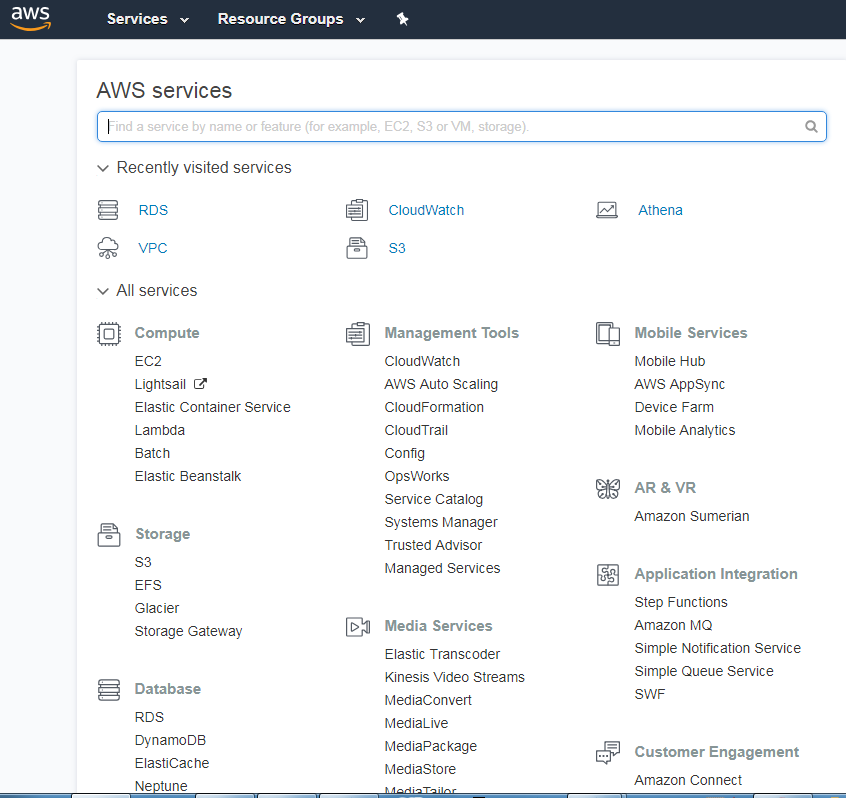
DBA MONITORING AWS

ATS DBA

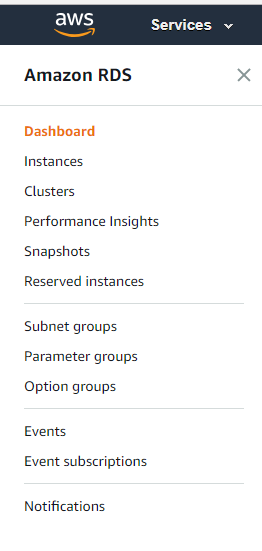
June 2018

1. RDS

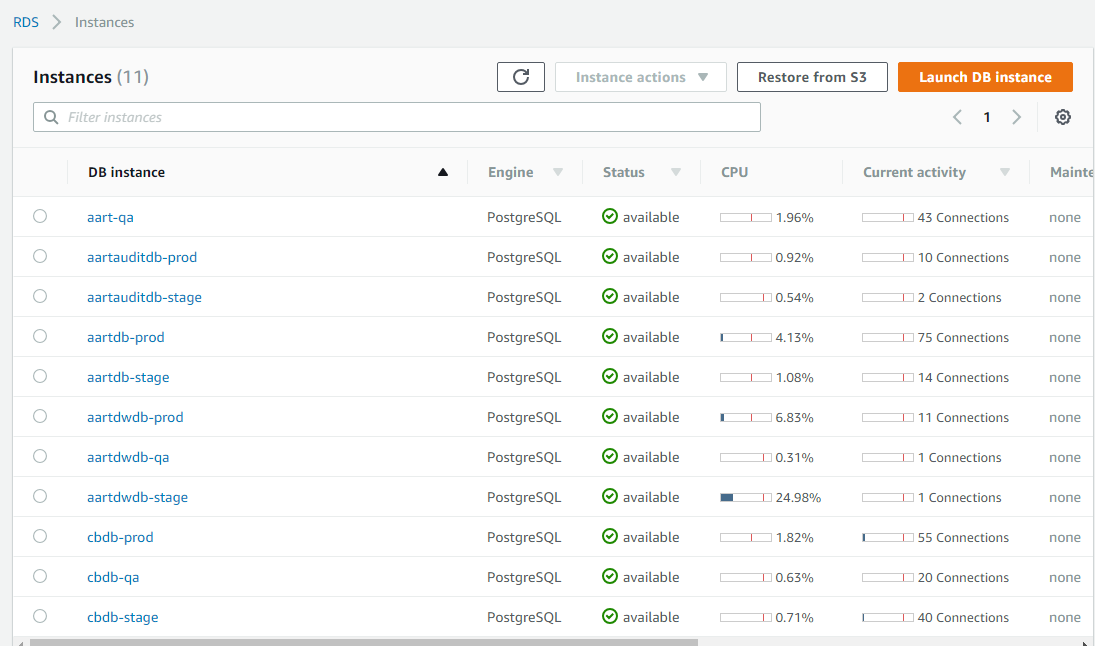
Login AWS

c

Click on RDS



Click on Instances

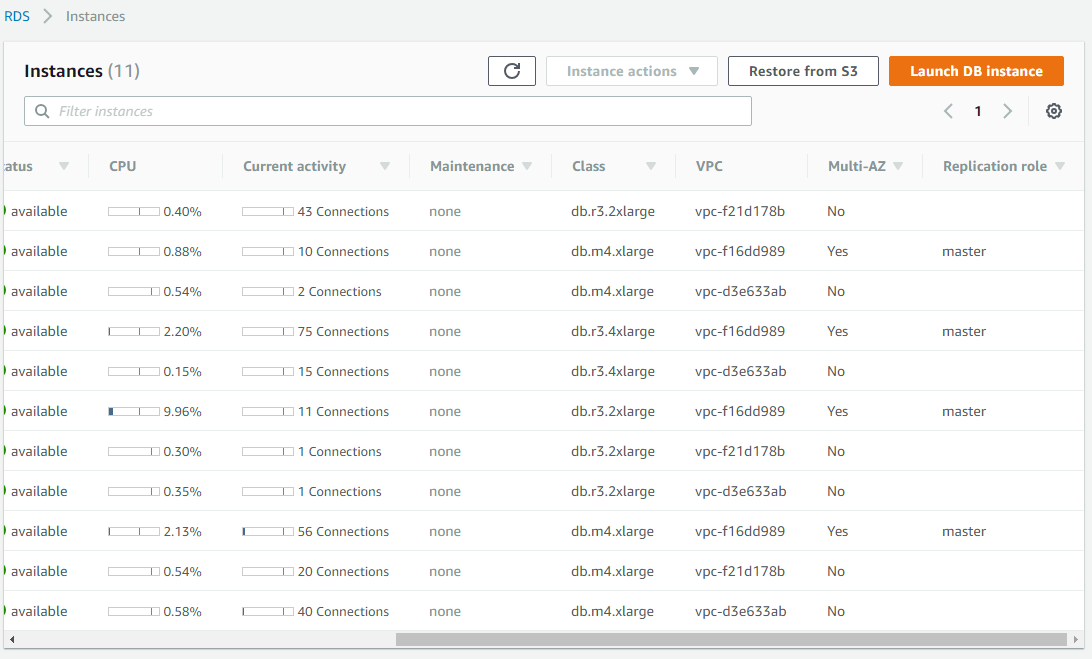


Observer CPU levels and connection numbers, to see if there are any alarming activities.

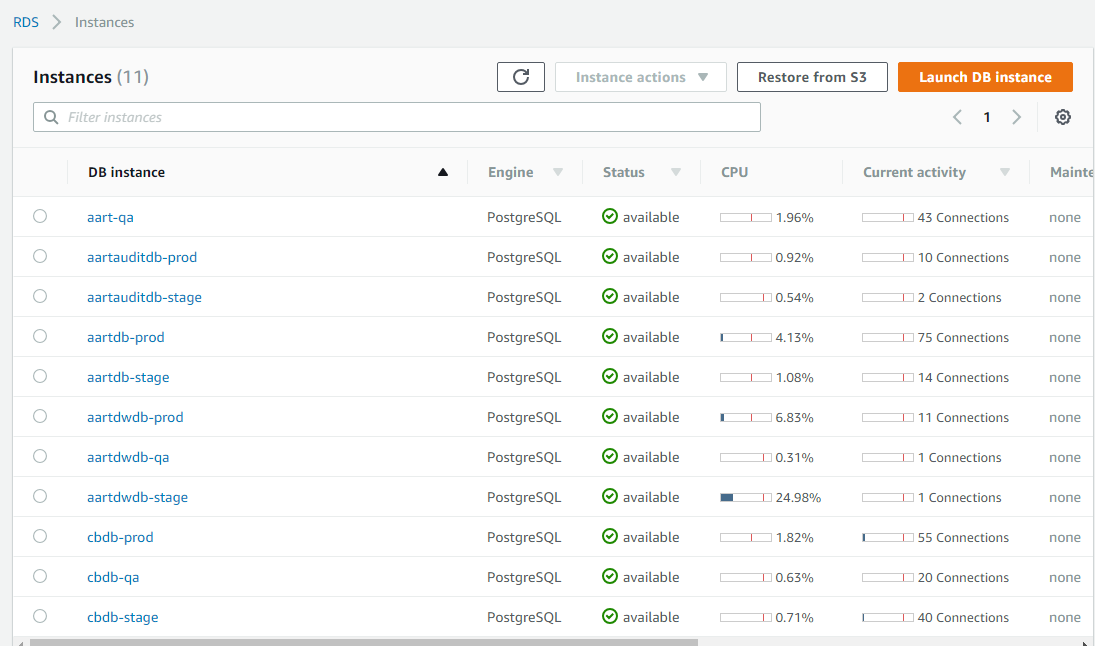
You may also run the query (with PgAdmin or PSQL) to see query activity in a specific database

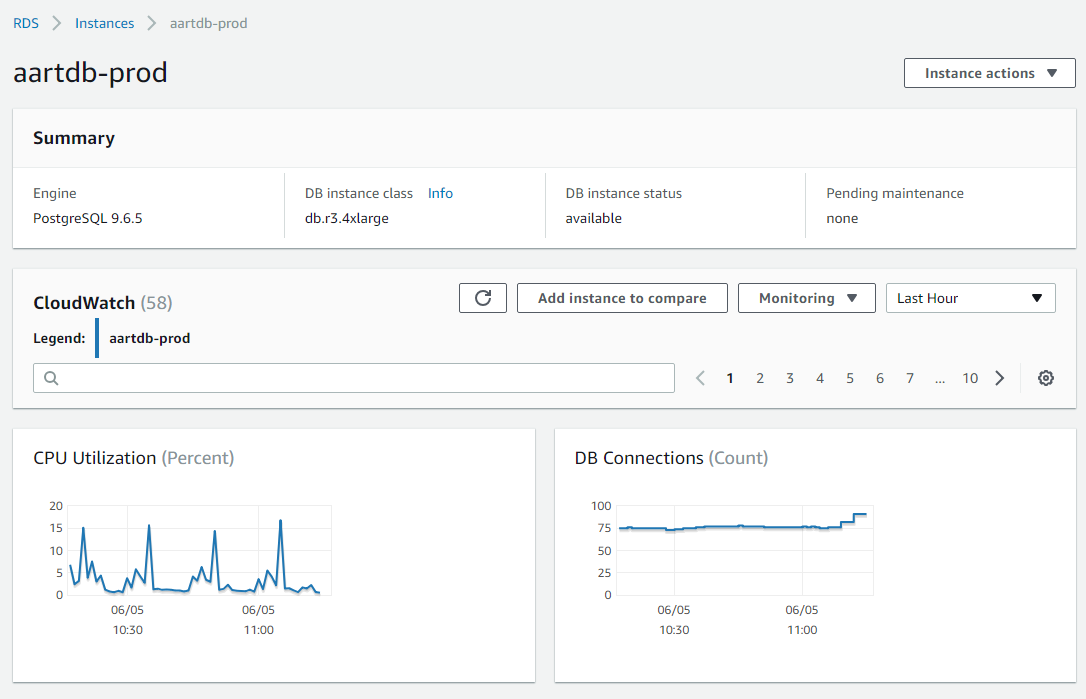
**select \* from pg\_stat\_activity**

If needed, scroll to the right to observe additional information. Please note that instances with “master” replication role have a replica, currently in Oregon.



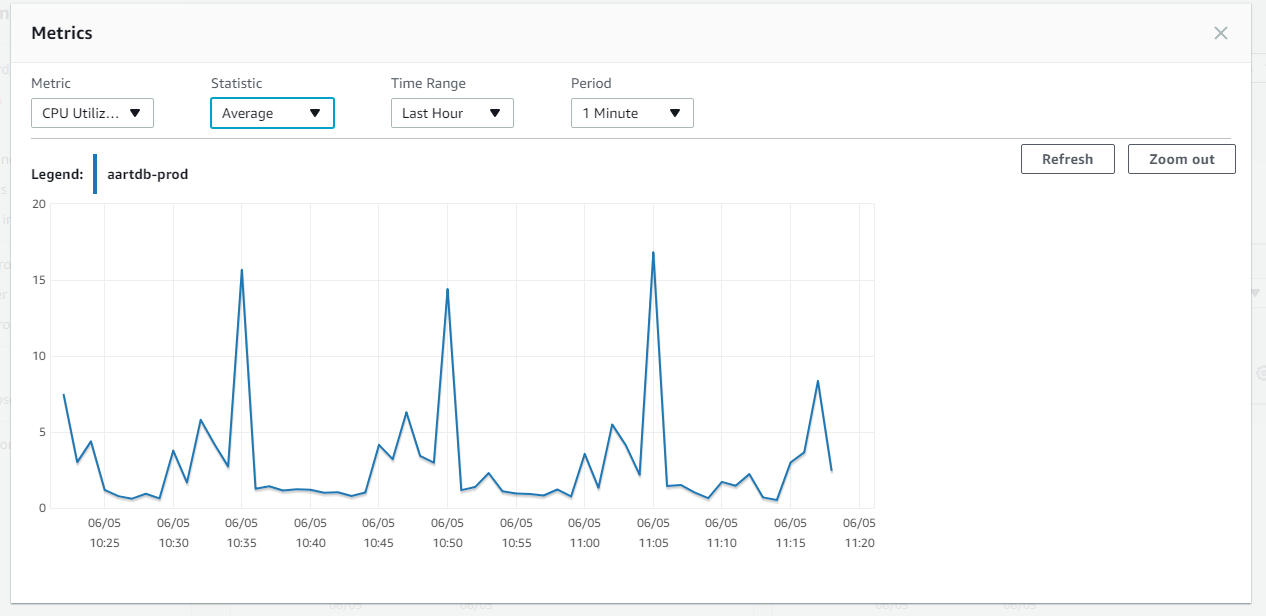
Click on any instance that you want to check, for example aartdb-prod





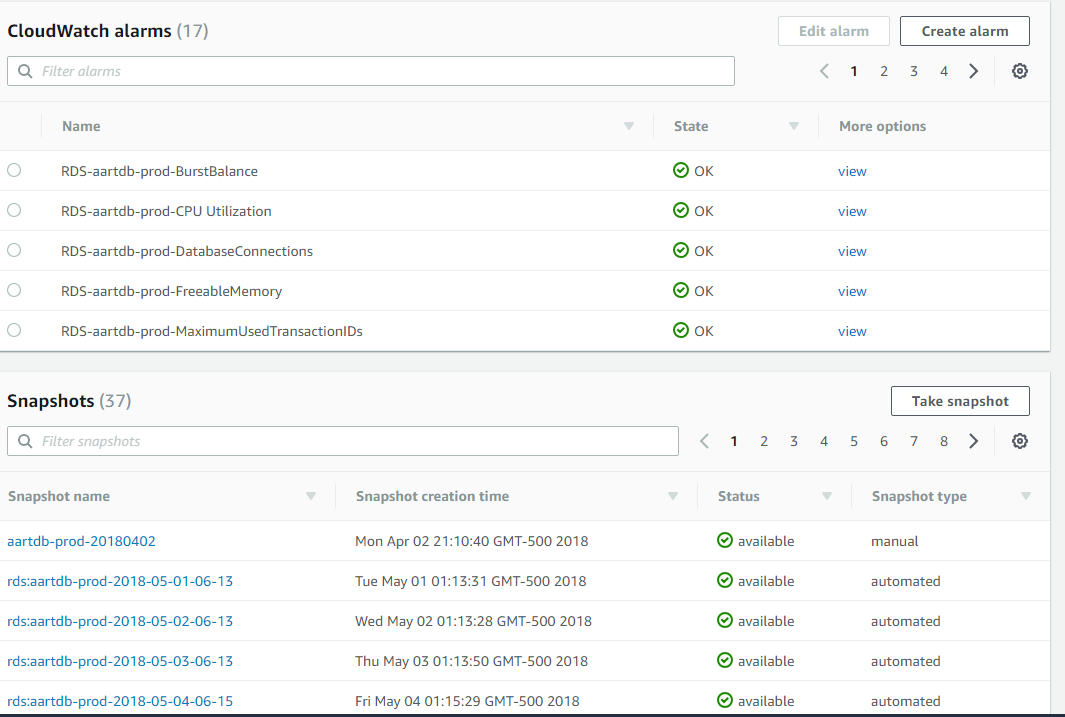
You can see performance metrics, such as, CPU Utilization, Database Connections, and if scroll down you can see other metrics. Also, you can click on the numbers under CloudWatch to open more metrics graphs.

If you click on a metric window and then change parameters, such as Statistic (average, min, max, sum), time range (from last hour to last two weeks), and period (from 1minute to 1 day).

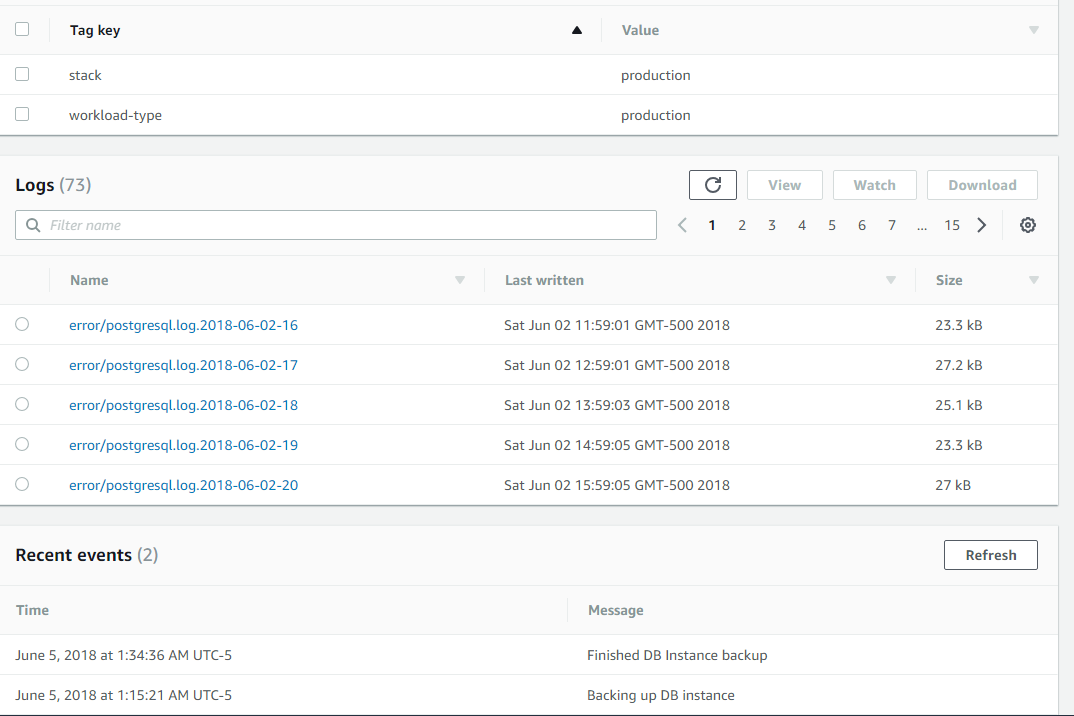


Click on the “X” to close the window.

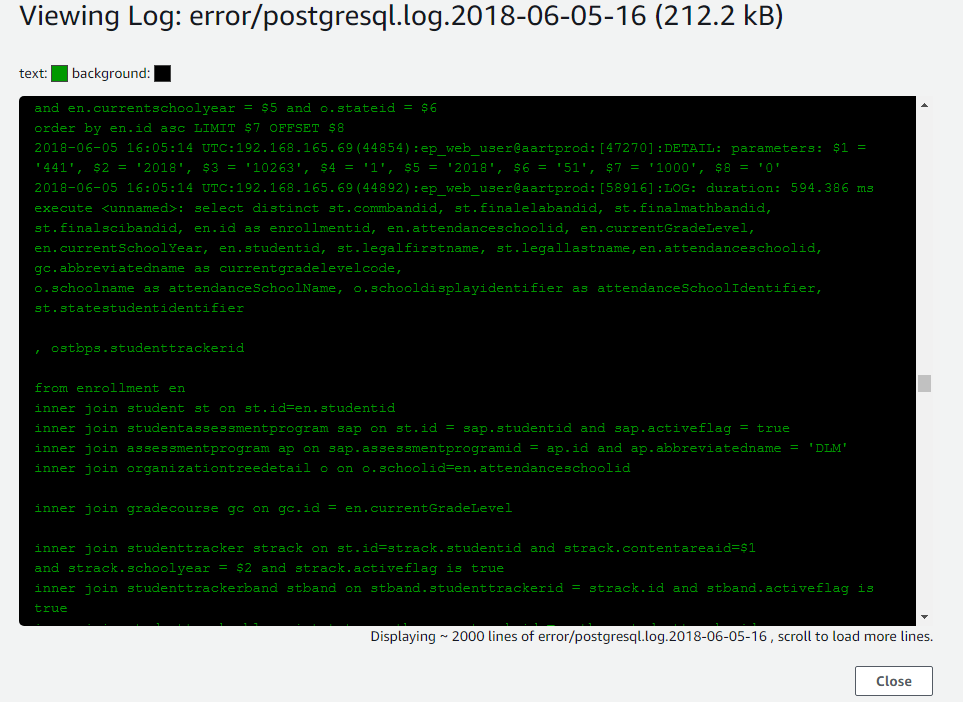
When you further scroll down you can see a list of CloudWatch alarms pertaining to the istance, and snapshots of the instance. Click on a highest number in the Snapshots list to see the latest snapshot. Verify that snapshots are being taken automatically as needed. There should be 35 automated snapshots for production instances.



When you further scroll down you can see logs and events.

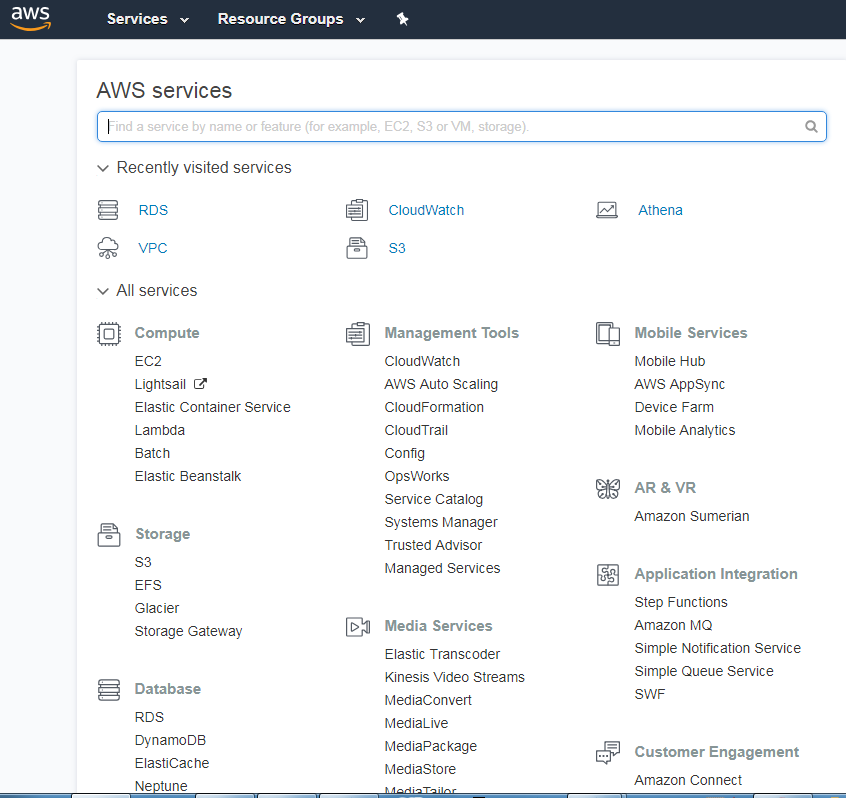


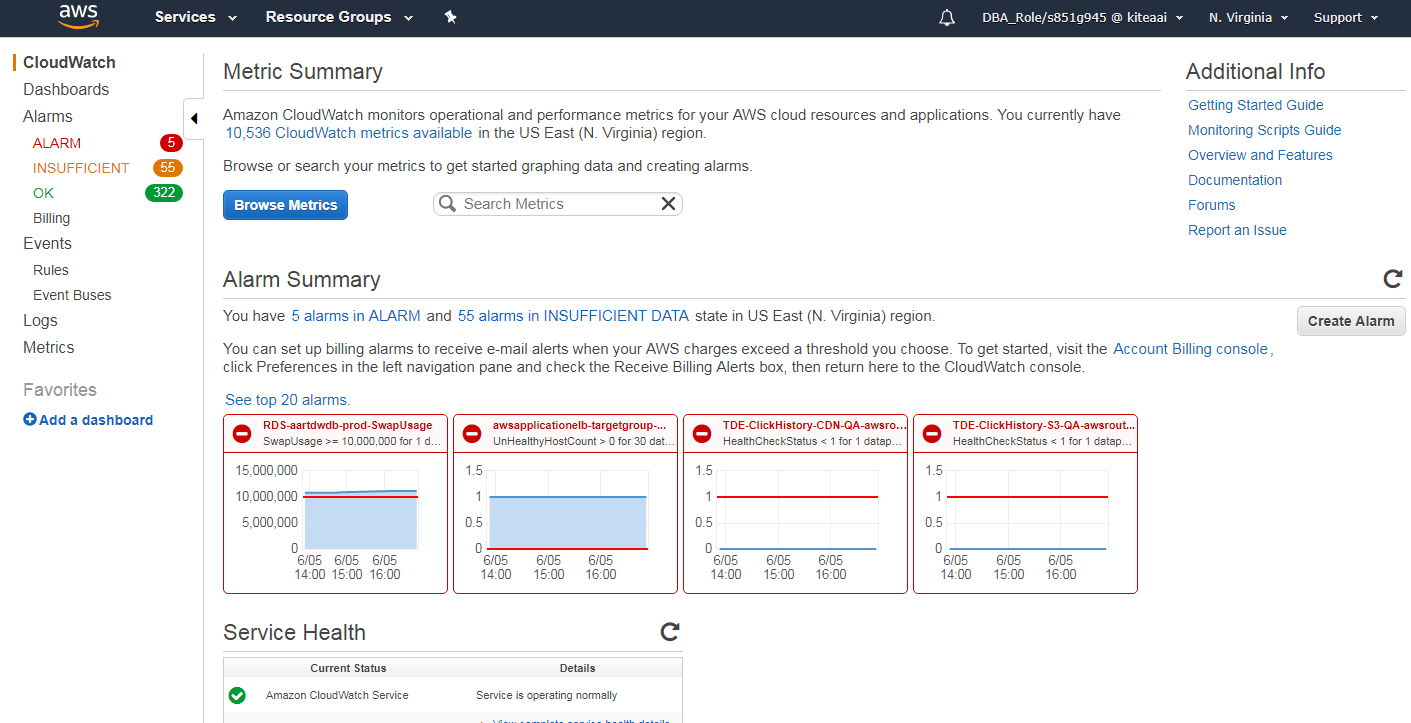
You may chose a specific log, or click on the highest number in the Logs list to see the latest logs. Click on any log of interest. When done, scroll down and click “Close”.



1. **CLOUDWATCH**

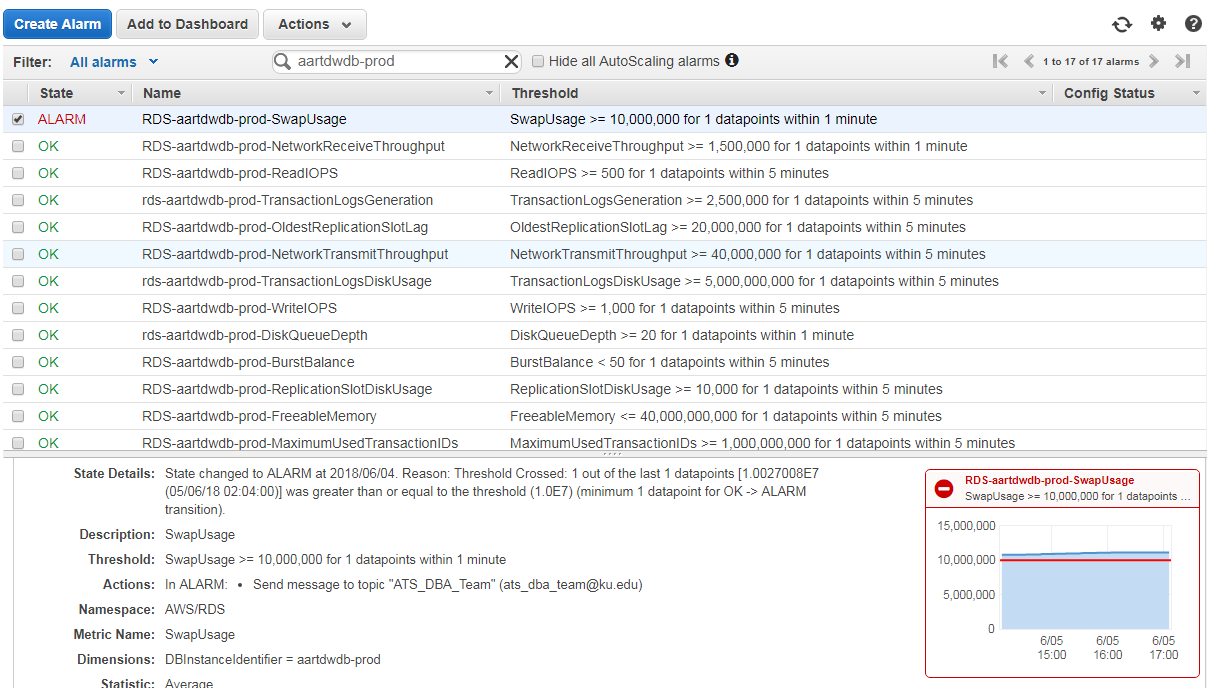
Login AWS and click on CloudWatch





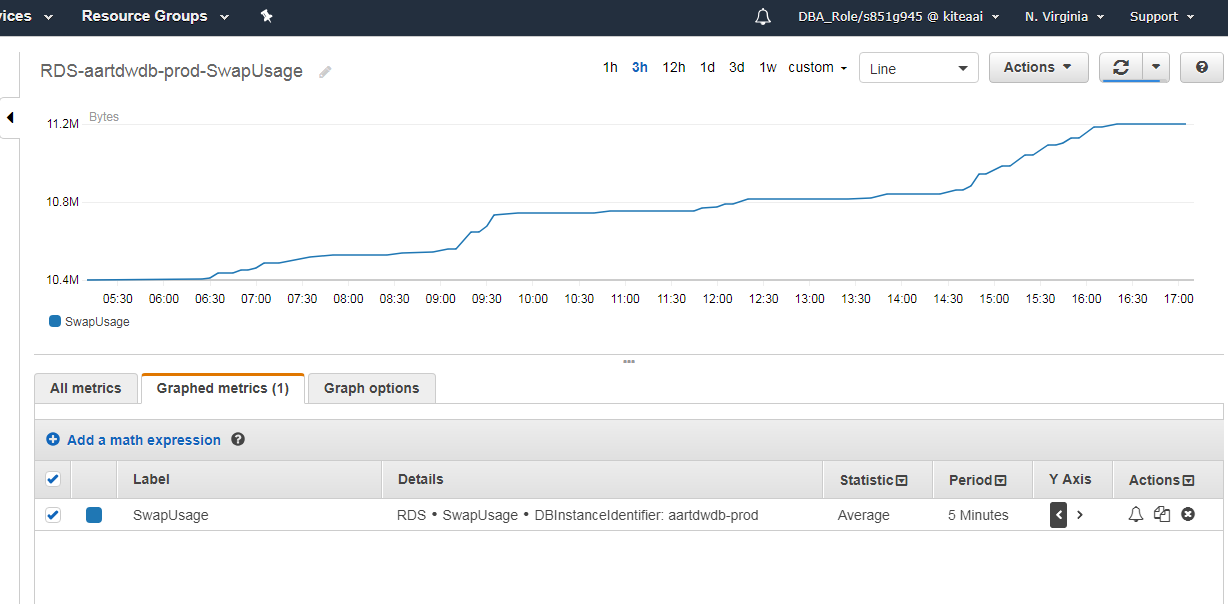
You may click on “ALARM” to see current alarms, which are not just for RDS, but for all.

You may click on “Alarms” and then narrow down the list according to the characters you enter in the search window. Here we see a list based on “aartdwdb-prod”.

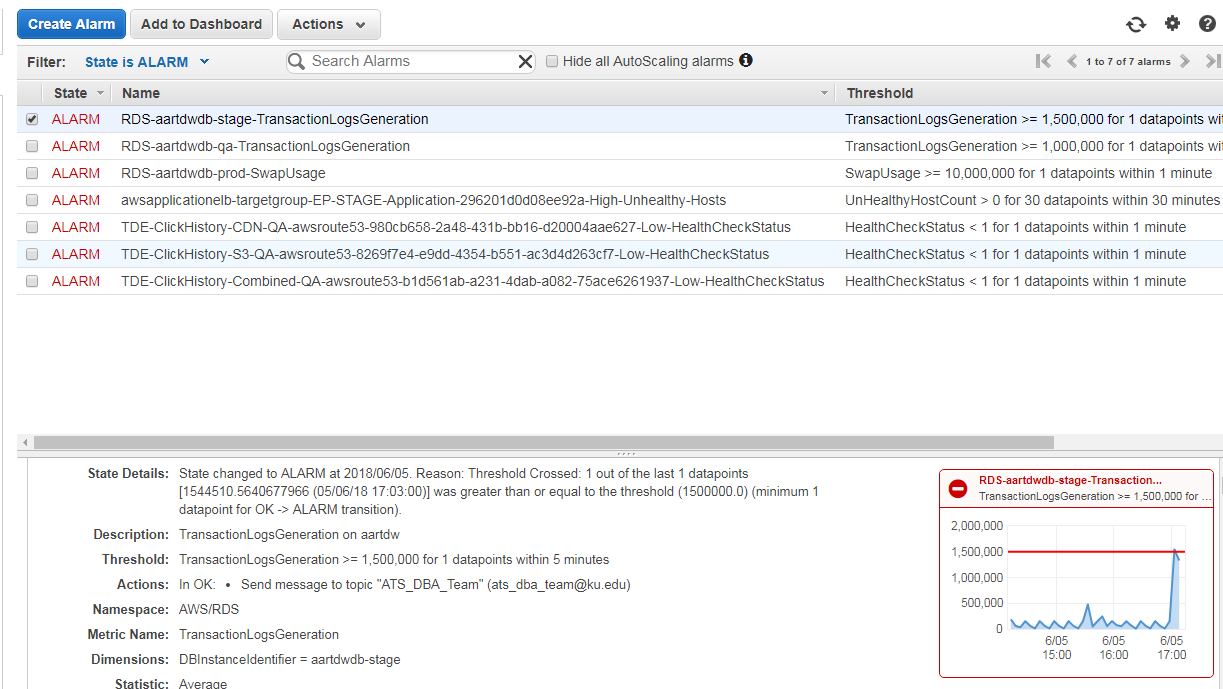


You can click on a specific metric to see more details. Here we have clicked on a metric that happens to have an active alarm, which is recommended to check.

You can also click on the graph to see more details and change parameters, as in the instance metrics described before.



Another example

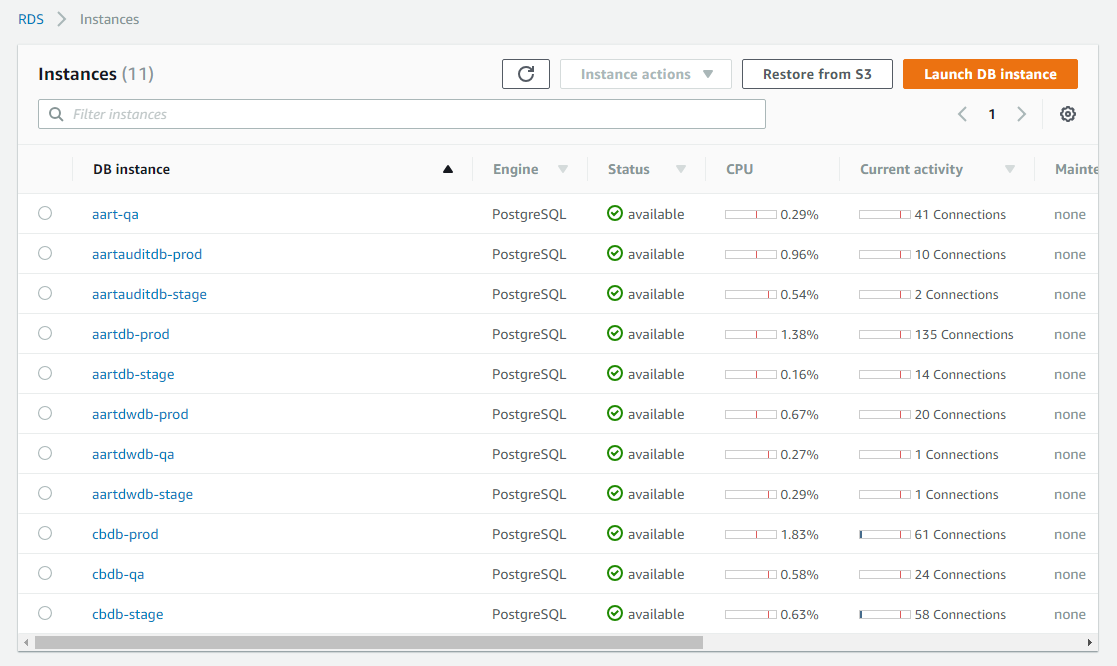


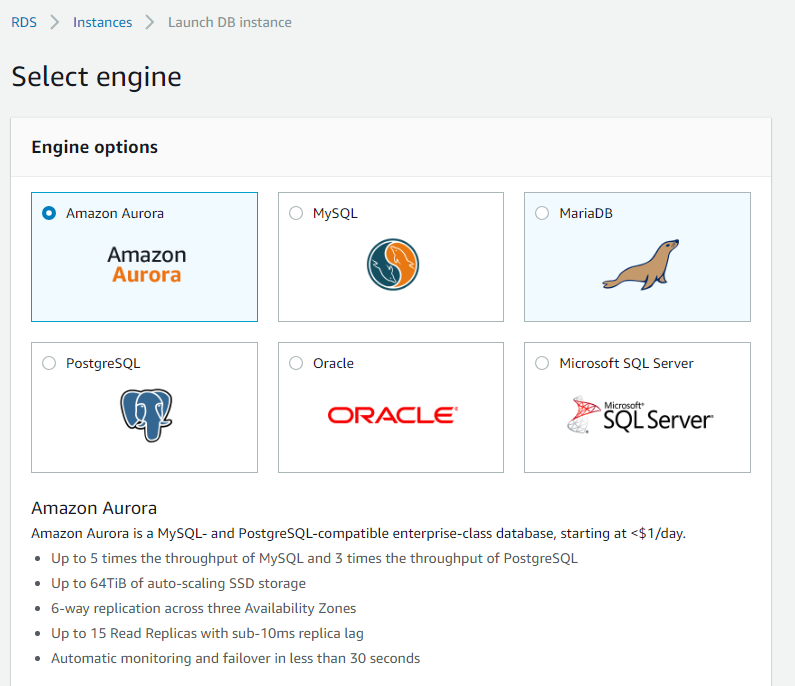
1. **CREATING COUDWATCH ALARMS**

We’ll cover it by performing an example.

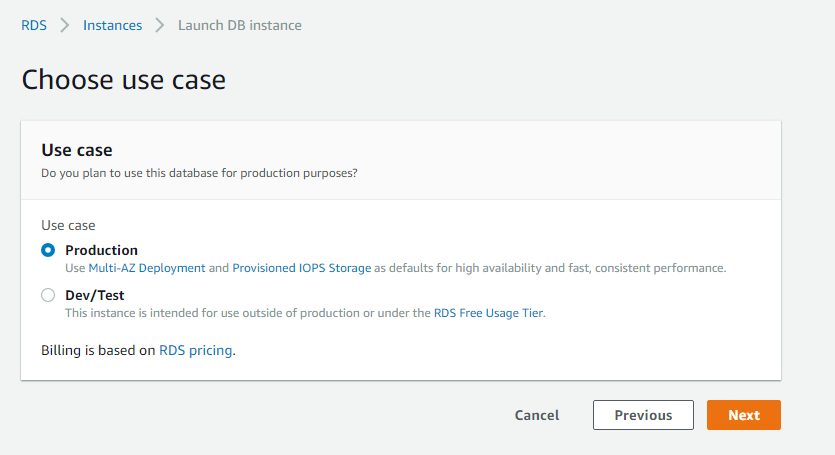
1. **MONITOR DATABSE UPGARDES/UPDATES**

In AWS click on RDS, then click on Launch DB Instance



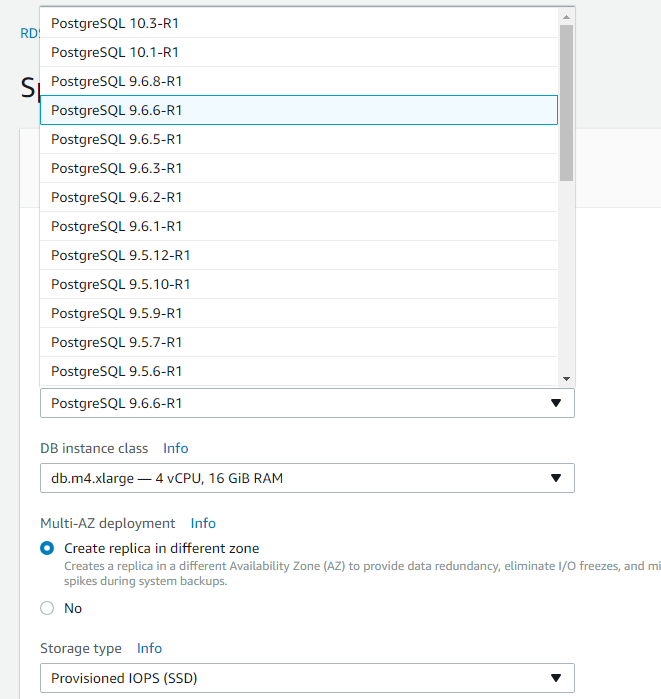


Choose PostgreSQL then click Next at the bottom of the page.



Select Production and click Next.

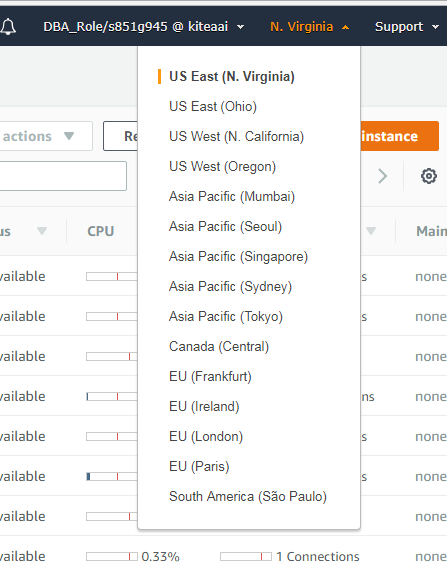
Drop down menu Db Engine Version offers the options for PostgreSQL version/revision.

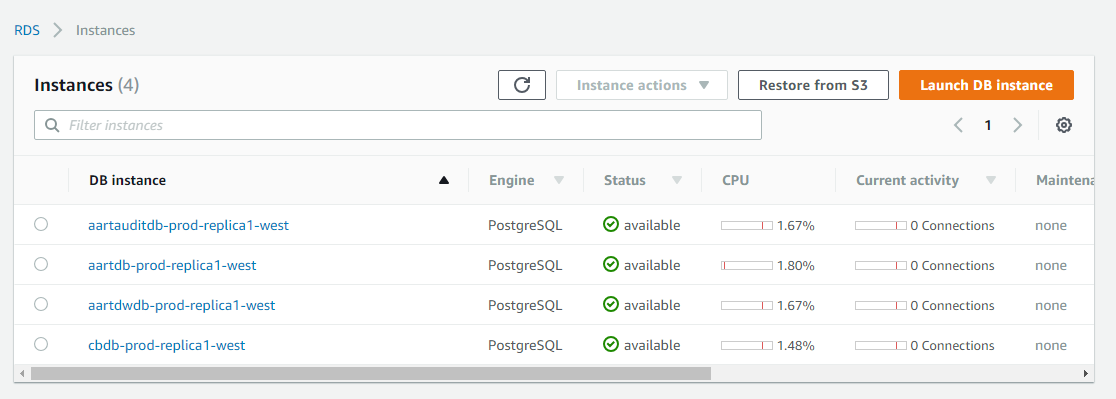


Cancel the create process. Do not save.

1. **MONITOR PRODUCTION REPLICAS**

Launch RDS. Select US West (Oregon) from the region drop down menu.





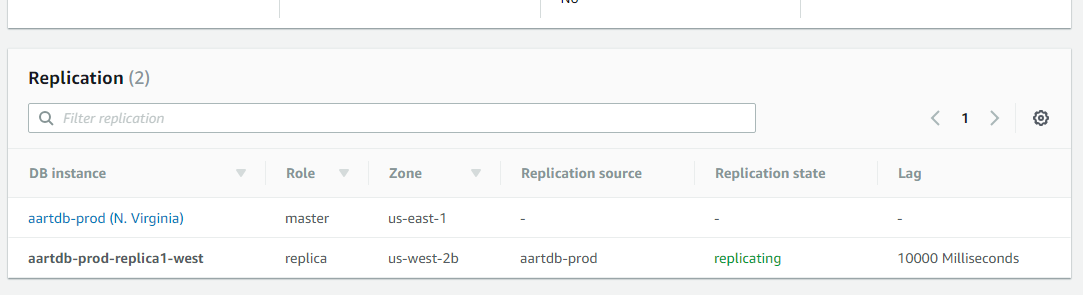
Replicas exist for production instances only.

Check activities and performance as for the RDS instances in East N. Virginia. There should be small CPU level used, and no connections. Any different behavior should be investigated.

Occasionally, restore an instance and check connectivity and database consistency (per the source), using PgAdmin or PSQL.

Open the Oregon region, got to RDS, and click on Instances.

Expand an instance and scroll down.



Here you can check that the process is replicating, and see how the replication lags behind the source. Make sure it is not excessive.

You may also login the replica database and execute

SELECT extract(epoch from now() - pg\_last\_xact\_replay\_timestamp()) AS slave\_lag

This will give you the lag time.

1. **CHECK DATABSE BLOAT**

You can run the bloating query with PgAdmin or PSQL.

In (aart) EP databases use “public” for schema name, while (lportal) CB uses “cb”

The query is

SELECT current\_database(), schemaname, tblname, bs\*tblpages AS real\_size, (tblpages-est\_tblpages)\*bs AS extra\_size, CASE WHEN tblpages - est\_tblpages > 0 THEN 100 \* (tblpages - est\_tblpages)/tblpages::float ELSE 0 END AS extra\_ratio, fillfactor, (tblpages-est\_tblpages\_ff)\*bs AS bloat\_size, CASE WHEN tblpages - est\_tblpages\_ff > 0 THEN 100 \* (tblpages - est\_tblpages\_ff)/tblpages::float ELSE 0 END AS bloat\_ratio, is\_na -- , (pst).free\_percent + (pst).dead\_tuple\_percent AS real\_fragFROM ( SELECT ceil( reltuples / ( (bs-page\_hdr)/tpl\_size ) ) + ceil( toasttuples / 4 ) AS est\_tblpages, ceil( reltuples / ( (bs-page\_hdr)\*fillfactor/(tpl\_size\*100) ) ) + ceil( toasttuples / 4 ) AS est\_tblpages\_ff, tblpages, fillfactor, bs, tblid, schemaname, tblname, heappages, toastpages, is\_na -- , stattuple.pgstattuple(tblid) AS pst FROM ( SELECT ( 4 + tpl\_hdr\_size + tpl\_data\_size + (2\*ma) - CASE WHEN tpl\_hdr\_size%ma = 0 THEN ma ELSE tpl\_hdr\_size%ma END - CASE WHEN ceil(tpl\_data\_size)::int%ma = 0 THEN ma ELSE ceil(tpl\_data\_size)::int%ma END ) AS tpl\_size, bs - page\_hdr AS size\_per\_block, (heappages + toastpages) AS tblpages, heappages, toastpages, reltuples, toasttuples, bs, page\_hdr, tblid, schemaname, tblname, fillfactor, is\_na FROM ( SELECT tbl.oid AS tblid, ns.nspname AS schemaname, tbl.relname AS tblname, tbl.reltuples, tbl.relpages AS heappages, coalesce(toast.relpages, 0) AS toastpages, coalesce(toast.reltuples, 0) AS toasttuples, coalesce(substring( array\_to\_string(tbl.reloptions, ' ') FROM '%fillfactor=#"\_\_#"%' FOR '#')::smallint, 100) AS fillfactor, current\_setting('block\_size')::numeric AS bs, CASE WHEN version()~'mingw32' OR version()~'64-bit|x86\_64|ppc64|ia64|amd64' THEN 8 ELSE 4 END AS ma, 24 AS page\_hdr, 23 + CASE WHEN MAX(coalesce(null\_frac,0)) > 0 THEN ( 7 + count(\*) ) / 8 ELSE 0::int END + CASE WHEN tbl.relhasoids THEN 4 ELSE 0 END AS tpl\_hdr\_size, sum( (1-coalesce(s.null\_frac, 0)) \* coalesce(s.avg\_width, 1024) ) AS tpl\_data\_size, bool\_or(att.atttypid = 'pg\_catalog.name'::regtype) AS is\_na FROM pg\_attribute AS att left JOIN pg\_class AS tbl ON att.attrelid = tbl.oid left JOIN pg\_namespace AS ns ON ns.oid = tbl.relnamespace left JOIN pg\_stats AS s ON s.schemaname=ns.nspname AND s.tablename = tbl.relname AND s.inherited=false AND s.attname=att.attname LEFT JOIN pg\_class AS toast ON tbl.reltoastrelid = toast.oid WHERE att.attnum > 0 AND NOT att.attisdropped AND tbl.relkind = 'r' GROUP BY 1,2,3,4,5,6,7,8,9,10, tbl.relhasoids ORDER BY tblname--2,3 ) AS s ) AS s2 ) AS s3 where schemaname='public' order by real\_size

Enter the results in a spreadsheet and save records for specific dates when you run it, to have a record for comparison.

Check the ratio between the extra size and real size.