**Assignment 23.2**

Top of Form

Bottom of Form

Top of Form

Bottom of Form

Top of Form

Bottom of Form

|  |
| --- |
|  |
| Hive data types are categorized into two types.   * **Primary data types** * **Complex data types** | |
|  | |
|  | |

|  |
| --- |
| * **Primary data types:** prime data types contains following types. |
|  |
|  |

|  |
| --- |
| • Numeric Types |
|  |

|  |
| --- |
| • String Types |
|  |

|  |
| --- |
| • Date/Time Types |
|  |

|  |
| --- |
| • Miscellaneous Types |
|  |

|  |
| --- |
|  |
|  |

|  |
| --- |
| **Numeric Data Types** |
|  |

|  |
| --- |
| • Integral types are – TINYINT, SMALLINT, INT & BIGINT |
| Example :100,1000,50000 |
|  |

|  |
| --- |
| • Floating types are – FLOAT, DOUBLE & DECIMAL. |
| Example :1500.00,750000.00,DECIMAL(5,2) |
|  |

|  |
| --- |
|  |
|  |

|  |
| --- |
|  |
|  |

|  |
| --- |
|  |
|  |

|  |
| --- |
| **String Data Types** |
|  |

|  |
| --- |
| • STRING |
|  |

|  |
| --- |
| String literals can be expressed with either single quotes (') or double quotes (") |
| Example : ‘Hello…welcome to hadoop session’ |
|  |

|  |
| --- |
|  |
|  |

|  |
| --- |
| • VARCHAR |
|  |

|  |
| --- |
| Varchar types are created with a length specifier (between 1 and 65355), which defines the maximum number of characters allowed in the character string. |
|  |

|  |
| --- |
|  |
|  |

|  |
| --- |
| • CHAR |
|  |

|  |
| --- |
| Char types are similar to Varchar but they are fixed-length meaning that values shorter than the specified length value are padded with spaces but trailing spaces are not important during comparisons. |
|  |

|  |
| --- |
|  |
|  |

|  |
| --- |
| **Date/Time Types** |
|  |

|  |
| --- |
|  |
|  |

|  |
| --- |
| • Hive provides DATE and TIMESTAMP data types in traditional UNIX time stamp format for date/time related fields in hive. |
|  |

|  |
| --- |
|  |
|  |

|  |
| --- |
| • DATE values are represented in the form YYYY-MM-DD. Date ranges allowed are 0000-01-01 to 9999-12-31. |
|  |

|  |
| --- |
|  |
|  |

|  |
| --- |
| • TIMESTAMP use the format yyyy-mm-dd hh:mm:ss[.f...]. |
|  |

|  |
| --- |
|  |
|  |

|  |
| --- |
| • We can also cast the String, Time-stamp values to Date format if they match format. |
|  |

|  |
| --- |
|  |
|  |

|  |
| --- |
| Miscellaneous Types |
|  |

|  |
| --- |
|  |
|  |

|  |
| --- |
| • Hive supports two more primitive data types, BOOLEAN and BINARY. Similar to Java’s Boolean, BOOLEAN in hive stores true or false values only. |
|  |

|  |
| --- |
|  |
|  |

|  |
| --- |
| • BINARY is an array of Bytes and similar to VARBINARY in many RDBMSs |
| Example: DATE ‘2014-12-07’. |
|  |
|  |

|  |
| --- |
|  |
|  |

|  |
| --- |
|  |
|  |

|  |
| --- |
| * **Complex data types:** |
| * These data types are built on using the primitive data types. |
|  |

|  |
| --- |
| * Currently Hive supports four complex data types. They are: |
|  |

|  |
| --- |
| **ARRAY** |
|  |

|  |
| --- |
| **MAP** |
|  |

|  |
| --- |
| **STRUCT** |
|  |

|  |
| --- |
| **UNIONTYPE** |
|  |
|  |

|  |
| --- |
|  |
| * **ARRAY**   ARRAY<datatype> | |
| Contain a list of elements of the same data type. These elements are accessed by using an index. | |
|  | |

|  |
| --- |
|  |
|  |

|  |
| --- |
|  |
|  |

|  |
| --- |
|  |

|  |
| --- |
| It is similar to arrays in Java. |
|  |

|  |
| --- |
| Example :  An array, “fruits”, containing a list of elements [‘apple’, ’mango’, ‘orange’], the element “apple” in the array can be accessed by specifying fruits[1]. |
|  |

|  |
| --- |
|  |

|  |
| --- |
|  |
|  |

|  |
| --- |
| * **MAP** |
|  |

|  |
| --- |
| MAP<primitive\_type,data\_type> |
|  |

|  |
| --- |
| Collection of key-value pairs. |
|  |

|  |
| --- |
| Fields are accessed using array notation of keys ([‘key’]).  Example : a map, “pass\_list” containing the “user name” as key and “password” as value, the password of the user can be accessed by specifying pass\_list[‘username’] |
|  |
|  |

|  |
| --- |
|  |
|  |

|  |
| --- |
| * **STRUCT** |
|  |

|  |
| --- |
| STRUCT <col\_name:data\_type[COMMENT col\_comment] |
|  |

|  |
| --- |
| It is similar to STRUCT in C language. |
|  |

|  |
| --- |
| It is a record type which encapsulates a set of named fields that can be any primitive data type. |
|  |

|  |
| --- |
| Elements in STRUCT type are accessed using the DOT (.) notation. |
|  |

|  |
| --- |
| Example – For a column c of type STRUCT {d INT; f INT} the a field is accessed by the expression c.a  In a struct, ”car”, the color of the car can be retrieved as specifying car.color. |
|  |

|  |
| --- |
|  |
|  |

|  |
| --- |
| * **UNIONTYPE** |
|  |

|  |
| --- |
| UNIONTYPE <data\_type,data\_type,….> |
|  |

|  |
| --- |
| It is similar to Unions in C. |
|  |

At any point of time, an Union Type can hold any one (exactly one) data type from its specified data types.

* The create table statement containing the complex type is shown below.

CREATE TABLE complex\_data\_types

(

vegetables ARRAY<string>,

Pass\_list MAP<string,string>,

car STRUCT<color:string, wheel\_size:float>

);