Troubleshooting Best Practice

Troubleshooting Topics

1. General Practice
2. Query Problems
3. Loading Problems
4. Schema Change Problems
5. GraphStudio Problems
6. Gbar Problems
7. Installation Problems

General Practice

General Practice

General Check

Before any investigation. Always make sure:

1. All services are up. (>gadmin status)
2. All servers are getting enough disk space (>df -lh)
3. All servers have enough memory (>free -g)
4. Memory usage is normal (>tsar)
5. No OOM or other stuff (>dmesg -T | tail)

General Practice

Locate the Root Folder

The root folder of Tigergraph can be found with command:

```
>gadmin --dump-config | grep root
```

```
tigergraph@ubuntu:~/tigergraph/zk$ gadmin --dump-config | grep root
tigergraph.log.root: /home/tigergraph/tigergraph/logs
tigergraph.root.dir: /home/tigergraph/tigergraph
```

Note:
Not all Tigergraph systems are installed under /home/tigergraph/tigergraph. But we will assume that in these slides

General Practice

Check Error Logs

When services are down? Check the corresponding logs, hopefully we can get some stacktrace or error message

use >gadmin log to list all logs

```
tigergraph@ubuntu:~/tigergraph/logs/nginx$ gadmin log
GPE : /home/tigergraph/tigergraph/logs/gpe/gpe_1_1.out
GPE : /home/tigergraph/tigergraph/logs/GPE_1_1/log.INFO
GSE : /home/tigergraph/tigergraph/logs/gse/gse_1_1.out
GSE : /home/tigergraph/tigergraph/logs/GSE_1_1/log.INFO
RESTPP : /home/tigergraph/tigergraph/logs/restpp/restpp_1_1.out
RESTPP : /home/tigergraph/tigergraph/logs/RESTPP-LOADER_1_1/log.INFO
RESTPP : /home/tigergraph/tigergraph/logs/RESTPP_1_1/log.INFO
GSQL : /home/tigergraph/tigergraph/dev/gdk/gsql/logs/GSQL_LOG
NGINX : /home/tigergraph/tigergraph/logs/nginx/nginx1.out
NGINX : /home/tigergraph/tigergraph/logs/nginx/nginx_1.error.log
NGINX : /home/tigergraph/tigergraph/logs/nginx/nginx_1.access.log
VIS : /home/tigergraph/tigergraph/logs/gui/gui_ADMIN.log
VIS : /home/tigergraph/tigergraph/logs/gui/gui_INFO.log
```

*.out logs are for errors, **check this one**

log.INFO logs are for normal behaviours

Different components

Zookeeper can be found at ~/tigergraph/zk/zookeeper.out.*

Kafka logs can be found at ~/tigergraph/kafka/kafka.out

TigerGraph

Query Problems

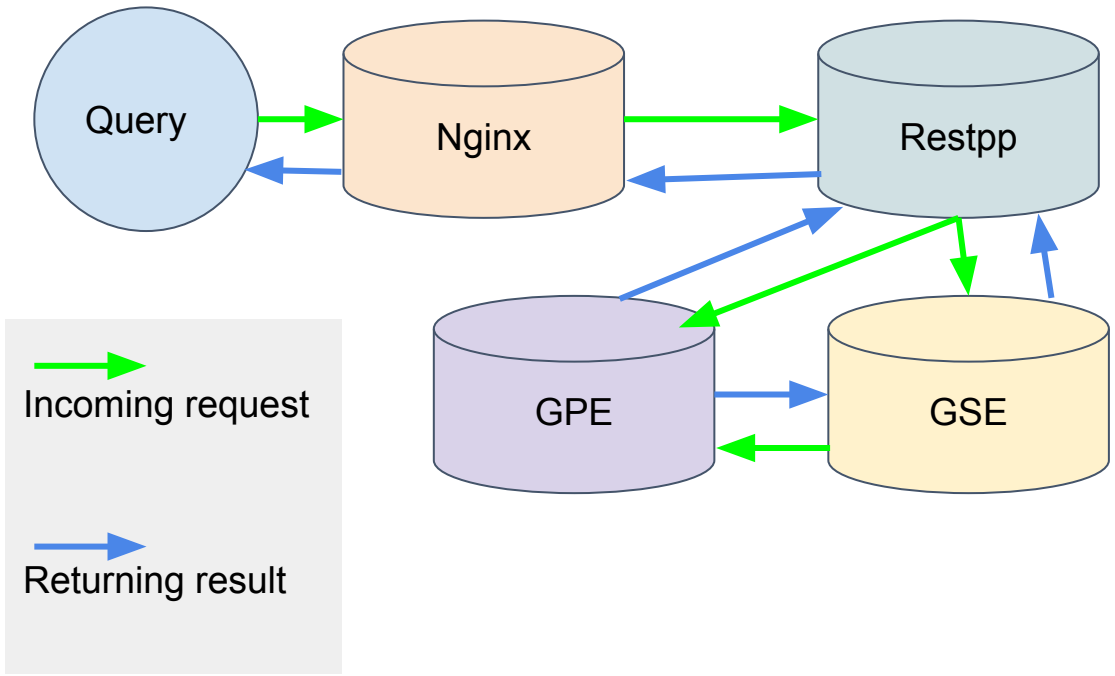
Query Problems

1. Log Checking
2. Query Slow
3. Query Hangs
4. Query No Result
5. Query Installation Fail

TigerGraph

Query Problems -- Log Checking

Query Execution Flow Chart



1. Nginx receives request
2. Nginx send request to Restpp
3. Restpp send ID translation task to GSE, and query request to GPE
4. GSE send translated ID to GPE, GPE starts to process query
5. GPE send result to restpp, GPE send translation task to GSE, GSE send translation result to Restpp
6. Restpp send result back to Nginx
7. Nginx send the response

TigerGraph

Query Problems -- Log Checking

Nginx receives the request

```
>grep QUERY_NAME ~/tigergraph/logs/nginx/nginx_1.access.log
```

```
tigergraph@ubuntu:~/tigergraph/logs/nginx$ grep InvitedUserBehavior  nginx_1.access.log
127.0.0.1 - - [21/Feb/2019:15:11:42 -0800] "GET /engine/query/AntiFraud/InvitedUserBehavior?inputUser=11 HTTP/1.1" 202 67 "http
://localhost:14240/" "Mozilla/5.0 (X11; Linux x86_64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/59.0.3071.86 Safari/537.36"
```

Nginx send request to Restpp

```
>grep QUERY_NAME /home/tigergraph/tigergraph/logs/RESTPP_1_1/log.INFO
```

```
grep InvitedUserBehavior /home/tigergraph/tigergraph/logs/RESTPP_1_1/log.INFO
I0221 15:11:42.138013  9181 handler.cpp:235] Engine_req|RawRequest|196610:RESTPP_1_1:1550790702138|GET|url = /query/AntiFraud/InvitedUserBehavior?inputUser
=11&|payload_data.size() = 2|api = v2
```

Request ID

Note:
Here **1550790702138** is the request ID. With request ID all logs in Restpp, GPE and GSE can be found.

TigerGraph

Query Problems -- Log Checking

GPE Process Query

GPE log is very important, most of time of a query is spent in GPE, GPE log gives you the detailed info of query Execution. Such as data amount has been processed, time elapsed in each ACCUM and POST-ACCUM clause.

```
>grep REQUEST_ID /home/tigergraph/tigergraph/logs/GPE_1_1/log.INFO
```

```
tigergraph@ubuntu:~/tigergraph/logs/nginx$ grep 1550790702138 /home/tigergraph/tigergraph/logs/GPE_1_1/log.INFO
I0221 15:11:42.142649 10265 serviceapi.cpp:563] Request|default,196610:RESTPP_1_1:1550790702138:NYN,16,,0|Start_RunUDF
I0221 15:11:42.142686 10265 gtimer.cpp:134] (0.000 ms) |default,196610:RESTPP_1_1:1550790702138:NYN,16,,0|UDF::Start
I0221 15:11:42.142748 10265 gtimer.cpp:134] (0.066 ms) |default,196610:RESTPP_1_1:1550790702138:NYN,16,,0|UDF::Initialization
I0221 15:11:42.142918 10265 gtimer.cpp:134] (0.171 ms) |default,196610:RESTPP_1_1:1550790702138:NYN,16,,0|UDF::Finished iteration 1 which did 2 edge maps a
nd 0 vertex maps (in 0.104 ms), 2 vertex reduces (in 0.058 ms), and activated 2 vertices.
I0221 15:11:42.143066 10265 gtimer.cpp:134] (0.144 ms) |default,196610:RESTPP_1_1:1550790702138:NYN,16,,0|UDF::Finished iteration 2 which did 1 edge maps a
nd 0 vertex maps (in 0.104 ms), 1 vertex reduces (in 0.031 ms), and activated 1 vertices
I0221 15:11:42.143324 10265 gtimer.cpp:134] (0.257 ms) |default,196610:RESTPP_1_1:1550790702138:NYN,16,,0|UDF::Finished iteration 3 which did 5 edge maps a
nd 0 vertex maps (in 0.049 ms), and activated 5 vertices.
I0221 15:11:42.143424 10265 gtimer.cpp:134] (0.730 ms) |default,196610:RESTPP_1_1:1550790702138:NYN,16,,0|UDF::End
I0221 15:11:42.143424 10265 serviceapi.cpp:566] Request|default,196610:RESTPP_1_1:1550790702138:NYN,16,,0|Stop_RunUDF|0ms
I0221 15:11:42.143529 10265 enginejobrunner.cpp:320] Request|default,196610:RESTPP_1_1:1550790702138:NYN,16,,0|Finished in 4.218 ms|Response 1277 bytes|Id
conversion 9
```

Number of ACCUM clauses executed in each SELECT statement

Query Started

End of execution

Query execution time

Time spent in ACCUM clause

Number of POST-ACCUM clauses executed in each SELECT statement and its corresponding execution time

# of vertices that has been selected to the vertex set

SELECT statements executed

Sent task to GSE and response to Restpp

TigerGraph

Query Problems -- Log Checking

GSE Process ID Translation Tasks

```
>grep REQUEST_ID /home/tigergraph/tigergraph/logs/GSE_1_1/log.INFO
```

```
:~/tigergraph/logs/nginx$ grep 1550790702138 /home/tigergraph/tigergraph/logs/GSE_1_1/log.INFO
9353  4731 service_dispatcher.cpp:152] Engine_GSE|default,196610:RESTPP_1_1:1550790702138:NYN,16,,0|typeduid2vid|1034|1|1|42
9506  4711 service_combiner.cpp:159] Engine_GSE|default,196610:RESTPP_1_1:1550790702138:NYN,16,,0|sendResponse|GPE_1_1|1|25
5072  4732 service_dispatcher.cpp:163] Engine_GSE|default,196610:RESTPP_1_1:1550790702138:NYN,16,,0,GPE_1_1,Y|vid2uid|1035|9|9|65
5757  4712 service_combiner.cpp:159] Engine_GSE|default,196610:RESTPP_1_1:1550790702138:NYN,16,,0,GPE_1_1,Y|sendResponse|RESTPP_1_1|9|88
:~/tigergraph/logs/nginx$
```

Log above means GSE has receive the request from Restpp and has send the response to GPE

uid2vid request from Restpp

Send response to GPE

vid2uid request from GPE

Send response to Restpp

Note:
Here **uid** means external string ID, and **vid** means internal vertex ID.

TigerGraph

Query Problems -- Log Checking

Restpp return the result to Nginx

```
>grep REQUEST_ID /home/tigergraph/tigergraph/logs/RESTPP_1_1/log.INFO
```

```
tigergraph@ubuntu:~/tigergraph/logs/nginx$ grep 1550790702138 /home/tigergraph/tigergraph/logs/RESTPP_1_1/log.INFO
I0221 15:11:42.138013  9181 handler.cpp:235] Engine_req|RawRequest|196610:RESTPP_1_1:1550790702138|GET|url = /query/AntiFraud/InvitedUserBehavior?inputUser
=11&|payload_data.size() = 2|api = v2
I0221 15:11:42.146179  9182 requestrecord.cpp:221] Engine_req|ReturnResult|196610:RESTPP_1_1:1550790702138|1167
```

Return to Nginx

Nginx send out the response

```
>grep QUERY_NAME ~/tigergraph/logs/nginx/nginx_1.access.log
```

```
tigergraph@ubuntu:~/tigergraph/logs/nginx$ grep InvitedUserBehavior  nginx_1.access.log
127.0.0.1 - - [21/Feb/2019:15:11:42 -0800] "GET /engine/query/AntiFraud/InvitedUserBehavior?inputUser=11 HTTP/1.1" 202 67 "http
://localhost:14240/" "Mozilla/5.0 (X11; Linux x86_64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/59.0.3071.86 Safari/537.36"
127.0.0.1 - - [21/Feb/2019:15:11:42 -0800] "GET /query/AntiFraud/InvitedUserBehavior?inputUser=11 HTTP/1.1" 200 1167 "-" "-"
```


TigerGraph

Query Problems -- Log Checking

Other useful commands

Check recently executed query:

```
>grep UDF:: /home/tigergraph/tigergraph/logs/GPE_1_1/log.INFO | tail -n 50
```

Get the number of queries executed recently:

```
>grep UDF::End /home/tigergraph/tigergraph/logs/GPE_1_1/log.INFO | wc -l
```

Note:
Due to log rotation, query logs may also logged in previous log files

Grep distributed query log:

```
>grep “Action done” /home/tigergraph/tigergraph/logs/GPE_1_1/log.INFO | tail -n 50
```

Grep logs from all servers:

```
>grun all “grep UDF:: /home/tigergraph/tigergraph/logs/GPE_*/log.INFO | tail -n 50”
```


TigerGraph

Query Problems -- Query Slow

Query performance may be slow for these reasons:

1. Insufficient Memory

When a query takes up too much memory, the engine will start to put data onto disk, and memory swap will also kick in. Use `>free -g` to check available memory and swap status.

Resolution: Optimize query data structure or increase physical memory size

2. GSQL Logic

For usual case a single server can process about 1 to 20 million edges per second. If the number gets really low, for most of the cases this happens because of the query logic. Please refer to slides Query Writing Best Practice for details on query performance tuning.

Query Problems -- Query Slow

3. Disk IO

When the query writes result to local disk. The disk IO might become the bottleneck of the query performance. Disk IO can be checked by using command `>sar 1 10`.

Resolution: the lines being printed can be stored in a data structure first. Then print the data structure. For details please refer to Query Writing Best Practice slides **Prints to File.**

4. Huge JSON response

When the JSON response is too big, it takes longer time to compose and transfer the JSON than traverse the graph. This can be identified by

```
grep UDF::End /home/tigergraph/tigergraph/logs/GPE_1_1/log.INFO
```

If the query is already end in GPE and haven't been sent out , along with about 200% CPU usage, then it probably be this case.

Resolution: reduce the JSON being printed

Query Problems -- Query Slow

5. Memory Leak

This is a rare case, when it happens the query gets slower and slower and the memory usage of GPE increases over time.

Resolution: report to engineering team to support.

6. Network Issue

When intense communication happens among the servers, the query will be slowed down. To identify this issue, you can observe that the CPU usage stays at a very low rate, and the GPE log.INFO keeps printing ???, meanwhile network IO is very high (???) (TODO)

Resolution: ????

Query Problems -- Query Slow

7. Frequent Data Ingestion in small batches

small batch of data ingestion increase the loading overhead and query processing workload.

Resolution: increase batch size

Query Problems -- Query Hangs

Query can hang for reasons:

1. Services are Down.

Please refer to General Practice->Check Error Log

2. Query Infinite Loop

- Refer to Query Problems->Log Checking, see if the iteration logs are continually produced. And the edgeMaps gives the same edge number in every few iterations.
- The iteration number is much larger than you expected.
- Check the CPU usage (>top), if a query is running it should be high.

Resolution:

Once confirmed, restart gpe to stop the query (>gadmin restart gpe -fy)

Check all **WHILE** statement in the query, see if it can properly stopped.

Note:
Stop GPE with caution.

TigerGraph

Query Problems -- Query Hangs

3. Query is Still Running, Just Slow

Be patient and refer to Query Problems->Query Slow

4. GraphStudio Error

If you are running the query from GraphStudio, it might keep rolling as if the query hasn't finished. Right click on the page, select **inspect->console**. See if is there any suspicious error.

TigerGraph

Query Problems -- Query No Result

It is a common problem happens during both PoC implementation and technical support.When your query logic is supposed to be correct but no result has been returned. The possible reasons are:

```
[
  {
    "@@circleEdgeTuples": []
  }
]
```

1. Data not Loaded

From the Load Data tab of GraphStudio, the numbers of edges and vertices can be checked.

Please make sure that all edge/vertex types that are needed for this query are loaded.

Type	Number
Total Vertex	20,741
Total Edge	56,096
Vertex "Payment_Instrument"	1,001
Vertex "Device_Token"	1,002
Vertex "User"	8,738
Vertex "Transaction"	10,000
Edge "User_Transfer_Transaction"	10,000

TigerGraph

Query Problems -- Query No Result

2. Properties not Loaded

By referring to Query Problems->Log Checking, The numbers of vertices/edges traversed can be observed from GPE log.INFO, for one of the iterations you might see **activated 0 vertices**. This means no target vertex satisfies your searching condition. Such as in one select statement, the query failed to pass where clause checking or having clause checking.

In a select statement log, if you got **0 vertex reduces** while the edge map number is not 0, that means all edges has been filtered out by **WHERE** clause that no vertex entered the **POST-ACCUM** phase. If you got more than 0 vertex reduced but **activated 0 vertices.** That means all the vertices were filtered out by the **HAVING** clause.

To confirm, pick a few vertices/edges that should have satisfied the condition check from GraphStudio. See what was wrong with their attributes.

Query Problems -- Installation Fail

When installing the query, it might fail at certain percentage. Different percentage indicates different reasons.

Usually, Installation Failure happens within this range because of a compilation issue.

check /home/tigergraph/tigergraph/dev/gdk/gsql/logs/GSQL_LOG and search for the last **error**. It will point to you the C++ compilation error.

Loading Problems

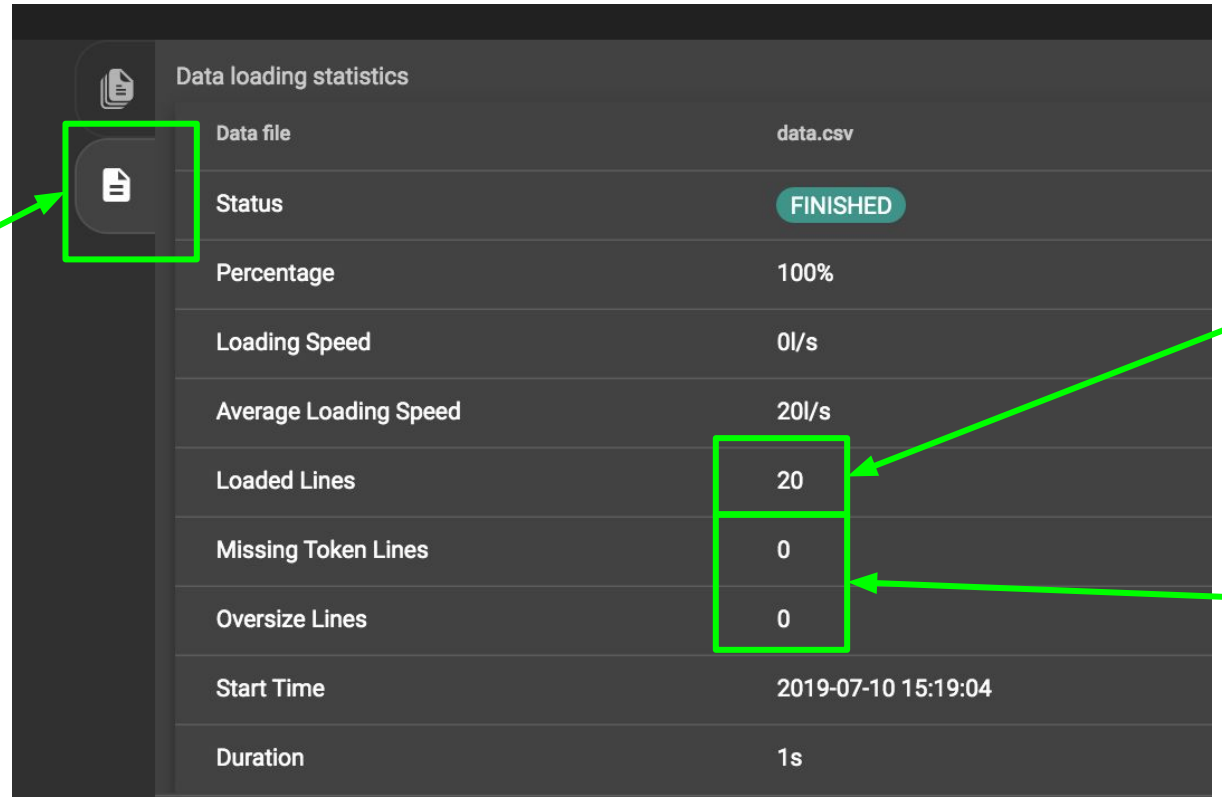
Loading Problems

1. Log Checking
2. Loading Slow
3. Loading Hangs
4. Data Not Loaded
5. Loading Fail

TigerGraph

Log Checking

Check log from GraphStudio



Click on the data source then click on the second tab

Make sure this equals to you total line number

Make sure these are 0 Missing Token: the header referred to doesn't exist

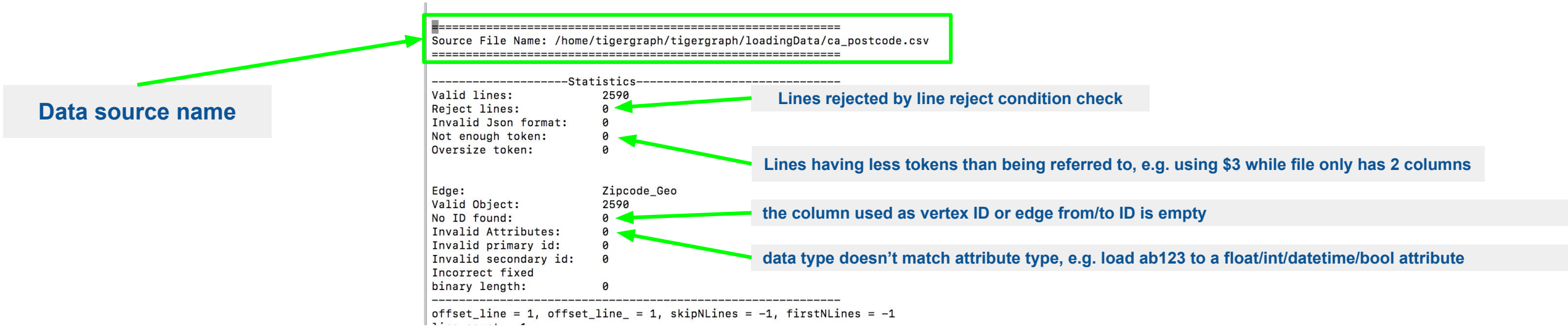
TigerGraph

Log Checking

Check log from command line

By default the log is located at **~/tigergraph/logs/restpp/restpp_loader_logs/GRAPH_NAME/**

The latest .log file is the log we want to look at.



TigerGraph

Log Checking

Check log from command line

```
============================================================
Source File Name: /home/tigergraph/tigergraph/loadingData/healthcare_facility_locations.csv
============================================================

--------------------Statistics------------------------------
Valid lines:              10771
Reject lines:             0
Invalid Json format:      0
Not enough token:         2 [ERROR]
Oversize token:           0


Edge:                     Facility_Geo
Valid Object:             10755
No ID found:              0
Invalid Attributes:       16 [ERROR] (e.g. 120:LATITUDE,482:LATITUDE,340:LATITUDE,314:LATITUDE
ATITUDE,943:LATITUDE,432:LATITUDE,460:LATITUDE,504:LATITUDE)
Invalid primary id:       0
Invalid secondary id:     0
Incorrect fixed
binary length:            0
------------------------------------------------------------
```

line number in the input file and the vertex/edge attribute had the mismatch

TigerGraph

Loading Slow

In normal situation, a single server TigerGraph can have loading performance **100k to 1000k** lines per second or **100GB to 200GB** of data per hour. However these numbers varies a lot based on many factors:

1. Loading logic. How many vertices/edges was generated each line.
2. Data format, is it JSON or CSV? Does it have multi-level delimiters. Is temp table intensively used?
3. Hardware configuration, SSD or HD, how many CPU cores.
4. The data is POST remotely or local, how is the network.
5. Size of each input file, many small files slows the loading down.
6. Value with high cardinality has been loaded to a string compress attribute

```
Using graph 'Geospatial'
The job loadGeo is created.
[Tip: Use "CTRL + C" to stop displaying the loading status update, then use "SHOW LOADING STATUS jobid" to track the loading progress again]
[Tip: Manage loading jobs with "ABORT/RESUME LOADING JOB jobid"]
Starting the following job, i.e.
  JobName: loadGeo, jobid: Geospatial.loadGeo.file.m1.1568930423085
  Loading log: '/home/tigergraph/tigergraph/logs/restpp/restpp_loader_logs/Geospatial/Geospatial.loadGeo.file.m1.1568930423085.log'

Job "Geospatial.loadGeo.file.m1.1568930423085" loading status
[FINISHED] m1 ( Finished: 3 / Total: 3 )
  [LOADED]
  +-------------------------------------------------------------------------------------------------------------------+
  |                                                                FILENAME |   LOADED LINES |    AVG SPEED |   DURATION|
  |                                 /home/tigergraph/tigergraph/loadingData/ca_postcode.csv |           2591 |     2 kl/s |     1.00 s|
  |/home/tigergraph/tigergraph/loadingData/healthcare_facility_locations.csv |          10774 |    10 kl/s |     1.02 s|
  |                                    /home/tigergraph/tigergraph/loadingData/patient.csv |          10774 |    10 kl/s |     1.02 s|
  +-------------------------------------------------------------------------------------------------------------------+
```

Loading log

Loading speed

TigerGraph

Loading Slow (Solution 💡)

To improve loading performance:

1. When CPU has many cores. We can consider to increase restpp load handlers.

   >gadmin --config handler
   Increase number of **Restpp-Loader.HandlerCount**
   Save
   >gadmin config-apply

2. Separate ~/tigergraph/kafka from ~/tigergraph/gstore, config them to different disks.

3. Do distributed loading.

4. Do offline batch loading

5. Combine smaller files into a larger file

Loading Slow (Solution 💡)

6. gadmin --config runtime
"GPE":"RebuildDeltaCountLimit=0"

7. gadmin --config rebuild_threads

Loading Hangs

Loading hanging can be due to reasons:

1. gpe is down.

```
>gadmin status gpe
```

```
>gadmin log -v gpe
```

2. memory is full.

```
>free -g
```

3. Disk is full

```
>df -lh
```

3. Kafka is down.

```
>gadmin status kafka
```

```
>vim ~/tigergraph/kafka/kafka.out
```


4. Kafka Loader config, by default it only allows one job

Data Not Loaded

Things to check:

- Any invalid lines in the log. If any input value format doesn't match vertex/edge attribute type. The corresponding vertex/edge won't be created.

- Is any field within quotes, while using quotes is not specified in the loading job. Delimiters within the quotes will mess up the tokenization.

- If after loading all vertices attributes weren't populated.
  Are edges loaded in the correct order? E.g. from id and to id are reversed.

Loading Fail

Possible causes:

1. Loading job hangs for 600s, reached timeout
2. Port is occupied. Port 8500

Schema Change Problems

Schema Change Problems

1. Schema Change Mechanism
2. Schema Change Fail

TigerGraph

Schema Change Mechanism

Schema change is a three-step process which is controlled by the admin server.

1. DSC Drain: This step stops traffic from coming into RESTPP, GPE.
2. DSC Validation: This step verifies that there is no running query and all services are good to go.
3. DSC Apply: This step does the actual work and changes the schema for all services.
4. DSC Resume: This step resumes traffic after DSC.

Schema change is a cluster-wise operation that affects all partitions and replicas of all services.

Schema Change Fail

Signs of DSC Failure (GSQL)

- Creating a graph fails
- Global schema change fails
- Local schema change fails
- Drop graph failure
- GPE/RESTPP failed to start due to YAML error (Bug alert!)

When the above errors occur, first check the GSQL log to see what is the error code. There are two error codes 8 and 310. One is 8, this means the system is in critical non-auto recoverable error. We need manual resolution in this case. **We have never seen error 8 in practice**. The most common error code is 310. When schema change fails with 310, it means that schema change failed and the proposed change has not taken effect **at all**. E.g. the user tries to add an edge type and delete a vertex type. 310 means both failed, the result will not be only the edge type is added or only the vertex type is deleted.

Schema Change Fail