**Assignment 29.3**

Explain Brief of the following in brief

● Hive UDF

● Hive UDAF

● Hive UDTF

● Thrift server

**● Hive UDF**

UDFs works on a single row in a table and produces a single row as output. Its one to one relationship between input and output of a function. **e.g Hive built in TRIM() function.**

Hive allows us to define our own UDFs as well.

**How to Write a UDF function in Hive?**

To write a UDF, start by extending the UDF class and implements and the evaluate() function. • During query processing, an instance of the class is instantiated for each usage of the function in a query. • The evaluate() is called for each input row. • The result of evaluate() is returned to Hive. • It is legal to overload the evaluate method. • Hive will pick the method that matches in a similar way to Java method overloading. • Finally to use UDF, create jar and register the class as temporary function.

ADD JAR ; --ADD JAR /home/acadgild/hive/hive-udf.jar; CREATE TEMPORARY FUNCTION as ''; --CREATE TEMPORARY FUNCTION up AS 'udf.ToUpper';

**● Hive UDAF**

**UDAF:** User defined aggregate functions works on more than one row and gives single row as output. **e.g Hive built in MAX() or COUNT() functions**. here the relation is many to one. Lets say we have a table with students name, id and total marks, so here if we have 10 rows in the table and if we have to find student who got maximum number then our query need to check each 10 row to find the maximum but ultimately we get only one output which is the maximum. Hope this justifies the many to one relationship.Hive allows us to define our own UDAFs.

**● Hive UDTF**

**UDTF:** User defined tabular function works on one row as input and returns multiple rows as output. So here the relation in one to many. **e.g Hive built in EXPLODE() function.**Now lets take an array column USER\_IDS as ARRAY10,12,5,45> then SELECT EXPLODE(USER\_IDS) as ID FROM T\_USER. will give 10,12,5,45 as four different rows in output. UDTF can be used to split a column into multiple column. Here alias "AS" clause is mandatory .

**Steps to Create UDTFs in Hive:**

We can create a custom [Hive UDTF](https://acadgild.com/big-data/big-data-development-training-certification) by extending the GenericUDTF abstract class and then implementing the initialize, process, and possibly close methods.

**Initialize()**

The [Hive calls](https://acadgild.com/big-data/big-data-development-training-certification) the initialize method to notify the UDTF the argument types to expect. The UDTF must then return an object inspector corresponding to the row objects that the UDTF will generate.

**Process()**

Once initialize() method has been called, Hive will give rows to the UDTF using the process() method. While in process() function, the UDTF can produce and forward rows to other operators by calling forward() method.

**Close()**

Finally, Hive will call the close() method when all the rows have passed to the UDTF. This function allows for any cleanup that is necessary before returning from the User Defined Table Generating Function. It is important to note that we cannot write any records from this function.

● **Thrift server**

**Thrift** is an [interface definition language](https://en.wikipedia.org/wiki/Interface_definition_language) and [binary communication protocol](https://en.wikipedia.org/wiki/Binary_protocol)that is used to define and create [services](https://en.wikipedia.org/wiki/Service_(systems_architecture)) for numerous languages.It is used as a [remote procedure call](https://en.wikipedia.org/wiki/Remote_procedure_call) (RPC) framework and was developed at [Facebook](https://en.wikipedia.org/wiki/Facebook) for "scalable cross-language services development". It combines a software stack with a code generation engine to build [cross-platform](https://en.wikipedia.org/wiki/Cross-platform) services that can connect applications written in a variety of languages and frameworks,  Although developed at Facebook, it is now an [open source](https://en.wikipedia.org/wiki/Open_source) project in the [Apache Software Foundation](https://en.wikipedia.org/wiki/Apache_Software_Foundation).

Thrift includes a complete stack for creating clients and servers.[[8]](https://en.wikipedia.org/wiki/Apache_Thrift#cite_note-Architecture-8) The top part is generated code from the Thrift definition. From this file, the services generate client and processor code. In contrast to built-in types, created [data structures](https://en.wikipedia.org/wiki/Data_structure) are sent as result in generated code. The protocol and [transport layer](https://en.wikipedia.org/wiki/Transport_layer) are part of the [runtime library](https://en.wikipedia.org/wiki/Runtime_library). With Thrift, it is possible to define a service and change the protocol and transport without recompiling the code. Besides the client part, Thrift includes server infrastructure to tie protocols and transports together, like blocking, non-blocking, and multi-threaded servers. The underlying I/O part of the stack is implemented differently for different languages.