**Session 23**

**Assignment 2**

**Problem Statement:**

Explain Primary data types and complex data types in Hive with an example in brief.

**Part 1: Primary Datatypes:**

**Numeric Types**

* [**TINYINT**](https://cwiki.apache.org/confluence/display/Hive/LanguageManual+Types#LanguageManualTypes-tinyint) (1-byte signed integer, ranges from -128 to 127)

**Example:** 100

* [**SMALLINT**](https://cwiki.apache.org/confluence/display/Hive/LanguageManual+Types#LanguageManualTypes-smallint) (2-byte signed integer, ranges from -32,768 to 32,767)

**Example:** -20, 15

* [**INT**](https://cwiki.apache.org/confluence/display/Hive/LanguageManual+Types#LanguageManualTypes-int)**/INTEGER** (4-byte signed integer, ranges from -2,147,483,648 to 2,147,483,647) **Example:** 2020
* [**BIGINT**](https://cwiki.apache.org/confluence/display/Hive/LanguageManual+Types#LanguageManualTypes-bigint) (8-byte signed integer, ranges from -9,223,372,036,854,775,808(-9.2\*1018) to 9,223,372,036,854,775,807(9.2\*1018) )

**Example:** 15\*105

* **FLOAT** (4-byte single precision floating point number, ranges from 1.4e-45 to 3.4e38) **Example:** 15.26
* **DOUBLE** (8-byte double precision floating point number, ranges from 4.9e-.324 to 1.79e308)

**Example:** 0.79e32

* **DOUBLE PRECISION** (alias for DOUBLE, only available starting with Hive [2.2.0](https://issues.apache.org/jira/browse/HIVE-13556))
* [**DECIMAL**](https://cwiki.apache.org/confluence/display/Hive/LanguageManual+Types#LanguageManualTypes-decimal) (17-bytes precision up to 38 digits, ranges from -1038 +1 to 1038 -1)

**Example:** DECIMAL(10,3)

* + Introduced in Hive [0.11.0](https://issues.apache.org/jira/browse/HIVE-2693) with a precision of 38 digits.
  + Hive [0.13.0](https://issues.apache.org/jira/browse/HIVE-3976) introduced user definable precision and scale.

**Date/Time Types**

* [**TIMESTAMP**](https://cwiki.apache.org/confluence/display/Hive/LanguageManual+Types#LanguageManualTypes-timestamp) (This is a standard way of displaying the time. Format used to display Timestamp is ‘yyyy-mm-dd hh:mm:ss’. Note: Only available starting with Hive [0.8.0](https://issues.apache.org/jira/browse/HIVE-2272))
* [**DATE**](https://cwiki.apache.org/confluence/display/Hive/LanguageManual+Types#LanguageManualTypes-date)(This value is represented in the form ‘yyyy-mm-dd’, ranges from ‘0000-01-01’ to ‘9999-12-31’. Basically stored as String. Note: Only available starting with Hive [0.12.0](https://issues.apache.org/jira/browse/HIVE-4055)) **Example:** ‘2014-11-19’
* [**INTERVAL**](https://cwiki.apache.org/confluence/display/Hive/LanguageManual+Types#LanguageManualTypes-Intervals) (This is used often in finding the difference between two Dates or Timestamps. Note: Only available starting with Hive [1.2.0](https://issues.apache.org/jira/browse/HIVE-9792))

**String Types**

* [**STRING**](https://cwiki.apache.org/confluence/display/Hive/LanguageManual+Types#LanguageManualTypes-string) (Sequence of characters. Either single quotes( ‘ )or double quotes( “ ) can be used to enclose the characters) **Example:** “Welcome”
* [**VARCHAR**](https://cwiki.apache.org/confluence/display/Hive/LanguageManual+Types#LanguageManualTypes-varchar) (The maximum length of VARCHAR is specified in braces. Similar to SQL’s VARCHAR the maximum length allowed is 65355 bytes. Note: Only available starting with Hive [0.12.0](https://issues.apache.org/jira/browse/HIVE-4844))

**Example:** ‘gmail.com’

* [**CHAR**](https://cwiki.apache.org/confluence/display/Hive/LanguageManual+Types#LanguageManualTypes-char) (This is similar to the SQL’s CHAR with fixed length. That is values shorter than the specified length are padded with spaces. Note: Only available starting with Hive [0.13.0](https://issues.apache.org/jira/browse/HIVE-5191))

**Example:** ‘Hi’

**Miscellaneous Types**

Hive supports two more primitive datatype that falls under miscellaneous datatype. They are:

* **BOOLEAN** (In hive Boolean stores ‘TRUE’ or ‘FALSE’ values only)
* **BINARY** (In hive Binary is an array of bytes. Binary columns are stored within the records. The arbitrary bytes can be included in Binary column and these bytes are not parsed as numbers or strings by Hive. Note: Only available starting with Hive [0.8.0](https://issues.apache.org/jira/browse/HIVE-2380))

**Part 2: Complex Datatypes:**

The Complex datatype can be built up from primitive and other composite datatypes.

Generally Hive supports 4 types of complex data types

* ARRAY
* MAP
* STRUCT
* UNIONTYPE

**Hive Array:** It behaves same as Java Array. It is an ordered sequence or collection of similar datatype elements. All the elements in an array are indexable using zero-based integers. **Example:** array (‘one’, ‘two’, ‘three’)

**Hive Map:** It is an unordered collection of key-value pairs. Keys must be of primitive types. Values can be of any type. The fields can be accessed using array notation of keys.

**Example:** map(‘first’, ‘name’, ‘last’, ‘job’). In this ‘name’ can be accessed with map[‘first’]

**Hive STRUCT:** It is similar to STRUCT in C language. It is a collection of elements of different types. Any data type can be used to specify this STRUCT data type. Elements in STRUCT type are accessed using the DOT (.) notation.

**Example:** For a column x of type STRUCT {a int; b int} the field is accessed by the notation ‘x.a’.

**Hive UNIONTYPE:** It is collection of Heterogeneous data types. It is similar to Unions in C. At any point of time, a UnionType can hold any one (exactly one) data type from its specified data types.