# Implementing SCDType2 in Hive

## Author

Umesh Pawar
Associate - Projects
Cognizant Technology Solutions
Employee ID: 461757
Created on: December 18, 2015

# Contents

# Slowly Changing Dimension

Dimensions in data management and data warehousing contain relatively static data about such entities as geographical locations, customers or products. A Slowly Changing Dimension (SCD) is a dimension that stores and manages both current and historical data over time in a data warehouse. Data captured by Slowly Changing Dimensions (SCDs) changes slowly but unpredictably, rather than according to a regular schedule.

It is considered and implemented as one of the most critical ETL tasks in tracking the history of dimension records.

The three types of SCDs are:

**Type 1 SCD** - Overwriting

In a Type 1 SCD the new data overwrites the existing data. Thus the existing data is lost as it is not stored anywhere else. Any additional information is not needed to specified to create a Type 1 SCD.

**Type 2 SCD -** Creating another dimension record

A Type 2 SCD retains the full history of values. When the value of a chosen attribute changes, the current record is closed. A new record is created with the changed data values and this new record becomes the current active record. Each record contains the effective time and expiration time to identify the time period between which the record was active.

**Type 3 SCD** - Creating a current value field

A Type 3 SCD stores two versions of values for certain selected level attributes. Each record stores the previous value and the current value of the selected attribute. When the value of any of the selected attributes changes, the current value is stored as the old value and the new value becomes the current value.

## SCD2 logic in Hive

SCD2 by Example:

### Yesterday's snapshot: (main_table)

| ID(Primary Key) | Column1 | Column2 | eff_start_date | eff_end_date |
|---|---|---|---|---|
| 1 | Umesh | IND | 10-12-2014 | 31-12-9999 |
| 2 | Shital | IND | 10-12-2013 | 31-12-9999 |

### Today's source data: (delta_table)

Where today's business date is 25-12-2015

| ID(Primary Key) | Column1 | Column2 |
|---|---|---|
| 1 | Umesh | USA |
| 2 | Shital | IND |
| 3 | Dinesh | IND |

Update

No Change (discard)

Insert

### Expected Output of SCDType2 phase: (main_table)

| ID(Primary Key) | Column1 | Column2 | eff_start_date | eff_end_date |
|---|---|---|---|---|
| 1 | Umesh | USA | 25-12-2015 | 31-12-9999 |
| 1 | Umesh | IND | 10-12-2014 | 25-12-2015 |
| 2 | Shital | IND | 10-12-2013 | 31-12-9999 |
| 3 | Dinesh | IND | 25-12-2015 | 31-12-9999 |

# Pseudo code to implement SCDType2 in Scalding:

**Assumptions:**

1. Daily Incremental data has only Insert And Update records

**Steps:**

1. Perform left outer join on snapshot table (main_table) with daily delta table(delta_table) to identify updates and logically close old record.
2. Perform union all operation to merge the record got from step-1 and daily delta table(delta_table)
3. Insert step-2 result set into _temp table (temporary table)
4. Drop/archive main_table as per your requirement
5. Rename _temp table to main_table
6. You are done with scdType2

**Queries:**

**##HQL for main table**

```
CREATE EXTERNAL TABLE main_table(
id STRING,
column1 STRING,
column2 STRING,
eff_start_date STRING,
eff_end_date STRING)
STORED AS PARQUET
LOCATION '/warehouse/projects/bigdata/main_table';
```

**##HQL for delta table**

```
CREATE EXTERNAL TABLE delta_table(
id STRING,
column1 STRING,
column2 STRING
)
STORED AS PARQUET
LOCATION '/warehouse/projects/bigdata/delta_table';
```

**##HQL for temp table**

```
CREATE EXTERNAL TABLE temp_table(
id STRING,
column1 STRING,
column2 STRING,
eff_start_date STRING,
eff_end_date STRING)
STORED AS PARQUET
LOCATION '/warehouse/projects/bigdata/temp_table';
```

**## HQL to perform SCDTYPE2**

```
INSERT OVERWRITE TABLE temp_table
{{SELECT id,
column1,
column2,
${CURRENT_BUS_DATE}, --update value of current business date from variable
"9999-12-31" --high end date to make it active
FROM delta_table
}
UNION ALL
{ SELECT id,
column1,
column2,
eff_start_date,
${CURRENT_BUS_DATE} --logically closing records with current business date
FROM main_table main
LEFT OUTER JOIN delta_table delta
WHERE (main.id=delta.id && main.column1 != delta.column1 && main.column2 != delta.column2)
}}
```

**##HQL to drop main table**

```
DROP TABLE main_table;
```

**##HQL to rename temp table to main table.**

```
ALTER TABLE temp_table RENAME TO main_table;
```