PostgreSQL Cloud Performance

Oskari Saarenmaa

PGConf APAC 2018 - Singapore

Agenda

1. Introduction
2. Why cloud – why not cloud?
3. Running PostgreSQL in a cloud
4. Performance considerations
5. Benchmark methodology and setup
6. Results
7. Q & A

Speaker

- CEO, co-founder @ Aiven, a cloud DBaaS company
- Previously: database consultant, software architect
- PostgreSQL user since 1999 (rel 6.4)
    - Contributed bug fixes and features to core
    - Worked on extensions and tooling in the PG ecosystem

@OskariSaarenmaa

Aiven

- Independent Database as a Service provider
- Based in Helsinki and Boston
  - APAC presence later in 2018
- 7 DB products now available in 70 regions
  - including 23 in APAC
- First to offer PostgreSQL 10 as a service!

Why cloud?

It's someone else's computer:

- They buy the hardware and cover capital costs
- They install new and replace broken hardware
- Resources available on-demand, no waiting for procurement

When using DBaaS:

- They install, maintain, and backup the software
- You are provided integrated monitoring and metrics
- Backups, replication and other tooling is up in minutes

Why not cloud?

As it is “someone else’s computer”, you’ll have:

- Less control over details
- Operational concerns
  - Will there be someone to fix issues in case of problems?
- Compliance concerns
  - Someone else has physical access to the data
- Potentially higher operational costs
  - When only looking at the infrastructure costs
  - Assuming you can plan your hardware use well in advance

Roll your own or use a DBaaS?

Operate your own databases:

+ Lift & shift an existing production on-prem DB to cloud
+ Superuser access
+ All custom extensions

- Manage backups, plan for scaling
- Slower provisioning
- No built-in monitoring

Use a DB as a service provider:

+ Automatic provisioning and maintenance of systems
+ New clusters available in minutes
+ Integrated monitoring systems
+ Point-in-time recovery built in

- Limited PL/language support
- No superuser access (usually)

Performance considerations

Hardware: CPU, storage IO, network

Software: tuning for my workload?

Network: plan to access the database from the same network, typically fast access to data from the same region and availability zone – some differences in the top end

CPU: much the same across the clouds

Storage: not at all the same across the clouds

Storage systems

Latency to access storage systems in most scenarios:

CPU caches **<** RAM **<** Local disk **<** Network disk

Local disks (“instance storage”) in the cloud only available for the lifetime of a single VM instance – data durability must be guaranteed across node faults using other means:

- Replication
- Incremental backup of data as it’s written

Turns out we can do both reliably with PostgreSQL

Block storage system options

Network disks

- \+ Persistent past node lifetime
- \+ Almost infinitely scalable

- \- Really slow, or
- \- Quite expensive (PrIOPS)
- \- Compete with others over limited IO bandwidth
- \- Not free of faults

Local disks

- \+ Fast
- \+ Potentially really fast
- \+ Cheap

- \- Available in limited sizes
  - \- (or not at all)
- \- Ephemeral
  - \- Node shuts down: data is gone

Important considerations for benchmarking

1. Number of different things affect performance
2. None of the comparisons ever match your production workload
3. Repeat the benchmark process several times to ensure the numbers are stable

The presented benchmarks measure PostgreSQL performance under one specific benchmarking scenario using virtual machines provided by different vendors with as similar specifications as possible.

Methodology

1. Provision a benchmark host in the target cloud
    a. PostgreSQL 10.3
    b. Linux 4.15.9
2. Provision a DB instance from a DBaaS provider
    a. 16 GB RAM instances
    b. 64 GB RAM instances
3. Initialize with a large dataset
    a. Roughly 2x memory size
    b. Data encrypted on disk with SSL required for clients
    c. WAL archiving enabled
4. Run PGBench with a varying number of clients for 1 hour

Benchmarks: network disks

- 5 Infrastructure clouds in 2 APAC regions
- 2 Database sizes
- PostgreSQL 10
- PGBench

16 GB RAM instances, with network disks

AWS	GCP	Azure	DigitalOcean	UpCloud
m5.xlarge	n1-standard-4	Standard D3v2	s-6vcpu-16gb	6xCPU-16GB
4 vCPU	4 vCPU	4 vCPU	6 vCPU	6 vCPU
16 GB RAM	15 GB RAM	14 GB RAM	16 GB RAM	16 GB RAM
350 GB EBS	350 GB PD-SSD	350 GB P20	350 GB block store	350 GB MAXIOPS

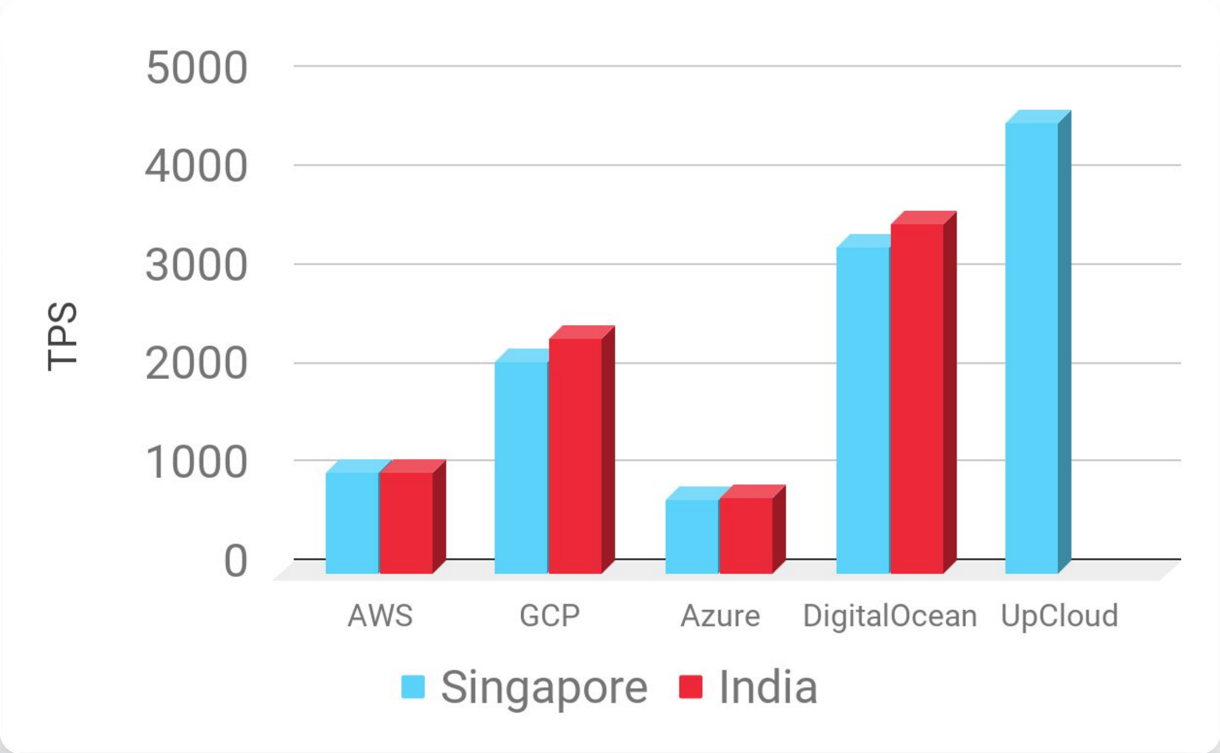
postgresql.conf settings

```
work_mem = 12MB
shared_buffers = 3GB
max_wal_size = 16GB
wal_level = replica
```

pgbench commands

```
pgbench --initialize --scale=2000
pgbench --jobs=4 --client=16 \
        --time=3600
```

16 GB RAM instances, with network disks



64 GB RAM instances, with network disks

AWS	GCP	Azure	DigitalOcean	UpCloud
m5.4xlarge	n1-standard-16	Standard D5v2	s-16vcpu-64gb	16xCPU-64GB
16 vCPU	16 vCPU	16 vCPU	16 vCPU	16 vCPU
64 GB RAM	60 GB RAM	56 GB RAM	64 GB RAM	64 GB RAM
1 TB EBS	1 TB PD-SSD	1 TB P30	1 TB block store	1 TB MAXIOPS

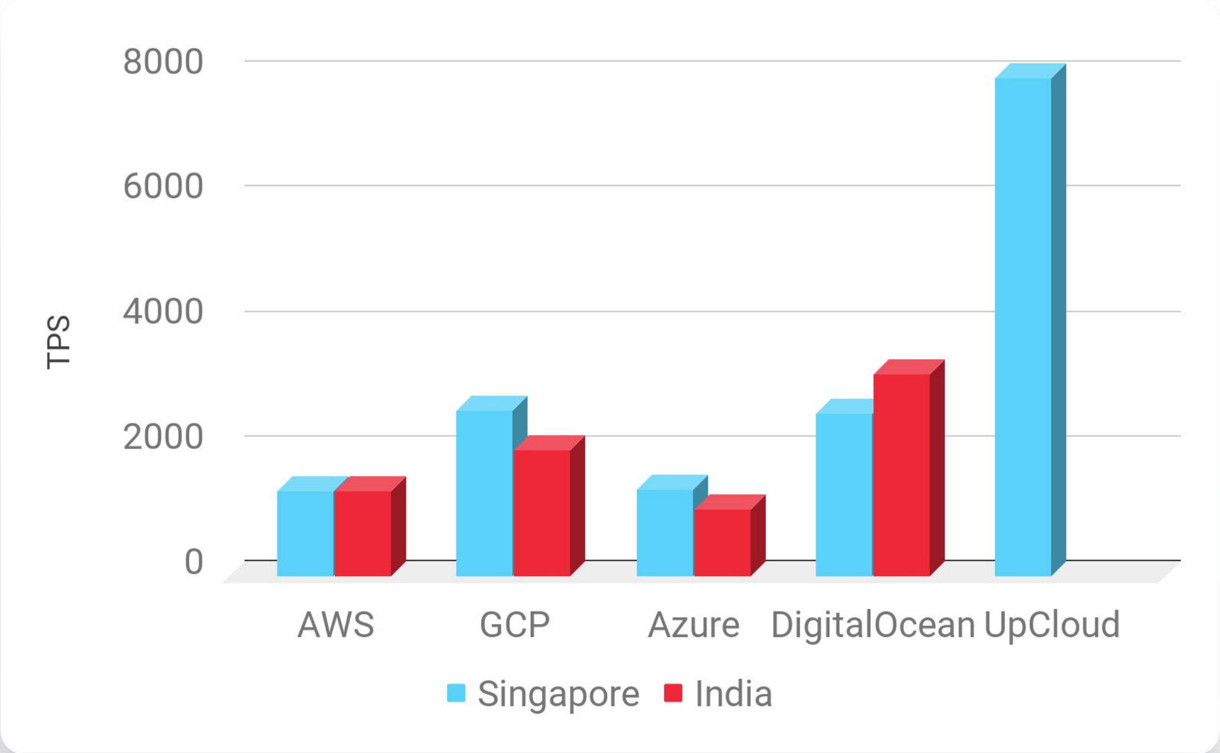
postgresql.conf settings

```
work_mem = 32MB
shared_buffers = 12GB
max_wal_size = 50GB
wal_level = replica
```

pgbench commands

```
pgbench --initialize --scale=8000
pgbench --jobs=4 --client=64 \
        --time=3600
```

64 GB RAM instances, with network disks



Benchmarks: local vs network disks

- Google Cloud: Up to 8 local NVMe disks attached to any instance type
- AWS: Fixed NVMe disks with i3.* instance types
- Other clouds: not applicable

16 GB RAM instances with local disks

AWS

i3.large

2 vCPU

15 GB RAM

350 GB NVMe
(max 475 GB)

GCP

n1-standard-4

4 vCPU

15 GB RAM

350 GB NVMe
(max 3 TB)

Azure

DigitalOcean

UpCloud

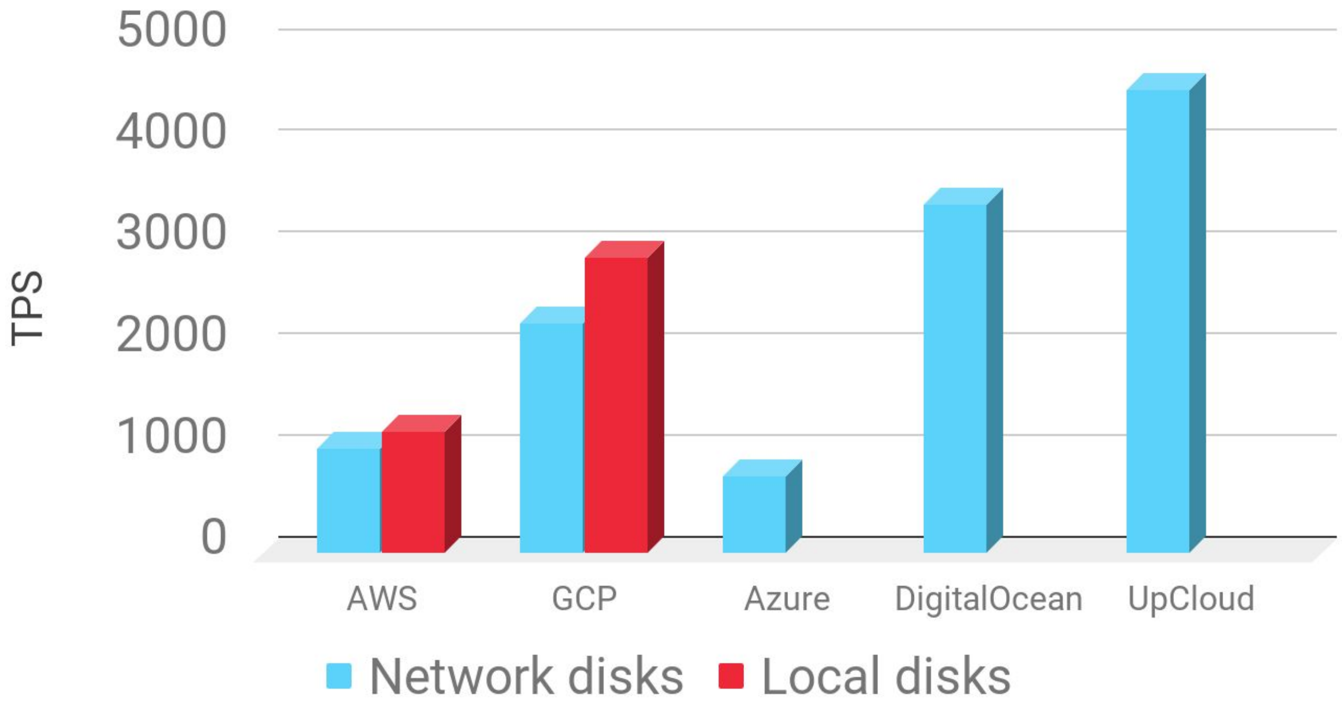
postgresql.conf settings

```
work_mem = 12MB
shared_buffers = 3GB
max_wal_size = 16GB
wal_level = replica
```

pgbench commands

```
pgbench --initialize --scale=2000
pgbench --jobs=4 --client=16 \
        --time=3600
```

16 GB RAM instances, network vs local disks



64 GB RAM instances, with local disks

AWS	GCP	Azure	DigitalOcean	UpCloud
i3.2xlarge 8 vCPU 61 GB RAM 1 TB NVMe (max 1900 GB)	n1-standard-16 16 vCPU 60 GB RAM 1 TB NVMe (max 3 TB)	✖	✖	✖

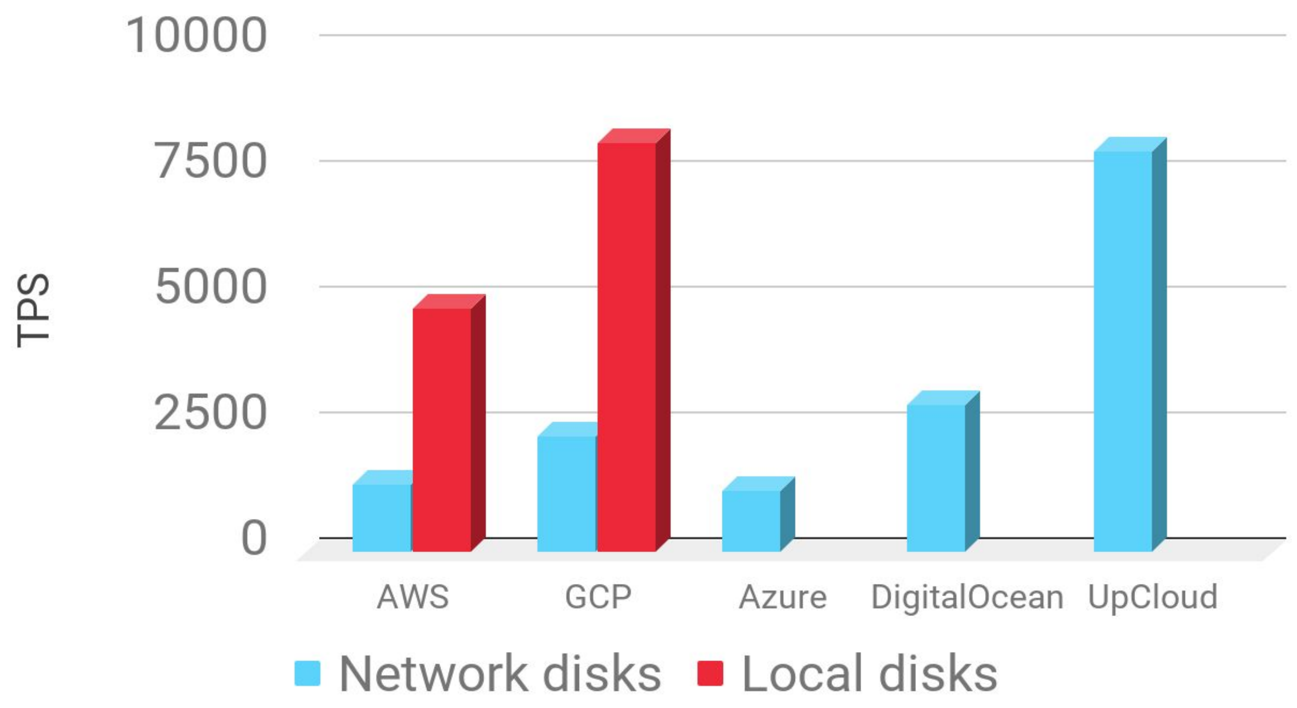
postgresql.conf settings

```
work_mem = 32MB
shared_buffers = 12GB
max_wal_size = 50GB
wal_level = replica
```

pgbench commands

```
pgbench --initialize --scale=8000
pgbench --jobs=4 --client=64 \
        --time=3600
```

64 GB RAM instances, network vs local disks



DBaaS comparison in AWS

- Aiven PostgreSQL in AWS (10.3)
- Amazon RDS for PostgreSQL (10.1)
- Amazon Aurora with PostgreSQL (10.1)

- AWS ap-southeast-1: Singapore

AWS DBaaS 16 GB RAM services

Aiven	RDS	Aurora
startup-16	db.m4.xlarge	db.r4.large
2 vCPU	4 vCPU	2 vCPU
15 GB RAM	16 GB RAM	15 GB RAM
350 GB NVMe	350 GB EBS	Transparently scalable storage

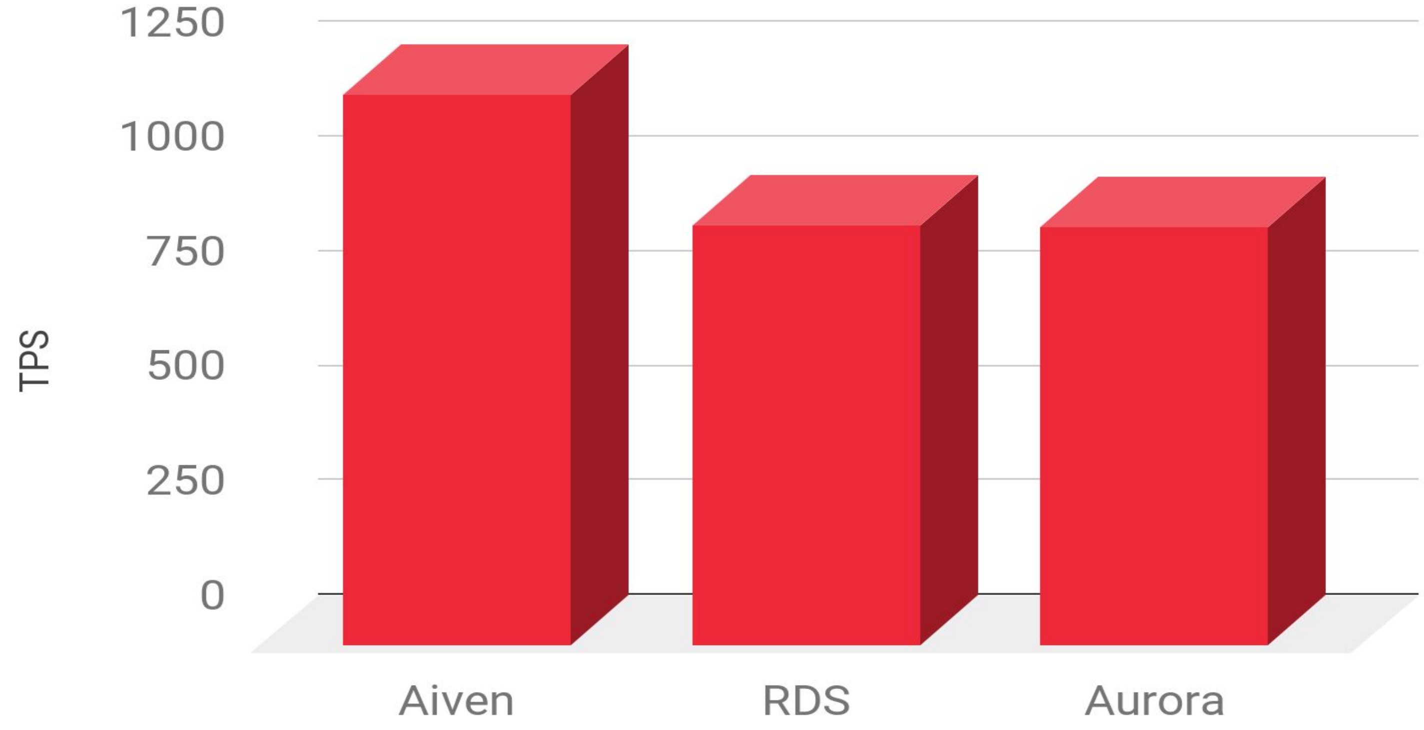
postgresql.conf settings

```
work_mem = 12MB
shared_buffers = 3GB
max_wal_size = 16GB
wal_level = replica
```

pgbench commands

```
pgbench --initialize --scale=2000
pgbench --jobs=4 --client=16 \
        --time=3600
```

AWS DBaaS 16 GB RAM services



AWS DBaaS 64 GB RAM services

Aiven	RDS	Aurora
startup-64	db.m4.4xlarge	db.r4.2xlarge
8 vCPU	16 vCPU	8 vCPU
61 GB RAM	60 GB RAM	61 GB RAM
1 TB NVMe	1 TB EBS	Transparently scalable storage

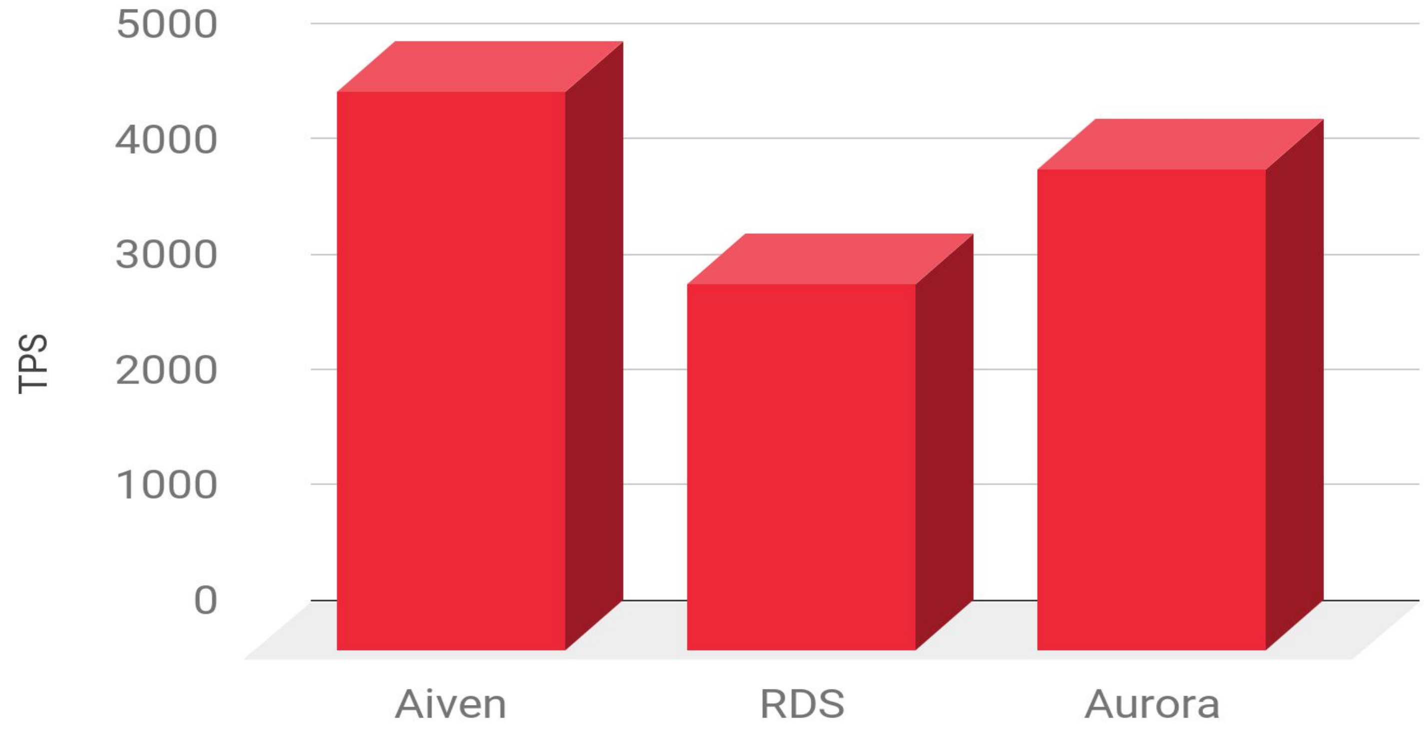
postgresql.conf settings

```
work_mem = 32MB
shared_buffers = 12GB
max_wal_size = 50GB
wal_level = replica
```

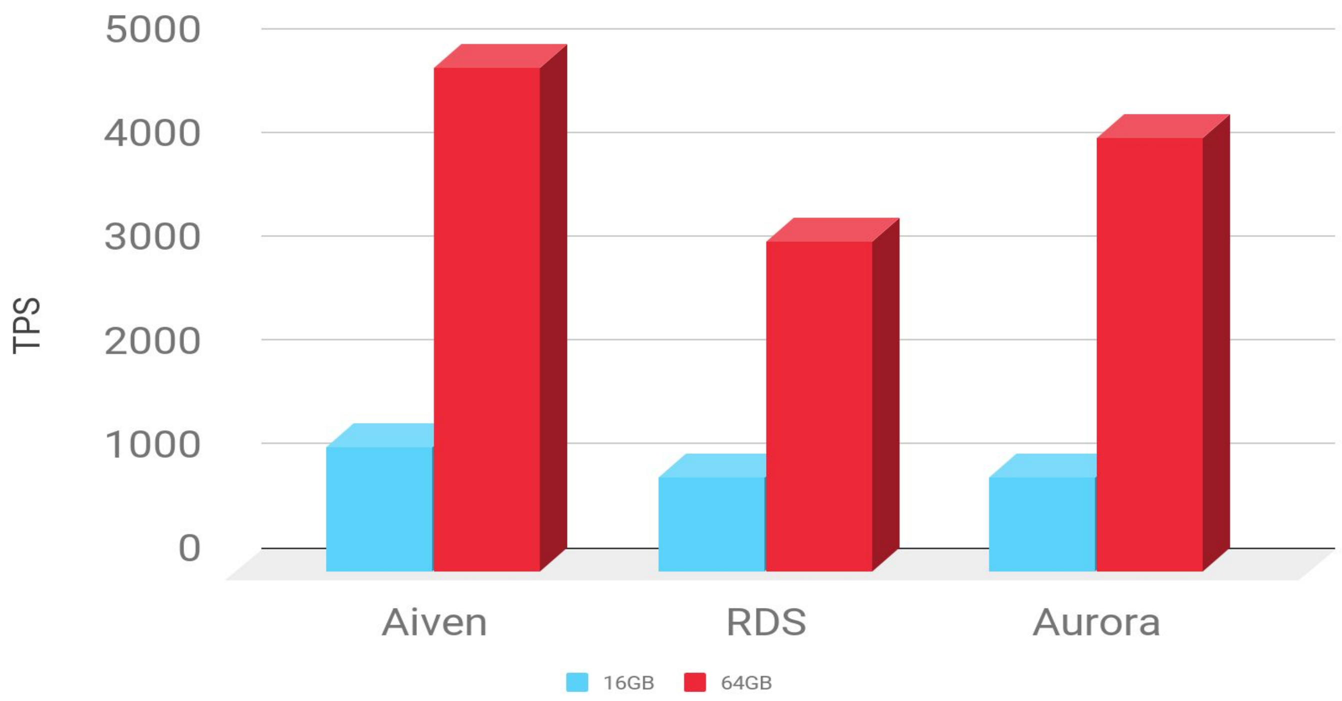
pgbench commands

```
pgbench --initialize --scale=8000
pgbench --jobs=4 --client=64 \
        --time=3600
```

AWS DBaaS 64 GB RAM services



AWS DBaaS 16 vs 64 GB RAM services



Questions?

Cool t-shirts for the first ones to ask a question!

P.S. try out Aiven and get a cool t-shirt even if you didn’t ask a question

aiven

Thanks!

https://aiven.io

@aiven_io

@OskariSaarenmaa