Comparisons** (`hive.strict.checks.type.safety`)
- **Restrict Queries with ORDER BY but no LIMIT clause** (`hive.strict.checks.orderby.no.limit`)
- **Restrict Partitioned Table Scans with no Partitioned Column Filter** (`hive.strict.checks.no.partition.filter`)
- **Restrict Cross Joins (Cartesian Products)** (`hive.strict.checks.cartesian.product`)

For more information about these configuration properties, see HIVE-12727, HIVE-15148, HIVE-18251, and HIVE-18552.

Java XML Serialization Has Been Removed

The configuration property `hive.plan.serialization.format` has been removed. Previously, this configuration property could be set to either `javaXML` or `kryo`. Now the default is `kryo` serialization, which cannot be changed. For more information, see HIVE-12609 and the Apache wiki.

Configuration Property Enabling Column Position Usage with GROUP BY and ORDER BY Separated into Two Properties

The configuration property `hive.groupby.orderby.position.alias`, which enabled using column position with the `GROUP BY` and the `ORDER BY` clauses has been removed and replaced with the following two configuration properties. These configuration properties enable using column position with `GROUP BY` and `ORDER BY` separately:

Configuration Property Name	Description/Default Setting	Possible Values
`hive.groupby.position.alias`	When set to `true`, specifies that columns can be referenced with their position when using `GROUP BY` clauses in queries. **Default Setting:** `false`. This behavior is turned off by default.	`true` | `false`
`hive.orderby.position.alias`	When set to `true`, specifies that columns can be referenced with their position when using `ORDER BY` clauses in queries. **Default Setting:** `true`. This behavior is turned on by default.	`true` | `false`

For more information, see HIVE-15797 and the Apache wiki entries for configuration properties, GROUP BY syntax, and ORDER BY syntax.

HiveServer2 Impersonation Property (hive.server2.enable.impersonation) Removed

In earlier versions of CDH, the following two configuration properties could be used to set impersonation for HiveServer2:

- `hive.server2.enable.impersonation`
- `hive.server2.enable.doAs`

In CDH 6.0, `hive.server2.enable.impersonation` is removed. To configure impersonation for HiveServer2, use the configuration property `hive.server2.enable.doAs`. To set this property in Cloudera Manager, select the Hive service and click on the **Configuration** tab. Then search for the **HiveServer2 Enable Impersonation** setting and select the checkbox to enable HiveServer2 impersonation. This property is enabled by default in CDH 6.

For more information about this property, see the Apache wiki documentation for HiveServer2 configuration properties.

Changed Default File Format for Storing Intermediate Query Results

The configuration property `hive.query.result.fileformat` controls the file format in which a query's intermediate results are stored. In CDH 6, the default setting for this property has been changed from `TextFile` to `SequenceFile`.

To change this configuration property in Cloudera Manager:

1. In the Admin Console, select the Hive service and click on the **Configuration** tab.
2. Then search for the **Hive Service Advanced Configuration Snippet (Safety Valve) for hive-site.xml** setting and add the following information:
   - **Name**: `hive.query.result.fileformat`
   - **Value**: Valid values are `TextFile`, `SequenceFile` (default), or `RCfile`
   - **Description**: Sets the file format in which a query's intermediate results are stored.
3. After you add this information, click **Save Changes** and restart the Hive service.

For more information about this parameter, see the Apache wiki.

HiveServer2 Thrift API Code Repackaged Resulting in Class File Location Changes

HiveServer2 Thrift API code has been repackaged in CDH 6.0, resulting in the following changes:

- All files generated by the Thrift API for HiveServer2 have moved from the following *old* namespace:

  ```
  org.apache.hive.service.cli.thrift
  ```

  To the following *new* namespace:

  ```
  org.apache.hive.service.rpc.thrift
  ```

- All files generated by the Thrift API for HiveServer2 have moved into a separate jar file called `service-rpc`.

As a result of these changes, all Java classes such as `TCLIService.java`, `TOpenSessionReq.java`, `TSessionHandle.java`, and `TGetSchemasReq.java` have changed locations. For more information, see HIVE-12442.

Values Returned for Decimal Numbers Are Now Padded with Trailing Zeroes to the Scale of the Specified Column

Decimal values that are returned in query results are now padded with trailing zeroes to match the specified scale of the corresponding column. For example, *before* this change, when Hive read a decimal column with a specified scale of 5, the value returned for zero was returned as `0`. *Now*, the value returned for zero is `0.00000`. For more information, see HIVE-12063.

Hive Logging Framework Switched to SLF4J/Log4j 2

The logging framework for Hive has switched to SLF4J (Simple Logging Facade for Java) and now uses Log4j 2 by default. Use of Log4j 1.x, Apache Commons Logging, and `java.util.logging` have been removed. To accommodate this change, write all Log4j configuration files to be compatible with Log4j 2.

For more information, see HIVE-12237, HIVE-11304, and the Apache wiki.

Deprecated Parquet Java Classes Removed from Hive

The deprecated parquet classes, `parquet.hive.DeprecatedParquetInputFormat` and `parquet.hive.DeprecatedParquetOutputFormat` have been removed from Hive because they resided outside of the `org.apache` namespace. Any existing tables that use these classes are automatically migrated to the new SerDe classes when the metastore is upgraded.

Use one of the following options for specifying the Parquet SerDe for new Hive tables:

- Specify in the `CREATE TABLE` statement that you want it stored as Parquet. For example:

```
CREATE TABLE <parquet_table_name> (col1 INT, col2 STRING) STORED AS PARQUET;
```

- Set the `INPUTFORMAT` to `org.apache.hadoop.hive.ql.io.parquet.MapredParquetInputFormat` and set the `OUTPUTFORMAT` to `org.apache.hadoop.hive.ql.io.parquet.MapredParquetOutputFormat`. For example:

```
CREATE TABLE <parquet_table_name> (col1 INT, col2 STRING)
STORED AS
     INPUTFORMAT "org.apache.hadoop.hive.ql.io.parquet.MapredParquetInputFormat"
     OUTPUTFORMAT "org.apache.hadoop.hive.ql.io.parquet.MapredParquetOutputFormat";
```

For more information, see HIVE-6757 and the Apache wiki.

Removed JDBC, Counter-based, and HBase-based Statistics Collection Mechanisms

Support for JDBC, counter-based, and HBase-based statistics collection mechanisms has been removed from Hive. The following configuration properties are no longer supported:

- `hive.stats.dbclass`
- `hive.stats.retries.wait`
- `hive.stats.retries.max`
- `hive.stats.jdbc.timeout`
- `hive.stats.dbconnectionstring`
- `hive.stats.jdbcdrive`
- `hive.stats.key.prefix.reserve.length`

This change also removed the `cleanUp(String keyPrefix)` method from the `StatsAggregator` interface.

Now all Hive statistics are collected on the default file system. For more information, see HIVE-12164, HIVE-12411, HIVE-12005, and the Apache wiki.

S3N Connector Is Removed from CDH 6.0

The S3N connector, which is used to connect to the Amazon S3 file system from Hive has been removed from CDH 6.0. To connect to the S3 file system from Hive in CDH 6.0, you must now use the S3A connector. There are a number of differences between the S3N and the S3A connectors, including configuration differences. See the Apache wiki page on integrating with Amazon Web Services for details.

Migration involves making the following changes:

- Changing all metastore data containing URIs that start with `s3n://` to `s3a://`. This change is performed automatically when you upgrade the Hive metastore.
- Changing all scripts containing URIs that start with `s3n://` to `s3a://`. You must perform this change manually.

Columns Added to TRowSet Returned by the Thrift TCLIService#GetTables Request

Six additional columns have been added to the `TRowSet` that is returned by the `TCLIService#GetTables` request. These columns were added to comply with the official JDBC API. For more information, see the documentation for java.sql.DatabaseMetaData.

The columns added are:

Column Name	Description
REMARKS	Explanatory comment on the table.
TYPE_CAT	Types catalog.
TYPE_SCHEMA	Types schema.
TYPE_NAME	Types name.
SELF_REFERENCING_COL_NAME	Name of the designed identifier column of a typed table.
REF_GENERATION	Specifies how values in the SELF_REFERENCING_COL_NAME column are created.

For more information, see HIVE-7575.

Support Added for Escaping Carriage Returns and New Line Characters for Text Files (LazySimpleSerDe)

Support has been added for escaping carriage returns and new line characters in text files by modifying the `LazySimpleSerDe` class. Without this change, carriage returns and new line characters are interpreted as delimiters, which causes incorrect query results.

This feature is controlled by the SerDe property `serialization.escape.crlf`. It is enabled (set to `true`) by default. If `serialization.escape.crlf` is enabled, 'r' or 'n' cannot be used as separators or field delimiters.

This change only affects text files and removes the `getNullString` method from the `LazySerDeParameters` class. For more information, see HIVE-11785.

Hive on Spark Changes in CDH 6.0

The following new features have been added to Hive on Spark in CDH 6.0:

- Dynamic RDD Caching
- Optimized Hash Tables Enabled

> **Note:** For more information about the release, see the CDH 6 Release Notes. For information about features, components, or functionality that have been deprecated or removed from the CDH 6 release, see Deprecated Items.

Dynamic RDD Caching for Hive on Spark

An optimization has been added to Hive on Spark that enables automatic caching of reused RDDs (Resilient Distributed Datasets). This optimization can improve query performance when the query or sub-query must scan a table multiple times. For example, TPC-DS query 39 is a query that requires multiple table scans. This optimization is disabled by default in CDH 6.0, but can be enabled by setting the `hive.combine.equivalent.work.optimization` property to `true` in the `hive-site.xml` file.

> **Important:** While dynamic RDD caching can improve performance, using Spark's RDD cache may add additional memory pressure to Spark executors. This might increase the chance that a Spark executor runs out of memory and crashes.

To configure this property in Cloudera Manager:

1. In the Admin Console, select the Hive service.
2. Click the Configuration tab.
3. Search for the **HiveServer2 Advanced Configuration Snippet (Safety Valve) for hive-site.xml**.
4. Enter the following property configuration information:

    - **Name**: `hive.combine.equivalent.work.optimization`
    - **Value**: `true`
    - **Description**: `Enables dynamic RDD caching for HoS`

    To disable this configuration, set the **Value** field to `false`.

    To set this configuration property in the XML editor, enter the following code:

```
<property>
     <name>hive.combine.equivalent.work.optimization</name>
     <value>true</value>
     <description>Disables dynamic RDD caching for HoS</description>
</property>
```

5. Click **Save Changes**, and restart the service.

For more information see HIVE-10844 and HIVE-10550.

Optimized Hash Tables Enabled for Hive on Spark

Support has been added for optimized hash tables for Hive on Spark to reduce memory overhead. This feature is enabled by default in CDH 6.0, but can be disabled by setting the `hive.mapjoin.optimized.hashtable` property to `false` in the `hive-site.xml` file. To configure this property in Cloudera Manager:

1. In the Admin Console, select the Hive service.
2. Click the Configuration tab.
3. Search for the **HiveServer2 Advanced Configuration Snippet (Safety Valve) for hive-site.xml**.
4. Enter the following property configuration information:

    - **Name**: `hive.mapjoin.optimized.hashtable`
    - **Value**: `false`
    - **Description**: `Disables optimized hash tables for HoS`

    To enable this configuration, set the **Value** field to `true`.

    To set this configuration property in the XML editor, enter the following code:

```
<property>
     <name>hive.mapjoin.optimized.hashtable</name>
     <value>false</value>
     <description>Disables optimized hash tables for HoS</description>
</property>
```

5. Click **Save Changes**, and restart the service.

For more details, see HIVE-11182 and HIVE-6430.

Hive Unsupported Features in CDH 6.0

The following Hive features are not supported in CDH 6.0:

- AccumuloStorageHandler (HIVE-7068)
- ACID (HIVE-5317)
- Built-in `version()` function is not supported (CDH-40979)
- Cost-based Optimizer (CBO)
- Explicit Table Locking
- HCatalog - HBase plugin
- Hive Authorization (Instead, use Apache Sentry.)
- Hive on Apache Tez
- Hive Local Mode Execution

- Hive Metastore - Derby
- Hive Web Interface (HWI)
- HiveServer1 / JDBC 1
- HiveServer2 Dynamic Service Discovery (HS2 HA) (HIVE-8376)
- HiveServer2 - HTTP Mode (Use THRIFT mode.)
- HPL/SQL (HIVE-11055)
- LLAP (Live Long and Process framework)
- Scalable Dynamic Partitioning and Bucketing Optimization (HIVE-6455)
- Session-level Temporary Tables (HIVE-7090)
- Table Replication Across HCatalog Instances (HIVE-7341)

Note: For more information about the release, see the CDH 6 Release Notes. For information about features, components, or functionality that have been deprecated or removed from the CDH 6 release, see Deprecated Items.

Appendix: Apache License, Version 2.0

SPDX short identifier: Apache-2.0

Apache License
Version 2.0, January 2004
http://www.apache.org/licenses/

TERMS AND CONDITIONS FOR USE, REPRODUCTION, AND DISTRIBUTION

1. Definitions.

"License" shall mean the terms and conditions for use, reproduction, and distribution as defined by Sections 1 through 9 of this document.

"Licensor" shall mean the copyright owner or entity authorized by the copyright owner that is granting the License.

"Legal Entity" shall mean the union of the acting entity and all other entities that control, are controlled by, or are under common control with that entity. For the purposes of this definition, "control" means (i) the power, direct or indirect, to cause the direction or management of such entity, whether by contract or otherwise, or (ii) ownership of fifty percent (50%) or more of the outstanding shares, or (iii) beneficial ownership of such entity.

"You" (or "Your") shall mean an individual or Legal Entity exercising permissions granted by this License.

"Source" form shall mean the preferred form for making modifications, including but not limited to software source code, documentation source, and configuration files.

"Object" form shall mean any form resulting from mechanical transformation or translation of a Source form, including but not limited to compiled object code, generated documentation, and conversions to other media types.

"Work" shall mean the work of authorship, whether in Source or Object form, made available under the License, as indicated by a copyright notice that is included in or attached to the work (an example is provided in the Appendix below).

"Derivative Works" shall mean any work, whether in Source or Object form, that is based on (or derived from) the Work and for which the editorial revisions, annotations, elaborations, or other modifications represent, as a whole, an original work of authorship. For the purposes of this License, Derivative Works shall not include works that remain separable from, or merely link (or bind by name) to the interfaces of, the Work and Derivative Works thereof.

"Contribution" shall mean any work of authorship, including the original version of the Work and any modifications or additions to that Work or Derivative Works thereof, that is intentionally submitted to Licensor for inclusion in the Work by the copyright owner or by an individual or Legal Entity authorized to submit on behalf of the copyright owner. For the purposes of this definition, "submitted" means any form of electronic, verbal, or written communication sent to the Licensor or its representatives, including but not limited to communication on electronic mailing lists, source code control systems, and issue tracking systems that are managed by, or on behalf of, the Licensor for the purpose of discussing and improving the Work, but excluding communication that is conspicuously marked or otherwise designated in writing by the copyright owner as "Not a Contribution."

"Contributor" shall mean Licensor and any individual or Legal Entity on behalf of whom a Contribution has been received by Licensor and subsequently incorporated within the Work.

2. Grant of Copyright License.

Subject to the terms and conditions of this License, each Contributor hereby grants to You a perpetual, worldwide, non-exclusive, no-charge, royalty-free, irrevocable copyright license to reproduce, prepare Derivative Works of, publicly display, publicly perform, sublicense, and distribute the Work and such Derivative Works in Source or Object form.

3. Grant of Patent License.

Subject to the terms and conditions of this License, each Contributor hereby grants to You a perpetual, worldwide, non-exclusive, no-charge, royalty-free, irrevocable (except as stated in this section) patent license to make, have made, use, offer to sell, sell, import, and otherwise transfer the Work, where such license applies only to those patent claims licensable by such Contributor that are necessarily infringed by their Contribution(s) alone or by combination of their Contribution(s) with the Work to which such Contribution(s) was submitted. If You institute patent litigation against any entity (including a cross-claim or counterclaim in a lawsuit) alleging that the Work or a Contribution incorporated within the Work constitutes direct or contributory patent infringement, then any patent licenses granted to You under this License for that Work shall terminate as of the date such litigation is filed.

4. Redistribution.

You may reproduce and distribute copies of the Work or Derivative Works thereof in any medium, with or without modifications, and in Source or Object form, provided that You meet the following conditions:

1. You must give any other recipients of the Work or Derivative Works a copy of this License; and
2. You must cause any modified files to carry prominent notices stating that You changed the files; and
3. You must retain, in the Source form of any Derivative Works that You distribute, all copyright, patent, trademark, and attribution notices from the Source form of the Work, excluding those notices that do not pertain to any part of the Derivative Works; and
4. If the Work includes a "NOTICE" text file as part of its distribution, then any Derivative Works that You distribute must include a readable copy of the attribution notices contained within such NOTICE file, excluding those notices that do not pertain to any part of the Derivative Works, in at least one of the following places: within a NOTICE text file distributed as part of the Derivative Works; within the Source form or documentation, if provided along with the Derivative Works; or, within a display generated by the Derivative Works, if and wherever such third-party notices normally appear. The contents of the NOTICE file are for informational purposes only and do not modify the License. You may add Your own attribution notices within Derivative Works that You distribute, alongside or as an addendum to the NOTICE text from the Work, provided that such additional attribution notices cannot be construed as modifying the License.

You may add Your own copyright statement to Your modifications and may provide additional or different license terms and conditions for use, reproduction, or distribution of Your modifications, or for any such Derivative Works as a whole, provided Your use, reproduction, and distribution of the Work otherwise complies with the conditions stated in this License.

5. Submission of Contributions.

Unless You explicitly state otherwise, any Contribution intentionally submitted for inclusion in the Work by You to the Licensor shall be under the terms and conditions of this License, without any additional terms or conditions. Notwithstanding the above, nothing herein shall supersede or modify the terms of any separate license agreement you may have executed with Licensor regarding such Contributions.

6. Trademarks.

This License does not grant permission to use the trade names, trademarks, service marks, or product names of the Licensor, except as required for reasonable and customary use in describing the origin of the Work and reproducing the content of the NOTICE file.

7. Disclaimer of Warranty.

Unless required by applicable law or agreed to in writing, Licensor provides the Work (and each Contributor provides its Contributions) on an "AS IS" BASIS, WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied, including, without limitation, any warranties or conditions of TITLE, NON-INFRINGEMENT, MERCHANTABILITY, or FITNESS FOR A PARTICULAR PURPOSE. You are solely responsible for determining the appropriateness of using or redistributing the Work and assume any risks associated with Your exercise of permissions under this License.

8. Limitation of Liability.

In no event and under no legal theory, whether in tort (including negligence), contract, or otherwise, unless required by applicable law (such as deliberate and grossly negligent acts) or agreed to in writing, shall any Contributor be liable to You for damages, including any direct, indirect, special, incidental, or consequential damages of any character arising as a result of this License or out of the use or inability to use the Work (including but not limited to damages for loss of goodwill, work stoppage, computer failure or malfunction, or any and all other commercial damages or losses), even if such Contributor has been advised of the possibility of such damages.

9. Accepting Warranty or Additional Liability.

While redistributing the Work or Derivative Works thereof, You may choose to offer, and charge a fee for, acceptance of support, warranty, indemnity, or other liability obligations and/or rights consistent with this License. However, in accepting such obligations, You may act only on Your own behalf and on Your sole responsibility, not on behalf of any other Contributor, and only if You agree to indemnify, defend, and hold each Contributor harmless for any liability incurred by, or claims asserted against, such Contributor by reason of your accepting any such warranty or additional liability.

END OF TERMS AND CONDITIONS

APPENDIX: How to apply the Apache License to your work

To apply the Apache License to your work, attach the following boilerplate notice, with the fields enclosed by brackets "[]" replaced with your own identifying information. (Don't include the brackets!) The text should be enclosed in the appropriate comment syntax for the file format. We also recommend that a file or class name and description of purpose be included on the same "printed page" as the copyright notice for easier identification within third-party archives.

```
Copyright [yyyy] [name of copyright owner]

Licensed under the Apache License, Version 2.0 (the "License");
you may not use this file except in compliance with the License.
You may obtain a copy of the License at

  http://www.apache.org/licenses/LICENSE-2.0

Unless required by applicable law or agreed to in writing, software
distributed under the License is distributed on an "AS IS" BASIS,
WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.
See the License for the specific language governing permissions and
limitations under the License.
```