## Common Error Parsing Keywords :

### Postgres XL:

#### Log keywords to monitor :

Error / error

FATAL

failed

PANIC

could not

WARNING

#### What folders / files to monitor :

**GTM:** gtm.log

**GTM Proxy:** gtm\_proxy.log

**Coordinator:**

/eqx/logs/pg\_logs.log

<coordinator\_folder>/pg\_log/postgresql-YYYY-MM-DD\_TIME.log

**Datanode:**

<datanode\_folder>/pg\_log/postgresql-YYYY-MM-DD\_TIME.log

**Datanode Slave:**

<datanode\_slave\_folder>/pg\_log/postgresql-YYYY-MM-DD\_TIME.log

### Cassandra:

#### Log Keywords to monitor :

RuntimeException

Error / error

FATAL

failed

PANIC

could not

WARNING

#### What folders / files to monitor :

**$CASSANDRA\_HOME/logs/system.log**

**$CASSANDRA\_HOME/logs/hs\_err\_pid<pid\_number>.log** (This will get generated only in cases when there is core dump exception -- details of this exception will be in the generated file)

## 

## Postgres:

**psql: FATAL: database "root" does not exist**This is a common issue if the user is trying to access psql without specifying database name. Psql assumes the database name is the same as the username.

If this is not the name of an actual database you can ignore this.

**Failed to get pooled connections**This means the application cannot get pooled connections. Increase the max\_pool\_size and max\_connections in the postgresql.conf of the nodes

**“archive command failed with exit code “**The postgres node is set with archive mode enabled but the command given for archiving is failing.

Check the archive command on postgresql.conf (check the last key-value that is referred) --

* Make sure that the user, remote IP and more importantly the remote folder path is correct.
* That the user that is referenced on the remote machine has rights to the folder that is referenced.
* A slave node is started correctly on the remote node and is listening for replication.
* Try to run the rsync command directly from the command line with the same user(toro) as the database and check there are no issues.

Issues with archival will quickly start filling the pg\_xlog folder -- This is a clear sign that the archival is not happening. Once you start archival and it gets completed the pg\_xlog folder will shrink down.

**PANIC: could not write to file "pg\_xlog/<file name>": No space left on device**This would happen in a situation where the HDD or partition allocated for postgres data becomes full. In a situation where you don’t have space there are not too many folders you can safely delete without corrupting your install.

* You can take a backup of the pg\_log files on another drive or machine or delete them if they are very old.
* If you don’t have sufficient space you will not be able to start the postgres instance. In extreme cases where there is no space left and no log files you can delete then you can take the step of moving the pg\_xlog to a different drive, create a symlink to the pg\_xlog folder and then try to start the database. (Keep in mind, TAKE AND KEEP A SAFE BACKUP OF THE pg\_xlog folder first. If this is corrupted somehow you will not be able to get into your database)
* This situation normally happens when the archival to slave is configured but is not happening, the pg\_xlog will start filling up very quickly.

**FATAL: database "<database\_name>" does not exist  
The database subdirectory "base/<some number>" is missing.**This issue will showup if a database is created on a cluster later it is deleted when some of the nodes are not available you will see inconsistencies in some nodes.

If you see these errors on data nodes you will have to drop the database from the datanodes. You can not run queries on the datanodes directly, instead, what you have to do is from any coordinator run the following query:

execute direct on (dn1) ‘drop database dummy\_db’;

**database "testdb1" is being accessed by other users**If you try to drop a database and it is being accesssed by other users then you will not be able to drop it. You will need to ensure that all connections to this database are closed. Postgres-XL does internal data fetching using prepared\_transactions -- if any of the nodes have prepared transactions open on them, postgres will not allow you to drop the database.

On each coordinator and datanode you will need to first close the prepared transactions and then can you drop the database. Currently open transactions can be seen by running the query.

SELECT \* FROM pg\_prepared\_xacts WHERE database='mydb';

To rollback a particular transactions, you can run the following:

ROLLBACK PREPARED 'GID';

**GTM error, could not obtain snapshot. Current XID =**This will often be seen on Coordinator and Datanode machines if either the GTM or the GTM Proxy is not reachable.

If this happens on all the machines it is more than likely an issue the the GTM. If it happens on only some nodes that are connected to a particular proxy then it is an issue with that particular

**can not execute CREATE DATABASE "MyTest" in a read-only transaction.**

This may happen when you are trying to create the database,table, inserting the data on the datanode. You can do only selects but you can’t write on datanode. You can do all the transaction on coordinator node only

**"could not create socket: Too many open files"**

This is the OS level error. This may happen when ulimit for that user is set to less. Please set the ulimit more or unlimited

**"could not create temporary file base/<file name>:Too many open files"**

This may happen when ulimit is set to less to that particular user. Kindly set the ulimit more or unlimited for the postgres user.

**Pgxc\_ctl monitor all is slow**

This issue comes when the pgxc\_ctl try to get the PID from the nodes. It will slow down the status checking process of the node. Restart the node one from the pgxc\_ctl and try, if this not works, go to the node machine kill the process with pid and then remove the postmaster.pid(data and coordinator nodes)/gtm.pid(GTMs) and start the node from pgxc\_ctl. Then do the monitor all, this should work fine.

**ERROR: could not open file "base/6876991/17617753": No such file or directory**

Following may also be part of the error:

CONTEXT: writing block 0 of relation base/6876991/17617753

WARNING: could not write block 0 of base/6876991/17617753

DETAIL: Multiple failures --- write error might be permanent.

This indicates some corruption on the database level and can be a serious issue. This should be addressed as quickly as possible in the next available maintenance window.

Follow these steps :

* Copy out all important data from this datanode into CSV files so you have the data reference in case this datanode does not start again.
* Shut down the database on that datanode.
* Take a tar backup of the base, pg\_xlog and pg\_clog folders -- This is done so that there is a backup of the contents of the database.
* Change the log\_min\_messages to debug3 or debug5 --- this is a temporary change -- DO NOT FORGET TO CHANGE THIS SETTING BACK ONCE RECOVERY IS COMPLETE. Otherwise the logs will be flooded and you will use up all the drive space.
* The database may now go into recovery mode.
* In the logs you will be able to see the transaction that the node is trying to process, based on the starting point of transactions, and the number of transactions that need to be processed it will take that much time to recover. This is a not an exact estimate, but a pg\_xlog folder size of 22GB or so should take about 45 - 50 mins.
* Remember: If you stop the recovery process in the middle and try to restart it, the database will again start from the beginning. If there is progress, you have to be patient and let the database do what it is doing. Basically the database has gotten corrupted but Postgres is trying to replay all the transactions to get your database back to the correct state as it was before.
* Once the recovery is complete the datanode should start as normal automatically. You will also notice the size of the pg\_xlog folder shrink substantially.
* If it takes much longer than the estimate you can start the database in a single user mode. This is like a safemode for the database :
  + To start in the single user mode, stop the database and start it using the following command:

/opt/damocles/lib/postgres-xl/bin/postgres --single DBNAME=alltrade -D /eqx/dn6\_slave -d 3 -r /tmp/out\_put\_format

* Here if you have other problems with the database you may be able to recover, like corrupt indexes etc.

## Postgres XL - pgxc\_ctl :

**monitor all is very slow or stops after showing one or 2 nodes.**

* If this is a new cluster or this happens on a newly added node, you want to check the following things :
  + Make sure there is passwordless SSH between this machine of the cluster. If this is not working correctly then you need to fix this first before checking anything else.
  + Make sure there is no iptables / selinux stopping communication between the pgxc\_ctl machine and the machines in the cluster.
* If your cluster was working before and this is happening then :
  + This happens normally when there is a problem with the GTM or GTM Proxy. The problem is the Coordinator may be up and may not be able to get a transaction ID from the GTM. You can check this from looking at the logs of the component that is supposed to show in the list next for the monitor all or the last component that is shown on the list.
  + For GTM transaction ID issues you can just restart the GTM and the GTM Proxy and that should fix the issue. If this is an issue on a coordinator or datanode, you will have to check the logs for that component.

**pgxc\_ctl.conf is corrupted.**

When you start pgxc\_ctl you may see this error -- or this may come up when you add / remove / change a node on the cluster. This means some entry on the pgxc\_ctl.conf is incorrect. The thing to remember about this file is.

* It’s a normal text config file. The variables used in the end are the ones that get used -- so if a value is declared on top and again declared in the end, the one in the end will be used.
* pgxc\_ctl utility tracks the cluster using this file -- if you remove the entries from here then pgxc\_ctl will not be able to know what servers you have in the cluster.
* When pgxc\_ctl makes changes to this file, *it will always append things in the end. Pgxc\_ctl* will also put a date and time reference on when it made the change.This is a good indicator to look for problems if they happen automatically and someone didn’t manually change things look at the date reference and see if you are able to locate mismatched number of entries like this:

datanodeSlave=y

datanodeSlaveServers=( 10.198.71.49 10.198.71.50 10.198.71.51 10.198.71.52 10.198.71.53 10.198.71.31 10.198.71.55 )

datanodeSlavePorts=( 40101 40102 40103 40104 40107 )

datanodeSlavePoolerPorts=( 40111 40112 40113 40114 40106 40117 )

datanodeArchLogDirs=( /eqx/dn\_archlog1 /eqx/dn\_archlog2 /eqx/dn\_archlog3 /eqx/dn\_archlog4 /eqx/dn\_archlog5 /eqx/dn\_archlog6 /eqx/dn\_archlog7 )

datanodeSlaveDirs=( /eqx/dn1\_slave /eqx/dn2\_slave /eqx/dn3\_slave /eqx/dn4\_slave /eqx/dn5\_slave /eqx/dn6\_slave /eqx/dn7\_slave )

datanodeSlaveWALDirs=( none none none none none none none )

There are 7 values in the datanodeSlaveServers and 5 values in datanodeSlavePorts and 6 values in datanodeSlavePoolerPorts -- This is a wrong configuration and what you will expect to see as problems when pgxc\_ctl complains of config file corruption issues.

If you changed something manually, then you can reference what you changed. Remember, before changing anything --- Please take backup of the conf file first.

## Cassandra

1. ERROR [HintedHandoff:xx]

This is the error when you any pending hints.

This you can clear by ***nodetool truncatehints*** tools which will clear the any pending hints in the node

1. java.lang.RuntimeException: Insufficient disk space to write 572 bytes

* This is the very common error when disk is full.
* This you can avoid by checking the disk space regularly.
* You can move the logs to the different folder, which are already zipped by the cassandra.
* Create the data folder in the other directory which is having sufficient disk space and create the symlink.
* In most inevitable situation add the node to that datacenter and do nodetool rebuild and nodetool cleanup

1. ERROR [SSTableBatchOpen:1]

This when sstable is of the tables is corrupted or broken. You will see this when you restart the cassandra. Nodetool is having the tool to scrub the corrupt sstable, when you start the cassandra commitlog will replay all the data to sstable

sstablescrub <key\_space> <column\_family>

Keyspace → Database in which the sstable is corrupt

Column\_family → Table in which the sstable is corrupt

1. Unable to find java executable. Check JAVA\_HOME and PATH environment variables.

This is may happen when JDK is not installed, not set in the user path or you are trying to login with different user.

* Install the java version "1.8.0\_121" and above
* If Java is installed check the path in bashrc and the user

1. Connection error: ('Unable to connect to any servers', {'192.168.0.45': AuthenticationFailed(u'Failed to authenticate to 192.168.0.45: code=0100 [Bad credentials] message="org.apache.cassandra.exceptions.UnavailableException: Cannot achieve consistency level QUORUM"',)})

* This error will come when you configure the cassandra for password authentication and forget to replicate the system\_auth keyspace
* Change the cassandra.yaml below settings to the default values

***authenticator: PasswordAuthenticator***

***#authenticator: AllowAllAuthenticator #DEFAULT value***

***authorizer: CassandraAuthorizer***

***#authorizer: AllowAllAuthorizer #DEFAULT value***

* Use the cqlsh and change the replication of system\_auth equal to your nodes and data centers as below example
* ALTER KEYSPACE system\_auth WITH replication = {'class': 'NetworkTopologyStrategy', 'dc1': 4, 'dc2': 2, 'dc3': 3 }
* After alter exit the cqlsh change the values of parameter in cassandra.yaml to real value of the authentication.
* Restart the cassandra and login into cqlsh with user-name and password.

1. Connection error: ('Unable to connect to any servers', {'10.138.32.76': OperationTimedOut('errors=Timed out creating connection (5 seconds), last\_host=None',)})

* This is timeout error while starting the cassandra. Either you can try after sometime or you can use the --request-timeout=<int> switch for example

***cqlsh -p cassandra -u cassandra -h 10.138.32.76 9042 -k all\_trade --request-timeout=200***

Binary stopped and no error in the system.log -- You will have a file with the name like **hs\_err\_pid<pid\_number>.log --** in the cassandra logs folder.

Error inside this file will be something like:

Failed to write core dump. Core dumps have been disabled. To enable core dumping, try "ulimit -c unlimited" before starting Java again

* Check for user level limits by

ulimit -a

core file size (blocks, -c) 0  
data seg size (kbytes, -d) unlimited  
scheduling priority (-e) 0  
file size (blocks, -f) unlimited  
pending signals (-i) 2066215  
max locked memory (kbytes, -l) 64  
max memory size (kbytes, -m) unlimited  
open files (-n) 4096  
pipe size (512 bytes, -p) 8  
POSIX message queues (bytes, -q) 819200  
real-time priority (-r) 0  
stack size (kbytes, -s) 10240

cpu time (seconds, -t) unlimited

max user processes (-u) 2066215

virtual memory (kbytes, -v) unlimited

file locks (-x) unlimited

Look for any low values like the one highlighted above -- Make sure you are not setting these values on temporary basis on the OS -- otherwise after restart the values will get reset and the application will crash with the same values.

toro - memlock unlimited  
toro - nofile 100000  
toro - nproc 32768  
toro - as unlimited

Also make sure the following settings are correct in the /etc/sysctl.conf

###### net.core.rmem\_max = 16777216 net.core.wmem\_max = 16777216 net.core.rmem\_default = 16777216 net.core.wmem\_default = 16777216 net.core.optmem\_max = 40960 net.ipv4.tcp\_rmem = 4096 87380 16777216 net.ipv4.tcp\_wmem = 4096 65536 16777216