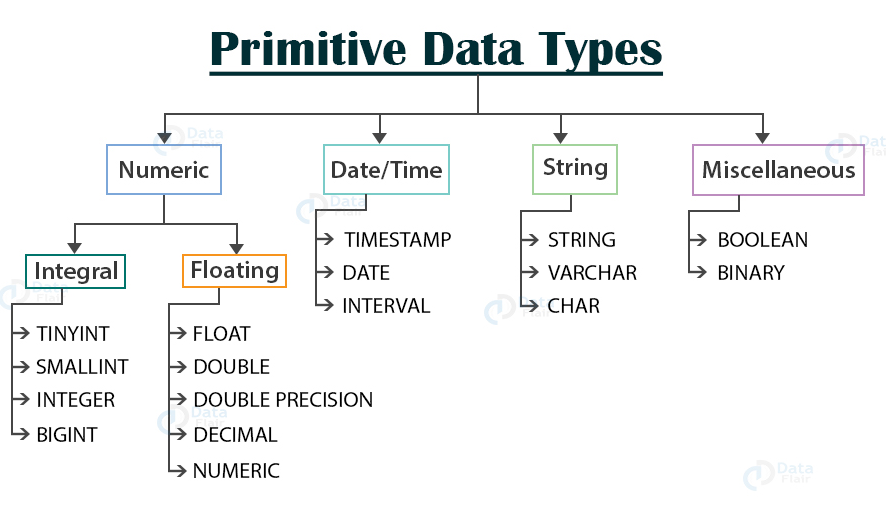
# Hive Data Types

Data Types in Hive specifies the column/field type in the Hive table. It specifies the type of values that can be inserted into the specified column.

## Hive Primitive Data Type



### Numeric Type

The Numeric data type in Hive is categorized into

* Integral data type
* Floating data type

#### Integral data type

****a. TINYINT**** – (1-byte signed integer ranging from -128 to 127)

****b. SMALLINT**** – (2-byte signed integer ranging from -32, 768 to 32, 767)

****c. INTEGER**** – (4-byte signed integer ranging from -2, 147, 483, 648 to 2, 147, 483, 647)

****d. BIGINT**** – (8-byte signed integer ranging from -9, 223, 372, 036, 854, 775, 808 to 9, 223, 372, 036, 854, 775, 807)

In Hive, Integral literals are assumed to be INTEGER by default unless they cross the range of INTEGER values. If we want to use a low integral value like 100 to be treated as TINYINT, SMALLINT, or BIGINT, then we will use the following postfixes (shown in the below table) with the number.

|  |  |  |
| --- | --- | --- |
| **Type** | **Postfix** | **Example** |
| TINYINT | Y | 100Y |
| SMALLINT | S | 100S |
| BIGINT | L | 100L |

#### Floating data type

****a. FLOAT****

It is a 4-byte single-precision floating-point number.

****b. DOUBLE****

It is an 8-byte double-precision floating-point number.

****c. DOUBLE PRECISION****

It is an alias for DOUBLE. It is only available starting with Hive 2.2.0

****d. DECIMAL****

It was introduced in Hive 0.11.0. It is based on Java’s BigDecimal. DECIMAL types support both scientific and non-scientific notations.

In Hive 0.11.0 and 0.12, the precision of the DECIMAL type is fixed and limited to 38 digits.

As of Hive 0.13, user can specify the scale and precision during table creation using the syntax:

**DECIMAL**(precision, scale)

If precision is not specified, then by default, it is equal to 10.

If the scale is not specified, then by default, it is equal to 0.

DECIMAL provides more precise values and greater range than DOUBLE.

****e. NUMERIC****

It started with [Hive 3.0.0.](https://issues.apache.org/jira/browse/HIVE-16764) The NUMERIC data type is the same as the DECIMAL type.

### Date/Time data type:

****a. TIMESTAMP****

Timestamps were introduced in Hive 0.8.0. It supports traditional UNIX timestamp with the optional nanosecond precision.

The supported Timestamps format is yyyy-mm-dd hh:mm:ss[.f…] in the text files.

If they are in any other format, declare them as the appropriate type and use **[UDF(User Defined Function)](https://data-flair.training/blogs/hive-udf/)** to convert them to timestamps.

The supported conversions are:

|  |  |
| --- | --- |
| **Integer numeric type** | UNIX timestamp in seconds |
| **Floating-point numeric type** | UNIX timestamp in seconds with decimal precision |
| **Strings** | java.sql.Timestamp format “YYYY-MM-DD HH:MM:SS.fffffffff” (9 decimal place precision) |

****b. DATE****

Dates were introduced in Hive 0.12.0. DATE value describes a particular year/month/day in the form of YYYY-MM-DD.

For example- DATE ‘2020-02-04’

It does not have a time of day component. The range of value supported for the DATE type is 0000-01-01 to 9999-12-31.

****c. INTERVAL****

Hive Interval data types are available only after starting with Hive version 1.2 or above.

Hive accepts the interval syntax with unit specifications. We have to specify the units along with the interval value.

For example, INTERVAL ‘1’ DAY refers to the day time.

### String data type

****a. STRING****

In Hive, String literals are represented either with the single quotes(‘ ’) or with double-quotes(“ ”).

Hive uses C-style escaping.

****b. VARCHAR****

In Hive, VARCHAR data types are of different lengths, but we have to specify the maximum number of characters allowed in the character string.

If the string value assigned to the varchar is less than the maximum length, then the remaining space will be freed out.

Also, if the string value assigned is more than the maximum length, then the string is silently truncated.

The length of the varchar is between(****1 to 65535****).

Trailing whitespace is important in varchar and will affect the comparison results.

****c. CHAR****

CHAR data types are fixed-length.

The values shorter than the specified length are padded with the spaces.

Unlike VARCHAR, trailing spaces are not significant in CHAR types during comparisons.

The maximum length of CHAR is fixed at ****255.****

### Miscellaneous data type

****a. BOOLEAN****

Boolean types in Hive store either true or false.

****b. BINARY****

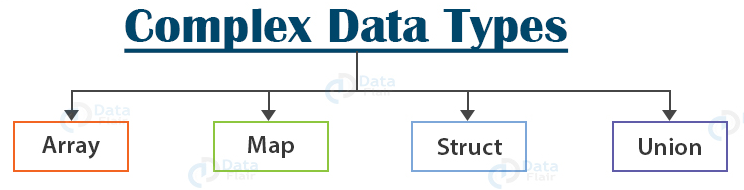
BINARY type in Hive is an array of bytes.

This is all about Hive Primitive Data Types. Let us now study Hive Complex Data Types.

## **Hive Complex Data Type**

Complex Data Types are built on the top of Primitive Data Type.

The Hive Complex Data Type are categorized as:



#### 1. arrays

Array in Hive is an ordered sequence of similar type elements that are indexable using the zero-based integers.

Arrays in Hive are similar to the arrays in JAVA.

array<datatype>

****Example****: array(‘Data’,’Flair’). The second element is accessed as array[1].

#### 2. maps

Map in Hive is a collection of key-value pairs, where the fields are accessed using array notations of keys (e.g., [‘key’]).

map<primitive\_type, data\_type>

****Example:**** ‘first’ -> ‘John’, ‘last’ -> ‘Deo’, represented as map(‘first’, ‘John’, ‘last’, ‘Deo’). Now ‘John’ can be accessed with map[‘first’].

#### 3. structs

STRUCT in Hive is similar to the STRUCT in C language. It is a record type that encapsulates a set of named fields, which can be any primitive data type.

We can access the elements in STRUCT type using DOT (.) notation.

STRUCT <col\_name : data\_type [ COMMENT col\_comment], ...>

****Example:**** For a column c3 of type STRUCT {c1 INTEGER; c2 INTEGER}, the c1 field is accessed by the expression c3.c1.

#### 4. union

UNION type in Hive is similar to the UNION in C. UNION types at any point of time can hold exactly one data type from its specified data types.

The full support for UNIONTYPE data type in Hive is still incomplete.

UNIONTYPE<data\_type, data\_type, ...>

## **Handling of NULL Values**

In Hive data types, the missing values are represented by the special value NULL.