ADVANCED HIVE

DS8003 – MGT OF BIG DATA AND TOOLS

Ryerson University

Instructor: Kanchana Padmanabhan

Today's Outline

1. Complex Data Types
2. Partitioning
3. Bucketing
4. Join Algorithms
5. Advanced Functions

Hive Complex Data Types

Complex Data Types

Complex Type	*Description*	*Literal Syntax*
Array	*Ordered sequences of the same type that are indexable using zero-based integers. Only single dimensional Arrays are supported*	*array(lat,lon)* *[-48.01234, 93.44444]* *location[0] → -48.01234*

Map	*An unordered collection of key-value tuples.*	*map('lat', lat, 'lon', lon)*
		{'lat': -48.01234, 'lon': 93.44444 }
		location['lat'] → -48.01234

Struct (Think! Mini Table)	*More structured data type, like a table. Fields can be accessed using the "dot" notation*	*struct(lat, lon)*
		{'lat': -48.01234, 'lon': 93.44444 }
		location.lat → -48.01234

Arrays

- It is an ordered collection of elements.The elements in the array must be of the same type.
- **array** allows you to store *n* number of values of the same data type
- Array Index uses zero-based integers
- Create table with column of type Array

```
create table test_array(product bigint , product_colors array<string>)
row format delimited
fields terminated by ','
 collection items terminated by '$';
```

- Load into Table from File

```
Load data  inpath '/user/root/array_test.txt' overwrite into table test_array;
```

- Query Array Column

```
Select product_colors from test_array;

Select product_colors[0], product_colors[1] from test_array;
```

Array Functions

- size(Array<T>)

  Select size(product_colors) from test_array;

- array_contains(Array<T>) (Boolean: True/False)

  Select array_contains(product_colors, ‘red’) from test_array;

- sort_array(Array<T>)

  Select sort_array(product_colors) from test_array;

Explode (Built-in Table-Generating Functions (UDTF))

explode() takes in an array (or a map) as an input and outputs the elements of the array (map) as separate rows

Product_Colors
[‘blue, ‘green’]
[‘yellow’,’red’]

Colors
Blue
Green
Yellow
red

```
SELECT explode(product_colors) AS colors FROM test_array;
```

https://cwiki.apache.org/confluence/display/Hive/LanguageManual+UDF#LanguageManualUDF-explode

Collect_List/Collect_Set

- Prereq:

```
Create table test_array_temp as
SELECT explode(product_colors) AS colors FROM test_array;
```

- Converts a column into Array object
- Collect_List: Returns a list of objects with duplicates.

```
Select collect_list(colors) as color from test_array_temp;
```

- Collect_Set: Returns a set of objects with duplicate elements eliminated

```
Select collect_set(colors) as color from test_array_temp;
```

Explode + Lateral View

- Lateral view used in conjunction with explode generates zero or more output rows for each input row
- Lateral view + Explode

ID	Product_Colors
1	[‘blue, ‘green’]
2	[‘yellow’,’red’]

```
SELECT product, colors
FROM test_array LATERAL VIEW
explode(product_colors) test_array AS colors;
```

ID	Colors
1	Blue
1	Green
2	Yellow
2	red

https://cwiki.apache.org/confluence/display/Hive/LanguageManual+LateralView

Map

- It is an unordered collection of key-value pairs.
- Keys must be of primitive types.
- Values can be of any type. (e.g., values can be Array)
- Create table with column of type Map

```
create table test_map(product bigint , product_parts_color array<string,string>)
row format delimited
fields terminated by ','
 collection items terminated by '$‘,
map keys terminated by '#';
```

- Load into Table from File
  - Load data inpath '/user/root/map_test.txt' overwrite into table test_map;
- Query Map Column

```
Select product_parts_color from test_map;
Select product_parts_color[‘screen’], product_parts_color[‘keyboard’] from test_msp;
```

Map Functions

- size(Map<k,v>)

  Select size(product_parts_color) from test_map;

- map_keys(Map<K,V>)

  Select map_keys(product_parts_color) from test_map;

- map_values(Map<K,V>)

  Select map_values(product_parts_color) from test_map;

- Explode (work similar to the explode for Arrays)
  - Explodes Map into two Columns; one for key and other for the value
  - SELECT explode(product_parts_color) AS (product_name, color) FROM test_map;

Struct

- □ Object-like collections wherein each item is made up of multiple pieces of data each with its own data type
- □ It is a record type which encapsulates a set of named fields that can be any primitive data type
- □ Create table with column of type Struct

```
create table test_struct(product bigint , product_info struct<size:int,color:string>)
row format delimited
fields terminated by ','
 collection items terminated by '$‘;
```

- □ Load into Table from File

```
Load data  inpath '/user/root/struct_test.txt' overwrite into table
test_struct;
```

- □ Query Struct Column

```
Select product_parts_color from test_struct;

Select product_info,.size, product_info.color from test_struct;
```

Hive Documentation

- https://cwiki.apache.org/confluence/display/Hive/Home#Home-UserDocumentation
    - Hive Tutorial
    - Language Manual
    - Hive/NoSQL Integrations
    - Hive Installation/Configurations
    - Many other resources

Partitioning

- Dividing up the table based on values in certain columns (Example: Country, ZipCode)
- Hive will segregate input records into different directories based on chosen column
- The division can be based on one or more columns (For example: we can partition by country and then by state)
- Columns chosen for partition are categorical with some finite set of possible value

http://blog.cloudera.com/blog/2014/08/improving-query-performance-using-partitioning-in-apache-hive/

Partitioning

- Partition will improve query performance since only reading required subdirectories instead of scanning the entire table

  Example: if table is partitioned on countries and where clause says "WHERE=US" then only data pertaining to USA will be loaded into MapReduce job and processed

- However, **a query across all partitions** could trigger an enormous MapReduce job if the table data and number of partitions are large
  - A highly suggested safety measure is putting Hive into **"strict" mode,** which prohibits queries of partitioned tables without a WHERE clause that filters on partitions.

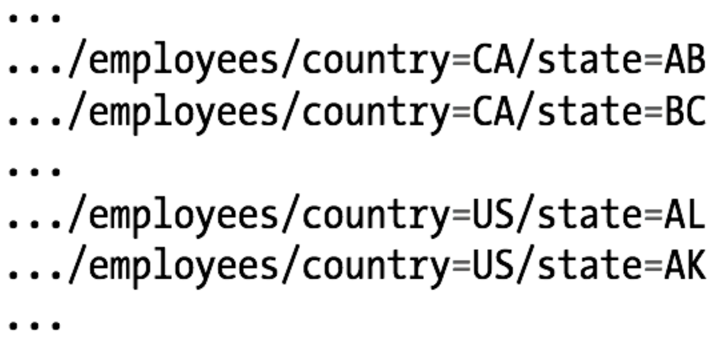
http://blog.cloudera.com/blog/2014/08/improving-query-performance-using-partitioning-in-apache-hive/

Partitioned Tables

- Create a hive table partitioned by two fields ***country*** and ***state***

```
CREATE TABLE employees(
            firstname VARCHAR(64),
            lastname  VARCHAR(64)
            )
            PARTITIONED BY (country VARCHAR(64), state VARCHAR(64));
```

- Sub-diretories on HDFS reflecting the partitioning structure



```
...
.../employees/country=CA/state=AB
.../employees/country=CA/state=BC
...
.../employees/country=US/state=AL
.../employees/country=US/state=AK
...
```

http://hadooptutorial.info/partitioning-in-hive/

Working with Partitions: Static

- Static Partitioning :
  - Data has to be loaded on a “per” partition bases.
  - Data has to be pre-processed into separate files based on state and country

```
LOAD DATA LOCAL INPATH ‘employee_CA.txt‘
 INTO TABLE employees
PARTITION (country = 'US', state = 'CA');
```

  - This statement will create the directory

“/user/hive/warehouse/employee/country=US/state=CA/” in HDFS

Working with Partitions: Dynamic

- □ Data does not have to be preprocessed
- □ Load the raw un-partitioned data into an temporary table (example: temp_employee)
- □ Load data from temporary table to final portioned table
- □ Advantage: We need to specify the exact country and state while loading
- □ Use File: employees_big.txt

```
INSERT INTO TABLE employees
        PARTITION (country, state)
        SELECT  firstname ,
                lastname   ,
                country    ,
                state
        FROM temp_employee;
```

- □ Requires setting of the following properties
  - set hive.exec.dynamic.partition=true;
  - set hive.exec.dynamic.partition.mode=nonstrict;

Working with Partitions

- The partition columns are typically used in “WHERE” clauses.

  SELECT firstname FROM employees

  WHERE country='US' AND state='CA'

  LIMIT 5;

  Think how much data needs to be processed by Map-Reduce jobs with and without partitions

  However, too many partitions will mean

  (1) Lots of metadata for the NameNode has to keep track

  (2) Too many map-reduce tasks when job uses data from multiple partitions (Example: if the table is partitioned on country, state, and zipcode and the most common analysis is run at Country or State level)

http://hadooptutorial.info/partitioning-in-hive/#Sample_Use_Case

Over Partitioning

- Over-partitioning can be detrimental to the performance for two reasons:
  - Millions of small files will overwhelm the NameNode
  - MapReduce processing converts a job into multiple tasks. In the default case, each task is a new JVM instance, requiring the overhead of start up and tear down. For small files, a separate task will be used for each file. The overhead of JVM start up and tear down can exceed the actual processing time!

Bucketing

- Partitions offer a convenient way to segregate data and to optimize queries. However, not all data sets lead to sensible partitioning, especially given the concerns raised earlier about appropriate sizing
- **Bucketing** is another technique for decomposing data sets into more manageable parts
- It is based on one or more columns
- Data is segregated by hashing the values of the columns into a fixed number of buckets
- Bucketing has several **advantages**.
  - The **number of buckets is fixed so it does not fluctuate with data.**
  - Buckets are **ideal for sampling.**
  - Bucketing also aids in doing **efficient mapside joins**

http://hadooptutorial.info/bucketing-in-hive/

Bucketing in Action

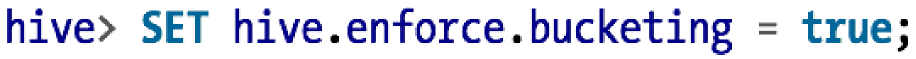
- □ Create a table partioned by date with N buckets by ***user_id***

```
CREATE TABLE employees_bucket(
               firstname VARCHAR(64),
         lastname  VARCHAR(64),
         country    VARCHAR(64),
      state      VARCHAR(64))
 CLUSTERED BY (state) SORTED BY (firstname) INTO 5 BUCKETS;
```

Employees with the same state will be stored in the same bucket!

Loading data into Bucketed Table

- □ Turn bucketing on in hive and insert data



```
hive> SET hive.enforce.bucketing = true;
```

- □ Load data into temp table
- □ Load data from temp table into bucketed table

```
INSERT INTO TABLE employees_bucket
        SELECT  firstname ,
                lastname  ,
                country   ,
                state
        FROM temp_employees;
```

Bucketing - Sampling

- It helps with sampling data based on values in bucketed columns
- Sample on specific bucket

  SELECT firstname, country, state FROM employees_bucket TABLESAMPLE(BUCKET 5 OUT OF 5 ON state);
- Sample of overall data

SELECT firstname, country, state, city FROM employees_bucket TABLESAMPLE(1 PERCENT);

Select * from employees_bucket TABLESAMPLE(2 ROWS);

http://myitlearnings.com/running-sampling-queries-in-hive/

http://thriveschool.blogspot.ca/2013/11/hive-bucketed-tables-and-sampling.html

http://hadooptutorial.info/bucketing-in-hive/#Table_Sampling_in_Hive

Join Algorithms

- Common Join
- Map Join
- Bucket Join
- Skew Join

- http://www.openkb.info/2014/11/understanding-hive-joins-in-explain.html

http://www.openkb.info/2014/11/understanding-hive-joins-in-explain.html

Reading Material

- □ http://grisha.org/blog/2013/04/19/mapjoin-a-simple-way-to-speed-up-your-hive-queries/
- □ https://cwiki.apache.org/confluence/download/attachments/27362054/Hive+Summit+2011-join.pdf
- □ https://netezzaadmin.wordpress.com/2013/09/25/hives-collection-data-types/
- □ https://joshuafennessy.com/2015/07/09/introduction-to-hive-complex-data-types-part-1-array/
- □ http://bigdatariding.blogspot.ca/2014/02/hive-complex-data-types-with-examples.html
- □ https://cwiki.apache.org/confluence/display/Hive/LanguageManual+LateralView
- □ https://dzone.com/articles/introduction-hives
- □ https://www.brentozar.com/archive/2013/03/introduction-to-hive-partitioning/
- □ http://www.joefkelley.com/736/

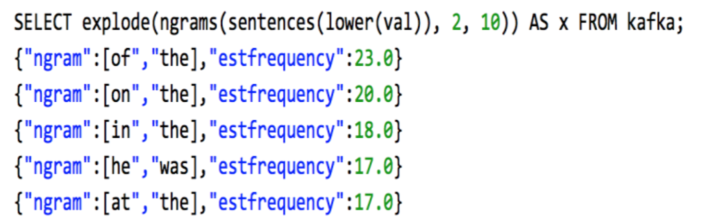
Hive Advanced Functions

Data – Geotagged Tweets

ID	DateTime	Latitude	Longitude	Tweet
USER_8d0e8566	2010-03-02T23:00:44	30.387524	-91.109663	Pre-workout prep has begun.
USER_8d0e8566	2010-03-02T23:04:20	30.387524	-91.109663	I really don't like that a college FB player's ON-FIELD production can be negated by a single workout. So proof's NOT in the pudding?
USER_87b48222	2010-03-02T23:23:29	37.530819	-77.475577	@USER_9bb099c2 15 pages??? fuck u mean!!?? damn.
USER_87b48222	2010-03-02T23:43:57	37.530819	-77.475577	@USER_e97d1292 lol do u know that song?
USER_01b8a291	2010-03-03T00:56:16	41.51179	-95.893286	HAHAHA OMG! I just found a baggie of weed that I hid from like four/five years ago!! Hahahaha!!!
USER_2e5f8774	2010-03-03T02:06:15	39.669307	-79.85002	@USER_2b2bd61b light skin free way and shit...lol Look like you sell bean pies
USER_942c68df	2010-03-03T02:21:36	41.220425	-85.861873	These judges are being hard this year.
USER_8d0e8566	2010-03-03T02:28:12	30.387524	-91.109663	@USER_b7cdabe3 People don't dance like that to get the burn anymore. Its frowned upon..LOL.
USER_8d0e8566	2010-03-03T02:29:39	30.399934	-91.121502	RT @USER_9c9e75e2: Officially getting rid of my iPhone with its dysfunctional button this weekend|Get a 9700 #BlackertheBerrytheSweetertheUse
USER_2e5f8774	2010-03-03T02:42:44	39.669307	-79.85002	@USER_7ac8dee6 Hey Cuz...Where u been at?
USER_8d0e8566	2010-03-03T02:43:01	30.393485	-91.110458	RT @USER_9c9e75e2: @USER_8d0e8566 I think that's the move!|Make it happen and we can play Word Mole against each other.
USER_8d0e8566	2010-03-03T02:53:19	30.393485	-91.110458	@USER_b7cdabe3 Oh, okay!
USER_942c68df	2010-03-03T02:55:36	41.234181	-85.812994	@USER_20c15b69 Me too.
USER_8d0e8566	2010-03-03T03:00:37	30.387524	-91.109663	The next 2hrs of tweets are @USER_fe579e73 for gibing me the idea with his #theory tweet
USER_942c68df	2010-03-03T03:14:53	41.234181	-85.812994	@USER_21fe08ea Aww that sucks. If ya dont mind me asking, whats ruining your relationship?
USER_8d0e8566	2010-03-03T03:26:46	30.387524	-91.109663	@USER_fe579e73 did u change ur settings to use twitlonger?
USER_8d0e8566	2010-03-03T03:29:41	30.387524	-91.109663	RT @USER_de057bc2: Twitter is jacked up tonight|Just on iPhones. #BlackertheBerrytheSweetertheUse
USER_8d0e8566	2010-03-03T03:33:47	30.387524	-91.109663	RT @USER_de057bc2: @USER_8d0e8566 EFF YO Blackberry|Sore Loser
USER_8d0e8566	2010-03-03T03:47:43	30.387524	-91.109663	@USER_45b5c066 @USER_2b5b12ff The body nice but that had to be a contest at a Bukket Nekked.
USER_8d0e8566	2010-03-03T03:57:23	30.387524	-91.109663	#PeterWisdom "If u wake up and ur gal or the gal ur in bed with is staring at u,take solace in knowing she'll be sleep when u escape." LOL
USER_87b48222	2010-03-03T03:59:01	37.530819	-77.475577	Where do you those rip away jeans?!! @USER_af454d84 and where can I get some?!
USER_8d0e8566	2010-03-03T04:17:29	30.387524	-91.109663	@USER_b7cdabe3 LOL
USER_8d0e8566	2010-03-03T04:37:07	30.387524	-91.109663	RT @USER_45b5c066: #FamilyGuy Meg and Brian make out. Meg stalks him like Misery &lt;&lt; did u just use a shag blog term?? #CLASSIC|Did I?

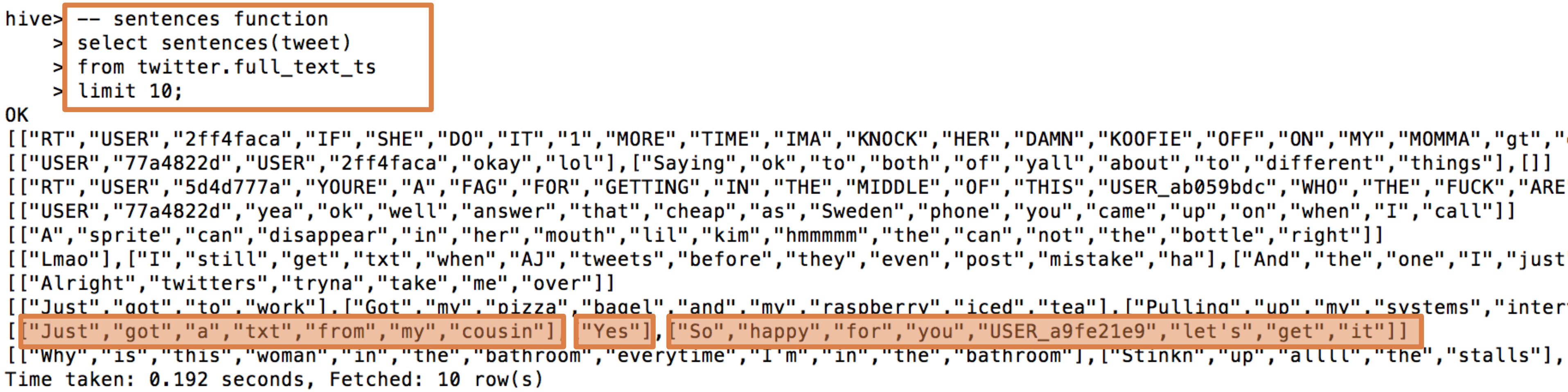
String Functions

String Func	Description	Syntax
split	Split strings around regex pattern	split(string str, string pat) split('big data tools ckme134', ' ')
sentences	Tokenizes a string of natural language text into words and sentences, where each sentence is broken at the appropriate sentence boundary and returned as an array of words	sentences('Hello there! How are you?') returns (("Hello", "there"), ("How", "are", "you"))

ngrams	Returns the top-k N-grams from a set of tokenized sentences, such as those returned by the sentences()	
		SELECT explode(ngrams(sentences(lower(val)), 2, 10)) AS x FROM kafka;
		{"ngram":[of","the],"estfrequency":23.0}
		{"ngram":[on","the],"estfrequency":20.0}
		{"ngram":[in","the],"estfrequency":18.0}
		{"ngram":[he","was],"estfrequency":17.0}
		{"ngram":[at","the],"estfrequency":17.0}

context_ngrams	Returns the top-k N-grams from a set of tokenized sentences, given a string of "context"	

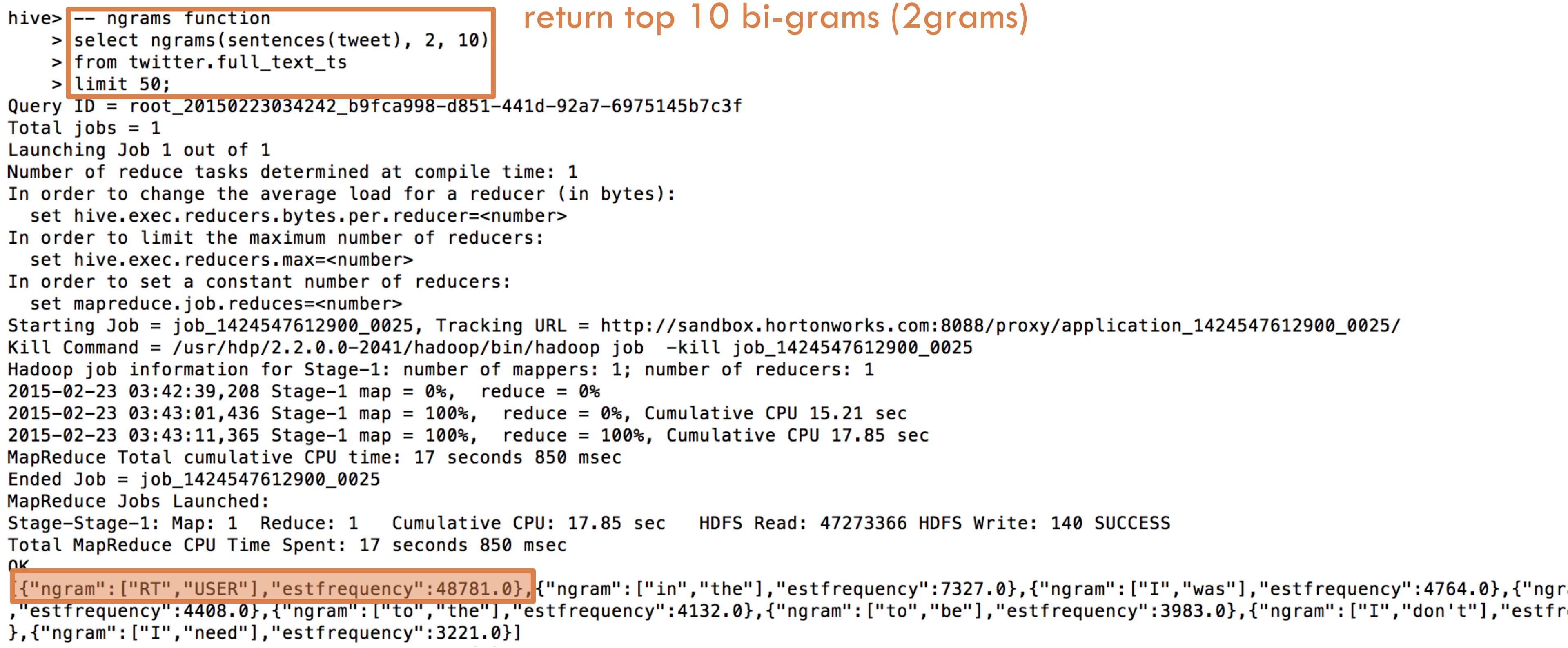
sentences() function



```
hive> -- sentences function
    > select sentences(tweet)
    > from twitter.full_text_ts
    > limit 10;
OK
[["RT","USER","2ff4faca","IF","SHE","DO","IT","1","MORE","TIME","IMA","KNOCK","HER","DAMN","KOOFIE","OFF","ON","MY","MOMMA","gt","
[["USER","77a4822d","USER","2ff4faca","okay","lol"],["Saying","ok","to","both","of","yall","about","to","different","things"],[]]
[["RT","USER","5d4d777a","YOURE","A","FAG","FOR","GETTING","IN","THE","MIDDLE","OF","THIS","USER_ab059bdc","WHO","THE","FUCK","ARE
[["USER","77a4822d","yea","ok","well","answer","that","cheap","as","Sweden","phone","you","came","up","on","when","I","call"]]
[["A","sprite","can","disappear","in","her","mouth","lil","kim","hmmmmm","the","can","not","the","bottle","right"]]
[["Lmao"],["I","still","get","txt","when","AJ","tweets","before","they","even","post","mistake","ha"],["And","the","one","I","just
[["Alright","twitters","tryna","take","me","over"]]
[["Just","got","to","work"],["Got","my","pizza","bagel","and","my","raspberry","iced","tea"],["Pulling","up","my","systems","inter
[["Just","got","a","txt","from","my","cousin"],["Yes"],["So","happy","for","you","USER_a9fe21e9","let's","get","it"]]
[["Why","is","this","woman","in","the","bathroom","everytime","I'm","in","the","bathroom"],["Stinkn","up","allll","the","stalls"],
Time taken: 0.192 seconds, Fetched: 10 row(s)
```

DEMO

ngrams() function

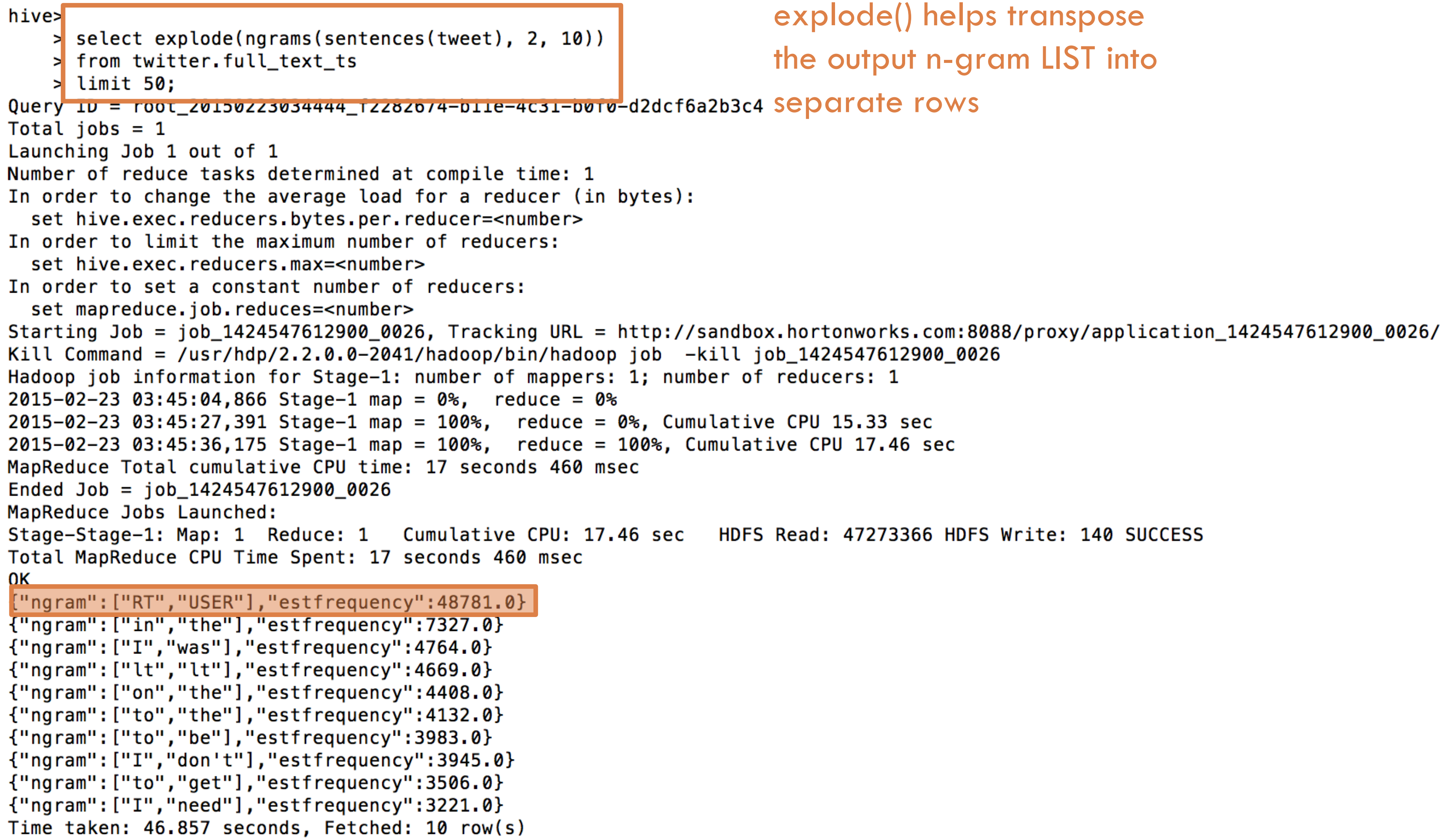


return top 10 bi-grams (2grams)

```
hive> -- ngrams function
    > select ngrams(sentences(tweet), 2, 10)
    > from twitter.full_text_ts
    > limit 50;
Query ID = root_20150223034242_b9fca998-d851-441d-92a7-6975145b7c3f
Total jobs = 1
Launching Job 1 out of 1
Number of reduce tasks determined at compile time: 1
In order to change the average load for a reducer (in bytes):
  set hive.exec.reducers.bytes.per.reducer=<number>
In order to limit the maximum number of reducers:
  set hive.exec.reducers.max=<number>
In order to set a constant number of reducers:
  set mapreduce.job.reduces=<number>
Starting Job = job_1424547612900_0025, Tracking URL = http://sandbox.hortonworks.com:8088/proxy/application_1424547612900_0025/
Kill Command = /usr/hdp/2.2.0.0-2041/hadoop/bin/hadoop job  -kill job_1424547612900_0025
Hadoop job information for Stage-1: number of mappers: 1; number of reducers: 1
2015-02-23 03:42:39,208 Stage-1 map = 0%,  reduce = 0%
2015-02-23 03:43:01,436 Stage-1 map = 100%,  reduce = 0%, Cumulative CPU 15.21 sec
2015-02-23 03:43:11,365 Stage-1 map = 100%,  reduce = 100%, Cumulative CPU 17.85 sec
MapReduce Total cumulative CPU time: 17 seconds 850 msec
Ended Job = job_1424547612900_0025
MapReduce Jobs Launched:
Stage-Stage-1: Map: 1  Reduce: 1   Cumulative CPU: 17.85 sec   HDFS Read: 47273366 HDFS Write: 140 SUCCESS
Total MapReduce CPU Time Spent: 17 seconds 850 msec
OK
[{"ngram":["RT","USER"],"estfrequency":48781.0},{"ngram":["in","the"],"estfrequency":7327.0},{"ngram":["I","was"],"estfrequency":4764.0},{"ngr
,"estfrequency":4408.0},{"ngram":["to","the"],"estfrequency":4132.0},{"ngram":["to","be"],"estfrequency":3983.0},{"ngram":["I","don't"],"estfr
},{"ngram":["I","need"],"estfrequency":3221.0}]
```

DEMO

ngrams() function



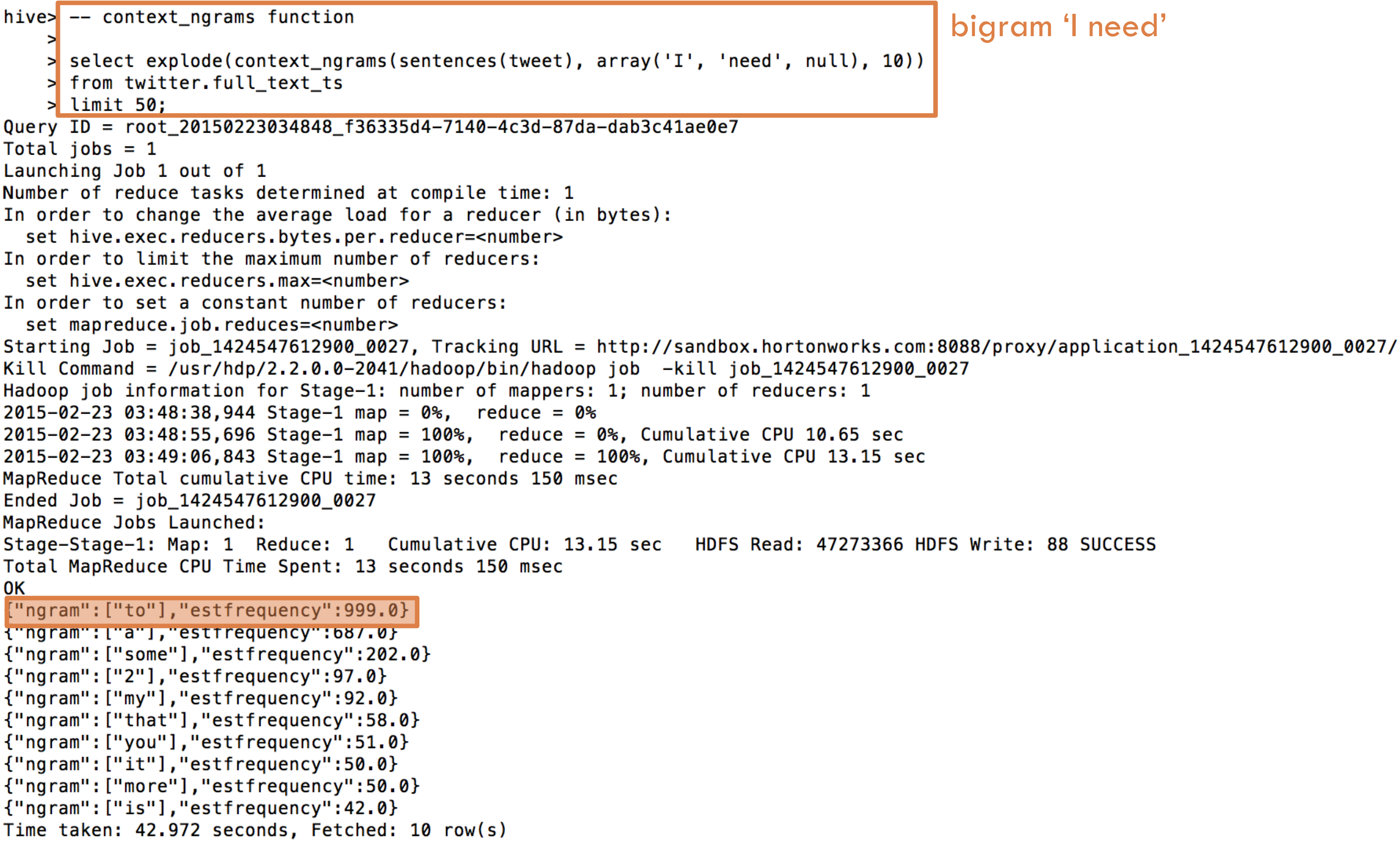
```
hive>
    > select explode(ngrams(sentences(tweet), 2, 10))
    > from twitter.full_text_ts
    > limit 50;
Query ID = root_20150223034444_f2282674-b11e-4c31-b0f0-d2dcf6a2b3c4
Total jobs = 1
Launching Job 1 out of 1
Number of reduce tasks determined at compile time: 1
In order to change the average load for a reducer (in bytes):
  set hive.exec.reducers.bytes.per.reducer=<number>
In order to limit the maximum number of reducers:
  set hive.exec.reducers.max=<number>
In order to set a constant number of reducers:
  set mapreduce.job.reduces=<number>
Starting Job = job_1424547612900_0026, Tracking URL = http://sandbox.hortonworks.com:8088/proxy/application_1424547612900_0026/
Kill Command = /usr/hdp/2.2.0.0-2041/hadoop/bin/hadoop job  -kill job_1424547612900_0026
Hadoop job information for Stage-1: number of mappers: 1; number of reducers: 1
2015-02-23 03:45:04,866 Stage-1 map = 0%,  reduce = 0%
2015-02-23 03:45:27,391 Stage-1 map = 100%,  reduce = 0%, Cumulative CPU 15.33 sec
2015-02-23 03:45:36,175 Stage-1 map = 100%,  reduce = 100%, Cumulative CPU 17.46 sec
MapReduce Total cumulative CPU time: 17 seconds 460 msec
Ended Job = job_1424547612900_0026
MapReduce Jobs Launched:
Stage-Stage-1: Map: 1  Reduce: 1   Cumulative CPU: 17.46 sec   HDFS Read: 47273366 HDFS Write: 140 SUCCESS
Total MapReduce CPU Time Spent: 17 seconds 460 msec
OK
{"ngram":["RT","USER"],"estfrequency":48781.0}
{"ngram":["in","the"],"estfrequency":7327.0}
{"ngram":["I","was"],"estfrequency":4764.0}
{"ngram":["lt","lt"],"estfrequency":4669.0}
{"ngram":["on","the"],"estfrequency":4408.0}
{"ngram":["to","the"],"estfrequency":4132.0}
{"ngram":["to","be"],"estfrequency":3983.0}
{"ngram":["I","don't"],"estfrequency":3945.0}
{"ngram":["to","get"],"estfrequency":3506.0}
{"ngram":["I","need"],"estfrequency":3221.0}
Time taken: 46.857 seconds, Fetched: 10 row(s)
```

explode() helps transpose the output n-gram LIST into separate rows

DEMO

context_ngrams() function

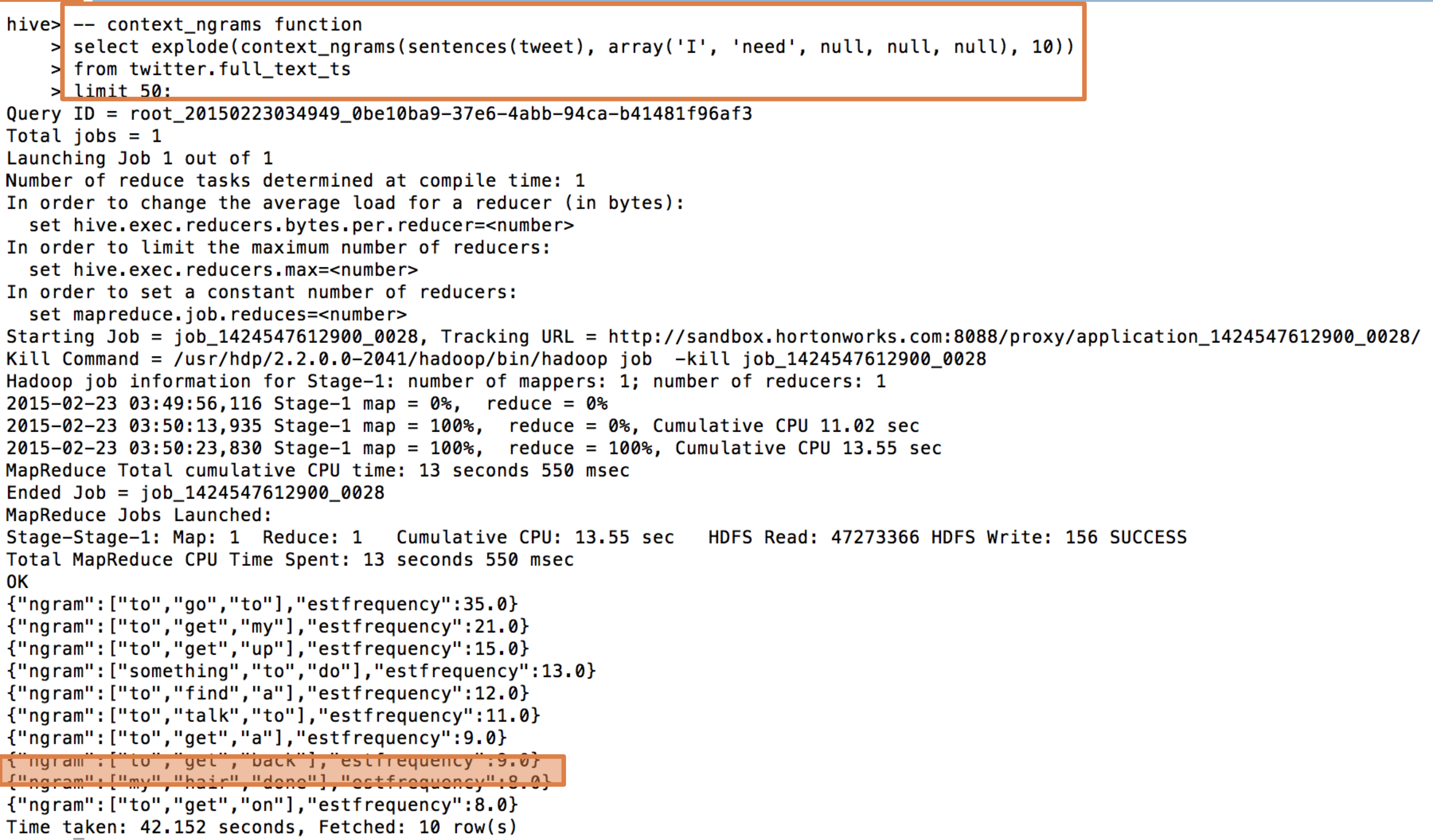
most popular word after bigram ‘I need’



```
hive> -- context_ngrams function
    >
    > select explode(context_ngrams(sentences(tweet), array('I', 'need', null), 10))
    > from twitter.full_text_ts
    > limit 50;
Query ID = root_20150223034848_f36335d4-7140-4c3d-87da-dab3c41ae0e7
Total jobs = 1
Launching Job 1 out of 1
Number of reduce tasks determined at compile time: 1
In order to change the average load for a reducer (in bytes):
  set hive.exec.reducers.bytes.per.reducer=<number>
In order to limit the maximum number of reducers:
  set hive.exec.reducers.max=<number>
In order to set a constant number of reducers:
  set mapreduce.job.reduces=<number>
Starting Job = job_1424547612900_0027, Tracking URL = http://sandbox.hortonworks.com:8088/proxy/application_1424547612900_0027/
Kill Command = /usr/hdp/2.2.0.0-2041/hadoop/bin/hadoop job  -kill job_1424547612900_0027
Hadoop job information for Stage-1: number of mappers: 1; number of reducers: 1
2015-02-23 03:48:38,944 Stage-1 map = 0%,  reduce = 0%
2015-02-23 03:48:55,696 Stage-1 map = 100%,  reduce = 0%, Cumulative CPU 10.65 sec
2015-02-23 03:49:06,843 Stage-1 map = 100%,  reduce = 100%, Cumulative CPU 13.15 sec
MapReduce Total cumulative CPU time: 13 seconds 150 msec
Ended Job = job_1424547612900_0027
MapReduce Jobs Launched:
Stage-Stage-1: Map: 1  Reduce: 1   Cumulative CPU: 13.15 sec   HDFS Read: 47273366 HDFS Write: 88 SUCCESS
Total MapReduce CPU Time Spent: 13 seconds 150 msec
OK
{"ngram":["to"],"estfrequency":999.0}
{"ngram":["a"],"estfrequency":687.0}
{"ngram":["some"],"estfrequency":202.0}
{"ngram":["2"],"estfrequency":97.0}
{"ngram":["my"],"estfrequency":92.0}
{"ngram":["that"],"estfrequency":58.0}
{"ngram":["you"],"estfrequency":51.0}
{"ngram":["it"],"estfrequency":50.0}
{"ngram":["more"],"estfrequency":50.0}
{"ngram":["is"],"estfrequency":42.0}
Time taken: 42.972 seconds, Fetched: 10 row(s)
```

DEMO

ngrams() function



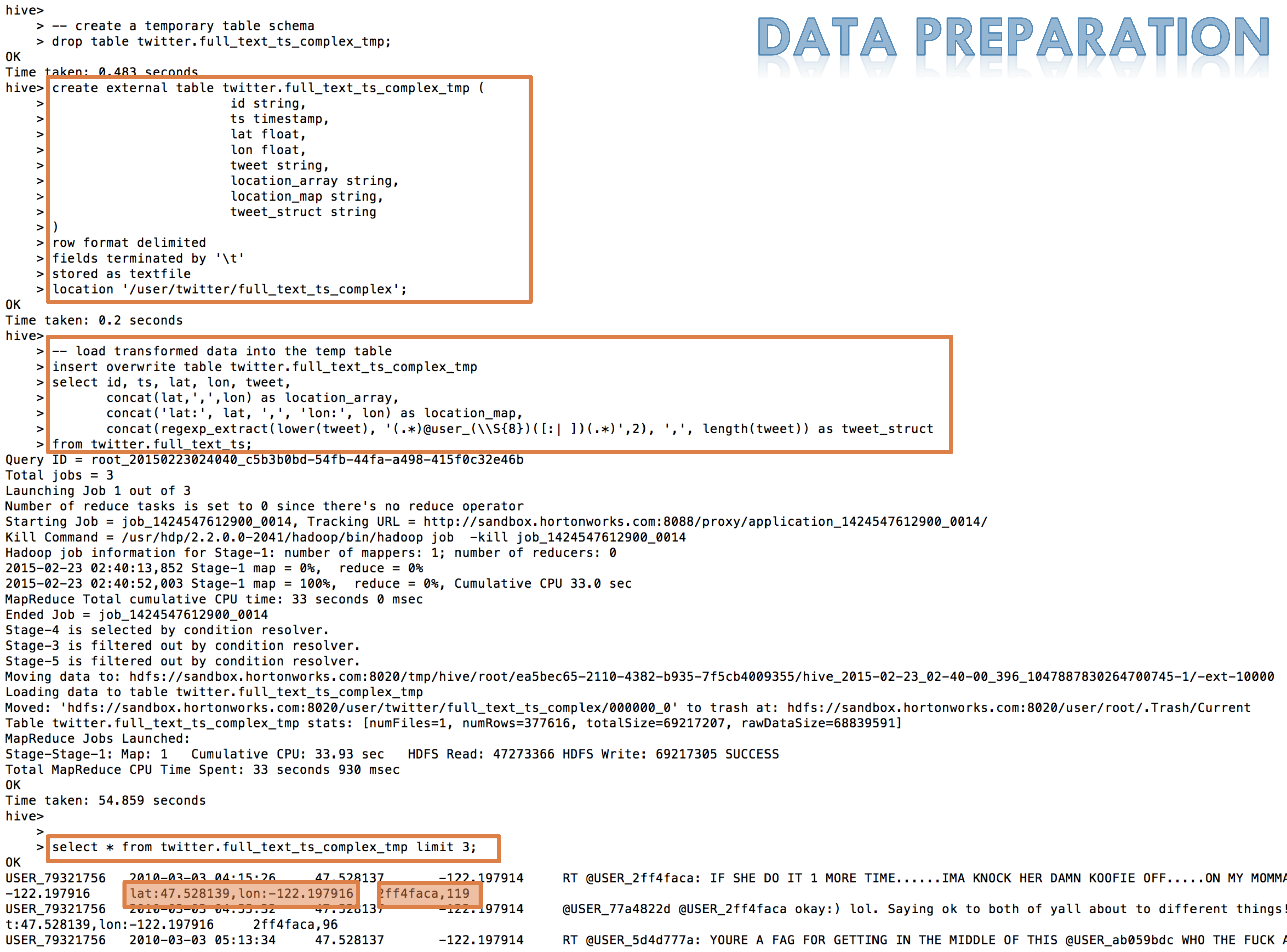
```
hive> -- context_ngrams function
    > select explode(context_ngrams(sentences(tweet), array('I', 'need', null, null, null), 10))
    > from twitter.full_text_ts
    > limit 50;
Query ID = root_20150223034949_0be10ba9-37e6-4abb-94ca-b41481f96af3
Total jobs = 1
Launching Job 1 out of 1
Number of reduce tasks determined at compile time: 1
In order to change the average load for a reducer (in bytes):
  set hive.exec.reducers.bytes.per.reducer=<number>
In order to limit the maximum number of reducers:
  set hive.exec.reducers.max=<number>
In order to set a constant number of reducers:
  set mapreduce.job.reduces=<number>
Starting Job = job_1424547612900_0028, Tracking URL = http://sandbox.hortonworks.com:8088/proxy/application_1424547612900_0028/
Kill Command = /usr/hdp/2.2.0.0-2041/hadoop/bin/hadoop job  -kill job_1424547612900_0028
Hadoop job information for Stage-1: number of mappers: 1; number of reducers: 1
2015-02-23 03:49:56,116 Stage-1 map = 0%,  reduce = 0%
2015-02-23 03:50:13,935 Stage-1 map = 100%,  reduce = 0%, Cumulative CPU 11.02 sec
2015-02-23 03:50:23,830 Stage-1 map = 100%,  reduce = 100%, Cumulative CPU 13.55 sec
MapReduce Total cumulative CPU time: 13 seconds 550 msec
Ended Job = job_1424547612900_0028
MapReduce Jobs Launched:
Stage-Stage-1: Map: 1  Reduce: 1   Cumulative CPU: 13.55 sec   HDFS Read: 47273366 HDFS Write: 156 SUCCESS
Total MapReduce CPU Time Spent: 13 seconds 550 msec
OK
{"ngram":["to","go","to"],"estfrequency":35.0}
{"ngram":["to","get","my"],"estfrequency":21.0}
{"ngram":["to","get","up"],"estfrequency":15.0}
{"ngram":["something","to","do"],"estfrequency":13.0}
{"ngram":["to","find","a"],"estfrequency":12.0}
{"ngram":["to","talk","to"],"estfrequency":11.0}
{"ngram":["to","get","a"],"estfrequency":9.0}
{"ngram":["to","get","back"],"estfrequency":9.0}
{"ngram":["my","hair","done"],"estfrequency":8.0}
{"ngram":["to","get","on"],"estfrequency":8.0}
Time taken: 42.152 seconds, Fetched: 10 row(s)
```

Built-in Aggregate Functions (UDAF)

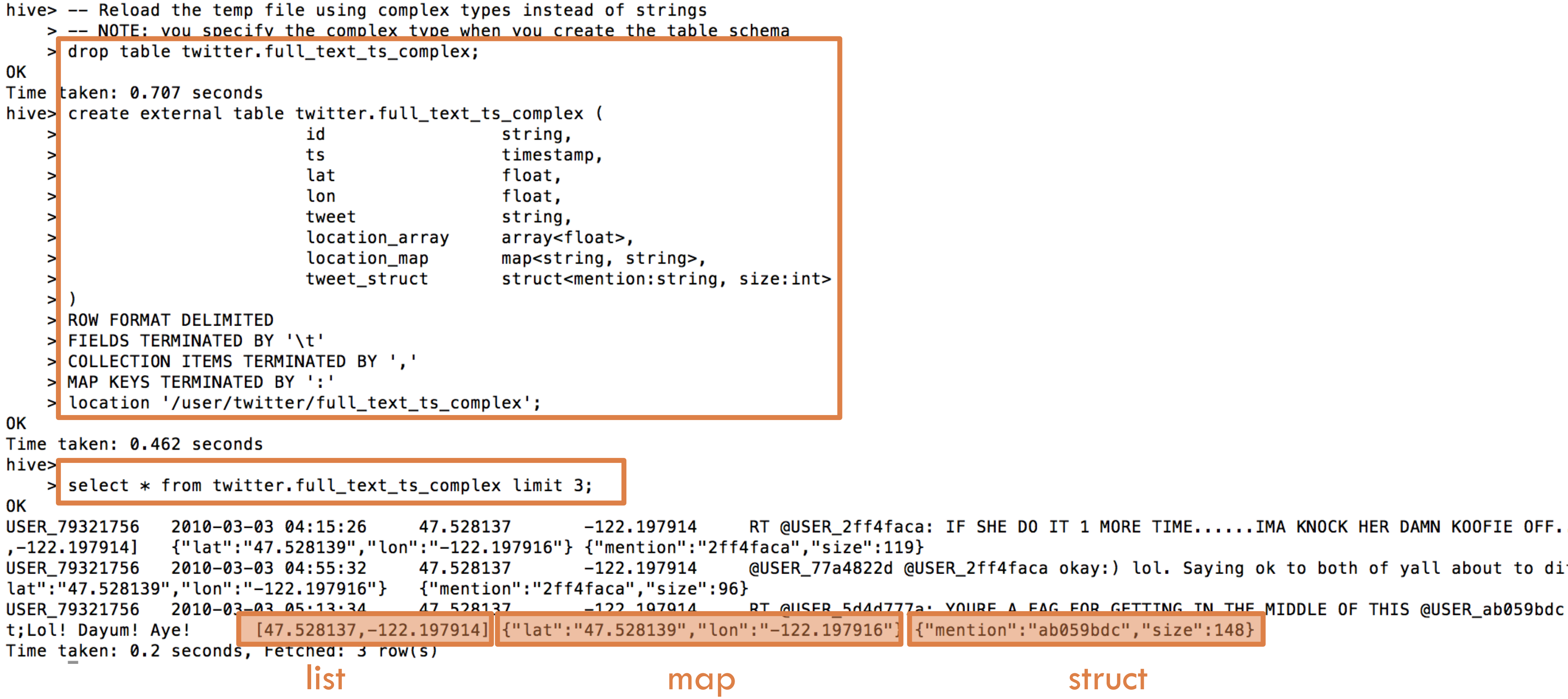
- count(*), count(distinct)
- sum, avg
- min, max
- percentile
- histogram_numeric
- collect_set
- collect_list

https://docs.treasuredata.com/articles/hive-aggregate-functions

DATA PREPARATION



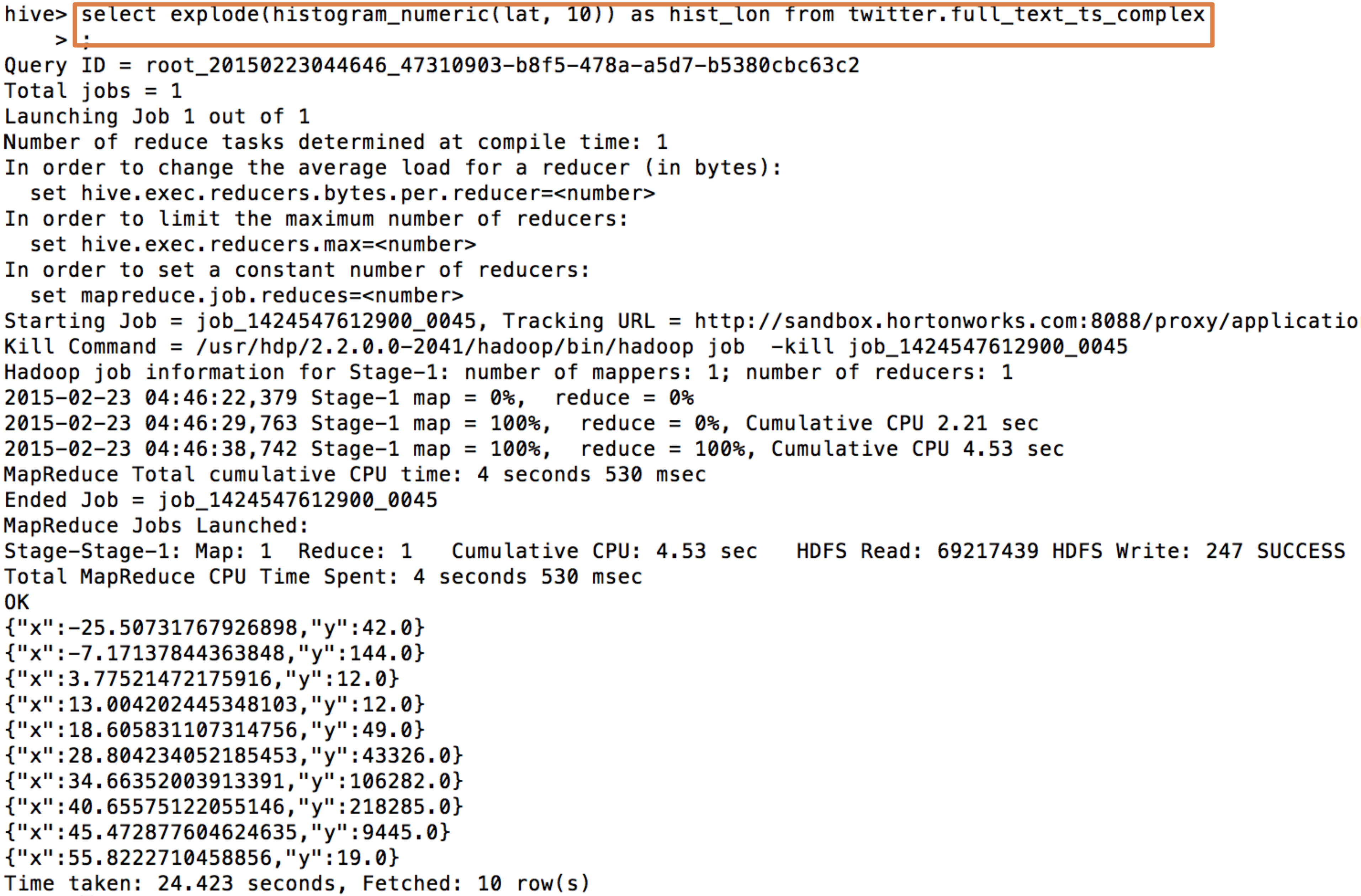
```
hive>
    > -- create a temporary table schema
    > drop table twitter.full_text_ts_complex_tmp;
OK
Time taken: 0.483 seconds
hive> create external table twitter.full_text_ts_complex_tmp (
    >                        id string,
    >                        ts timestamp,
    >                        lat float,
    >                        lon float,
    >                        tweet string,
    >                        location_array string,
    >                        location_map string,
    >                        tweet_struct string
    > )
    > row format delimited
    > fields terminated by '\t'
    > stored as textfile
    > location '/user/twitter/full_text_ts_complex';
OK
Time taken: 0.2 seconds
hive>
    > -- load transformed data into the temp table
    > insert overwrite table twitter.full_text_ts_complex_tmp
    > select id, ts, lat, lon, tweet,
    >        concat(lat,',',lon) as location_array,
    >        concat('lat:', lat, ',', 'lon:', lon) as location_map,
    >        concat(regexp_extract(lower(tweet), '(.*)@user_(\\S{8})([:| ])(.*)',2), ',', length(tweet)) as tweet_struct
    > from twitter.full_text_ts;
Query ID = root_20150223024040_c5b3b0bd-54fb-44fa-a498-415f0c32e46b
Total jobs = 3
Launching Job 1 out of 3
Number of reduce tasks is set to 0 since there's no reduce operator
Starting Job = job_1424547612900_0014, Tracking URL = http://sandbox.hortonworks.com:8088/proxy/application_1424547612900_0014/
Kill Command = /usr/hdp/2.2.0.0-2041/hadoop/bin/hadoop job  -kill job_1424547612900_0014
Hadoop job information for Stage-1: number of mappers: 1; number of reducers: 0
2015-02-23 02:40:13,852 Stage-1 map = 0%,  reduce = 0%
2015-02-23 02:40:52,003 Stage-1 map = 100%,  reduce = 0%, Cumulative CPU 33.0 sec
MapReduce Total cumulative CPU time: 33 seconds 0 msec
Ended Job = job_1424547612900_0014
Stage-4 is selected by condition resolver.
Stage-3 is filtered out by condition resolver.
Stage-5 is filtered out by condition resolver.
Moving data to: hdfs://sandbox.hortonworks.com:8020/tmp/hive/root/ea5bec65-2110-4382-b935-7f5cb4009355/hive_2015-02-23_02-40-00_396_1047887830264700745-1/-ext-10000
Loading data to table twitter.full_text_ts_complex_tmp
Moved: 'hdfs://sandbox.hortonworks.com:8020/user/twitter/full_text_ts_complex/000000_0' to trash at: hdfs://sandbox.hortonworks.com:8020/user/root/.Trash/Current
Table twitter.full_text_ts_complex_tmp stats: [numFiles=1, numRows=377616, totalSize=69217207, rawDataSize=68839591]
MapReduce Jobs Launched:
Stage-Stage-1: Map: 1   Cumulative CPU: 33.93 sec   HDFS Read: 47273366 HDFS Write: 69217305 SUCCESS
Total MapReduce CPU Time Spent: 33 seconds 930 msec
OK
Time taken: 54.859 seconds
hive>
    >
    > select * from twitter.full_text_ts_complex_tmp limit 3;
OK
USER_79321756   2010-03-03 04:15:26     47.528137       -122.197914     RT @USER_2ff4faca: IF SHE DO IT 1 MORE TIME......IMA KNOCK HER DAMN KOOFIE OFF.....ON MY MOMMA
-122.197916     lat:47.528139,lon:-122.197916   2ff4faca,119
USER_79321756   2010-03-03 04:55:32     47.528137       -122.197914     @USER_77a4822d @USER_2ff4faca okay:) lol. Saying ok to both of yall about to different things!
t:47.528139,lon:-122.197916     2ff4faca,96
USER_79321756   2010-03-03 05:13:34     47.528137       -122.197914     RT @USER_5d4d777a: YOURE A FAG FOR GETTING IN THE MIDDLE OF THIS @USER_ab059bdc WHO THE FUCK A
```



```
hive> -- Reload the temp file using complex types instead of strings
    > -- NOTE: you specify the complex type when you create the table schema
    > drop table twitter.full_text_ts_complex;
OK
Time taken: 0.707 seconds
hive> create external table twitter.full_text_ts_complex (
    >                       id                  string,
    >                       ts                  timestamp,
    >                       lat                 float,
    >                       lon                 float,
    >                       tweet               string,
    >                       location_array      array<float>,
    >                       location_map        map<string, string>,
    >                       tweet_struct        struct<mention:string, size:int>
    > )
    > ROW FORMAT DELIMITED
    > FIELDS TERMINATED BY '\t'
    > COLLECTION ITEMS TERMINATED BY ','
    > MAP KEYS TERMINATED BY ':'
    > location '/user/twitter/full_text_ts_complex';
OK
Time taken: 0.462 seconds
hive>
    > select * from twitter.full_text_ts_complex limit 3;
OK
USER_79321756   2010-03-03 04:15:26     47.528137       -122.197914     RT @USER_2ff4faca: IF SHE DO IT 1 MORE TIME......IMA KNOCK HER DAMN KOOFIE OFF.
,-122.197914]   {"lat":"47.528139","lon":"-122.197916"} {"mention":"2ff4faca","size":119}
USER_79321756   2010-03-03 04:55:32     47.528137       -122.197914     @USER_77a4822d @USER_2ff4faca okay:) lol. Saying ok to both of yall about to di
lat":"47.528139","lon":"-122.197916"}   {"mention":"2ff4faca","size":96}
USER_79321756   2010-03-03 05:13:34     47.528137       -122.197914     RT @USER_5d4d777a: YOURE A FAG FOR GETTING IN THE MIDDLE OF THIS @USER_ab059bdc
t;Lol! Dayum! Aye!   [47.528137,-122.197914] {"lat":"47.528139","lon":"-122.197916"} {"mention":"ab059bdc","size":148}
Time taken: 0.2 seconds, Fetched: 3 row(s)
```

histogram_numeric() function

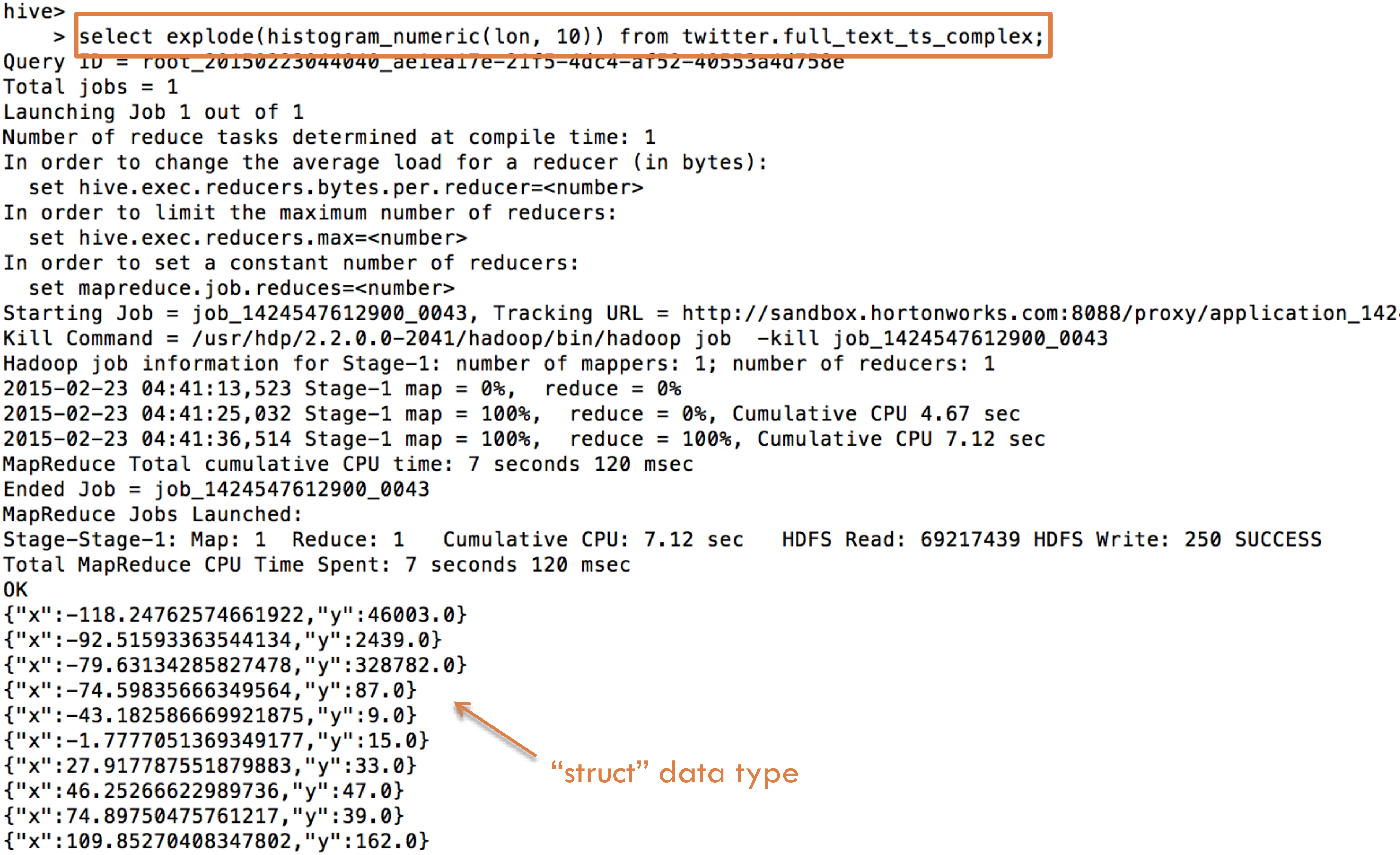
DEMO



```
hive> select explode(histogram_numeric(lat, 10)) as hist_lon from twitter.full_text_ts_complex
    > ;
Query ID = root_20150223044646_47310903-b8f5-478a-a5d7-b5380cbc63c2
Total jobs = 1
Launching Job 1 out of 1
Number of reduce tasks determined at compile time: 1
In order to change the average load for a reducer (in bytes):
  set hive.exec.reducers.bytes.per.reducer=<number>
In order to limit the maximum number of reducers:
  set hive.exec.reducers.max=<number>
In order to set a constant number of reducers:
  set mapreduce.job.reduces=<number>
Starting Job = job_1424547612900_0045, Tracking URL = http://sandbox.hortonworks.com:8088/proxy/applicatio
Kill Command = /usr/hdp/2.2.0.0-2041/hadoop/bin/hadoop job  -kill job_1424547612900_0045
Hadoop job information for Stage-1: number of mappers: 1; number of reducers: 1
2015-02-23 04:46:22,379 Stage-1 map = 0%,  reduce = 0%
2015-02-23 04:46:29,763 Stage-1 map = 100%,  reduce = 0%, Cumulative CPU 2.21 sec
2015-02-23 04:46:38,742 Stage-1 map = 100%,  reduce = 100%, Cumulative CPU 4.53 sec
MapReduce Total cumulative CPU time: 4 seconds 530 msec
Ended Job = job_1424547612900_0045
MapReduce Jobs Launched:
Stage-Stage-1: Map: 1  Reduce: 1   Cumulative CPU: 4.53 sec   HDFS Read: 69217439 HDFS Write: 247 SUCCESS
Total MapReduce CPU Time Spent: 4 seconds 530 msec
OK
{"x":-25.50731767926898,"y":42.0}
{"x":-7.17137844363848,"y":144.0}
{"x":3.77521472175916,"y":12.0}
{"x":13.004202445348103,"y":12.0}
{"x":18.605831107314756,"y":49.0}
{"x":28.804234052185453,"y":43326.0}
{"x":34.66352003913391,"y":106282.0}
{"x":40.65575122055146,"y":218285.0}
{"x":45.472877604624635,"y":9445.0}
{"x":55.8222710458856,"y":19.0}
Time taken: 24.423 seconds, Fetched: 10 row(s)
```

DEMO

histogram_numeric() function



```
hive>
    > select explode(histogram_numeric(lon, 10)) from twitter.full_text_ts_complex;
Query ID = root_20150223044040_ae1ea17e-21f5-4dc4-af52-40553a4d758e
Total jobs = 1
Launching Job 1 out of 1
Number of reduce tasks determined at compile time: 1
In order to change the average load for a reducer (in bytes):
  set hive.exec.reducers.bytes.per.reducer=<number>
In order to limit the maximum number of reducers:
  set hive.exec.reducers.max=<number>
In order to set a constant number of reducers:
  set mapreduce.job.reduces=<number>
Starting Job = job_1424547612900_0043, Tracking URL = http://sandbox.hortonworks.com:8088/proxy/application_142
Kill Command = /usr/hdp/2.2.0.0-2041/hadoop/bin/hadoop job  -kill job_1424547612900_0043
Hadoop job information for Stage-1: number of mappers: 1; number of reducers: 1
2015-02-23 04:41:13,523 Stage-1 map = 0%,  reduce = 0%
2015-02-23 04:41:25,032 Stage-1 map = 100%,  reduce = 0%, Cumulative CPU 4.67 sec
2015-02-23 04:41:36,514 Stage-1 map = 100%,  reduce = 100%, Cumulative CPU 7.12 sec
MapReduce Total cumulative CPU time: 7 seconds 120 msec
Ended Job = job_1424547612900_0043
MapReduce Jobs Launched:
Stage-Stage-1: Map: 1  Reduce: 1   Cumulative CPU: 7.12 sec   HDFS Read: 69217439 HDFS Write: 250 SUCCESS
Total MapReduce CPU Time Spent: 7 seconds 120 msec
OK
{"x":-118.24762574661922,"y":46003.0}
{"x":-92.51593363544134,"y":2439.0}
{"x":-79.63134285827478,"y":328782.0}
{"x":-74.59835666349564,"y":87.0}
{"x":-43.182586669921875,"y":9.0}
{"x":-1.7777051369349177,"y":15.0}
{"x":27.917787551879883,"y":33.0}
{"x":46.2526622989736,"y":47.0}
{"x":74.89750475761217,"y":39.0}
{"x":109.85270408347802,"y":162.0}
```

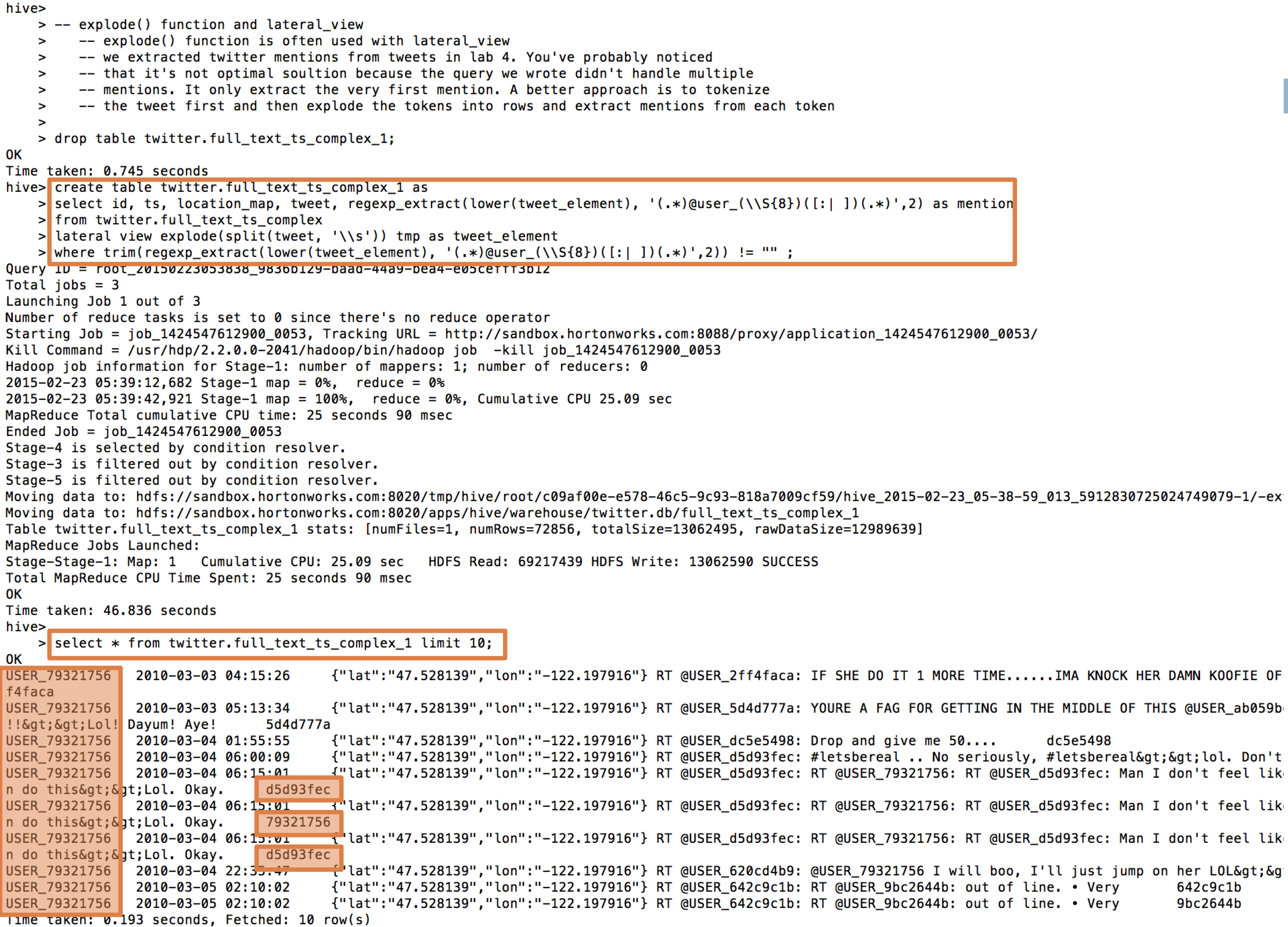
“struct” data type

Built-in Table-Generating Functions (UDTF)

- explode()
  - transposes list/map elements into multiple rows
  - usually used with lateral_view
- collect_set
  - transposes multiple rows associated with same key to a list/map
  - usually used with group by

DEMO

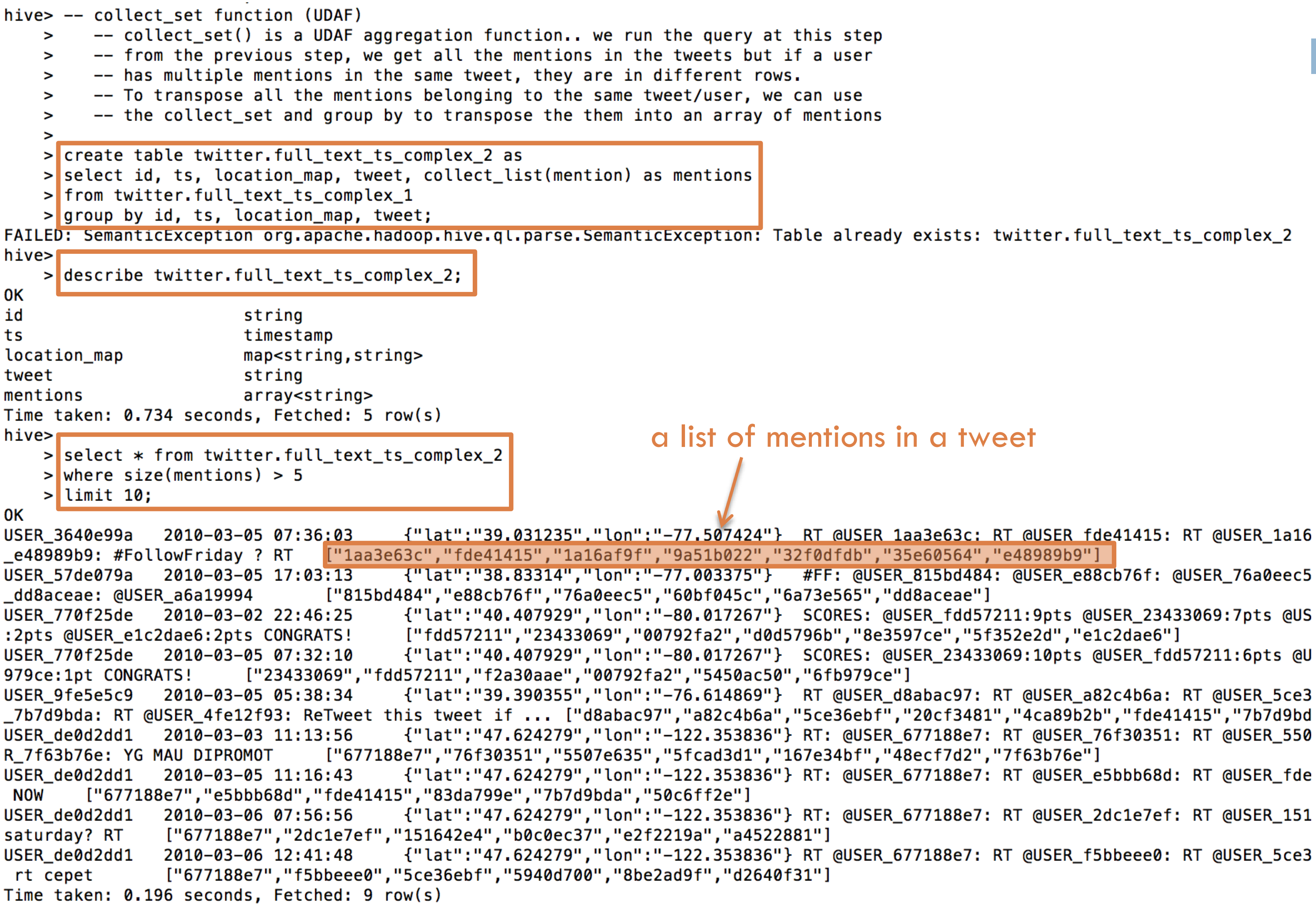
explode() function



```
hive>
    > -- explode() function and lateral_view
    >    -- explode() function is often used with lateral_view
    >    -- we extracted twitter mentions from tweets in lab 4. You've probably noticed
    >    -- that it's not optimal soultion because the query we wrote didn't handle multiple
    >    -- mentions. It only extract the very first mention. A better approach is to tokenize
    >    -- the tweet first and then explode the tokens into rows and extract mentions from each token
    >
    > drop table twitter.full_text_ts_complex_1;
OK
Time taken: 0.745 seconds
hive> create table twitter.full_text_ts_complex_1 as
    > select id, ts, location_map, tweet, regexp_extract(lower(tweet_element), '(.*)@user_(\\S{8})([:| ])(.*)',2) as mention
    > from twitter.full_text_ts_complex
    > lateral view explode(split(tweet, '\\s')) tmp as tweet_element
    > where trim(regexp_extract(lower(tweet_element), '(.*)@user_(\\S{8})([:| ])(.*)',2)) != "" ;
Query ID = root_20150223053838_9836b129-baad-44a9-bea4-e05cefff3b12
Total jobs = 3
Launching Job 1 out of 3
Number of reduce tasks is set to 0 since there's no reduce operator
Starting Job = job_1424547612900_0053, Tracking URL = http://sandbox.hortonworks.com:8088/proxy/application_1424547612900_0053/
Kill Command = /usr/hdp/2.2.0.0-2041/hadoop/bin/hadoop job  -kill job_1424547612900_0053
Hadoop job information for Stage-1: number of mappers: 1; number of reducers: 0
2015-02-23 05:39:12,682 Stage-1 map = 0%,  reduce = 0%
2015-02-23 05:39:42,921 Stage-1 map = 100%,  reduce = 0%, Cumulative CPU 25.09 sec
MapReduce Total cumulative CPU time: 25 seconds 90 msec
Ended Job = job_1424547612900_0053
Stage-4 is selected by condition resolver.
Stage-3 is filtered out by condition resolver.
Stage-5 is filtered out by condition resolver.
Moving data to: hdfs://sandbox.hortonworks.com:8020/tmp/hive/root/c09af00e-e578-46c5-9c93-818a7009cf59/hive_2015-02-23_05-38-59_013_5912830725024749079-1/-ex
Moving data to: hdfs://sandbox.hortonworks.com:8020/apps/hive/warehouse/twitter.db/full_text_ts_complex_1
Table twitter.full_text_ts_complex_1 stats: [numFiles=1, numRows=72856, totalSize=13062495, rawDataSize=12989639]
MapReduce Jobs Launched:
Stage-Stage-1: Map: 1   Cumulative CPU: 25.09 sec   HDFS Read: 69217439 HDFS Write: 13062590 SUCCESS
Total MapReduce CPU Time Spent: 25 seconds 90 msec
OK
Time taken: 46.836 seconds
hive>
    > select * from twitter.full_text_ts_complex_1 limit 10;
OK
USER_79321756   2010-03-03 04:15:26     {"lat":"47.528139","lon":"-122.197916"} RT @USER_2ff4faca: IF SHE DO IT 1 MORE TIME......IMA KNOCK HER DAMN KOOFIE OF
f4faca
USER_79321756   2010-03-03 05:13:34     {"lat":"47.528139","lon":"-122.197916"} RT @USER_5d4d777a: YOURE A FAG FOR GETTING IN THE MIDDLE OF THIS @USER_ab059b
!!&gt;&gt;Lol! Dayum! Aye!       5d4d777a
USER_79321756   2010-03-04 01:55:55     {"lat":"47.528139","lon":"-122.197916"} RT @USER_dc5e5498: Drop and give me 50....       dc5e5498
USER_79321756   2010-03-04 06:00:09     {"lat":"47.528139","lon":"-122.197916"} RT @USER_d5d93fec: #letsbereal .. No seriously, #letsbereal&gt;&gt;lol. Don't
USER_79321756   2010-03-04 06:15:01     {"lat":"47.528139","lon":"-122.197916"} RT @USER_d5d93fec: RT @USER_79321756: RT @USER_d5d93fec: Man I don't feel lik
n do this&gt;&gt;Lol. Okay.       d5d93fec
USER_79321756   2010-03-04 06:15:01     {"lat":"47.528139","lon":"-122.197916"} RT @USER_d5d93fec: RT @USER_79321756: RT @USER_d5d93fec: Man I don't feel lik
n do this&gt;&gt;Lol. Okay.       79321756
USER_79321756   2010-03-04 06:15:01     {"lat":"47.528139","lon":"-122.197916"} RT @USER_d5d93fec: RT @USER_79321756: RT @USER_d5d93fec: Man I don't feel lik
n do this&gt;&gt;Lol. Okay.       d5d93fec
USER_79321756   2010-03-04 22:35:47     {"lat":"47.528139","lon":"-122.197916"} RT @USER_620cd4b9: @USER_79321756 I will boo, I'll just jump on her LOL&gt;&g
USER_79321756   2010-03-05 02:10:02     {"lat":"47.528139","lon":"-122.197916"} RT @USER_642c9c1b: RT @USER_9bc2644b: out of line. • Very        642c9c1b
USER_79321756   2010-03-05 02:10:02     {"lat":"47.528139","lon":"-122.197916"} RT @USER_642c9c1b: RT @USER_9bc2644b: out of line. • Very        9bc2644b
Time taken: 0.193 seconds, Fetched: 10 row(s)
```

DEMO

collect_set() function



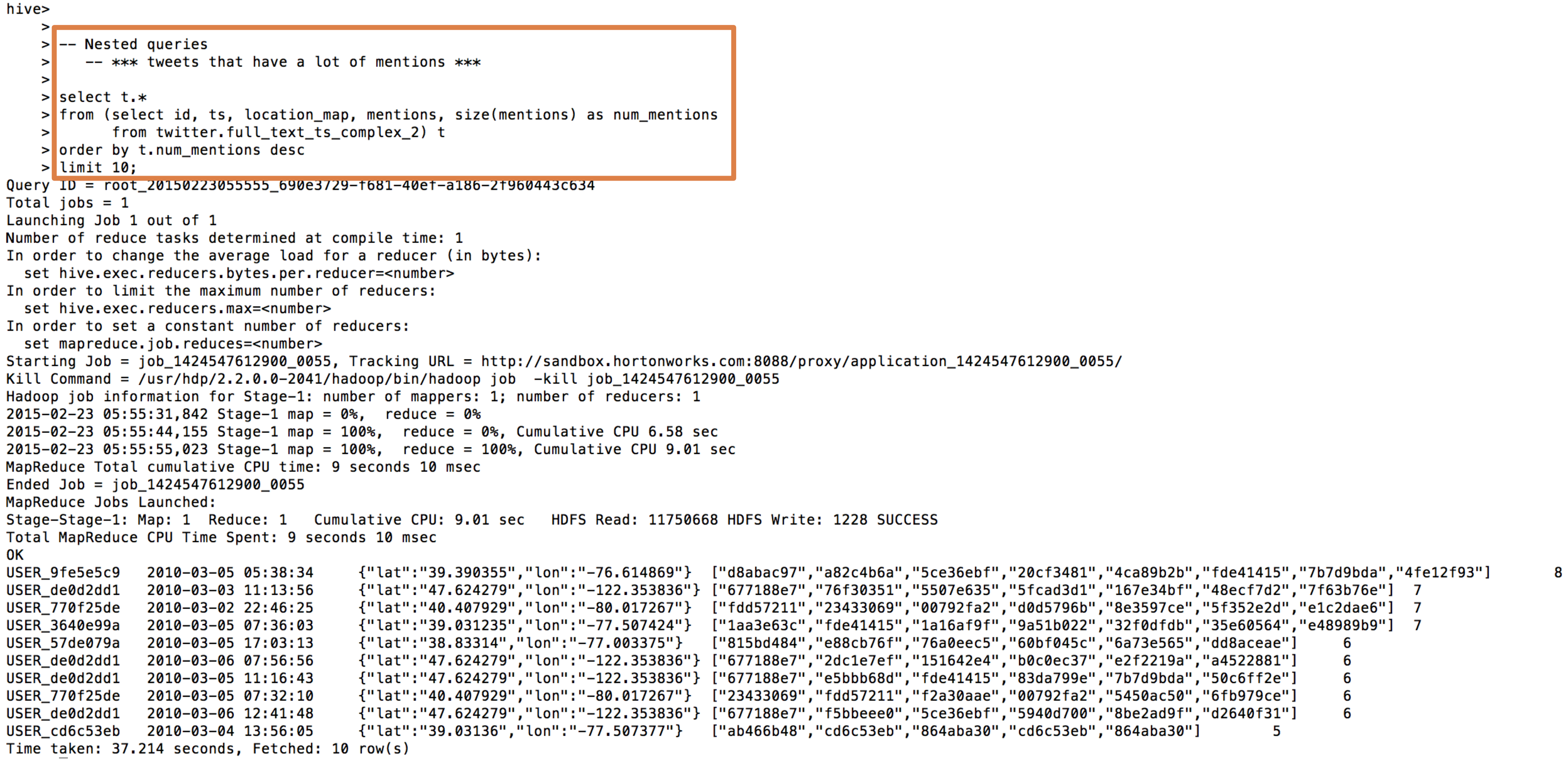
```
hive> -- collect_set function (UDAF)
    >    -- collect_set() is a UDAF aggregation function.. we run the query at this step
    >    -- from the previous step, we get all the mentions in the tweets but if a user
    >    -- has multiple mentions in the same tweet, they are in different rows.
    >    -- To transpose all the mentions belonging to the same tweet/user, we can use
    >    -- the collect_set and group by to transpose the them into an array of mentions
    >
    > create table twitter.full_text_ts_complex_2 as
    > select id, ts, location_map, tweet, collect_list(mention) as mentions
    > from twitter.full_text_ts_complex_1
    > group by id, ts, location_map, tweet;
FAILED: SemanticException org.apache.hadoop.hive.ql.parse.SemanticException: Table already exists: twitter.full_text_ts_complex_2
hive>
    > describe twitter.full_text_ts_complex_2;
OK
id                      string
ts                      timestamp
location_map            map<string,string>
tweet                   string
mentions                array<string>
Time taken: 0.734 seconds, Fetched: 5 row(s)
hive>
    > select * from twitter.full_text_ts_complex_2
    > where size(mentions) > 5
    > limit 10;
OK
USER_3640e99a   2010-03-05 07:36:03     {"lat":"39.031235","lon":"-77.507424"}  RT @USER_1aa3e63c: RT @USER_fde41415: RT @USER_1a16
_e48989b9: #FollowFriday ? RT   ["1aa3e63c","fde41415","1a16af9f","9a51b022","32f0dfdb","35e60564","e48989b9"]
USER_57de079a   2010-03-05 17:03:13     {"lat":"38.83314","lon":"-77.003375"}   #FF: @USER_815bd484: @USER_e88cb76f: @USER_76a0eec5
_dd8aceae: @USER_a6a19994       ["815bd484","e88cb76f","76a0eec5","60bf045c","6a73e565","dd8aceae"]
USER_770f25de   2010-03-02 22:46:25     {"lat":"40.407929","lon":"-80.017267"}  SCORES: @USER_fdd57211:9pts @USER_23433069:7pts @US
:2pts @USER_e1c2dae6:2pts CONGRATS!     ["fdd57211","23433069","00792fa2","d0d5796b","8e3597ce","5f352e2d","e1c2dae6"]
USER_770f25de   2010-03-05 07:32:10     {"lat":"40.407929","lon":"-80.017267"}  SCORES: @USER_23433069:10pts @USER_fdd57211:6pts @U
979ce:1pt CONGRATS!     ["23433069","fdd57211","f2a30aae","00792fa2","5450ac50","6fb979ce"]
USER_9fe5e5c9   2010-03-05 05:38:34     {"lat":"39.390355","lon":"-76.614869"}  RT @USER_d8abac97: RT @USER_a82c4b6a: RT @USER_5ce3
_7b7d9bda: RT @USER_4fe12f93: ReTweet this tweet if ... ["d8abac97","a82c4b6a","5ce36ebf","20cf3481","4ca89b2b","fde41415","7b7d9bd
USER_de0d2dd1   2010-03-03 11:13:56     {"lat":"47.624279","lon":"-122.353836"} RT: @USER_677188e7: RT @USER_76f30351: RT @USER_550
R_7f63b76e: YG MAU DIPROMOT     ["677188e7","76f30351","5507e635","5fcad3d1","167e34bf","48ecf7d2","7f63b76e"]
USER_de0d2dd1   2010-03-05 11:16:43     {"lat":"47.624279","lon":"-122.353836"} RT: @USER_677188e7: RT @USER_e5bbb68d: RT @USER_fde
 NOW    ["677188e7","e5bbb68d","fde41415","83da799e","7b7d9bda","50c6ff2e"]
USER_de0d2dd1   2010-03-06 07:56:56     {"lat":"47.624279","lon":"-122.353836"} RT: @USER_677188e7: RT @USER_2dc1e7ef: RT @USER_151
saturday? RT    ["677188e7","2dc1e7ef","151642e4","b0c0ec37","e2f2219a","a4522881"]
USER_de0d2dd1   2010-03-06 12:41:48     {"lat":"47.624279","lon":"-122.353836"} RT @USER_677188e7: RT @USER_f5bbeee0: RT @USER_5ce3
 rt cepet       ["677188e7","f5bbeee0","5ce36ebf","5940d700","8be2ad9f","d2640f31"]
Time taken: 0.196 seconds, Fetched: 9 row(s)
```

a list of mentions in a tweet

Hive Nested Queries

DEMO

Nested Queries



```
hive>
    >
    > -- Nested queries
    >    -- *** tweets that have a lot of mentions ***
    >
    > select t.*
    > from (select id, ts, location_map, mentions, size(mentions) as num_mentions
    >       from twitter.full_text_ts_complex_2) t
    > order by t.num_mentions desc
    > limit 10;
Query ID = root_20150223055555_690e3729-f681-40ef-a186-2f960443c634
Total jobs = 1
Launching Job 1 out of 1
Number of reduce tasks determined at compile time: 1
In order to change the average load for a reducer (in bytes):
  set hive.exec.reducers.bytes.per.reducer=<number>
In order to limit the maximum number of reducers:
  set hive.exec.reducers.max=<number>
In order to set a constant number of reducers:
  set mapreduce.job.reduces=<number>
Starting Job = job_1424547612900_0055, Tracking URL = http://sandbox.hortonworks.com:8088/proxy/application_1424547612900_0055/
Kill Command = /usr/hdp/2.2.0.0-2041/hadoop/bin/hadoop job  -kill job_1424547612900_0055
Hadoop job information for Stage-1: number of mappers: 1; number of reducers: 1
2015-02-23 05:55:31,842 Stage-1 map = 0%,  reduce = 0%
2015-02-23 05:55:44,155 Stage-1 map = 100%,  reduce = 0%, Cumulative CPU 6.58 sec
2015-02-23 05:55:55,023 Stage-1 map = 100%,  reduce = 100%, Cumulative CPU 9.01 sec
MapReduce Total cumulative CPU time: 9 seconds 10 msec
Ended Job = job_1424547612900_0055
MapReduce Jobs Launched:
Stage-Stage-1: Map: 1  Reduce: 1   Cumulative CPU: 9.01 sec   HDFS Read: 11750668 HDFS Write: 1228 SUCCESS
Total MapReduce CPU Time Spent: 9 seconds 10 msec
OK
USER_9fe5e5c9   2010-03-05 05:38:34     {"lat":"39.390355","lon":"-76.614869"}  ["d8abac97","a82c4b6a","5ce36ebf","20cf3481","4ca89b2b","fde41415","7b7d9bda","4fe12f93"]       8
USER_de0d2dd1   2010-03-03 11:13:56     {"lat":"47.624279","lon":"-122.353836"} ["677188e7","76f30351","5507e635","5fcad3d1","167e34bf","48ecf7d2","7f63b76e"]  7
USER_770f25de   2010-03-02 22:46:25     {"lat":"40.407929","lon":"-80.017267"}  ["fdd57211","23433069","00792fa2","d0d5796b","8e3597ce","5f352e2d","e1c2dae6"]  7
USER_3640e99a   2010-03-05 07:36:03     {"lat":"39.031235","lon":"-77.507424"}  ["1aa3e63c","fde41415","1a16af9f","9a51b022","32f0dfdb","35e60564","e48989b9"]  7
USER_57de079a   2010-03-05 17:03:13     {"lat":"38.83314","lon":"-77.003375"}   ["815bd484","e88cb76f","76a0eec5","60bf045c","6a73e565","dd8aceae"]     6
USER_de0d2dd1   2010-03-06 07:56:56     {"lat":"47.624279","lon":"-122.353836"} ["677188e7","2dc1e7ef","151642e4","b0c0ec37","e2f2219a","a4522881"]     6
USER_de0d2dd1   2010-03-05 11:16:43     {"lat":"47.624279","lon":"-122.353836"} ["677188e7","e5bbb68d","fde41415","83da799e","7b7d9bda","50c6ff2e"]     6
USER_770f25de   2010-03-05 07:32:10     {"lat":"40.407929","lon":"-80.017267"}  ["23433069","fdd57211","f2a30aae","00792fa2","5450ac50","6fb979ce"]     6
USER_de0d2dd1   2010-03-06 12:41:48     {"lat":"47.624279","lon":"-122.353836"} ["677188e7","f5bbeee0","5ce36ebf","5940d700","8be2ad9f","d2640f31"]     6
USER_cd6c53eb   2010-03-04 13:56:05     {"lat":"39.03136","lon":"-77.507377"}   ["ab466b48","cd6c53eb","864aba30","cd6c53eb","864aba30"]        5
Time taken: 37.214 seconds, Fetched: 10 row(s)
```

Union vs Union ALL

- https://cwiki.apache.org/confluence/display/Hive/LanguageManual+Union
- Union – Remove duplicates
- Union ALL – Does not remove duplicates

Hive Transform/MapReduce

- Hive also provides map(), reduce() and transform() capabilities that allows users to write more advanced and customized functions and thus have greater flexibility to control the map reduce jobs

```
-- word count in Hive with map and reduce
functions written in python
add file /root/lab/wc_mapper-2.py;
add file /root/lab/wc_reducer-2.py;

from (
        from raw_lines
        map raw_lines.line
        --call the mapper here
        using 'wc_mapper-2.py'
        as word, count
        cluster by word) map_output
insert overwrite table word_count
reduce map_output.word, map_output.count
--call the reducer here
using 'wc_reducer-2.py'
as word,count;
```

Hive manual on Transform: https://cwiki.apache.org/confluence/display/Hive/LanguageManual+Transform

Internal vs External Table

- **Internal Table**
  - Table Metadata and Data is under hive’s control
  - drop an internal table, it drops the data, and it also drops the metadata.
- **External Table**
  - Only metadata is under hive’s control
  - Drop an external table, it only drops the meta data
  - You can overlay multiple tables all pointing to the same raw data
  - External table files are accessible to anyone who has access to HDFS file structure and therefore security needs to be managed at the HDFS file/folder level.

```
create external table twitter.full_text_ts_complex_tmp (
id string,
ts timestamp,
tweet_struct string)
row format delimited
fields terminated by '\t'
stored as textfile
location '/user/root/full_text_ts_complex';
```

Readings

- https://cwiki.apache.org/confluence/display/Hive/Home#Home-UserDocumentation
  - Hive Tutorial
  - Language Manual
  - Hive/NoSQL Integrations
  - Hive Installation/Configurations
  - Many other resources
- https://cwiki.apache.org/confluence/display/Hive/LanguageManual+ORC
- https://www.mapr.com/blog/what-kind-hive-table-best-your-data
- https://acadgild.com/blog/apache-hive-file-formats/
- http://pyfunc.blogspot.ca/2012/03/external-tables-in-hive-are-handy.html

Hive Cheat Sheet

- http://hortonworks.com/wp-content/uploads/2013/05/hql_cheat_sheet.pdf