University of East London School of Architecture, Computing and Engineering (ACE)
Author: Dr Amin Karami (a.karami@uel.ac.uk) 19-20

Tutorial 5: KDDCup'99 Analysis using HIVE

CN7022 - Big Data Analytics

Dr Amin Karami (a.karami@uel.ac.uk)

LEARNING OUTCOMES: After completing this tutorial, you should:

- Have gotten a hands-on experience in using HIVE for data analysis
- Understand the real case studies applied in the Big Data platform for analysis
- Visualize the outcomes of queries into the graphical representations

Phase 1: Getting Ready

- This is the data set used for The Third International Knowledge Discovery and Data Mining Tools Competition, which was held in conjunction with KDD-99 The Fifth International Conference on Knowledge Discovery and Data Mining. The competition task was to build a network intrusion detector, a predictive model capable of distinguishing between "bad" connections, called intrusions or attacks, and "good" normal connections. This database contains a standard set of data to be audited, which includes a wide variety of intrusions simulated in a military network environment.
- The first task for big data analytics is to understand the data carefully. The featured are described here.
- This dataset has 22 types of bad connections: `back`, `buffer_overflow`, `ftp_write`, `guess_passwd`, `imap`, `ipsweep`, `land`, `loadmodule`, `multihop`, `neptune`, `nmap`, `perl`, `phf`, `pod`, `portsweep`, `rootkit`, `satan`, `smurf`, `spy`, `teardrop`, `warezclient`, `warezmaster`.
- The whole overview of the features are as follows:

feature name	*description*	*type*
`duration`	length (number of seconds) of the connection	continuous
`protocol_type`	type of the protocol, e.g. tcp, udp, etc.	discrete
`service`	network service on the destination, e.g., http, telnet, etc.	discrete
`src_bytes`	number of data bytes from source to destination	continuous
`dst_bytes`	number of data bytes from destination to source	continuous
`flag`	normal or error status of the connection	discrete
`land`	1 if connection is from/to the same host/port; 0 otherwise	discrete
`wrong_fragment`	number of "wrong" fragments	continuous
`urgent`	number of urgent packets	continuous
`host`	number of "host" indicators	continuous
`num_failed_logins`	number of failed login attempts	continuous

logged_in	1 if successfully logged in; 0 otherwise	discrete
num_compromised	number of “compromised” conditions	continuous
root_shell	1 if root shell is obtained; 0 otherwise	discrete
su_attempted	1 if “su root” command attempted; 0 otherwise	discrete
num_root	number of “root” accesses	continuous
num_file_creations	number of file creation operations	continuous
num_shells	number of shell prompts	continuous
num_access_files	number of operations on access control files	continuous
num_outbound_cmds	number of outbound commands in an ftp session	continuous
is_host_login	1 if the login belongs to the “host” list; 0 otherwise	discrete
is_guest_login	1 if the login is a guest login; 0 otherwise	discrete
count	number of connections to the same host as the current connection in the past two seconds	continuous
	Note: The following features refer to these same-host connections.	
serror_rate	% of connections that have “SYN” errors	continuous
rerror_rate	% of connections that have “REJ” errors	continuous
same_srv_rate	% of connections to the same service	continuous
diff_srv_rate	% of connections to different services	continuous
srv_count	number of connections to the same service as the current connection in the past two seconds	continuous
	Note: The following features refer to these same-service connections.	
srv_serror_rate	% of connections that have “SYN” errors	continuous
srv_rerror_rate	% of connections that have “REJ” errors	continuous
srv_diff_host_rate	% of connections to different hosts	continuous
connection_status	The status of the connection (class label)	Symbol

- A sample of data is:

```
tcp http REJ 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 3 0 0 1 0.33 1 0 0.67 3 255 1 0 0.33 0.15 0 0 1 0.79 normal.
tcp http REJ 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 1 0 0 1 1 1 0 0 13 255 1 0 0.08 0.15 0 0 1 0.79 normal.
tcp http REJ 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 1 0 0 1 1 1 0 0 23 255 1 0 0.04 0.14 0 0 1 0.79 normal.
tcp http REJ 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 2 0 0 1 1 1 0 1 33 255 1 0 0.03 0.13 0 0 1 0.79 normal.
tcp http REJ 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 1 0 0 1 1 1 0 0 43 255 1 0 0.02 0.12 0 0 1 0.79 normal.
tcp http REJ 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 1 0 0 1 1 1 0 0 53 255 1 0 0.02 0.11 0 0 1 0.79 normal.
tcp http REJ 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 2 0 0 1 1 1 0 1 63 255 1 0 0.02 0.09 0 0 1 0.79 normal.
tcp http REJ 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 1 0 0 1 1 1 0 0 73 255 1 0 0.01 0.07 0 0 1 0.79 normal.
tcp http REJ 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 2 0 0 1 1 1 0 1 83 255 1 0 0.01 0.07 0 0 1 0.81 normal.
tcp http REJ 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 1 0 0 1 1 1 0 0 1 255 1 0 1 0.07 0 0 1 0.85 normal.
icmp eco_i SF 8 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 29 0 0 0 0 1 0 1 2 2 1 0 1 0 0 0 0 0 ipsweep.
icmp eco_i SF 8 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 5 0 0 0 0 1 0 1 2 12 1 0 1 0.5 0 0 0 0 ipsweep.
icmp eco_i SF 8 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 10 0 0 0 0 1 0 1 2 22 1 0 1 0.5 0 0 0 0 ipsweep.
icmp eco_i SF 8 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 14 0 0 0 0 1 0 1 2 32 1 0 1 0.5 0 0 0 0 ipsweep.
icmp eco_i SF 8 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 19 0 0 0 0 1 0 1 2 42 1 0 1 0.5 0 0 0 0 ipsweep.
icmp eco_i SF 8 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 23 0 0 0 0 1 0 1 2 52 1 0 1 0.5 0 0 0 0 ipsweep.
```

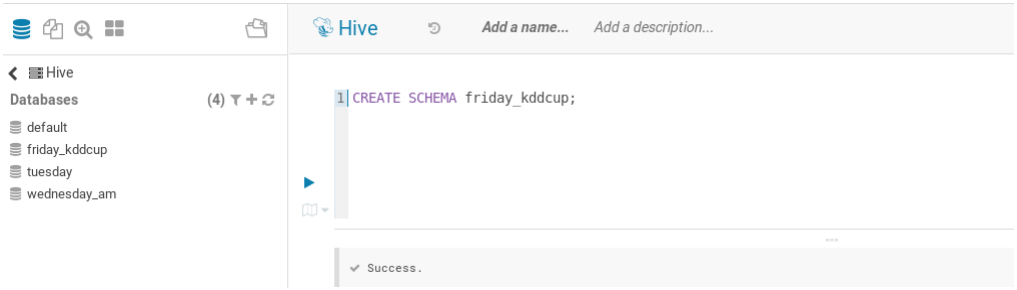
- Download the data from here. This is the KDDCup’99 data with 708MB. Then, copy this file into the Cloudera machine in **Desktop.**

Phase 2: Create a Database and Table in HIVE

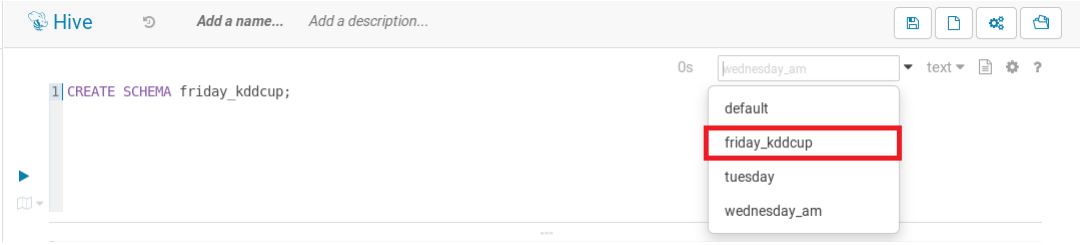
a) Create a database (Select your own database using the tutorial day):

Note: every group should create their own database: **friday, tuesday, wednesday_am, wednesday_pm**

```
CREATE SCHEMA friday_kddcup;
```



b) Refresh the page, then select the `friday_kddcup` database to work with it (Select your own database using the tutorial day):



c) Create a table:

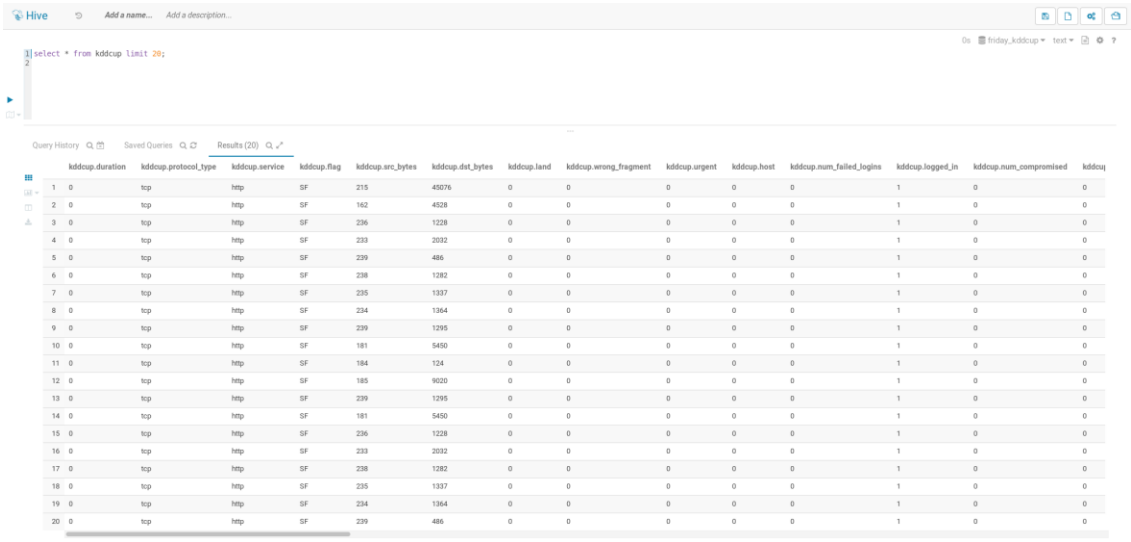
```
CREATE TABLE IF NOT EXISTS kddcup (duration int, protocol_type
string, service string, flag string, src_bytes int, dst_bytes
int, land int, wrong_fragment int, urgent int, host int,
num_failed_logins int, logged_in int, num_compromised int,
root_shell int, su_attempted int, num_root int,
num_file_creations int, num_shells int, num_access_files int,
num_outbound_cmds int, is_host_login int, is_guest_login int,
count int, srv_count int, serror_rate float, srv_serror_rate
float, rerror_rate float, srv_rerror_rate float, same_srv_rate
float, diff_srv_rate float, srv_diff_host_rate float,
dst_host_count int, dst_host_srv_count int,
dst_host_same_srv_rate float, dst_host_diff_srv_rate float,
dst_host_same_src_port_rate float, dst_host_srv_diff_host_rate
float, dst_host_serror_rate float, dst_host_srv_serror_rate
float, dst_host_rerror_rate float, dst_host_srv_rerror_rate
float, connection_status string) ROW FORMAT DELIMITED FIELDS
TERMINATED BY ',' LINES TERMINATED BY '\n' STORED AS TEXTFILE;
```

d) Load the data into table:

```
LOAD DATA LOCAL INPATH '/home/cloudera/Desktop/kddcup.data'
OVERWRITE INTO TABLE kddcup;
```

e) Let's show the first 10 records of data.

```
select * from kddcup limit 20;
```

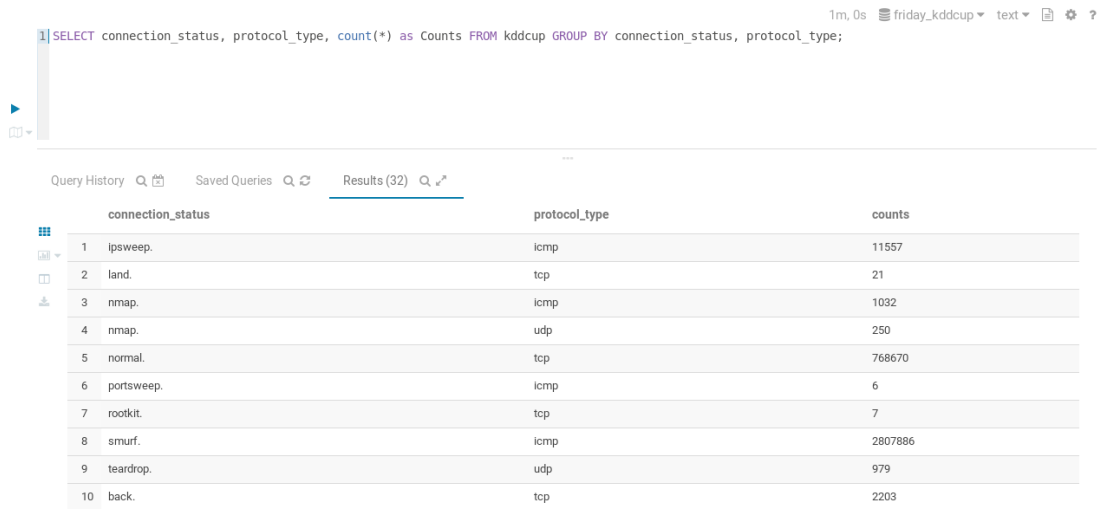


Phase 3: KDDCup Data Querying using HIVE

a) List the number of connections for different connections' statuses

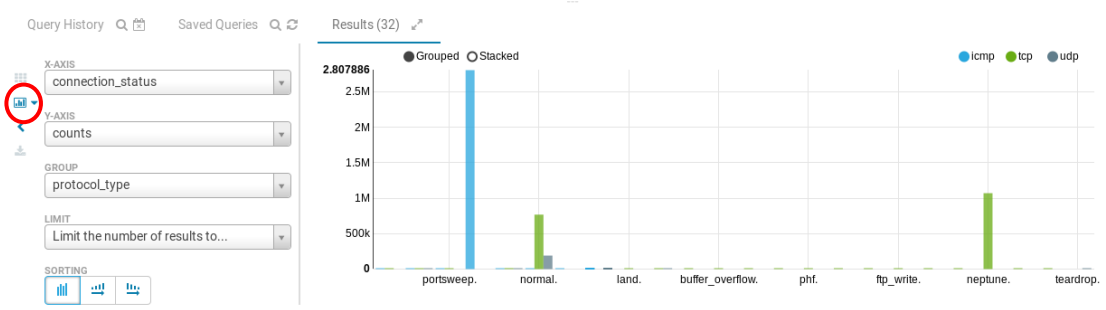
```
SELECT connection_status, protocol_type, count(*) as Counts FROM
kddcup GROUP BY connection_status, protocol_type;
```

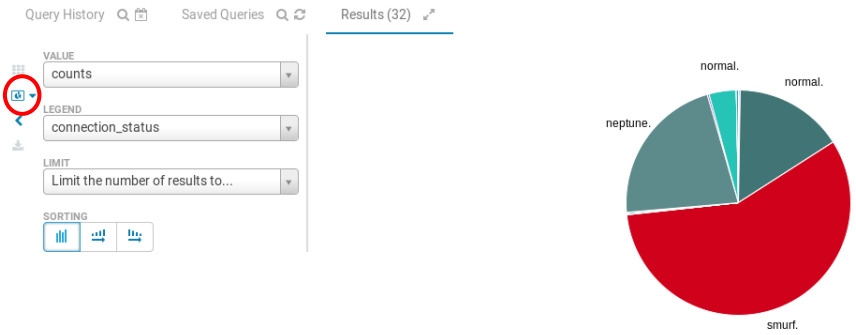
GROUP BY takes a while to be completed.



	connection_status	protocol_type	counts
1	ipsweep.	icmp	11557
2	land.	tcp	21
3	nmap.	icmp	1032
4	nmap.	udp	250
5	normal.	tcp	768670
6	portsweep.	icmp	6
7	rootkit.	tcp	7
8	smurf.	icmp	2807886
9	teardrop.	udp	979
10	back.	tcp	2203

Now, it is a time to present our work visually:

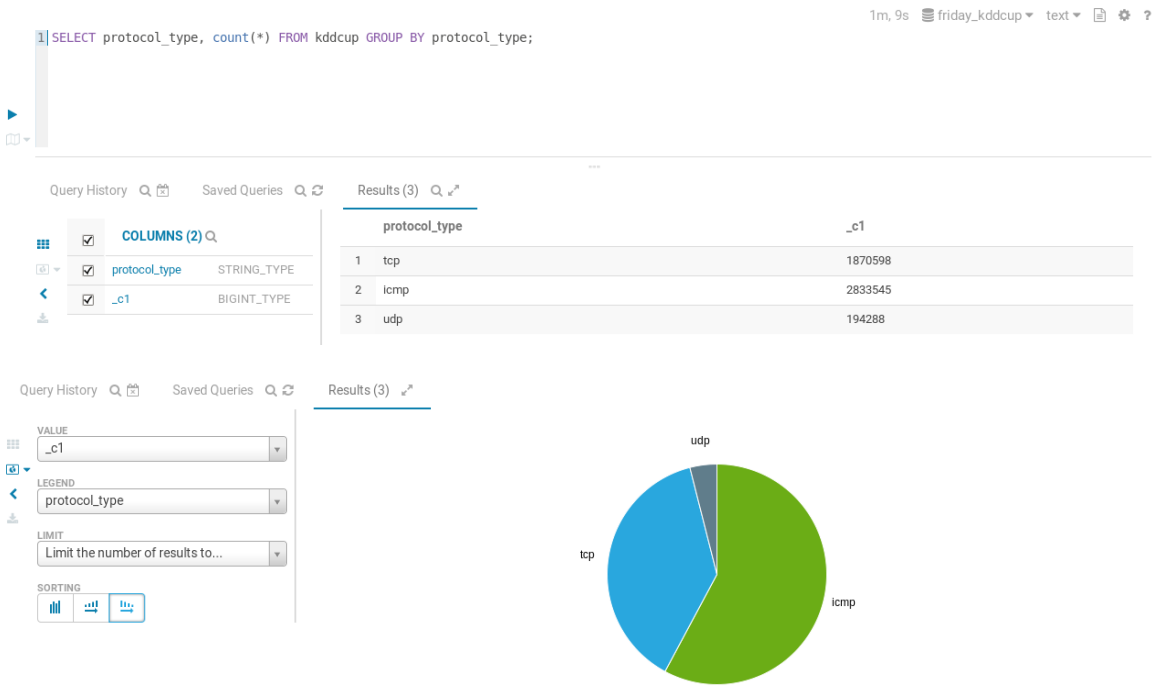


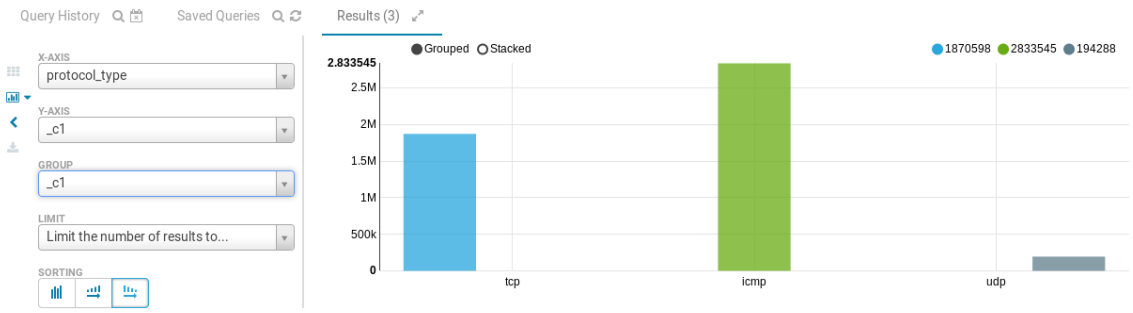


b) List the number of connections.

```
SELECT protocol_type, count(*) FROM kddcup GROUP BY protocol_type;
```

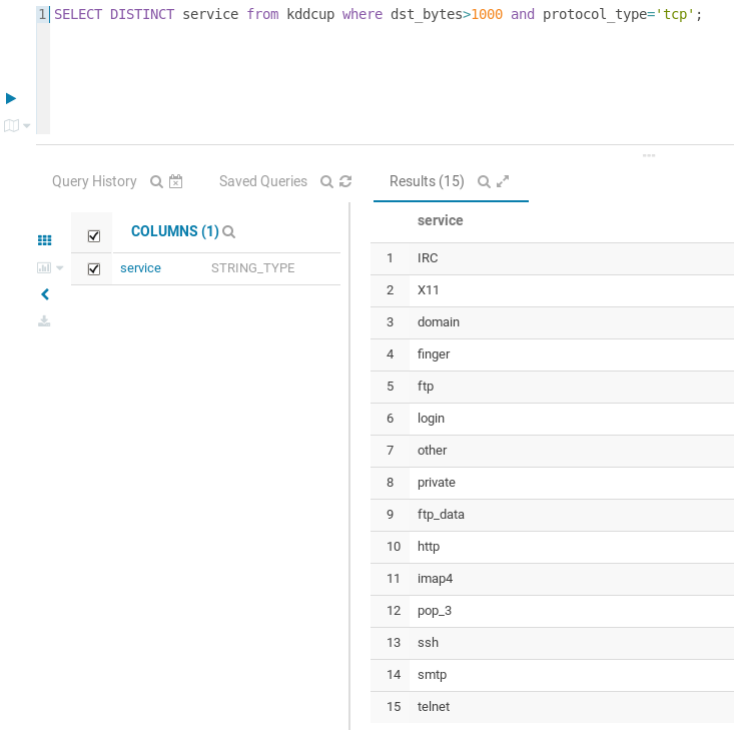
GROUP BY takes a while to be completed.





c) List the services based on the received destination packet's bytes and the `tcp` protocol.

```
SELECT DISTINCT service from kddcup where dst_bytes>1000 and protocol_type='tcp';
```



Since there is only one value/column, we cannot use 2D plots.

d) List the number of services based on the received destination packet's bytes and the `tcp` protocol.

```
SELECT service, count(*) from kddcup where dst_bytes>1000 and protocol_type='tcp' GROUP BY service;
```

```
1 SELECT service, count(*) from kddcup where dst_bytes>1000 and protocol_type='tcp' GROUP BY service;
```

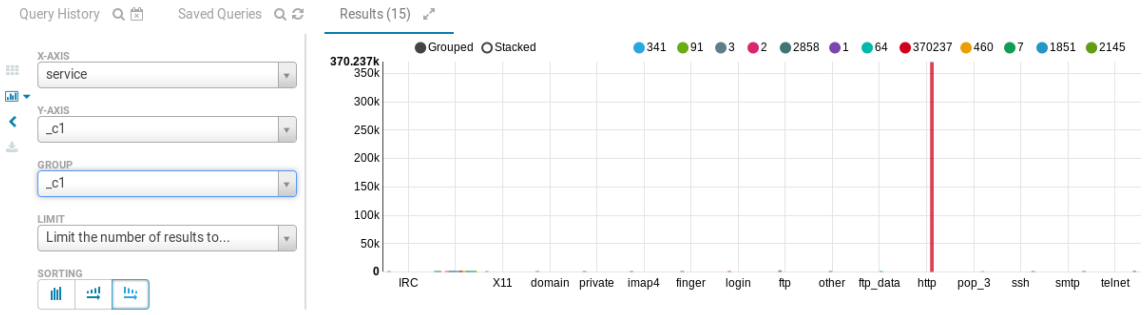
Query History | Saved Queries | Results (15)

COLUMNS (2)

☑	service	STRING_TYPE
☑	_c1	BIGINT_TYPE

	service	_c1
1	IRC	341
2	X11	91
3	domain	3
4	finger	2
5	ftp	2858
6	login	2
7	other	1
8	private	3
9	ftp_data	64
10	http	370237
11	imap4	3
12	pop_3	460
13	ssh	7
14	smtp	1851
15	telnet	2145

Query History | Saved Queries | Results (15)



Phase 4: Your turn to make more queries

It is your turn to practice HIVE queries using Analytic and Mathematical and many more functions. You should inquire the KDDCup dataset to create more information from this big sized dataset. For instance, provide useful information about the number of attacks (and sub-attacks) and normal connections. You can summarize the data based on the several features, such as the status of the connections, protocol types, services, server error rates, etc.