PERCONA

Databases run better with Percona

PERCONA

Running your databases on Kubernetes in the cloud:

What is the overhead?

Fernando Laudares Camargos

Senior Architect at Percona

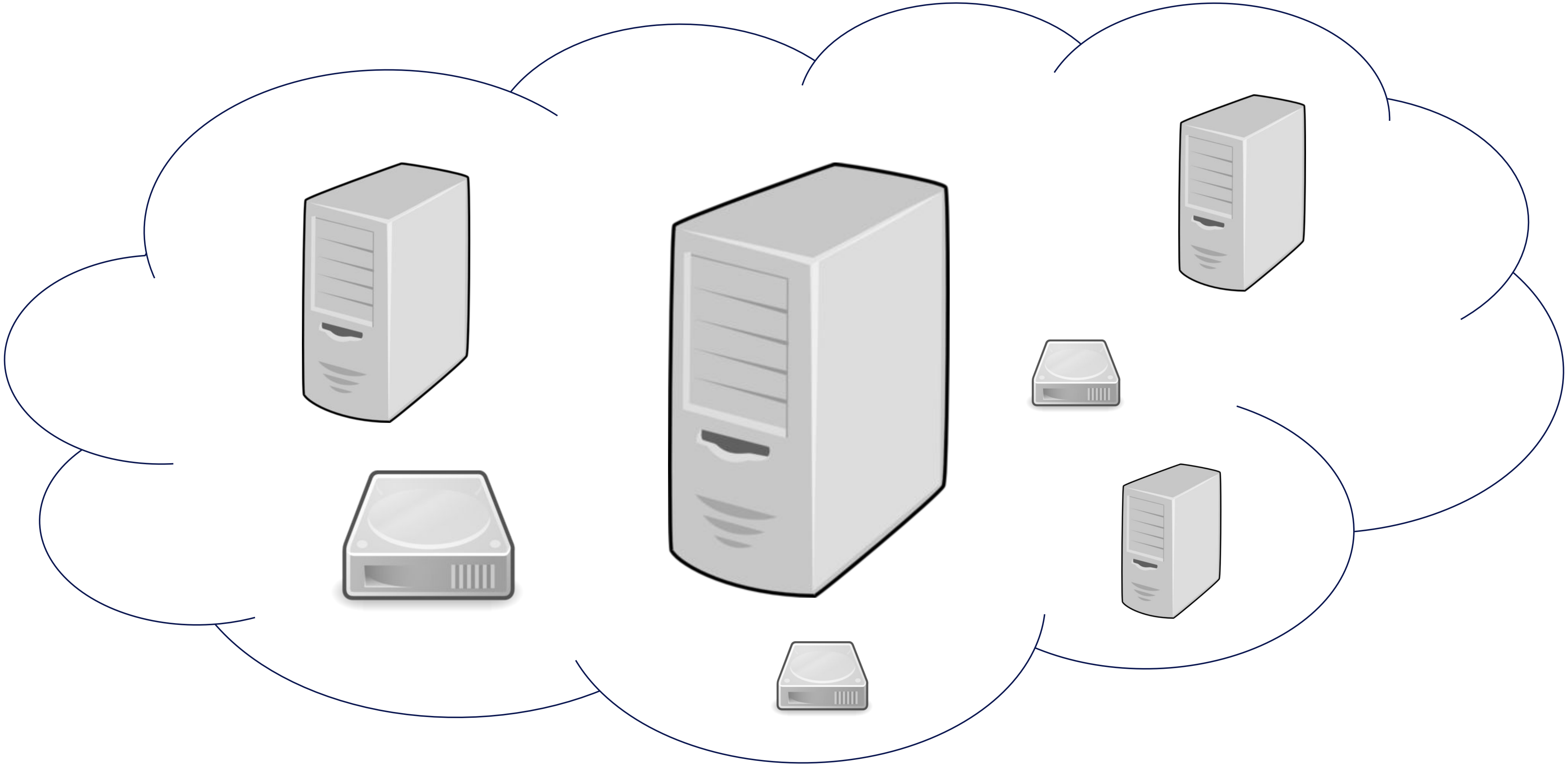
preFOSDEM
MySQL
Belgian Days 2026

Intro

- Joined Percona in 2013
- MySQL, PostgreSQL, K8s

Nando

IaaS



https://www.i2clipart.com/clipart-server-d59f
https://www.i2clipart.com/clipart-tango-drive-hard-disk-e1a0

IaaS databases (self-managed cloud databases)

A database hosted on a cloud providers':

Compute instance

vs

Kubernetes engine

If it's the same underlying hardware, does it perform in the same way?

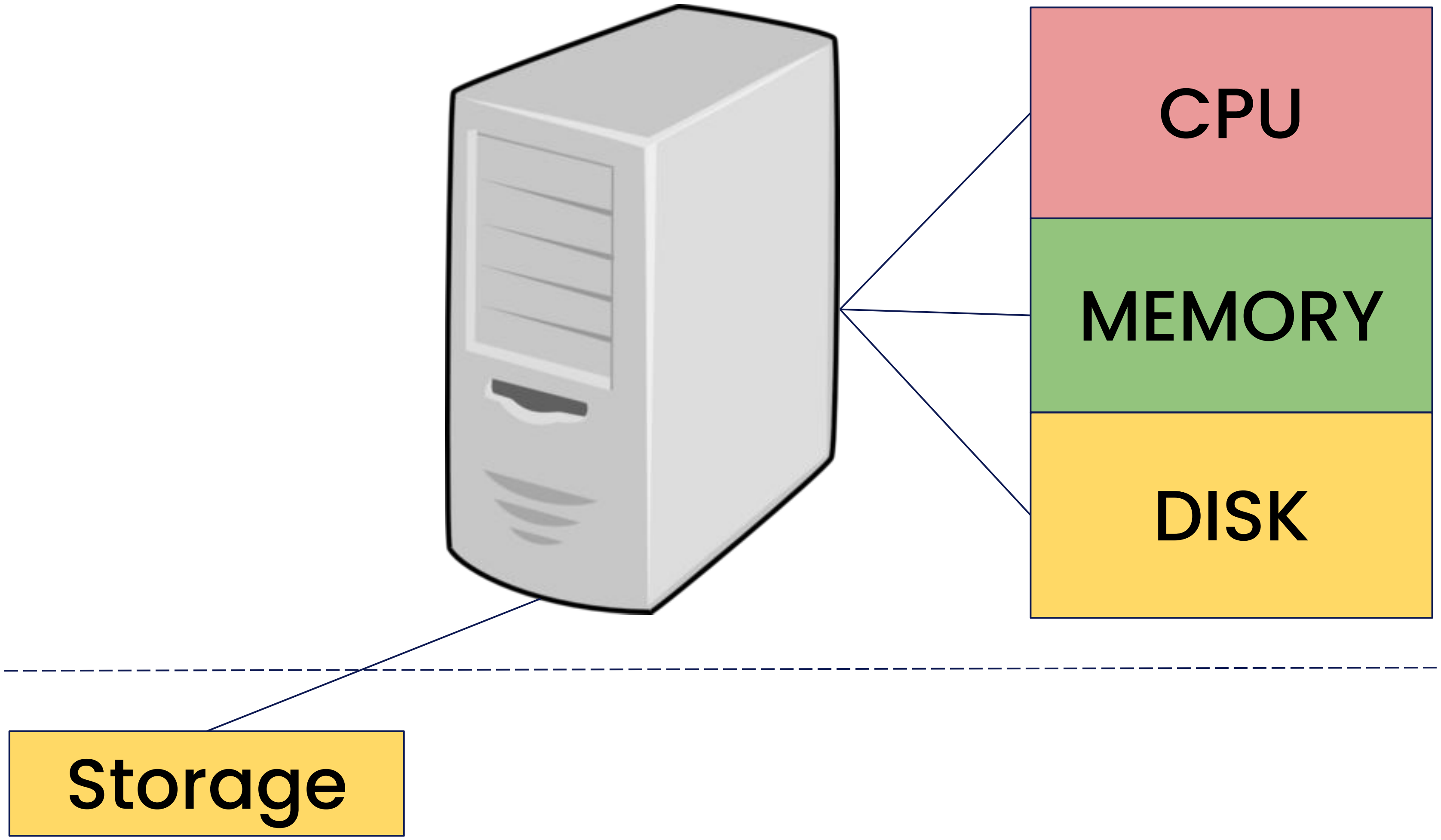
Kubernetes

Kubernetes

Virtualization + automation + orchestration

convenience

Resources



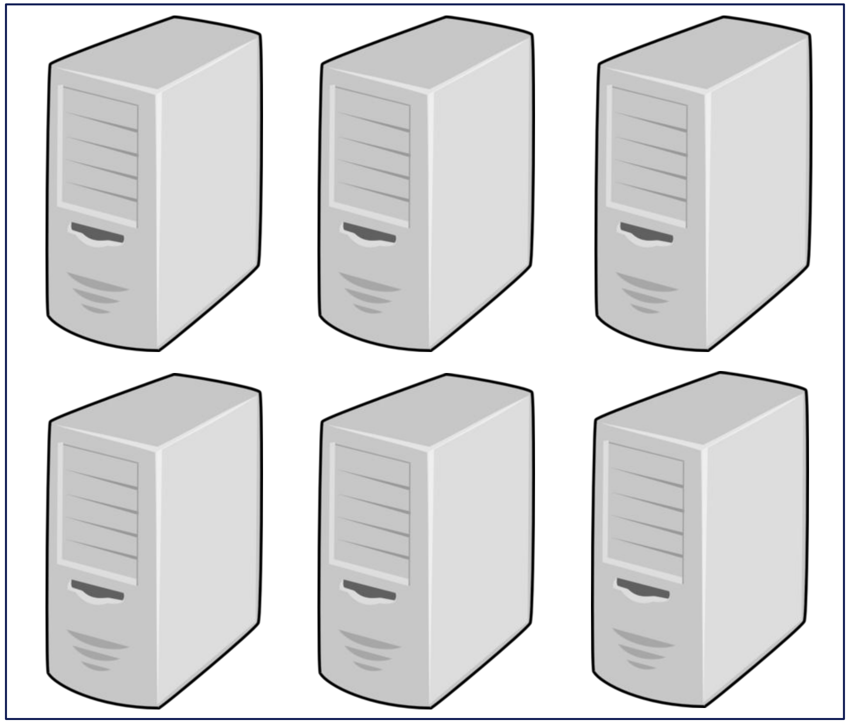
CPU

MEMORY

DISK

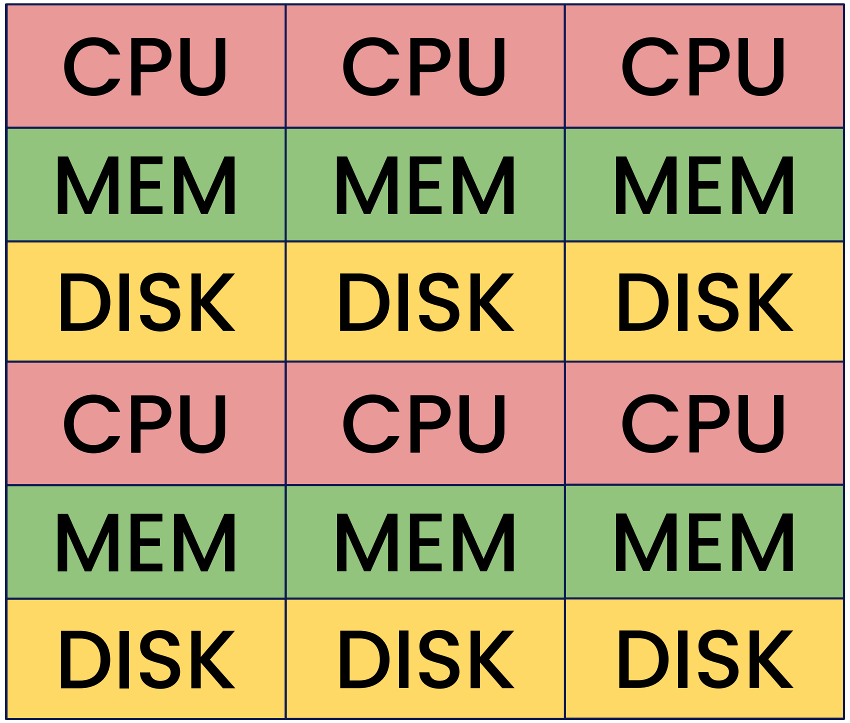
Storage

Virtualization + automation + orchestration



cluster

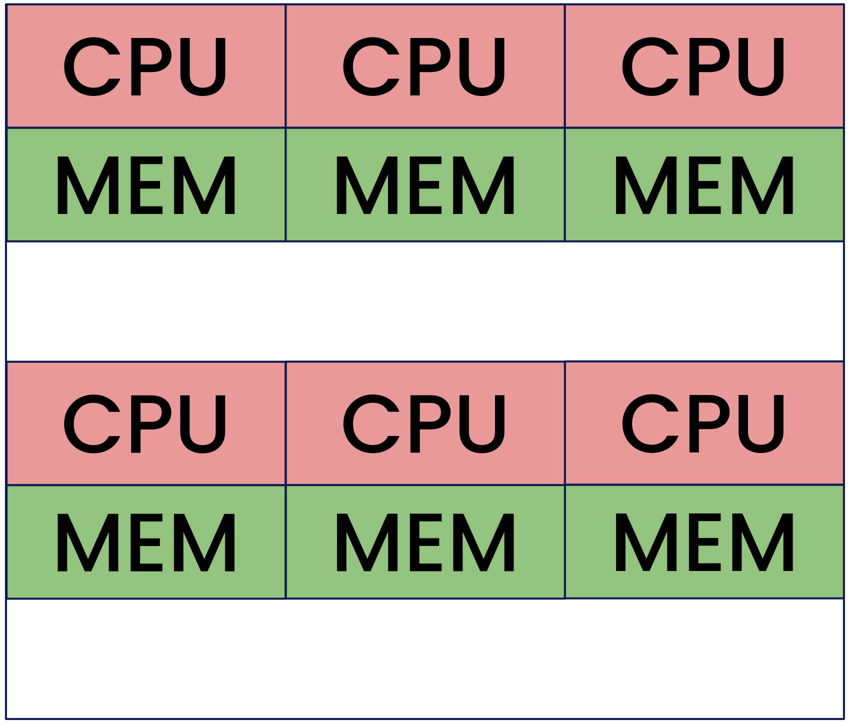
Virtualization + automation + orchestration



CPU	CPU	CPU
MEM	MEM	MEM
DISK	DISK	DISK
CPU	CPU	CPU
MEM	MEM	MEM
DISK	DISK	DISK

cluster

Virtualization + automation + orchestration



CPU CPU CPU

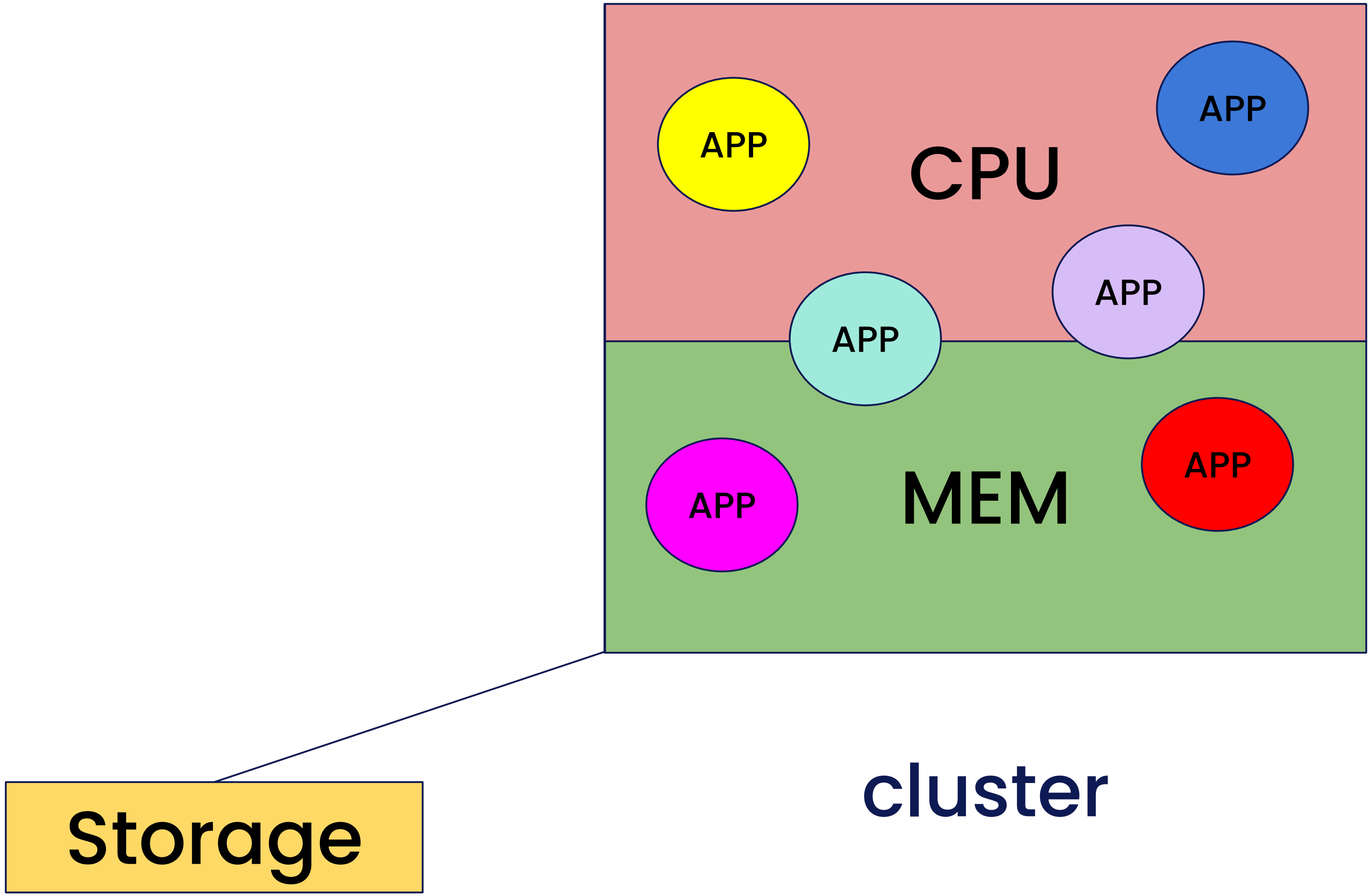
MEM MEM MEM

CPU CPU CPU

MEM MEM MEM

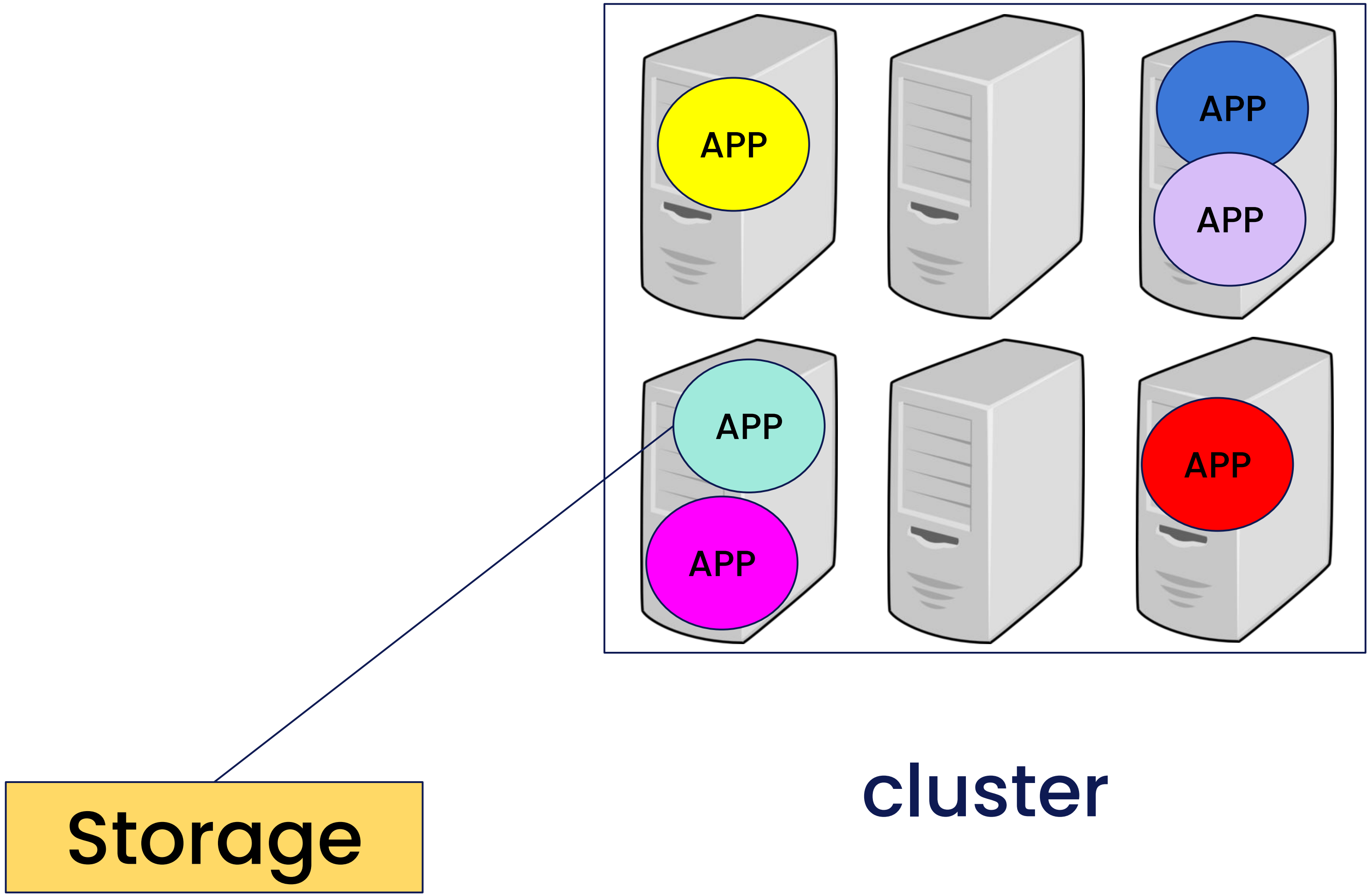
cluster

Virtualization + automation + orchestration



cluster

Virtualization + automation + orchestration



Storage

cluster

Kubernetes operators

“The *operator pattern* aims to capture the key aim of a human operator who is managing a service or set of services.”

“(...) how you can write code to automate a task beyond what Kubernetes itself provides.”

“(...) use automation to take care of repeatable tasks.”

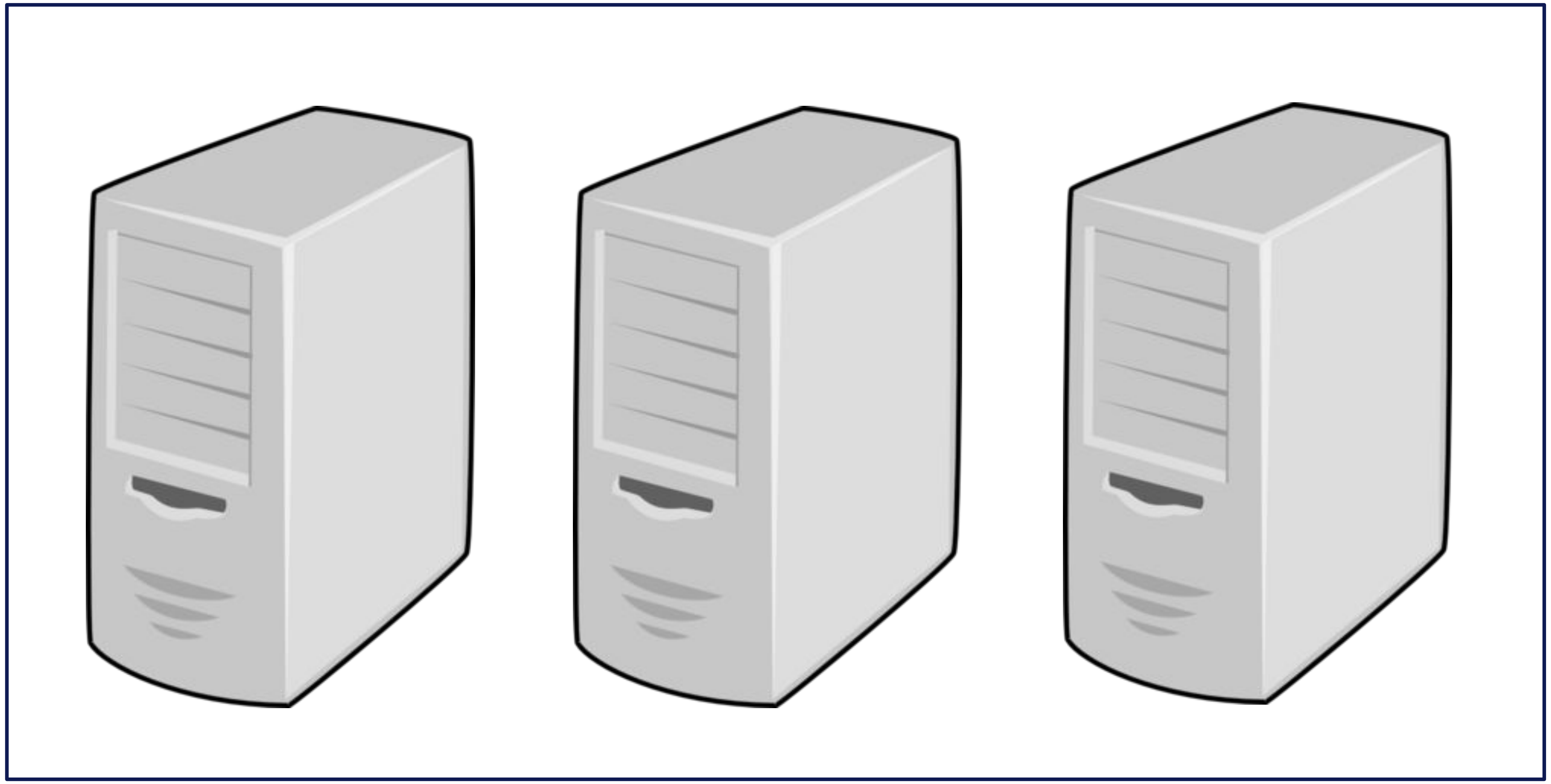
https://kubernetes.io/docs/concepts/extend-kubernetes/operator/

Kubernetes operators

- Database deployment
- High availability
- Backup and restore
- Scaling
- Monitoring integration

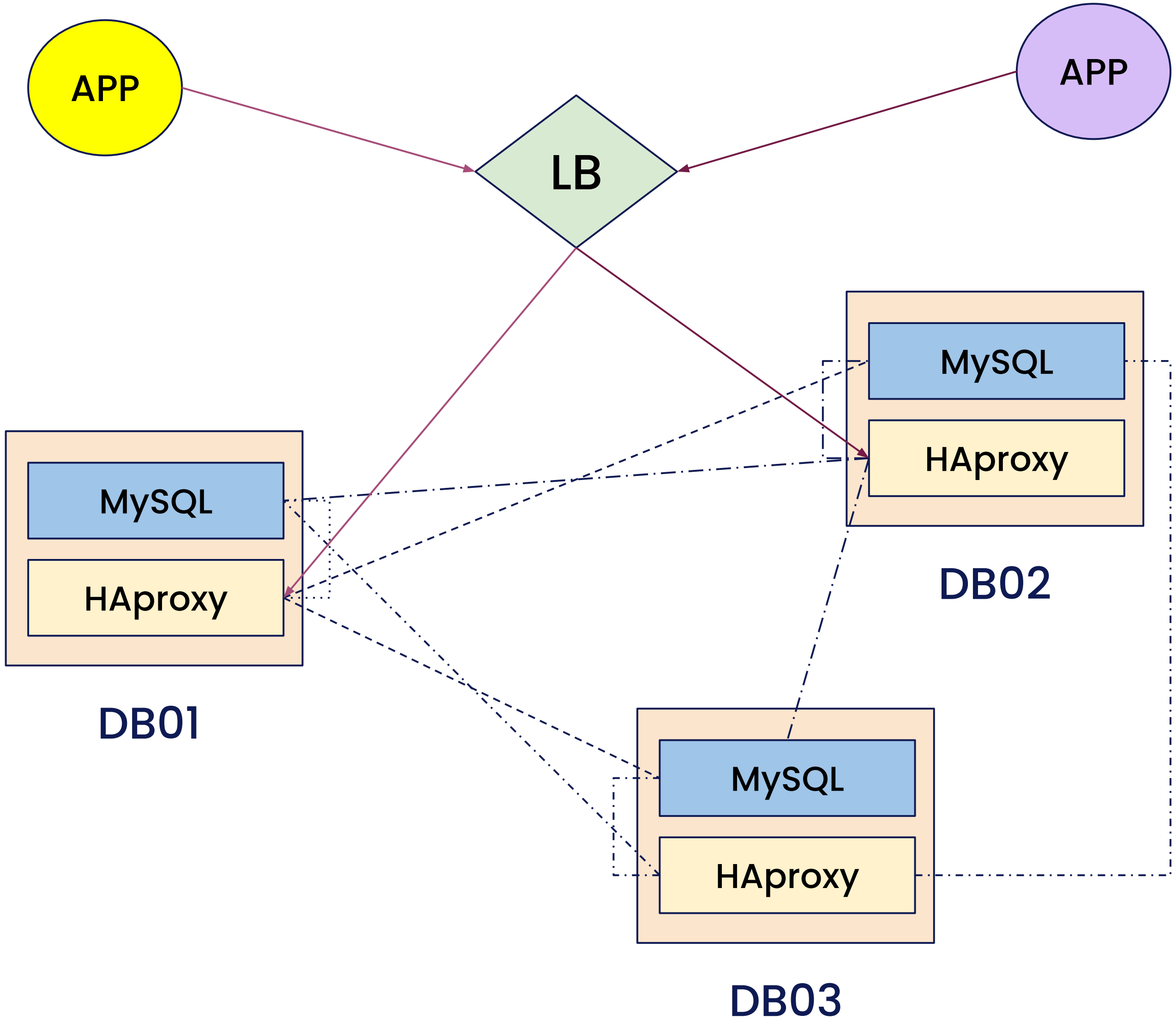
convenience

Percona Operator for MySQL

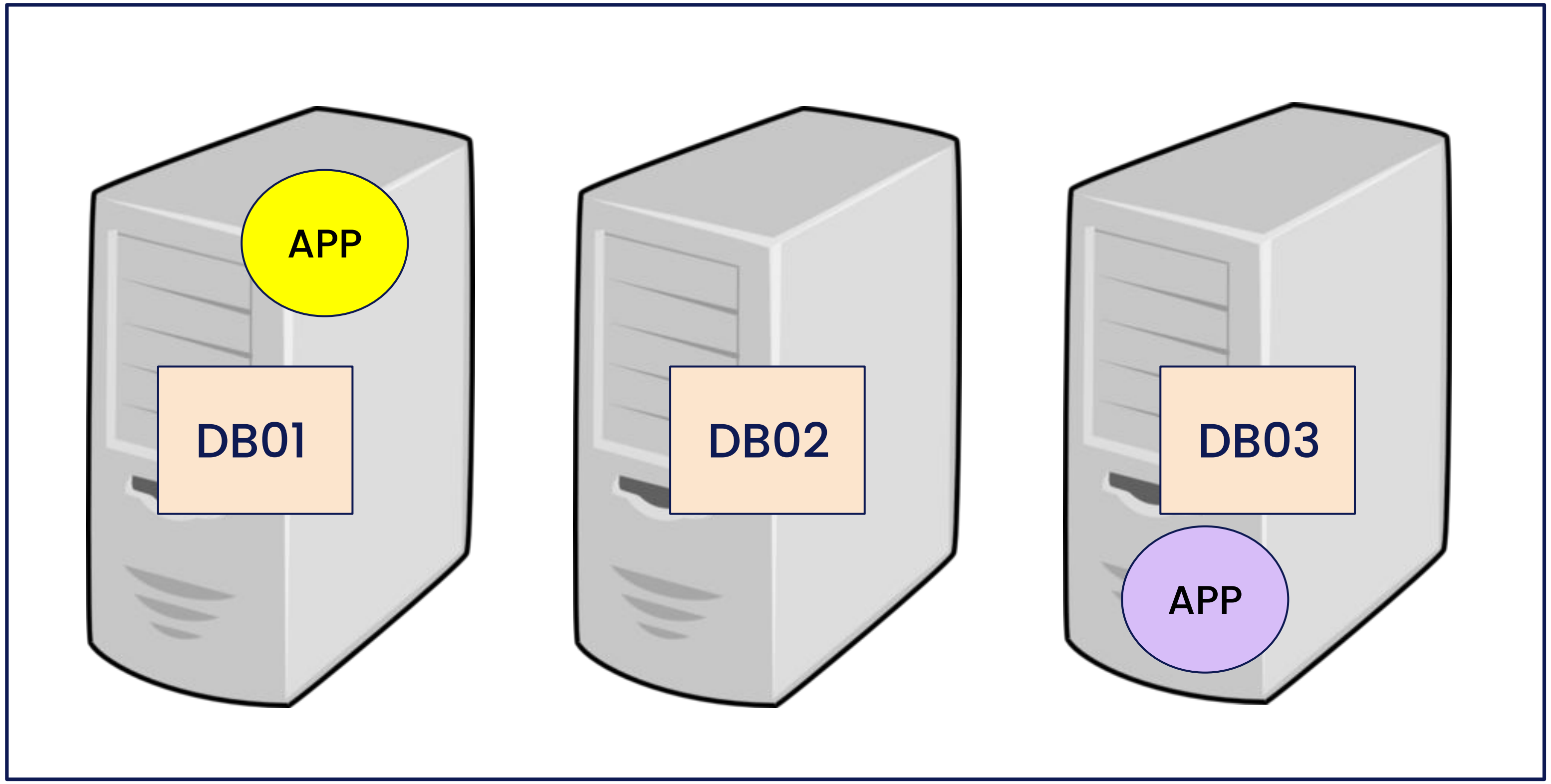


K8s cluster

Percona Operator for MySQL

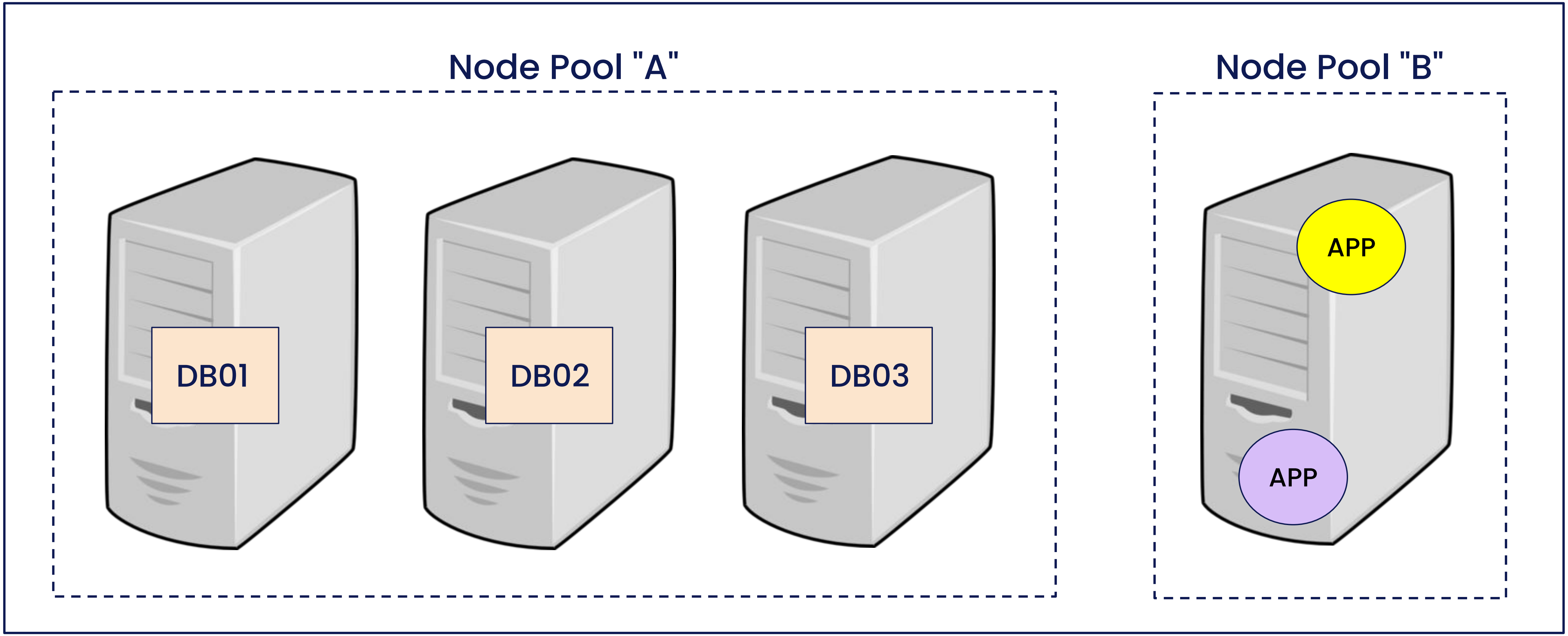


Percona Operator for MySQL



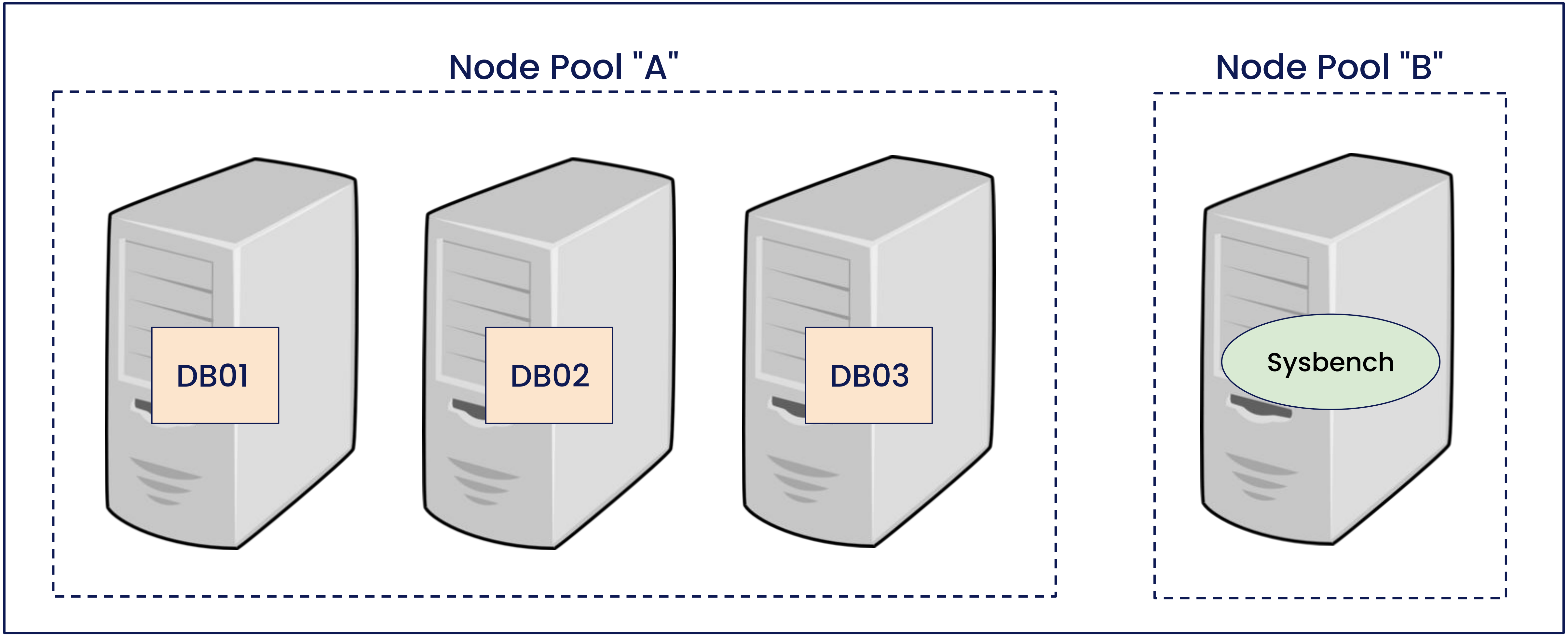
K8s cluster

Percona Operator for MySQL



K8s cluster

Percona Operator for MySQL



K8s cluster

Testing

Comparison

- 2 cloud providers:
  - AWS: EKS and "EC2"
  - Cloud "B"

- 2 workloads:
  - Sysbench OLTP
  - Sysbench TPC-C

- 2 datasets:
  - One that fits in memory (*small*)
  - One that doesn't (*big*)

https://smalldatum.blogspot.com/2021/11/dewitt-clause-vs-public-cloud.html

Small Datum

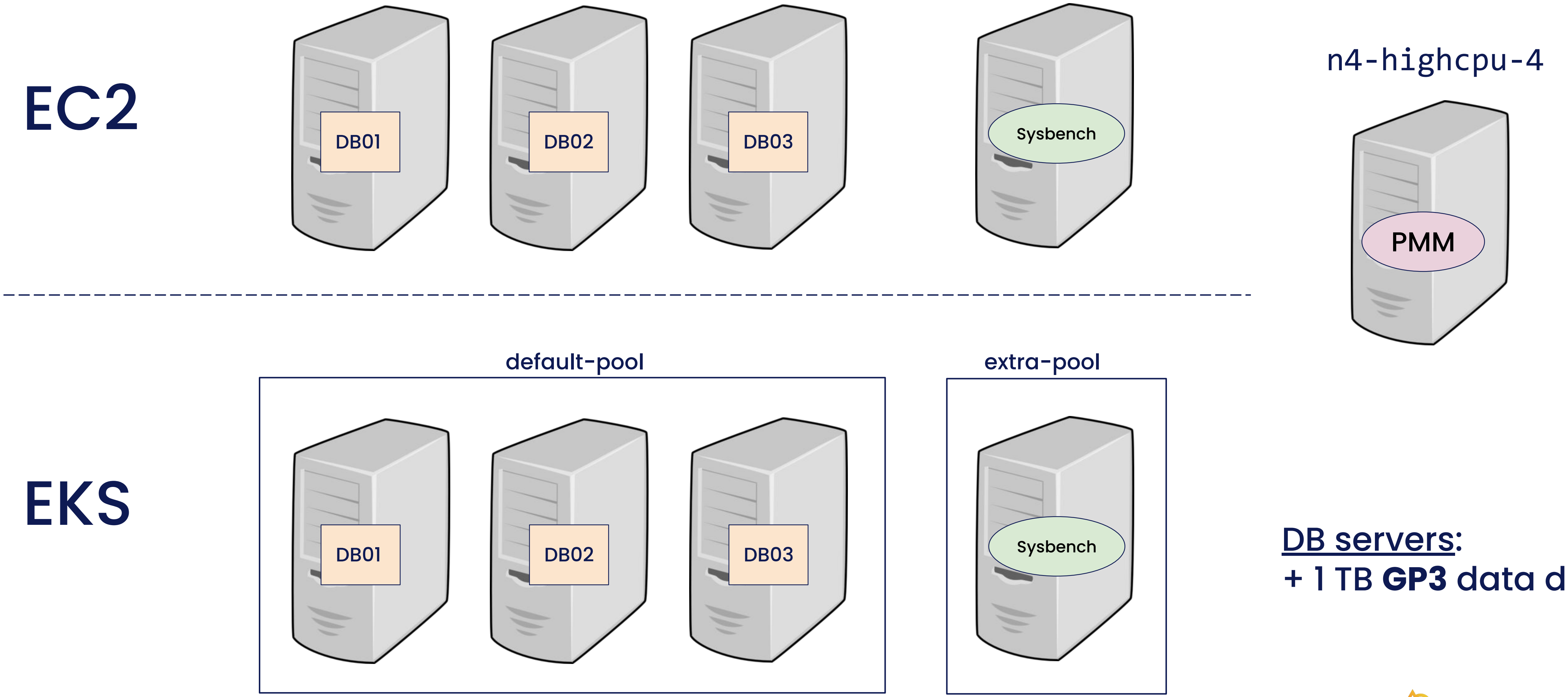
Saturday, November 13, 2021

DeWitt Clause vs the public cloud

Comparison

`m7i.2xlarge`

8 vCPUs
32 GB RAM



EC2

EKS

`n4-highcpu-4`

DB servers:
+ 1 TB **GP3** data disk

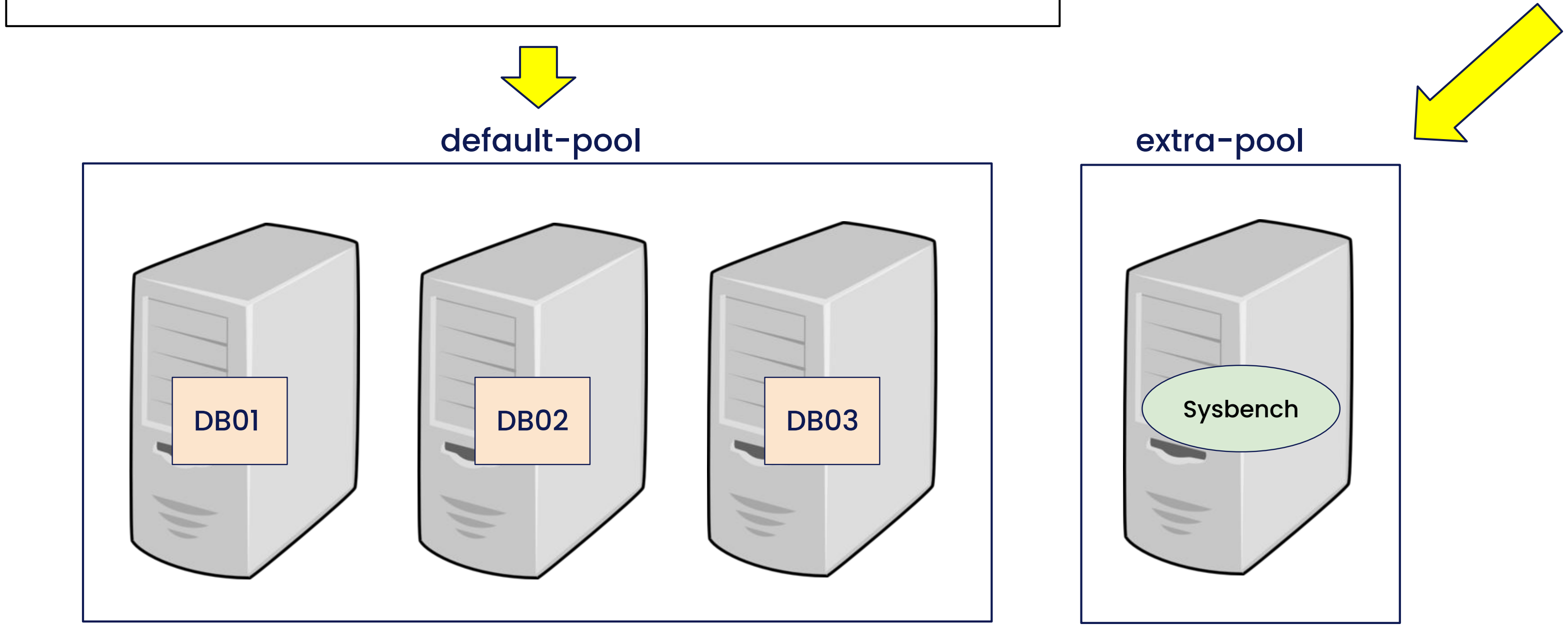
K8s: node selector

annotations

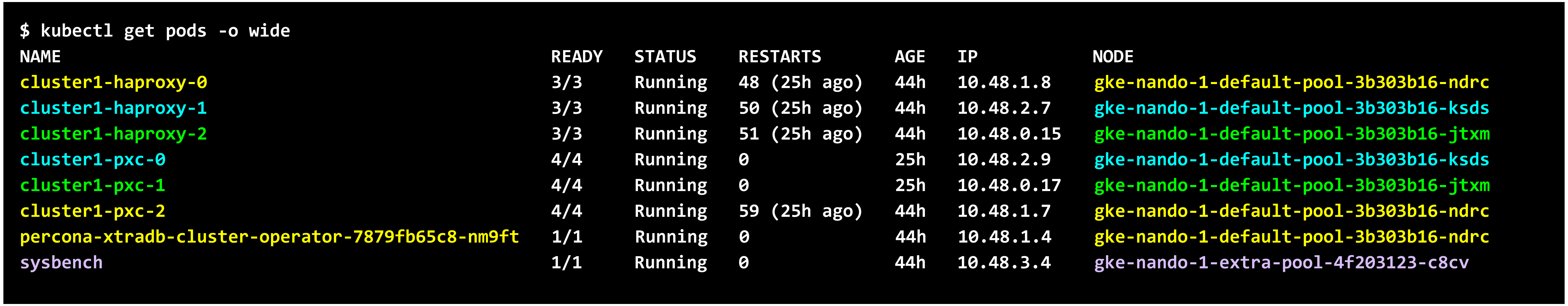
```
spec:
 pxc:
  affinity:
    antiAffinityTopologyKey: "kubernetes.io/hostname"
      advanced:
        nodeAffinity:
          requiredDuringSchedulingIgnoredDuringExecution:
            nodeSelectorTerms:
            - matchExpressions:
              - key: alpha.eksctl.io/nodegroup-name
                operator: In
                values:
                - default-pool
```

```
cat <<EOF | kubectl -n pxc apply -f -
apiVersion: v1
kind: Pod
metadata:
  name: sysbench
spec:
  nodeSelector:
    alpha.eksctl.io/nodegroup-name: extra-pool
  containers:
  - name: sysbench-client
    image: perconalab/sysbench
    imagePullPolicy: IfNotPresent
EOF
```

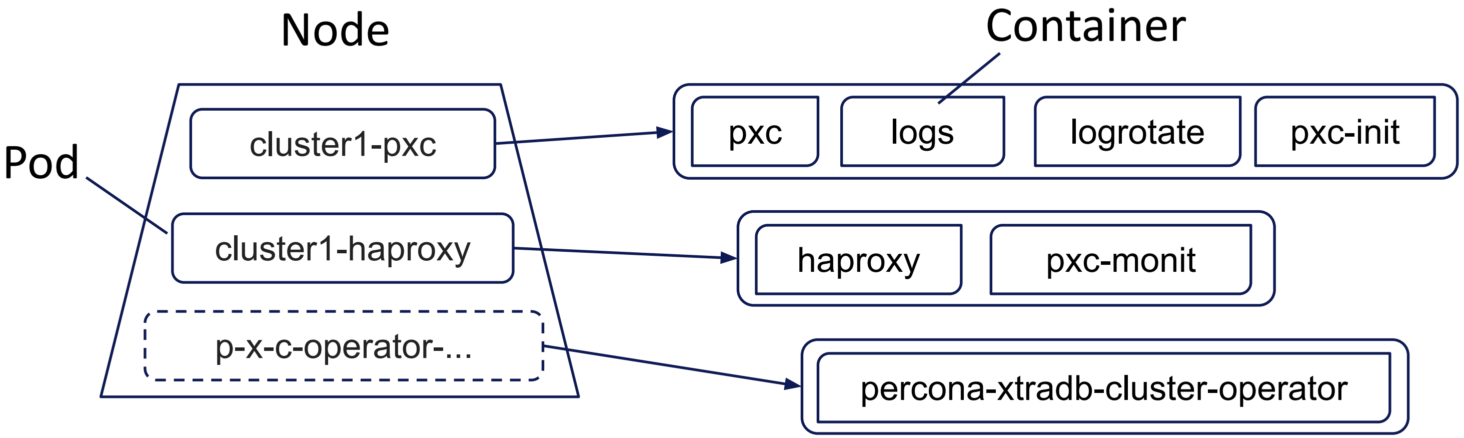
EKS



K8s: node selector



```
$ kubectl get pods -o wide
NAME                                              READY   STATUS    RESTARTS        AGE   IP           NODE
cluster1-haproxy-0                                3/3     Running   48 (25h ago)    44h   10.48.1.8    gke-nando-1-default-pool-3b303b16-ndrc
cluster1-haproxy-1                                3/3     Running   50 (25h ago)    44h   10.48.2.7    gke-nando-1-default-pool-3b303b16-ksds
cluster1-haproxy-2                                3/3     Running   51 (25h ago)    44h   10.48.0.15   gke-nando-1-default-pool-3b303b16-jtxm
cluster1-pxc-0                                    4/4     Running   0               25h   10.48.2.9    gke-nando-1-default-pool-3b303b16-ksds
cluster1-pxc-1                                    4/4     Running   0               25h   10.48.0.17   gke-nando-1-default-pool-3b303b16-jtxm
cluster1-pxc-2                                    4/4     Running   59 (25h ago)    44h   10.48.1.7    gke-nando-1-default-pool-3b303b16-ndrc
percona-xtradb-cluster-operator-7879fb65c8-nm9ft  1/1     Running   0               44h   10.48.1.4    gke-nando-1-default-pool-3b303b16-ndrc
sysbench                                          1/1     Running   0               44h   10.48.3.4    gke-nando-1-extra-pool-4f203123-c8cv
```



Configuration

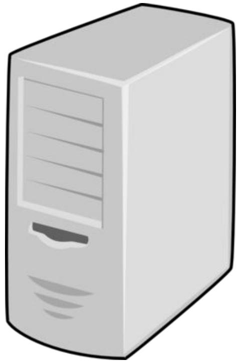
- Percona XtraDB Cluster 8.0.42
- Percona Operator for MySQL 1.18.0
    - Based on PXC 8.0.42

- Sysbench 1.0 (no SSL)
- PMM 3.5

```
[mysqld]
innodb_buffer_pool_size=22G
innodb_redo_log_capacity=40G
wsrep_provider_options="gcache.size=2G;
gcache.recover=yes"
```

Even though:

```
$ free -m
              total        used        free      shared  buff/cache   available
Mem:          32092       27400         446           1        4709        4691
Swap:             0           0           0
```



m7i.2xlarge

8 vCPUs
32 GB RAM

Sysbench

Rinse
&
repeat

Workload	Options	"Small"	"Big"
OLTP	--tables	8	8
	--table_size	15625000	125000000
TPCC	--tables	10	10
	--table_size	30	300

OLTP

1 prepare

2

```
sysbench /usr/share/sysbench/oltp_read_write.lua --db-driver=mysql --db-ps-mode=disable
--mysql-ignore-errors=all --skip_trx=on --mysql-user=sysbench --mysql-password=secret
--mysql-db=sbtest --mysql-host=cluster1-pxc-0.cluster1-pxc.pxc.svc.cluster.local --tables=8
--mysql-port=3306 --table_size=15625000 --report-interval=1 --threads=64 --time=3600 run
```

3 + oltp_read_write.lua x3

TPCC

4 prepare

```
LUA_PATH=/sysbench/sysbench-tpcc/?.lua /sysbench/sysbench-tpcc/tpcc.lua
--db-driver=mysql --mysql-user=sysbench --mysql-password=secret --mysql-db=sbtest
--mysql-host=192.168.81.6 --mysql-port=3306 --tables=10 --scale=30 --report-interval=1
--threads=64 --use_fk=0 --trx_level=RC --mysql-ignore-errors=all --force_pk=1
--time=3600 run
```

5

x3

Results

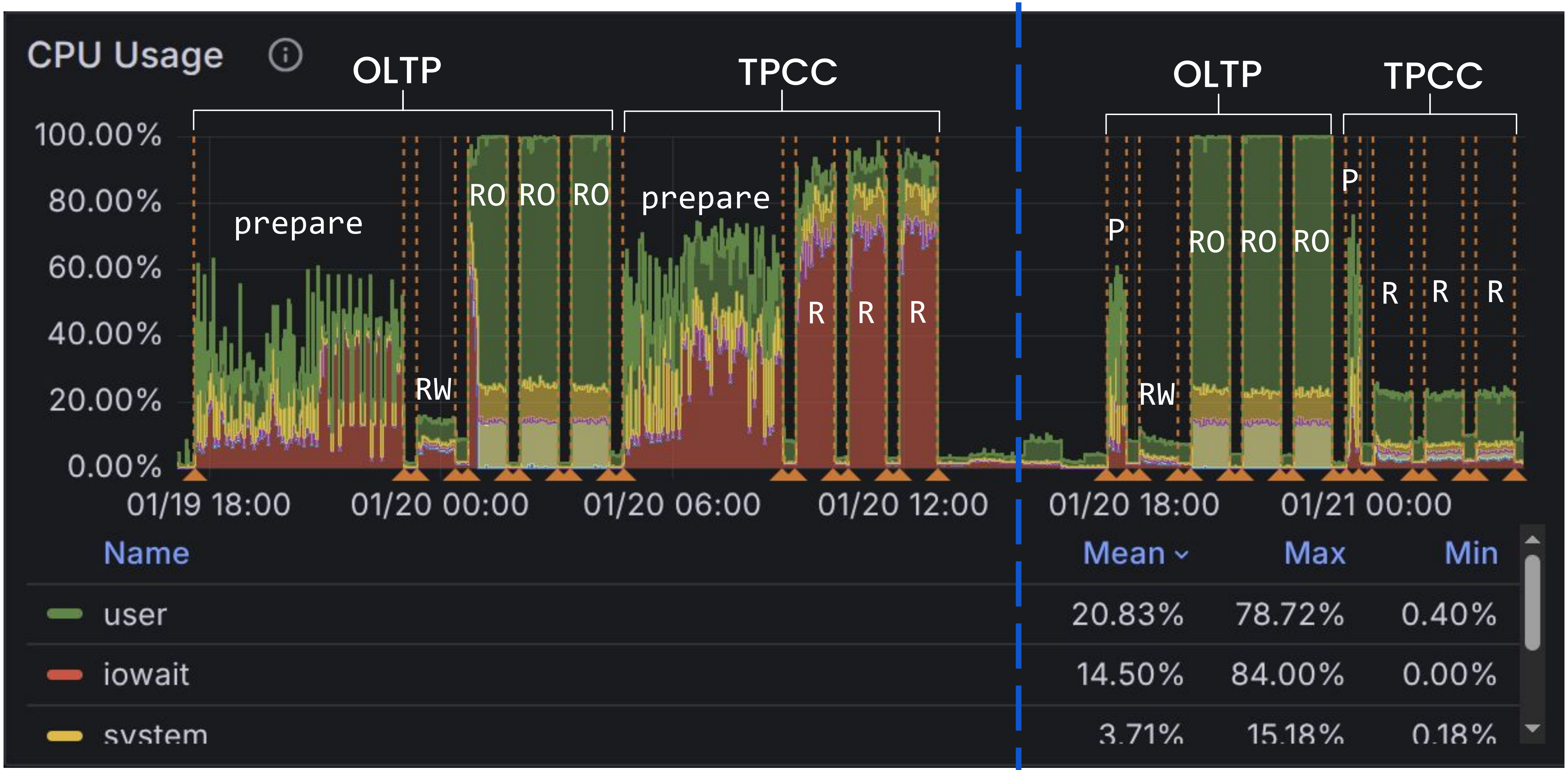
(previa)

Sysbench

Big

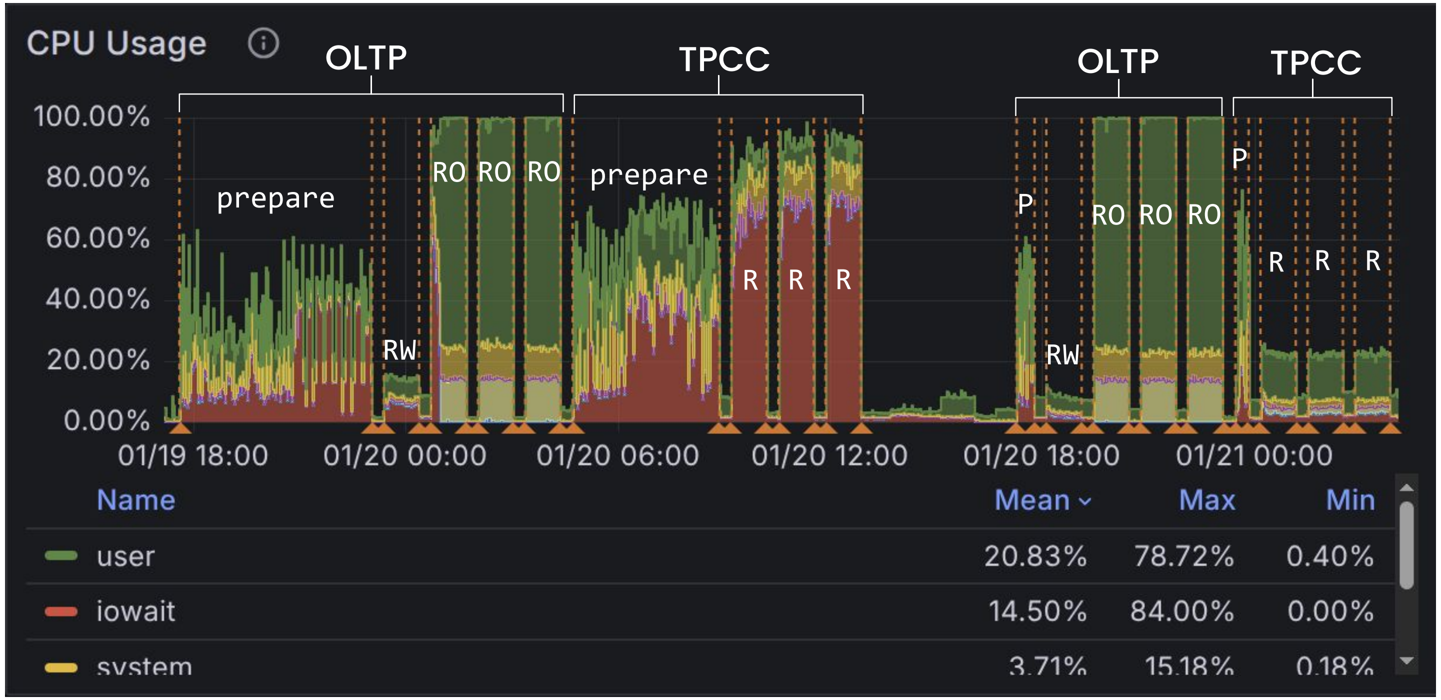
Small

EKS



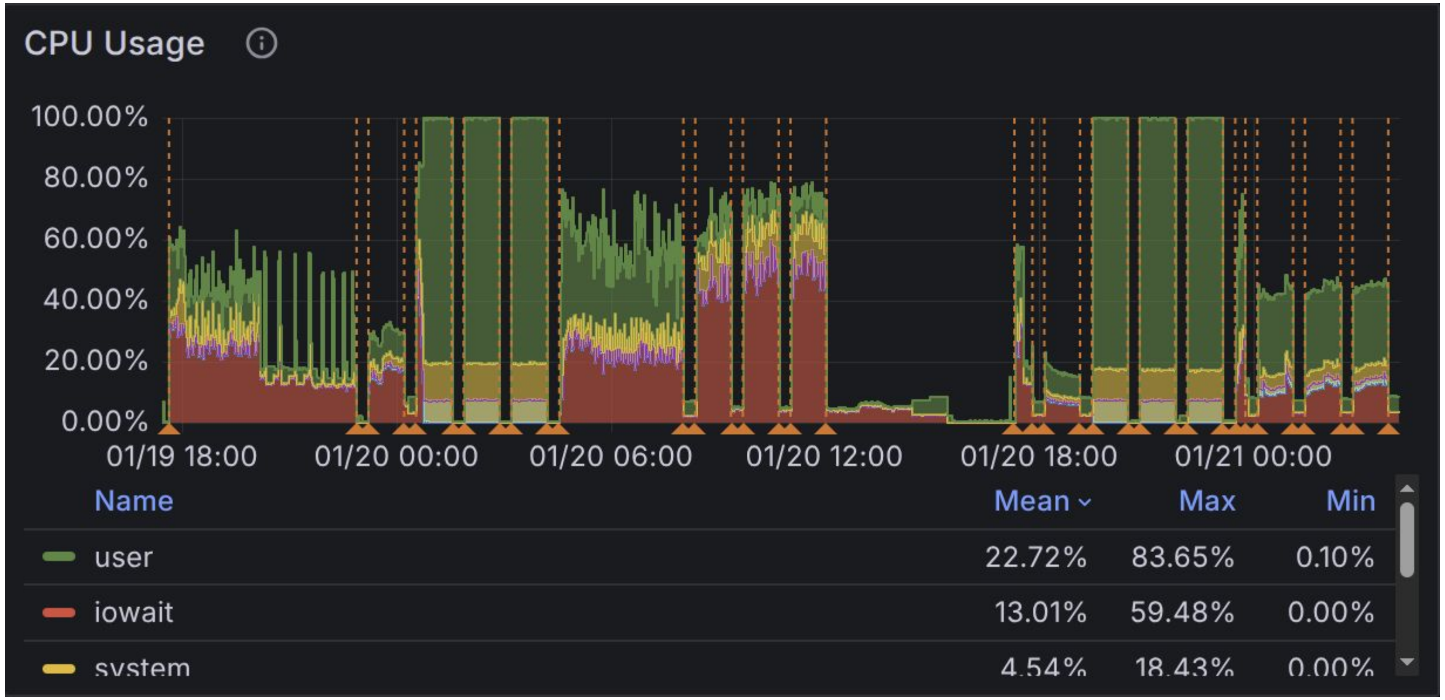
Name	Mean	Max	Min
🟩 user	20.83%	78.72%	0.40%
🟥 iowait	14.50%	84.00%	0.00%
🟨 system	3.71%	15.18%	0.18%

EKS



Name	Mean	Max	Min
user	20.83%	78.72%	0.40%
iowait	14.50%	84.00%	0.00%
system	3.71%	15.18%	0.18%

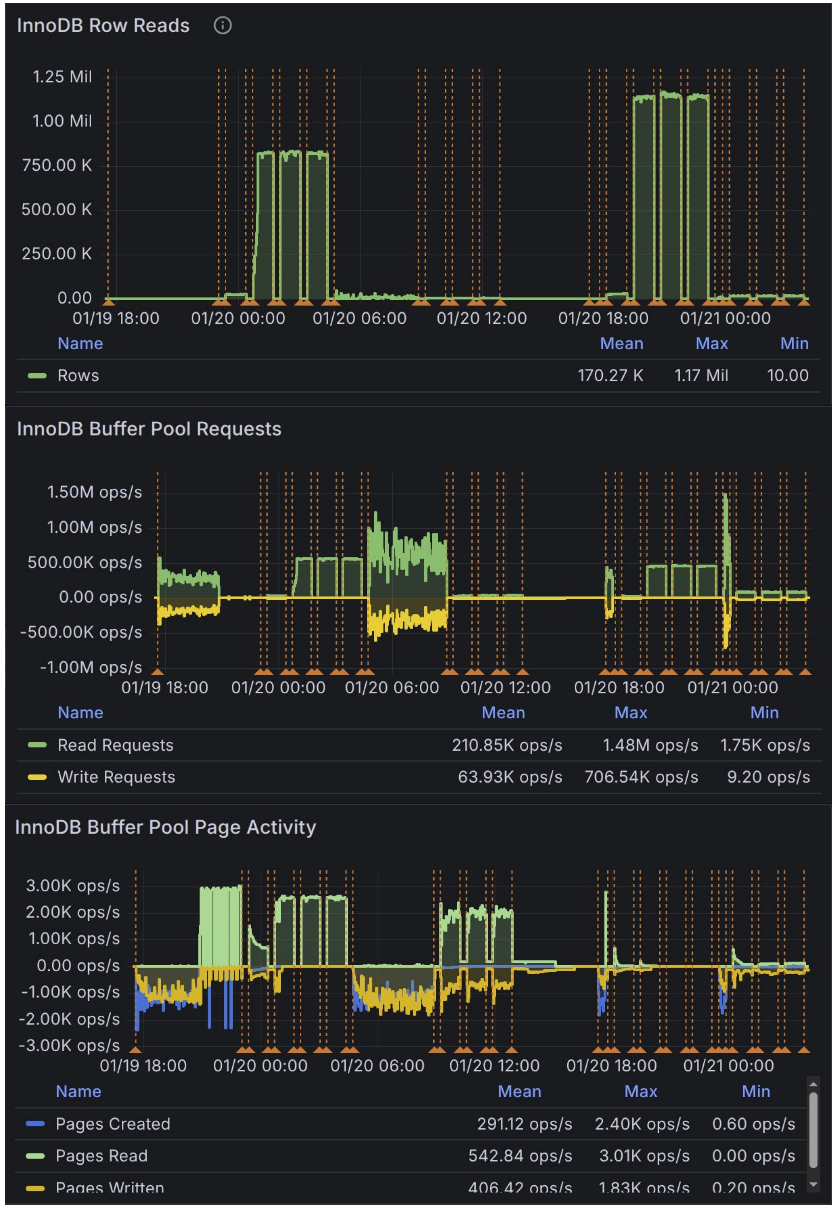
EC2



Name	Mean	Max	Min
user	22.72%	83.65%	0.10%
iowait	13.01%	59.48%	0.00%
system	4.54%	18.43%	0.00%

Disk reads

EKS



InnoDB Row Reads

Name	Mean	Max	Min
Rows	170.27 K	1.17 Mil	10.00

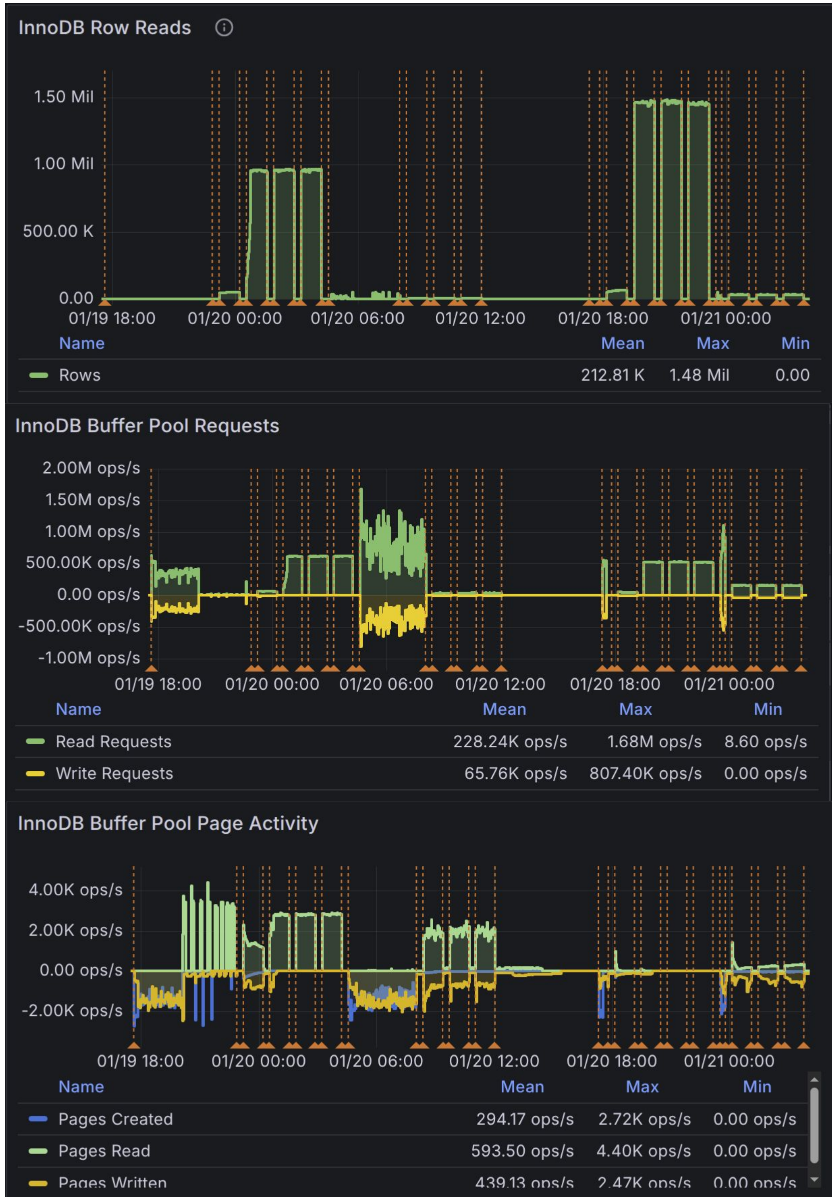
InnoDB Buffer Pool Requests

Name	Mean	Max	Min
Read Requests	210.85K ops/s	1.48M ops/s	1.75K ops/s
Write Requests	63.93K ops/s	706.54K ops/s	9.20 ops/s

InnoDB Buffer Pool Page Activity

Name	Mean	Max	Min
Pages Created	291.12 ops/s	2.40K ops/s	0.60 ops/s
Pages Read	542.84 ops/s	3.01K ops/s	0.00 ops/s
Pages Written	406.42 ops/s	1.83K ops/s	0.20 ops/s

EC2



InnoDB Row Reads

Name	Mean	Max	Min
Rows	212.81 K	1.48 Mil	0.00

InnoDB Buffer Pool Requests

Name	Mean	Max	Min
Read Requests	228.24K ops/s	1.68M ops/s	8.60 ops/s
Write Requests	65.76K ops/s	807.40K ops/s	0.00 ops/s

InnoDB Buffer Pool Page Activity

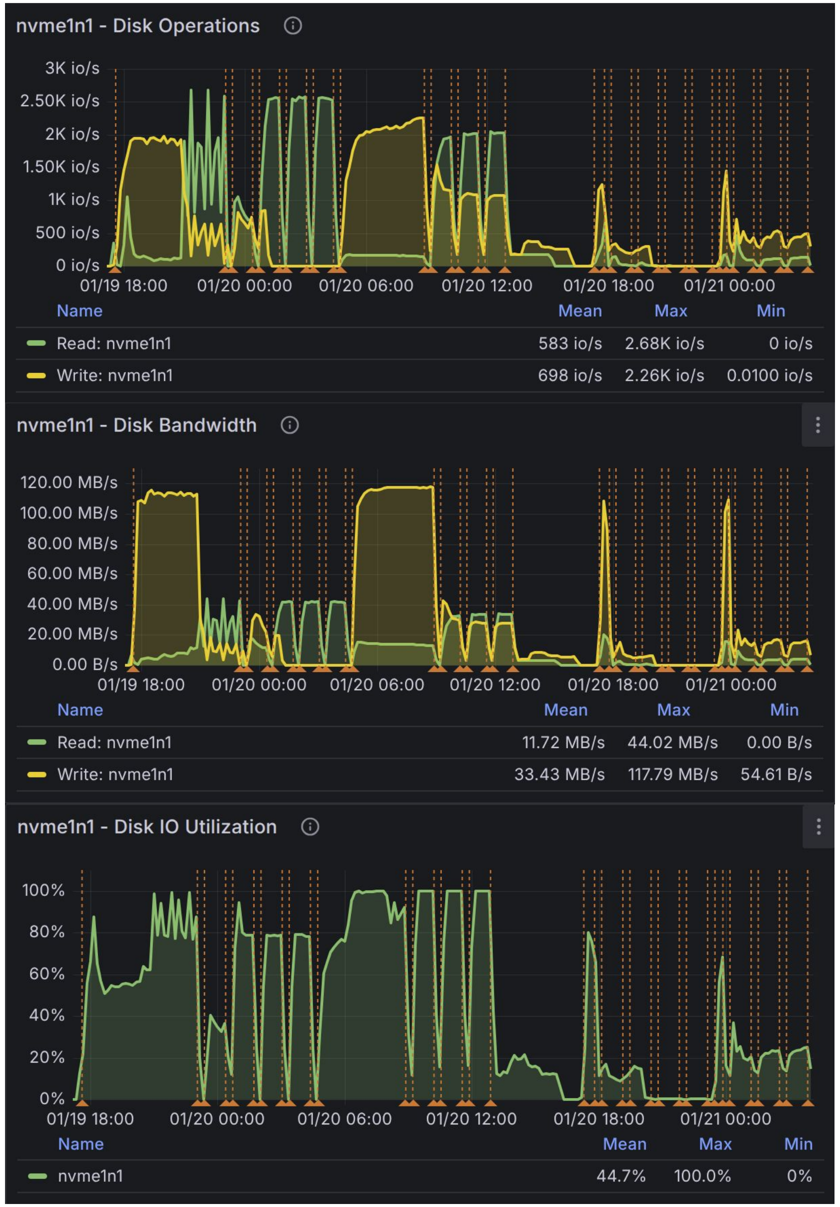
Name	Mean	Max	Min
Pages Created	294.17 ops/s	2.72K ops/s	0.00 ops/s
Pages Read	593.50 ops/s	4.40K ops/s	0.00 ops/s
Pages Written	439.13 ops/s	2.47K ops/s	0.00 ops/s

Data disk

"The new gp3 volumes deliver a baseline performance of **3,000 IOPS** and **125 MB/s** at any volume size."

https://aws.amazon.com/ebs/general-purpose/

EKS



nvme1n1 - Disk Operations

Name	Mean	Max	Min
Read: nvme1n1	583 io/s	2.68K io/s	0 io/s
Write: nvme1n1	698 io/s	2.26K io/s	0.0100 io/s

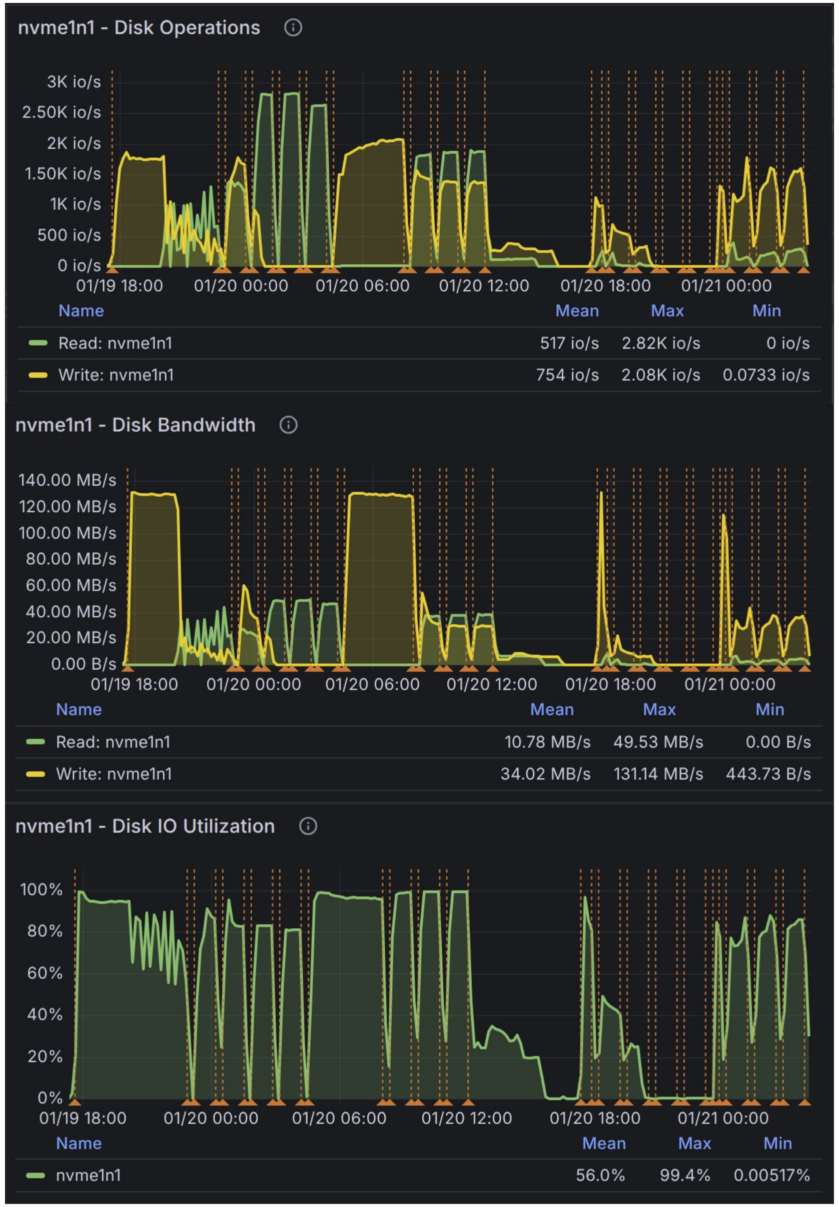
nvme1n1 - Disk Bandwidth

Name	Mean	Max	Min
Read: nvme1n1	11.72 MB/s	44.02 MB/s	0.00 B/s
Write: nvme1n1	33.43 MB/s	117.79 MB/s	54.61 B/s

nvme1n1 - Disk IO Utilization

Name	Mean	Max	Min
nvme1n1	44.7%	100.0%	0%

EC2



nvme1n1 - Disk Operations

Name	Mean	Max	Min
Read: nvme1n1	517 io/s	2.82K io/s	0 io/s
Write: nvme1n1	754 io/s	2.08K io/s	0.0733 io/s

nvme1n1 - Disk Bandwidth

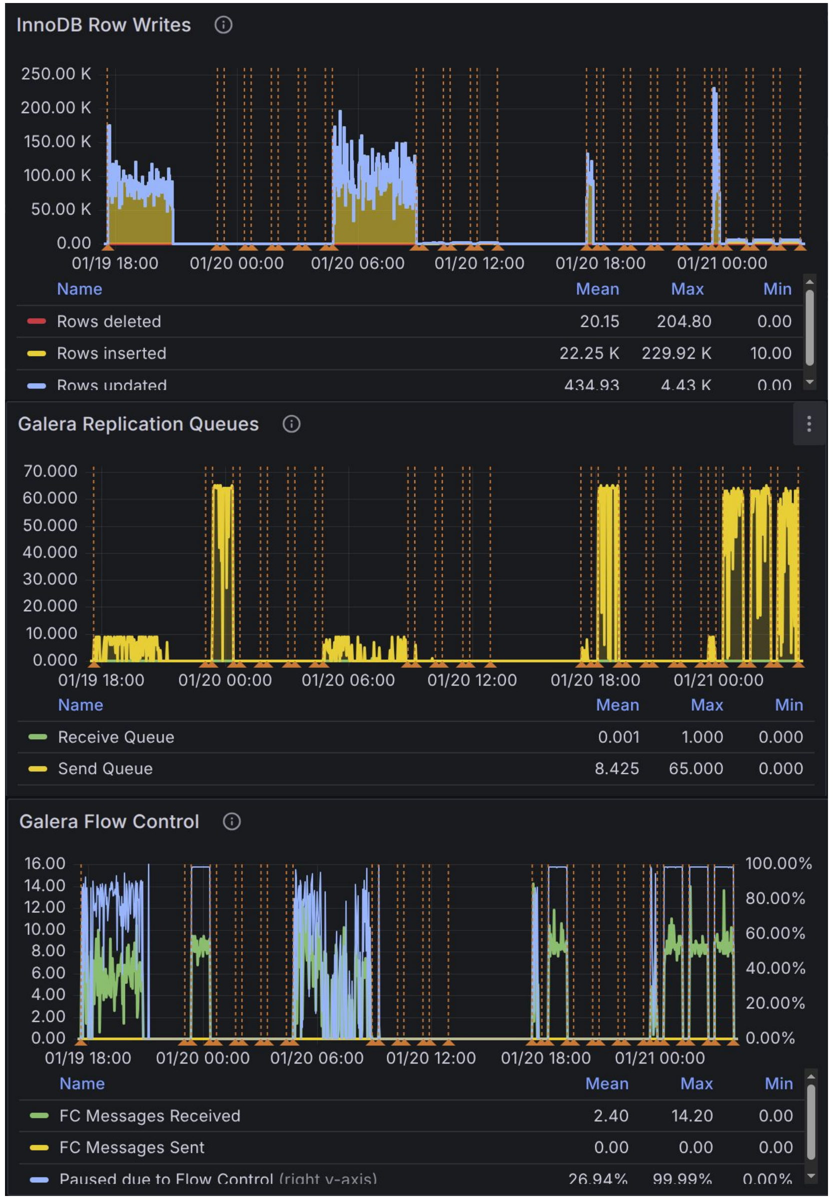
Name	Mean	Max	Min
Read: nvme1n1	10.78 MB/s	49.53 MB/s	0.00 B/s
Write: nvme1n1	34.02 MB/s	131.14 MB/s	443.73 B/s

nvme1n1 - Disk IO Utilization

Name	Mean	Max	Min
nvme1n1	56.0%	99.4%	0.00517%

Disk writes

EKS



InnoDB Row Writes

Name	Mean	Max	Min
Rows deleted	20.15	204.80	0.00
Rows inserted	22.25 K	229.92 K	10.00
Rows updated	434.93	4.43 K	0.00

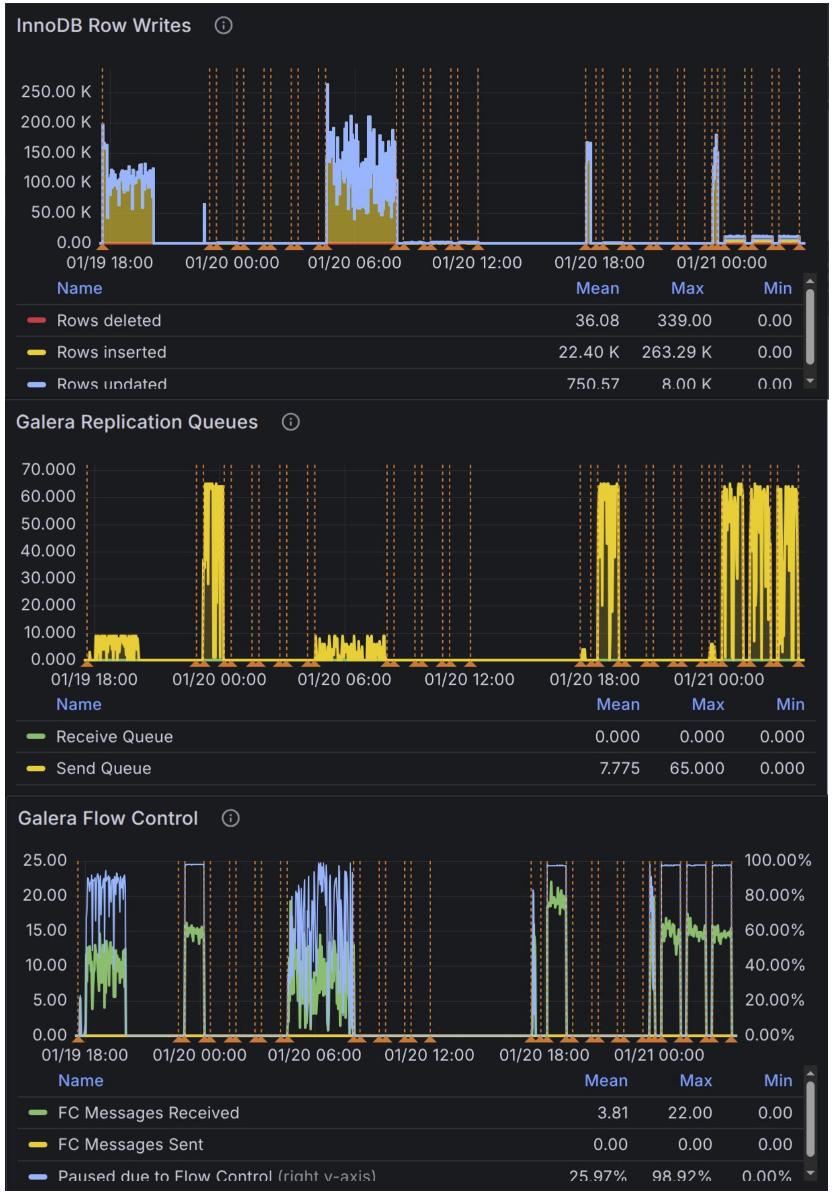
Galera Replication Queues

Name	Mean	Max	Min
Receive Queue	0.001	1.000	0.000
Send Queue	8.425	65.000	0.000

Galera Flow Control

Name	Mean	Max	Min
FC Messages Received	2.40	14.20	0.00
FC Messages Sent	0.00	0.00	0.00
Paused due to Flow Control (right y-axis)	26.94%	99.99%	0.00%

EC2



InnoDB Row Writes

Name	Mean	Max	Min
Rows deleted	36.08	339.00	0.00
Rows inserted	22.40 K	263.29 K	0.00
Rows updated	750.57	8.00 K	0.00

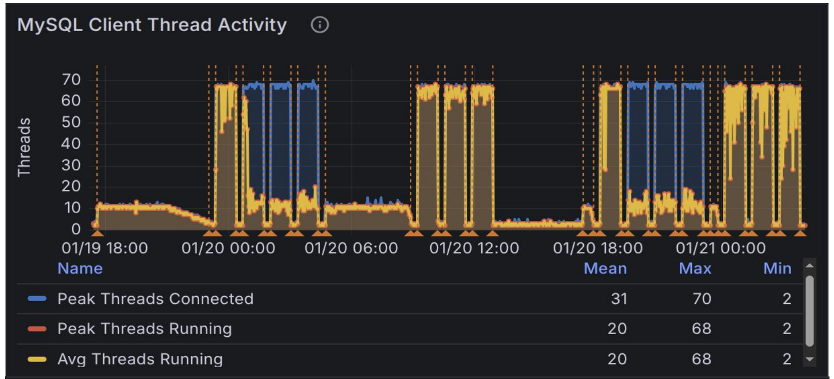
Galera Replication Queues

Name	Mean	Max	Min
Receive Queue	0.000	0.000	0.000
Send Queue	7.775	65.000	0.000

Galera Flow Control

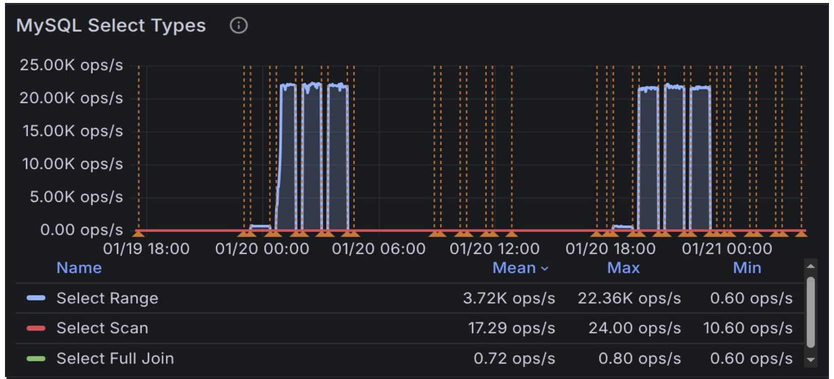
Name	Mean	Max	Min
FC Messages Received	3.81	22.00	0.00
FC Messages Sent	0.00	0.00	0.00
Paused due to Flow Control (right y-axis)	25.97%	98.92%	0.00%

Workload "anatomy"



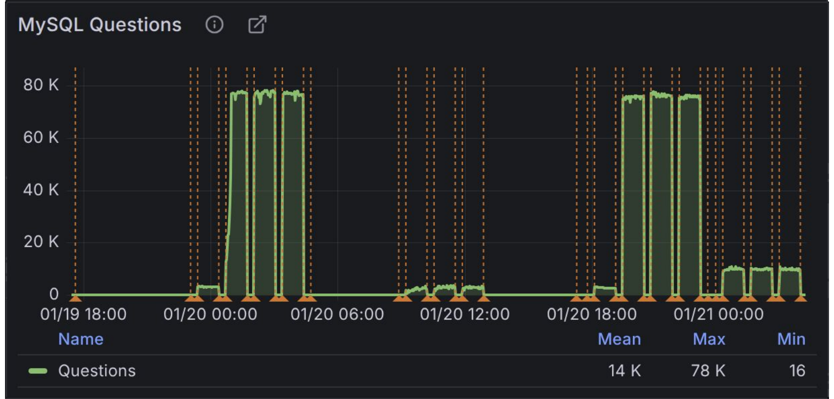
MySQL Client Thread Activity

Name	Mean	Max	Min
🟦 Peak Threads Connected	31	70	2
🟥 Peak Threads Running	20	68	2
🟨 Avg Threads Running	20	68	2



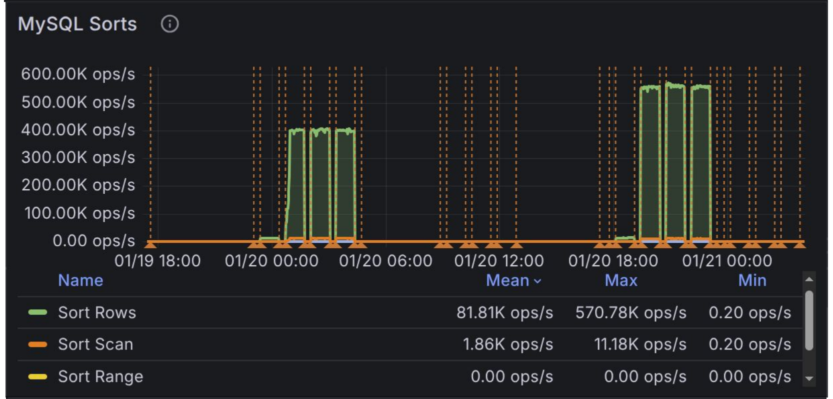
MySQL Select Types

Name	Mean	Max	Min
🟦 Select Range	3.72K ops/s	22.36K ops/s	0.60 ops/s
🟥 Select Scan	17.29 ops/s	24.00 ops/s	10.60 ops/s
🟩 Select Full Join	0.72 ops/s	0.80 ops/s	0.60 ops/s



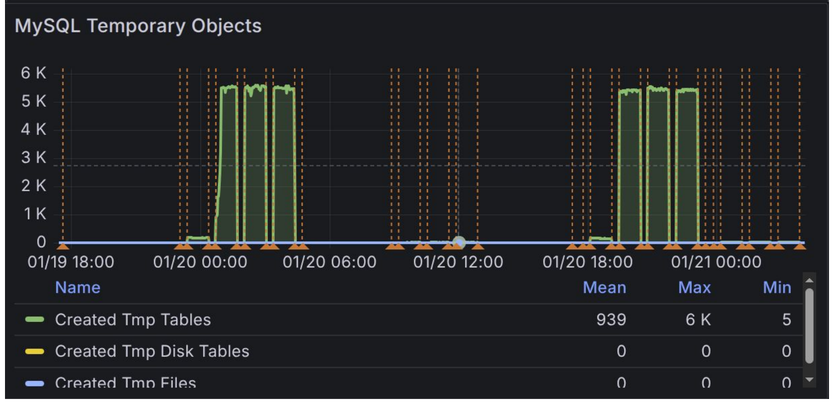
MySQL Questions

Name	Mean	Max	Min
🟩 Questions	14 K	78 K	16



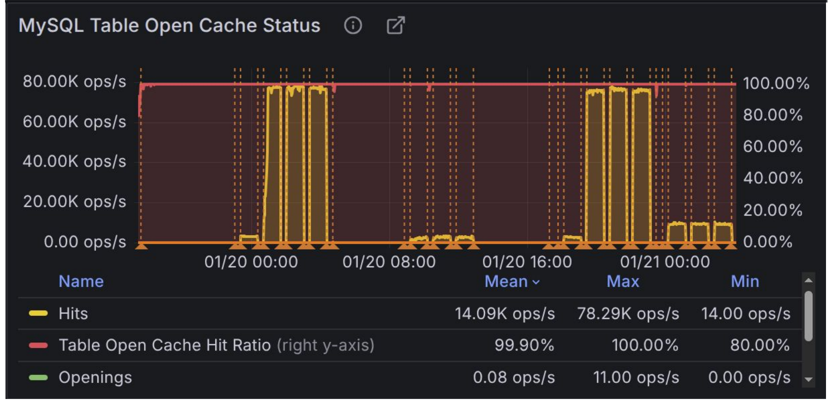
MySQL Sorts

Name	Mean	Max	Min
🟩 Sort Rows	81.81K ops/s	570.78K ops/s	0.20 ops/s
🟧 Sort Scan	1.86K ops/s	11.18K ops/s	0.20 ops/s
🟨 Sort Range	0.00 ops/s	0.00 ops/s	0.00 ops/s



MySQL Temporary Objects

Name	Mean	Max	Min
🟩 Created Tmp Tables	939	6 K	5
🟨 Created Tmp Disk Tables	0	0	0
🟦 Created Tmp Files	0	0	0



MySQL Table Open Cache Status

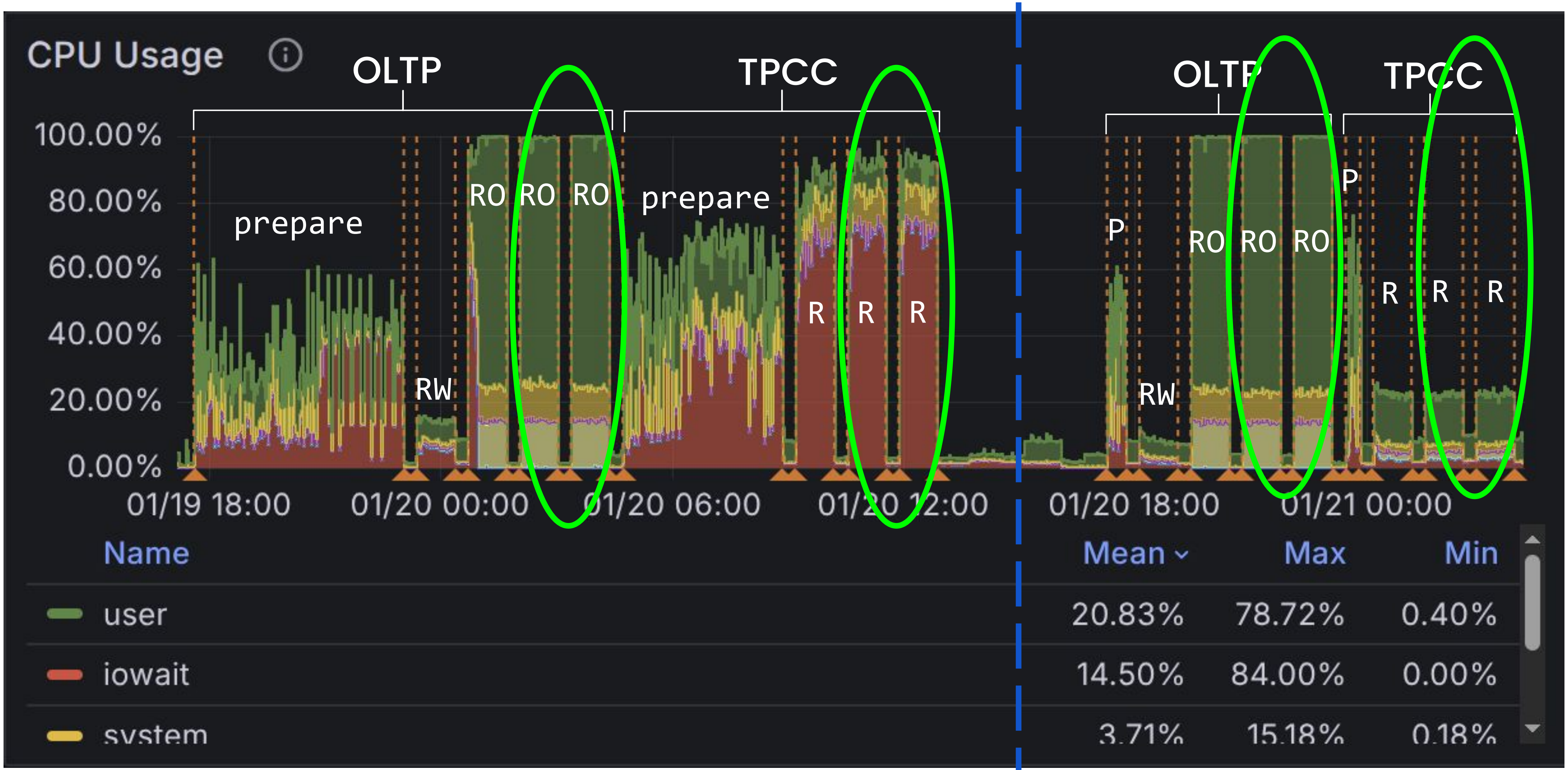
Name	Mean	Max	Min
🟨 Hits	14.09K ops/s	78.29K ops/s	14.00 ops/s
🟥 Table Open Cache Hit Ratio (right y-axis)	99.90%	100.00%	80.00%
🟩 Openings	0.08 ops/s	11.00 ops/s	0.00 ops/s

Results

Part 1

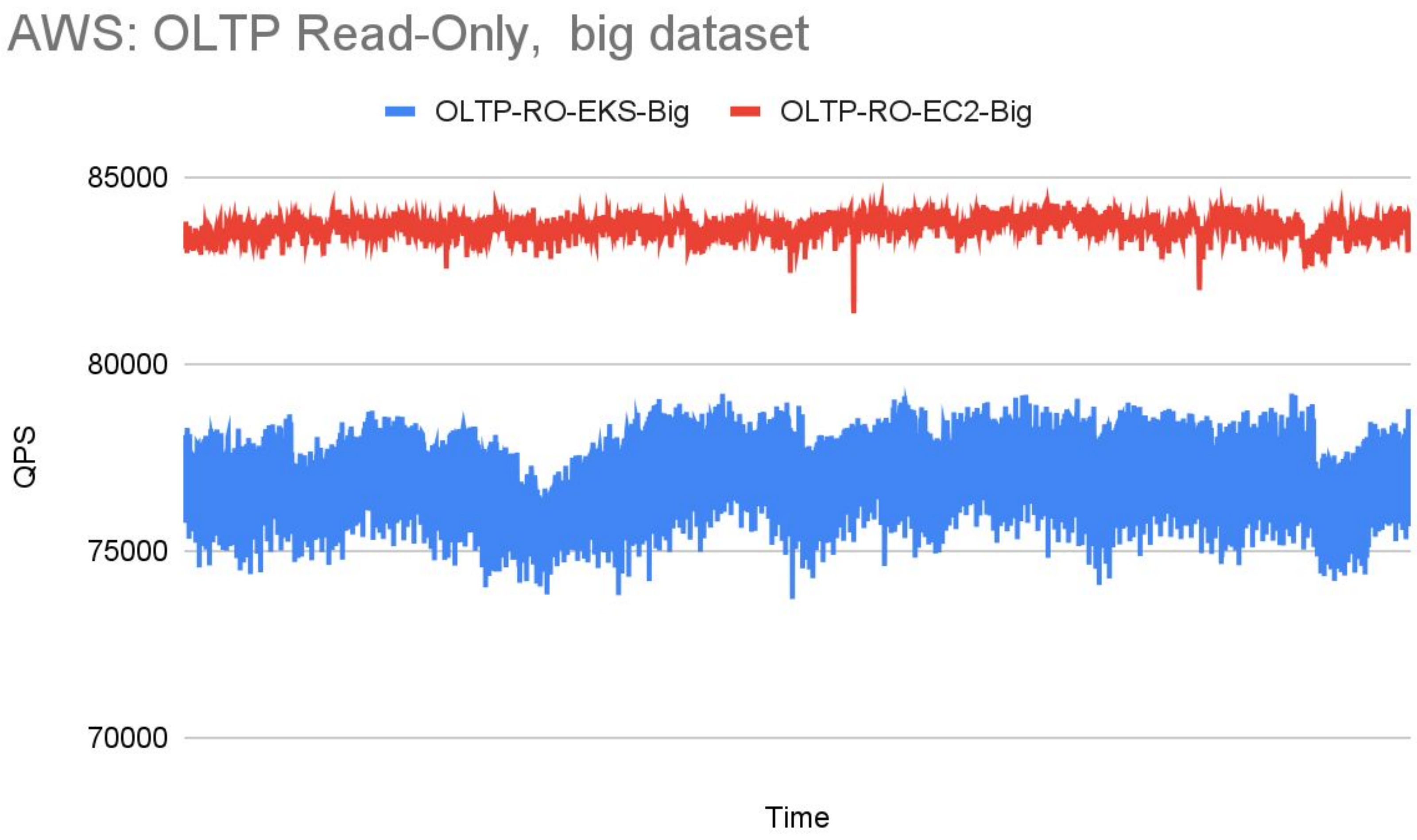
Big

Small

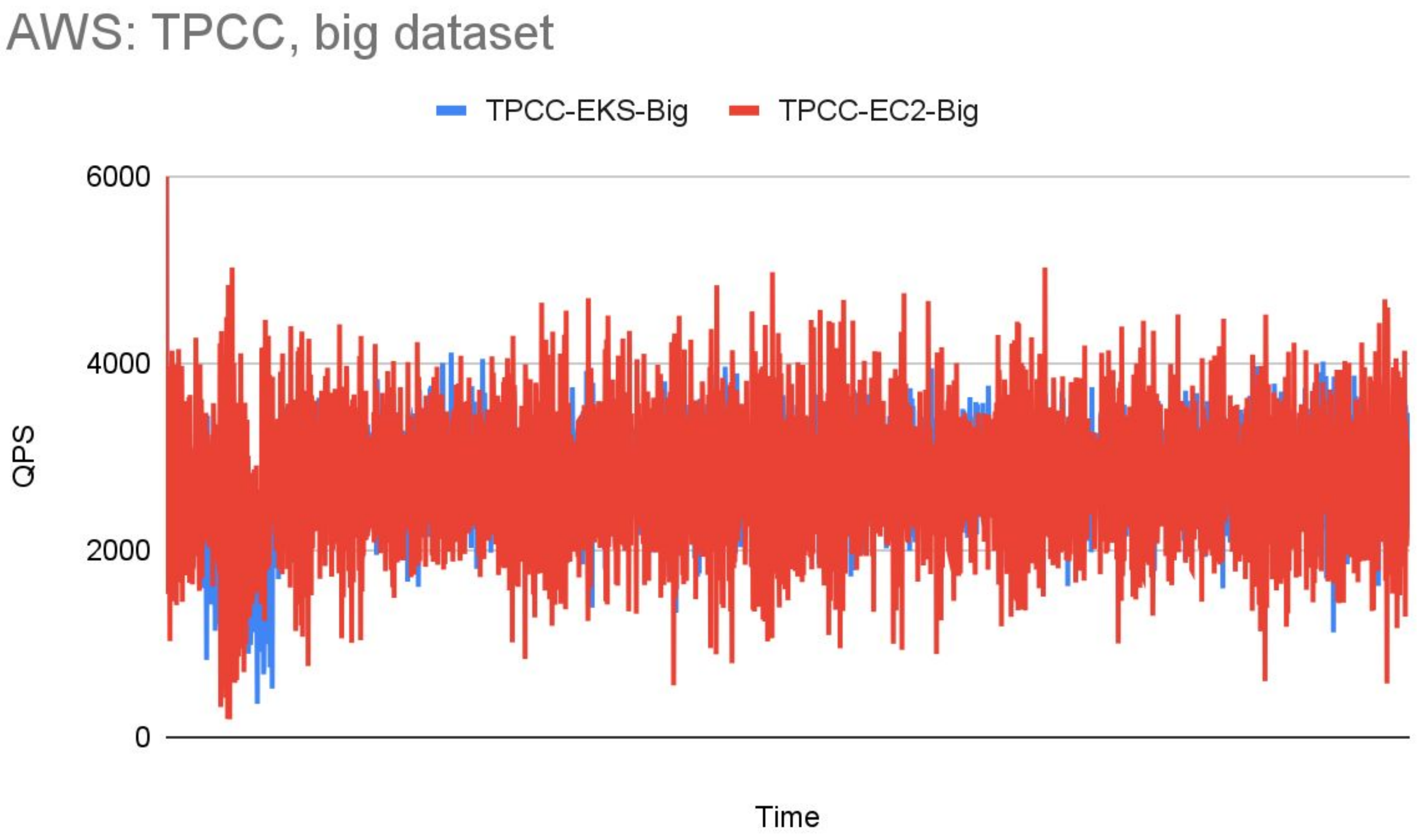


Name	Mean	Max	Min
🟩 user	20.83%	78.72%	0.40%
🟥 iowait	14.50%	84.00%	0.00%
🟨 system	3.71%	15.18%	0.18%

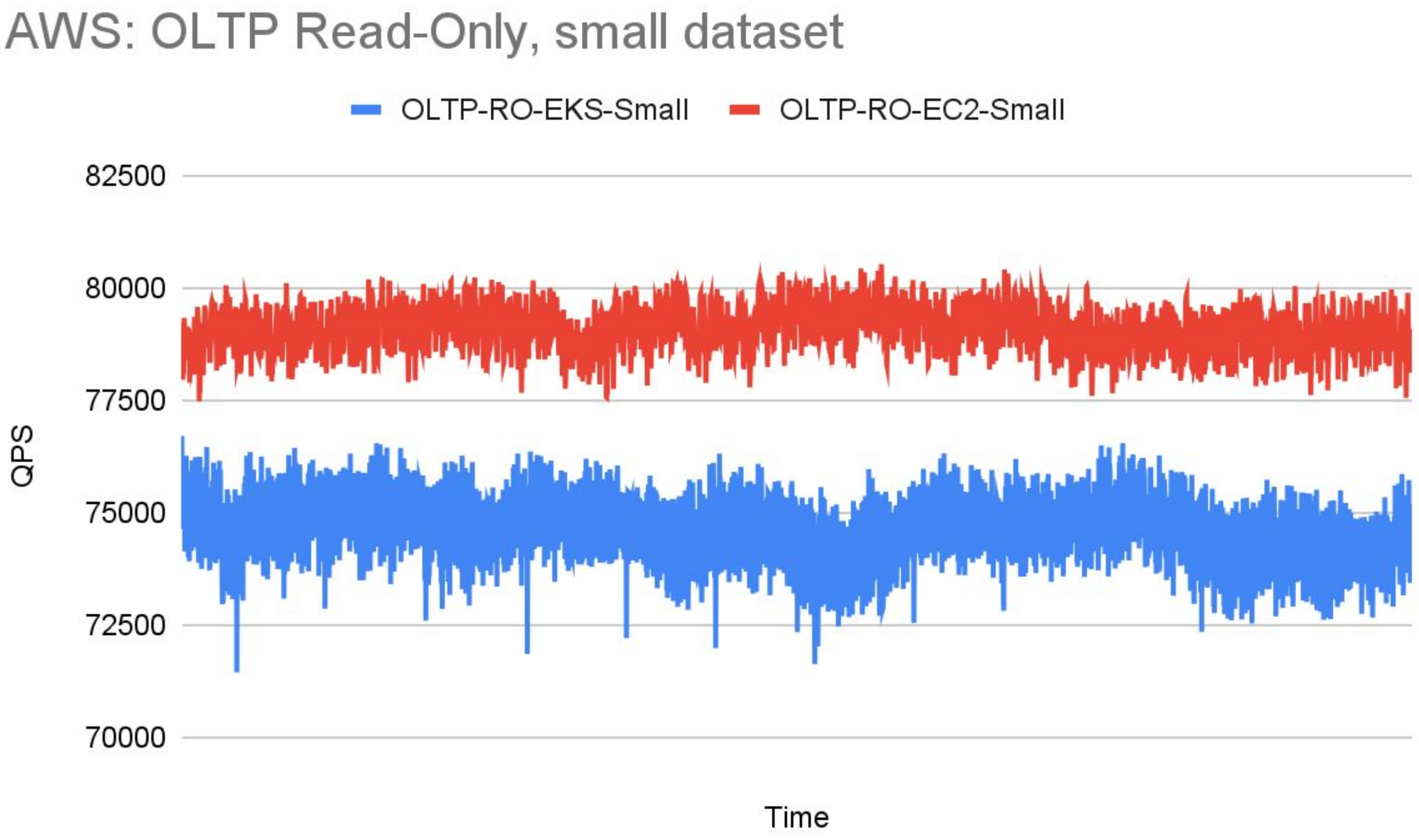
Part 1 - AWS


AWS: OLTP Read-Only, big dataset
OLTP-RO-EKS-Big
OLTP-RO-EC2-Big
85000
80000
75000
70000
QPS
Time

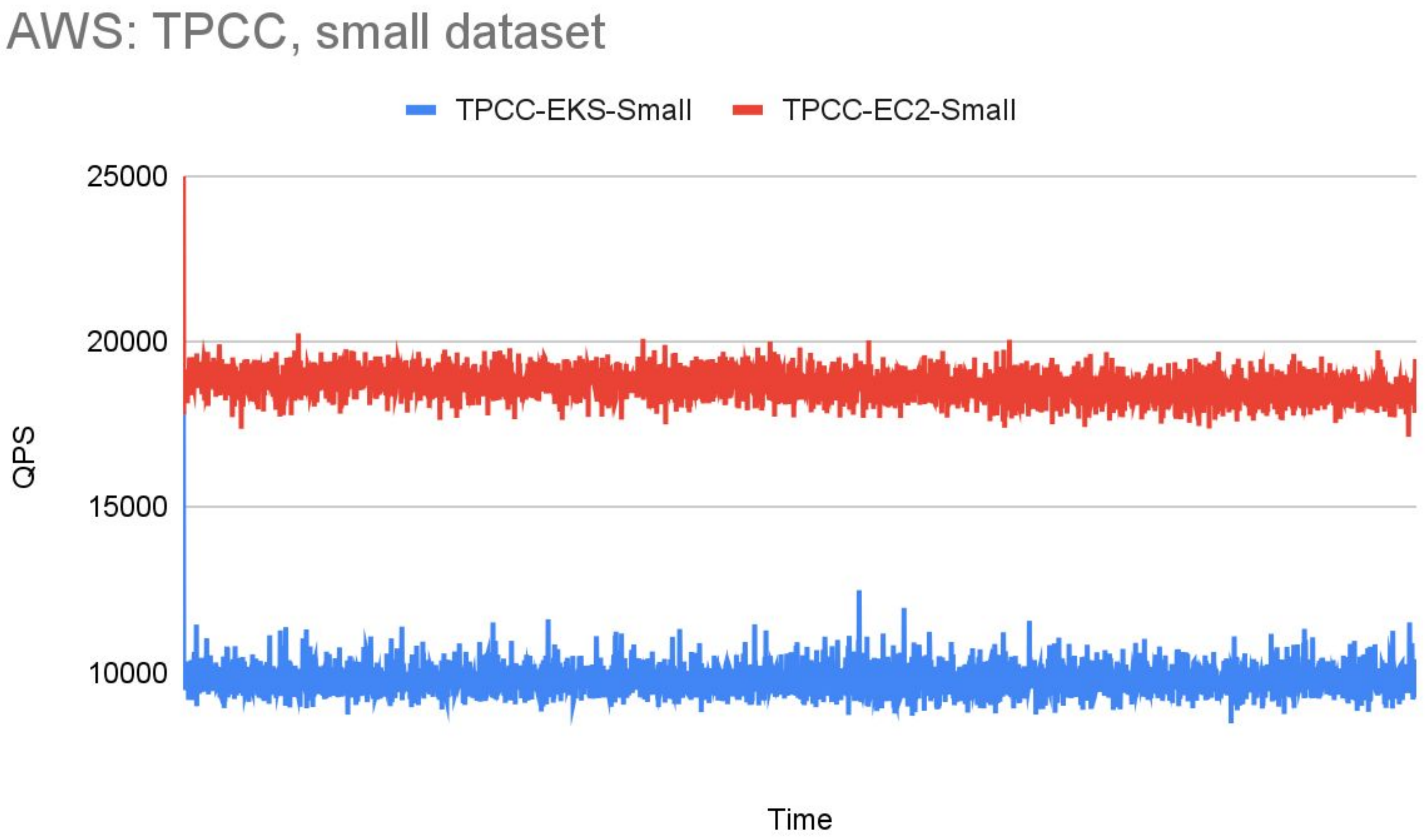
Part 1 - AWS


AWS: TPCC, big dataset
TPCC-EKS-Big
TPCC-EC2-Big
6000
4000
2000
0
QPS
Time

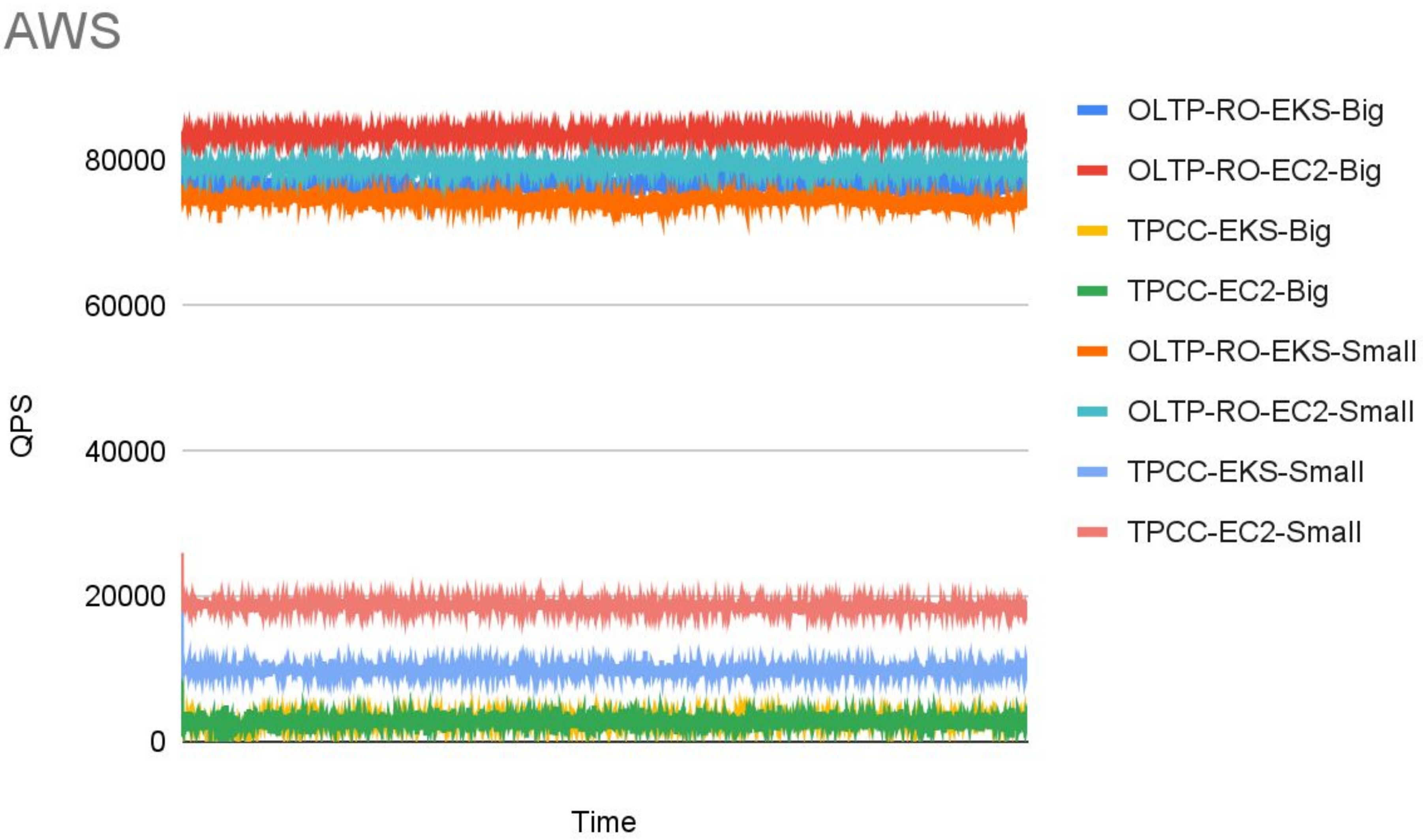
Part 1 - AWS


AWS: OLTP Read-Only, small dataset
OLTP-RO-EKS-Small
OLTP-RO-EC2-Small
82500
80000
77500
75000
72500
70000
QPS
Time

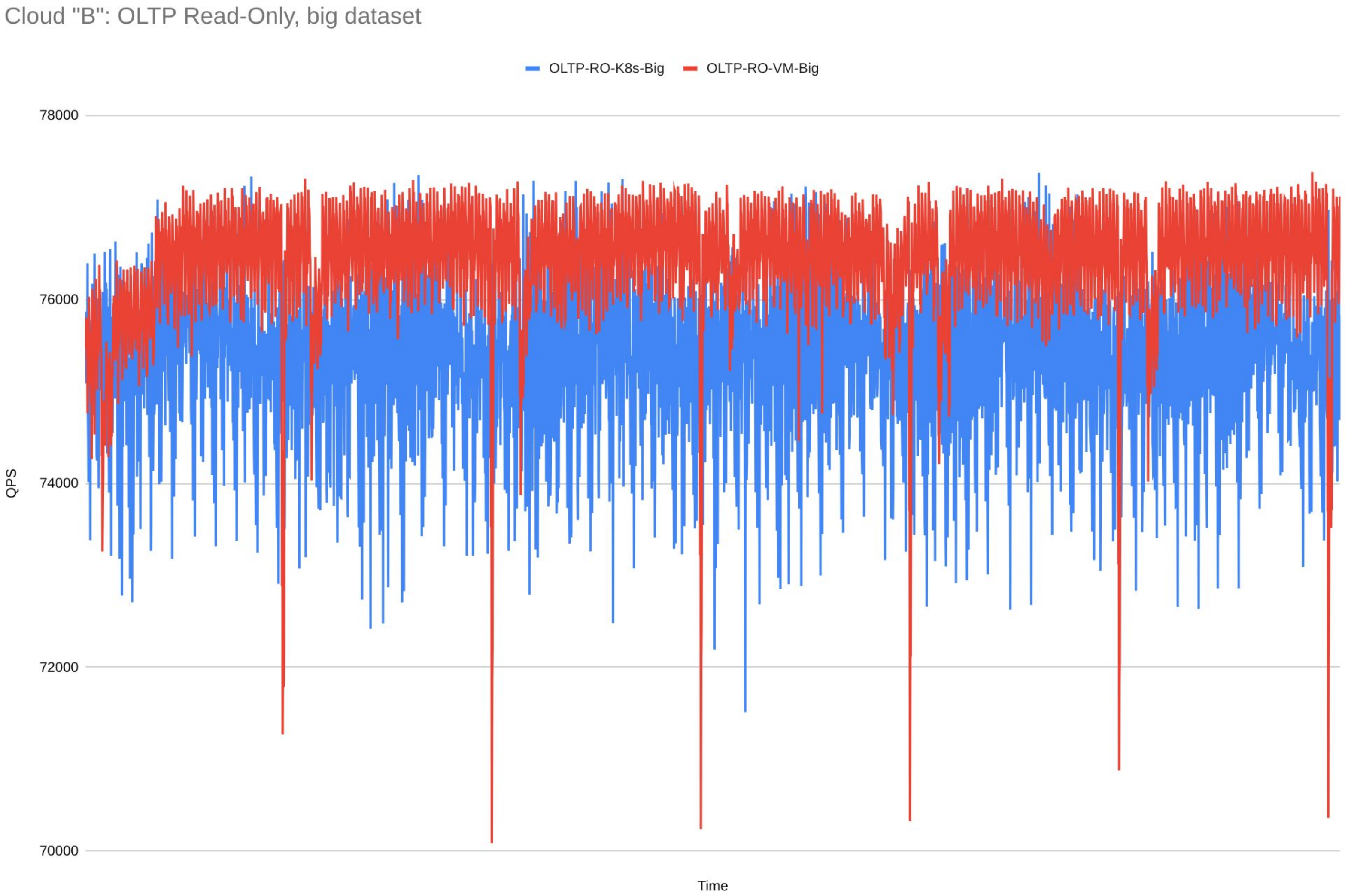
Part 1 - AWS


AWS: TPCC, small dataset
TPCC-EKS-Small
TPCC-EC2-Small
25000
20000
15000
10000
QPS
Time

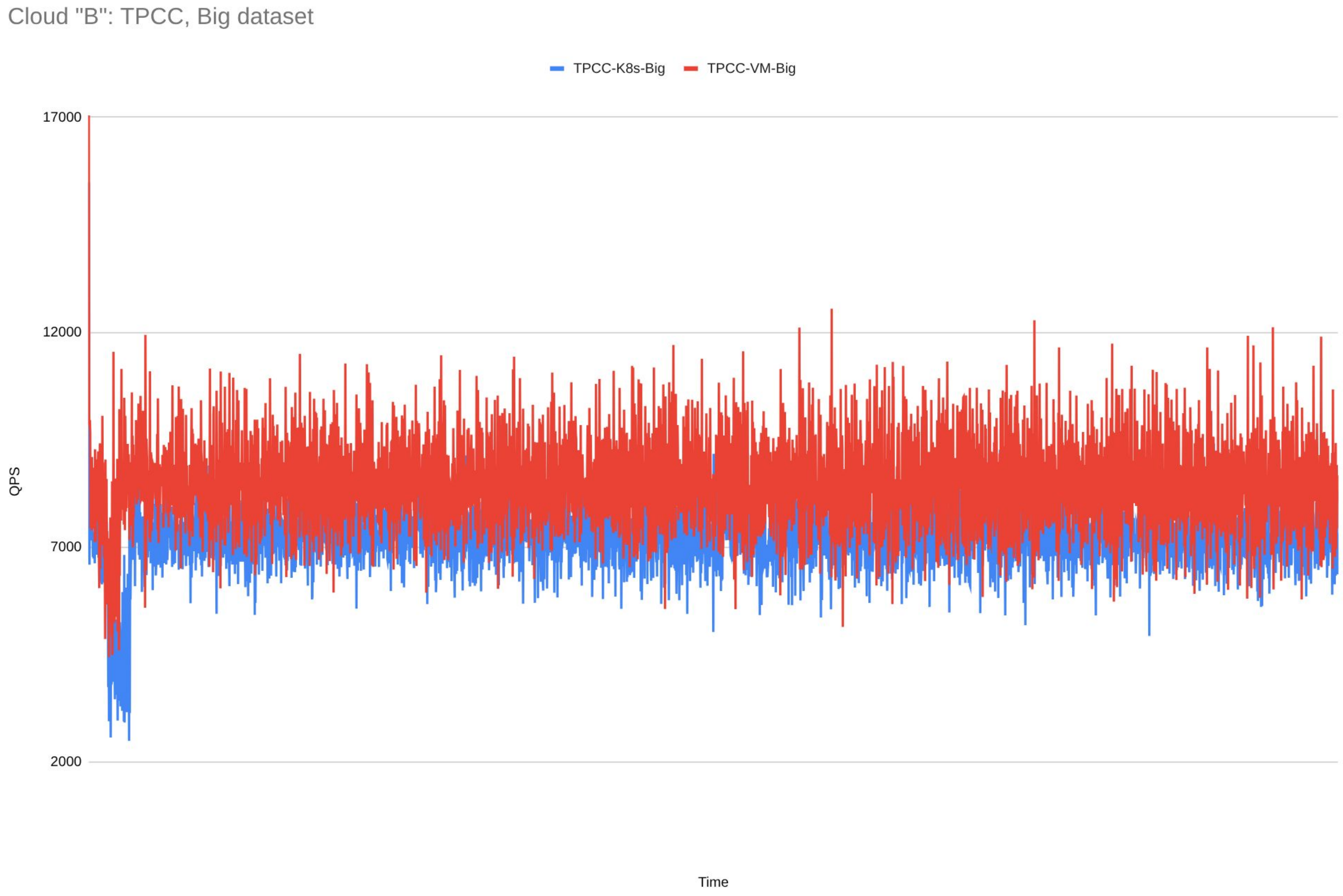
Part 1 - AWS


AWS
QPS
80000
60000
40000
20000
0
Time
OLTP-RO-EKS-Big
OLTP-RO-EC2-Big
TPCC-EKS-Big
TPCC-EC2-Big
OLTP-RO-EKS-Small
OLTP-RO-EC2-Small
TPCC-EKS-Small
TPCC-EC2-Small

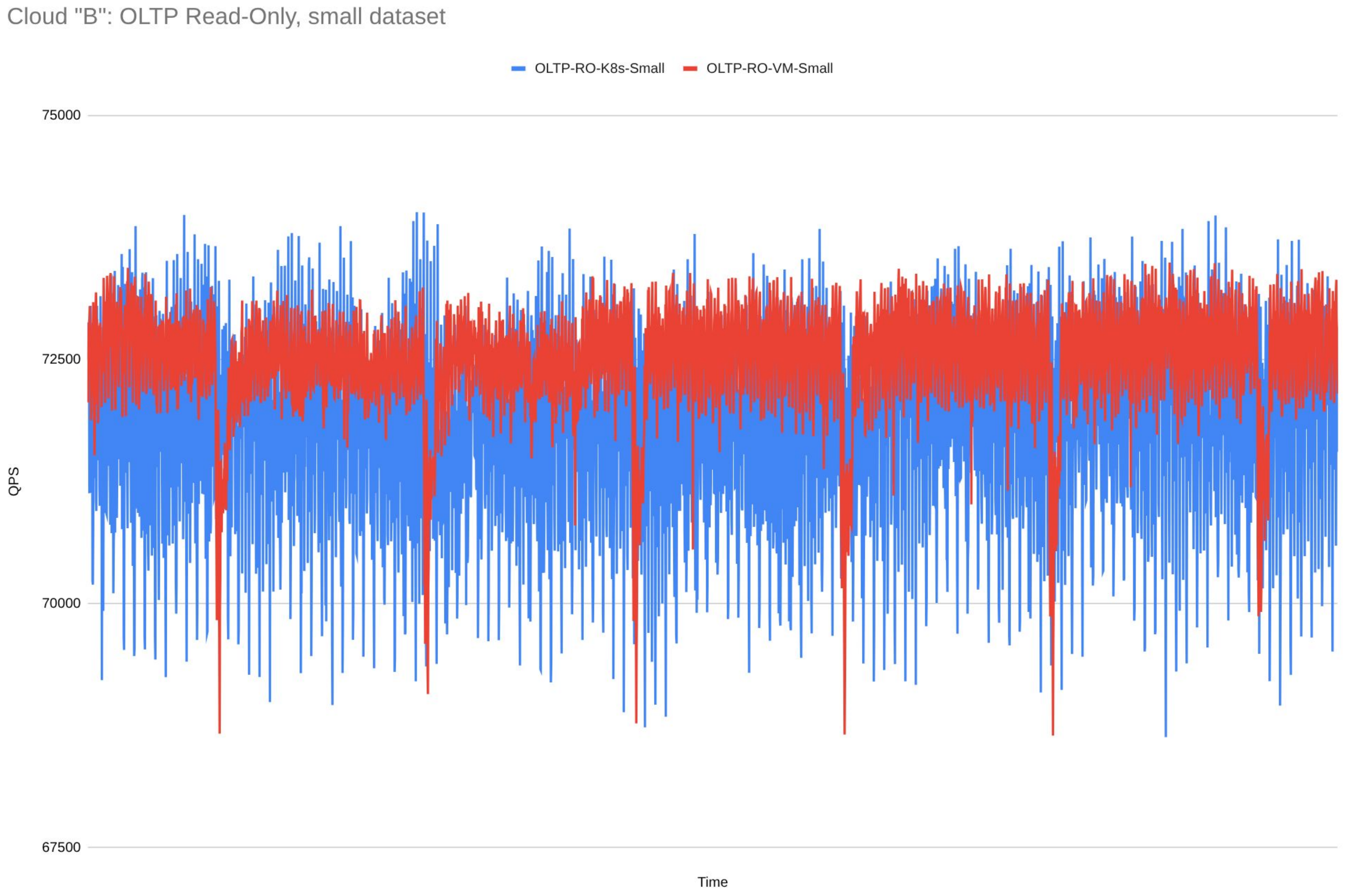
Part 1 - Cloud "B"



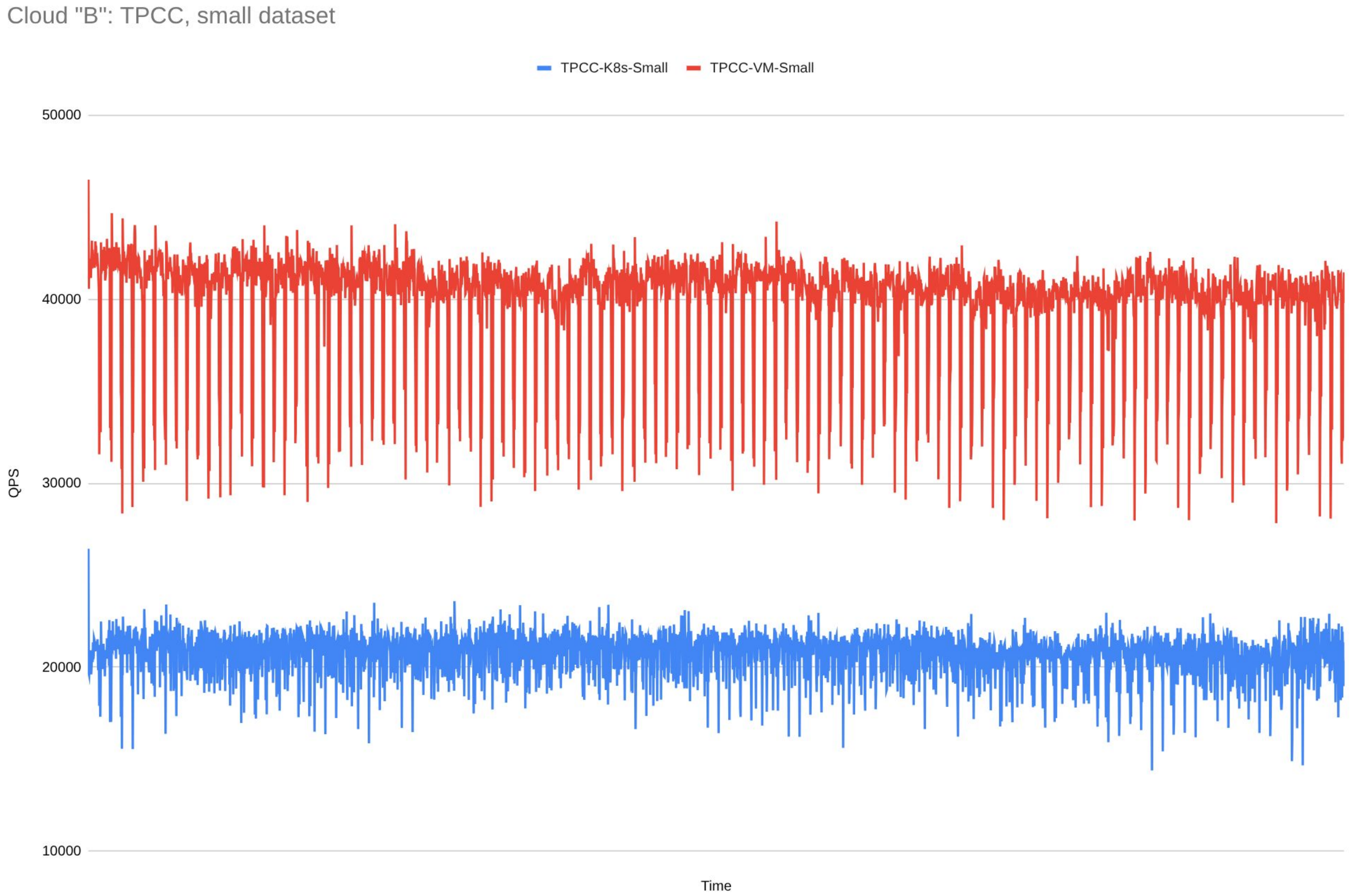
Part 1 - Cloud "B"



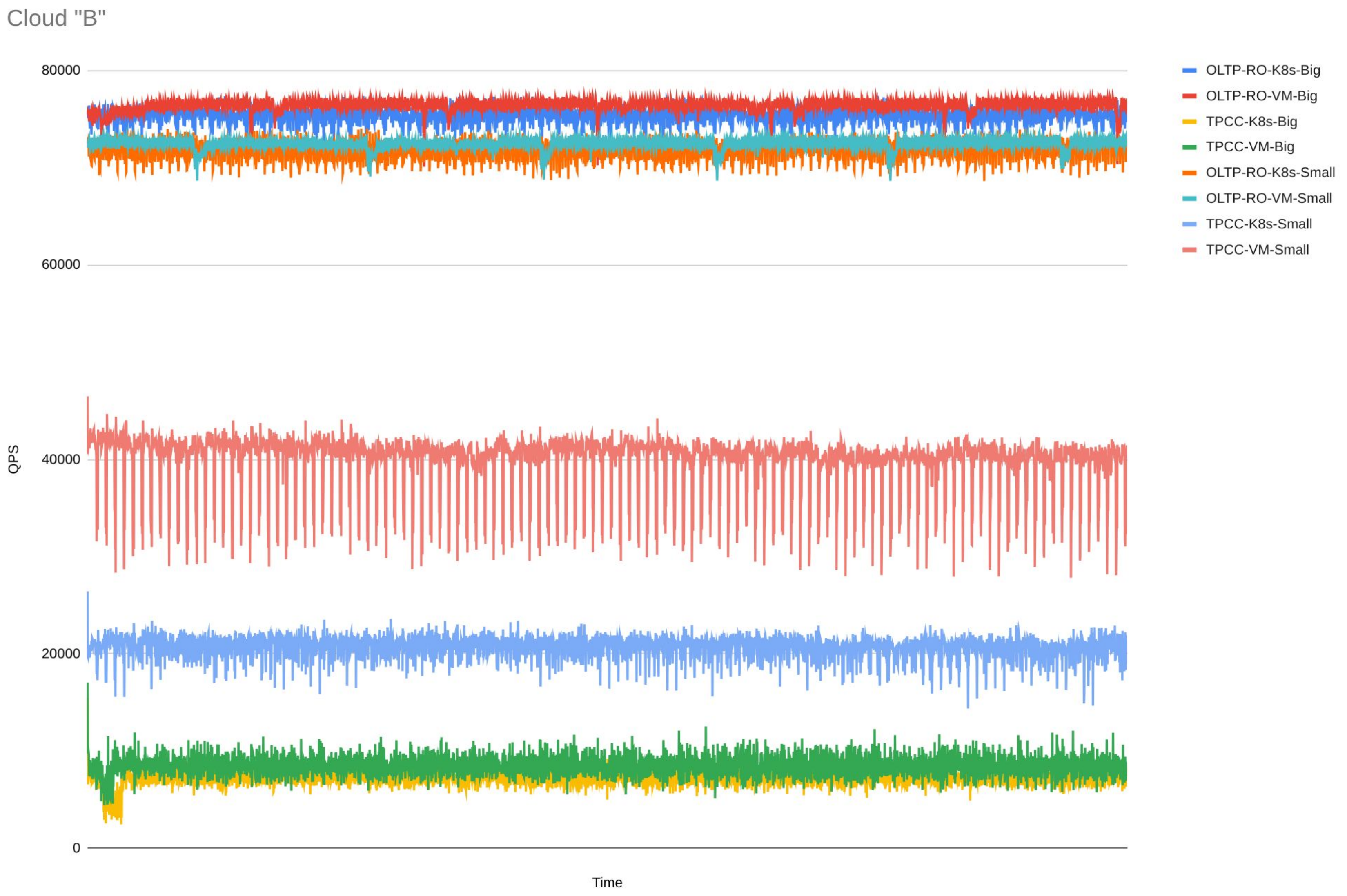
Part 1 - Cloud "B"



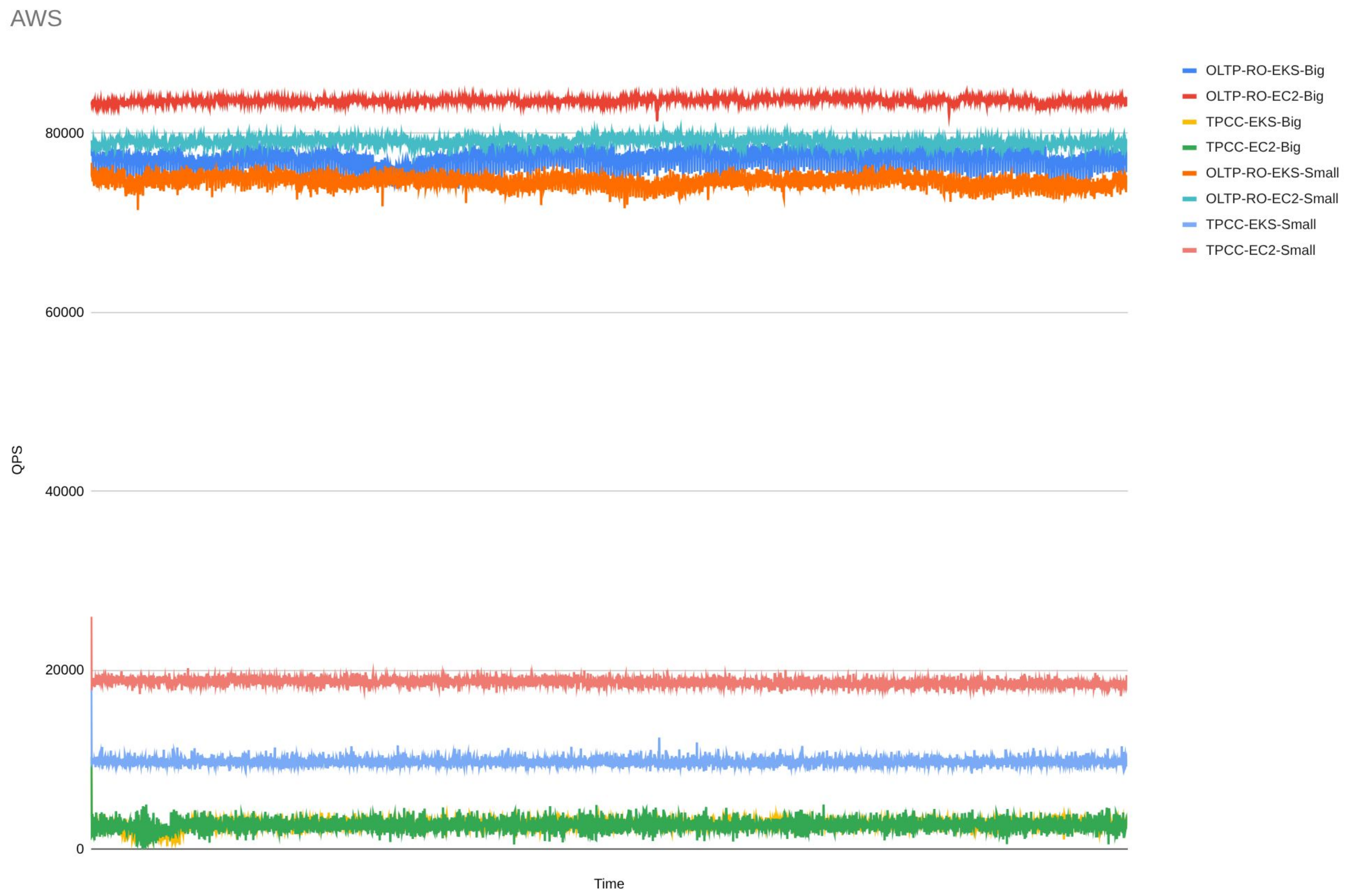
Part 1 - Cloud "B"



Part 1 - Cloud "B"



Part 1 - AWS

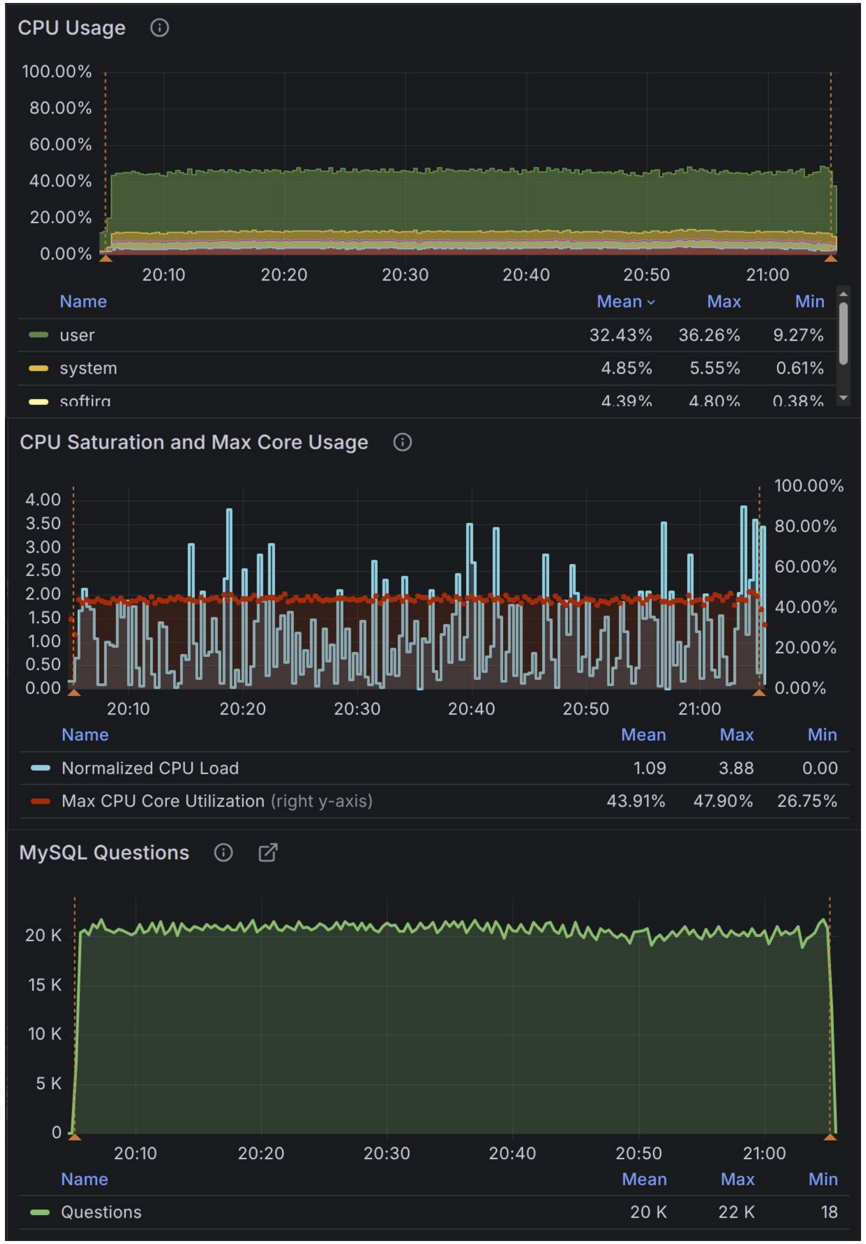

AWS
80000
60000
40000
20000
0
QPS
Time
OLTP-RO-EKS-Big
OLTP-RO-EC2-Big
TPCC-EKS-Big
TPCC-EC2-Big
OLTP-RO-EKS-Small
OLTP-RO-EC2-Small
TPCC-EKS-Small
TPCC-EC2-Small

Part 1 - Summary

	OLTP-RO-Big	TPCC-Big	OLTP-RO-Small	TPCC-Small
EKS	77303.06	2818.91	74635.4475	9811.6425
EC2	83652.8875	2751.245	79091.205	18691.4325
B-K8s	75349.675	7231.8375	71790.2625	20665.9375
B-VMs	76402.52	8469.1725	72490.96	39764.5325
EKS/EC2	0.92	1.02	0.94	0.52
B:K8s/VMs	0.99	0.85	0.99	0.52

Part 2 – TPCC, small dataset: Cloud B

K8s



CPU Usage

Name	Mean ⌄	Max	Min
user	32.43%	36.26%	9.27%
system	4.85%	5.55%	0.61%
softirq	4.39%	4.80%	0.38%

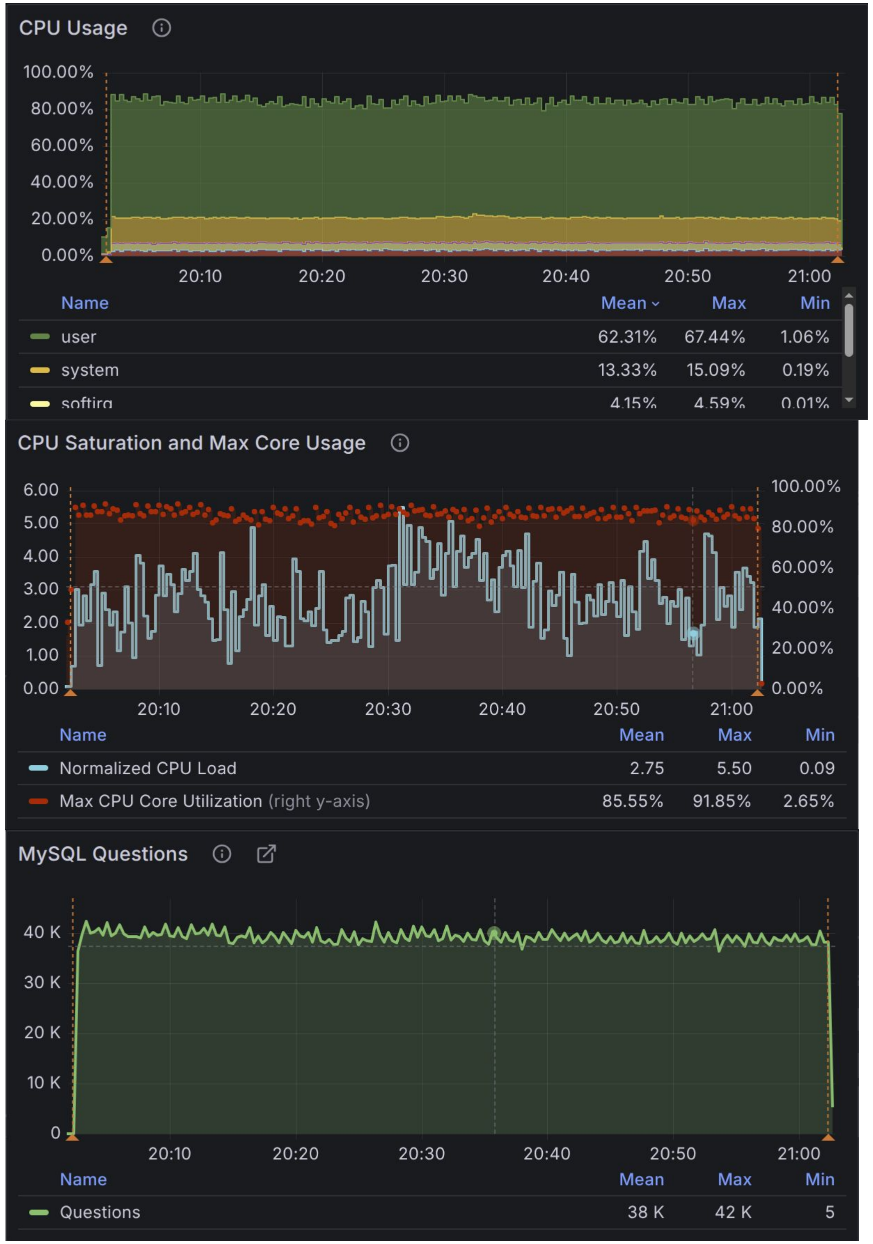
CPU Saturation and Max Core Usage

Name	Mean	Max	Min
Normalized CPU Load	1.09	3.88	0.00
Max CPU Core Utilization (right y-axis)	43.91%	47.90%	26.75%

MySQL Questions

Name	Mean	Max	Min
Questions	20 K	22 K	18

VMs



CPU Usage

Name	Mean ⌄	Max	Min
user	62.31%	67.44%	1.06%
system	13.33%	15.09%	0.19%
softirq	4.15%	4.59%	0.01%

CPU Saturation and Max Core Usage

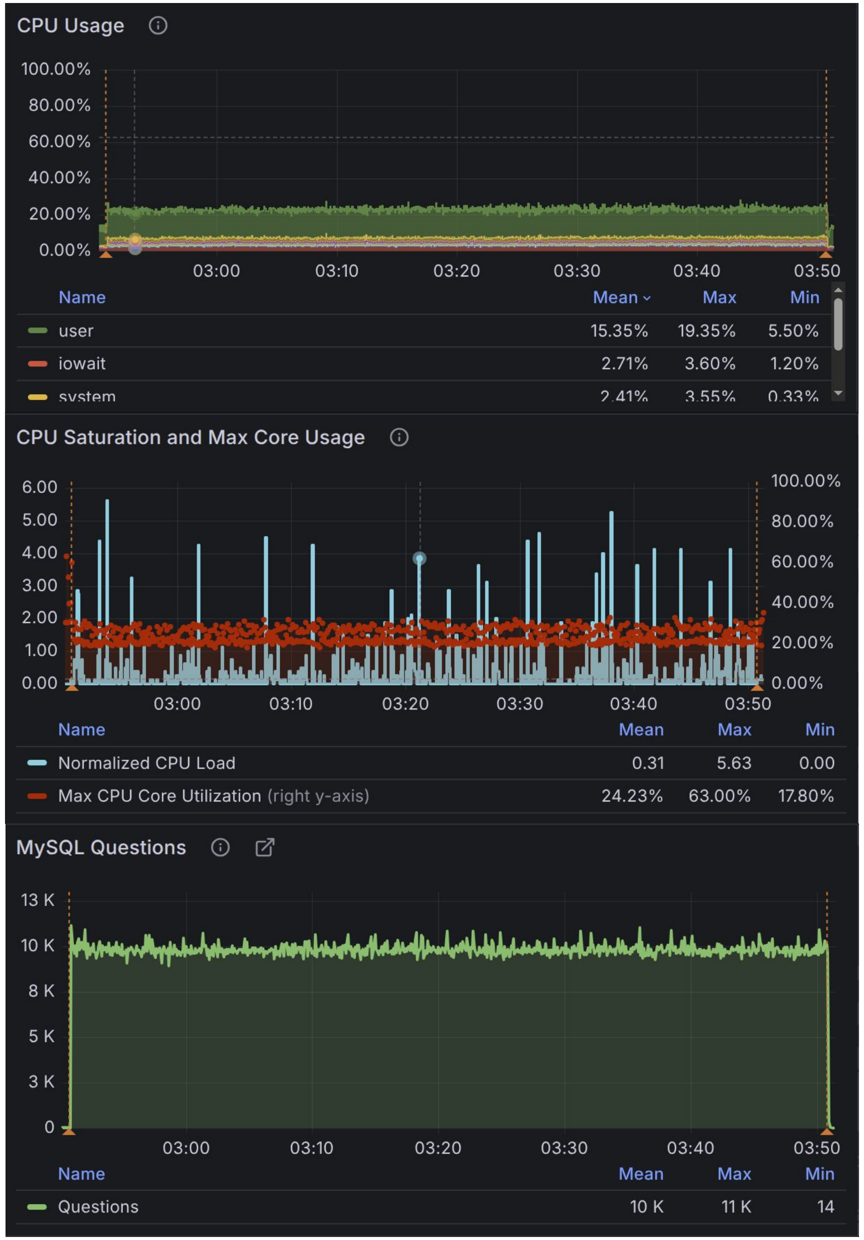
Name	Mean	Max	Min
Normalized CPU Load	2.75	5.50	0.09
Max CPU Core Utilization (right y-axis)	85.55%	91.85%	2.65%

MySQL Questions

Name	Mean	Max	Min
Questions	38 K	42 K	5

Part 2 – TPCC, small dataset: AWS

EKS



CPU Usage

Name	Mean	Max	Min
🟩 user	15.35%	19.35%	5.50%
🟥 iowait	2.71%	3.60%	1.20%
🟨 system	2.41%	3.55%	0.33%

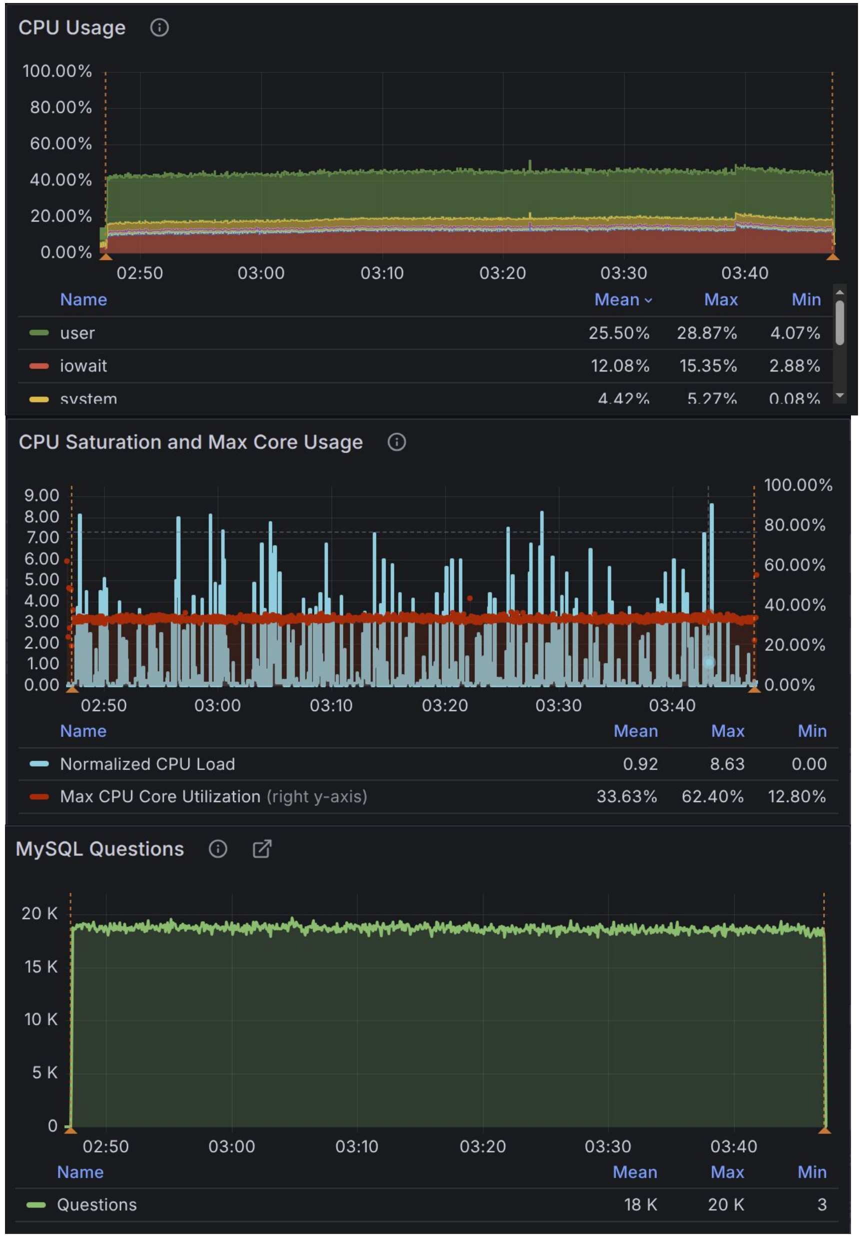
CPU Saturation and Max Core Usage

Name	Mean	Max	Min
Normalized CPU Load	0.31	5.63	0.00
Max CPU Core Utilization (right y-axis)	24.23%	63.00%	17.80%

MySQL Questions

Name	Mean	Max	Min
Questions	10 K	11 K	14

EC2



CPU Usage

Name	Mean	Max	Min
🟩 user	25.50%	28.87%	4.07%
🟥 iowait	12.08%	15.35%	2.88%
🟨 system	4.42%	5.27%	0.08%

CPU Saturation and Max Core Usage

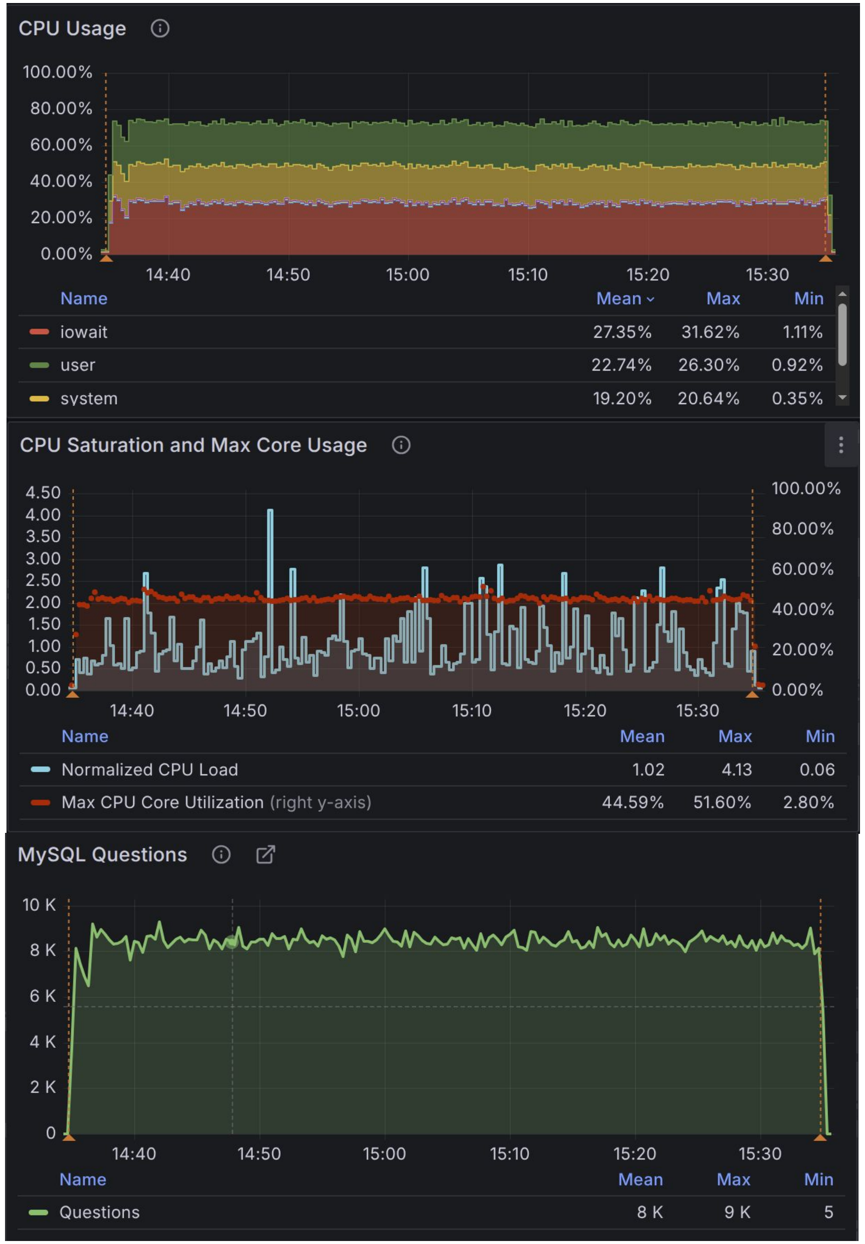
Name	Mean	Max	Min
Normalized CPU Load	0.92	8.63	0.00
Max CPU Core Utilization (right y-axis)	33.63%	62.40%	12.80%

MySQL Questions

Name	Mean	Max	Min
Questions	18 K	20 K	3

Part 2 – TPCC, big dataset: Cloud B

K8s



CPU Usage

Name	Mean	Max	Min
iowait	27.35%	31.62%	1.11%
user	22.74%	26.30%	0.92%
system	19.20%	20.64%	0.35%

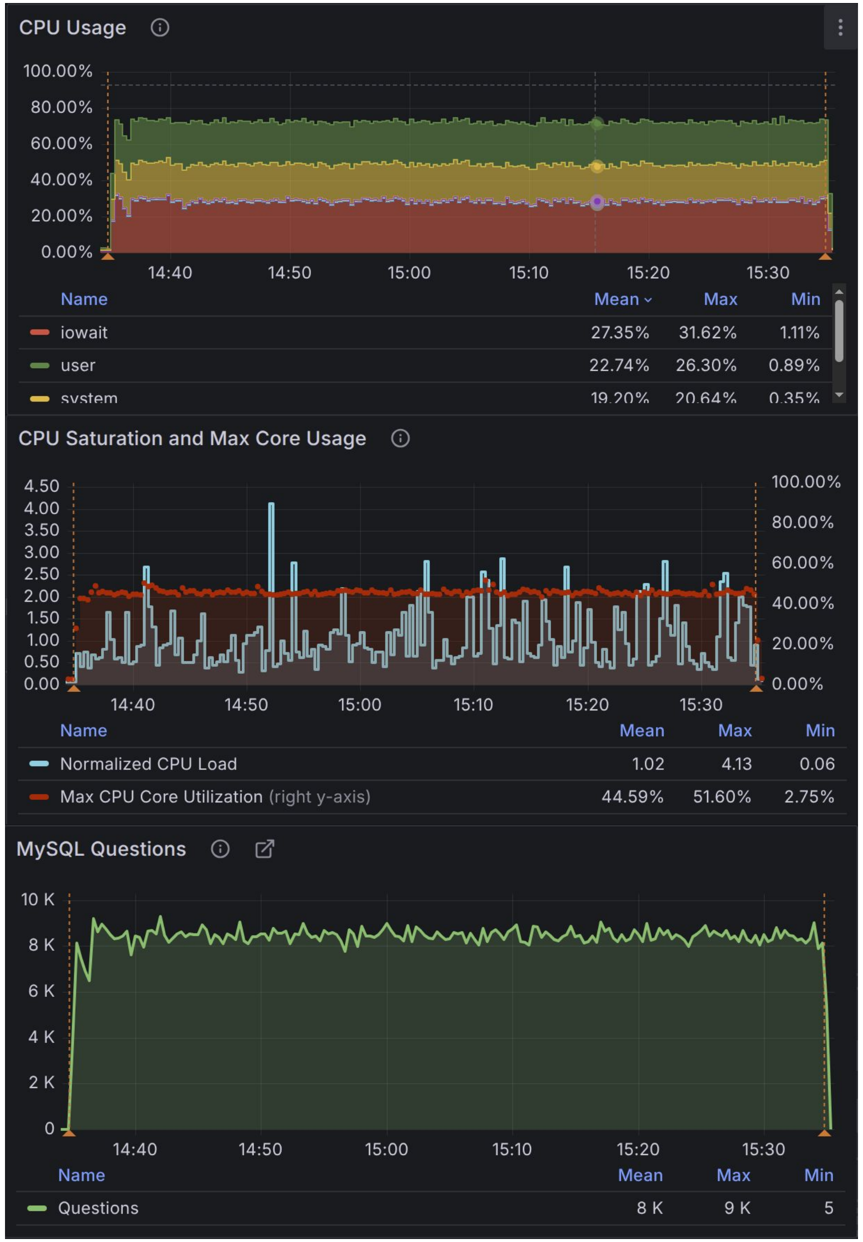
CPU Saturation and Max Core Usage

Name	Mean	Max	Min
Normalized CPU Load	1.02	4.13	0.06
Max CPU Core Utilization (right y-axis)	44.59%	51.60%	2.80%

MySQL Questions

Name	Mean	Max	Min
Questions	8 K	9 K	5

VMs



CPU Usage

Name	Mean	Max	Min
iowait	27.35%	31.62%	1.11%
user	22.74%	26.30%	0.89%
system	19.20%	20.64%	0.35%

CPU Saturation and Max Core Usage

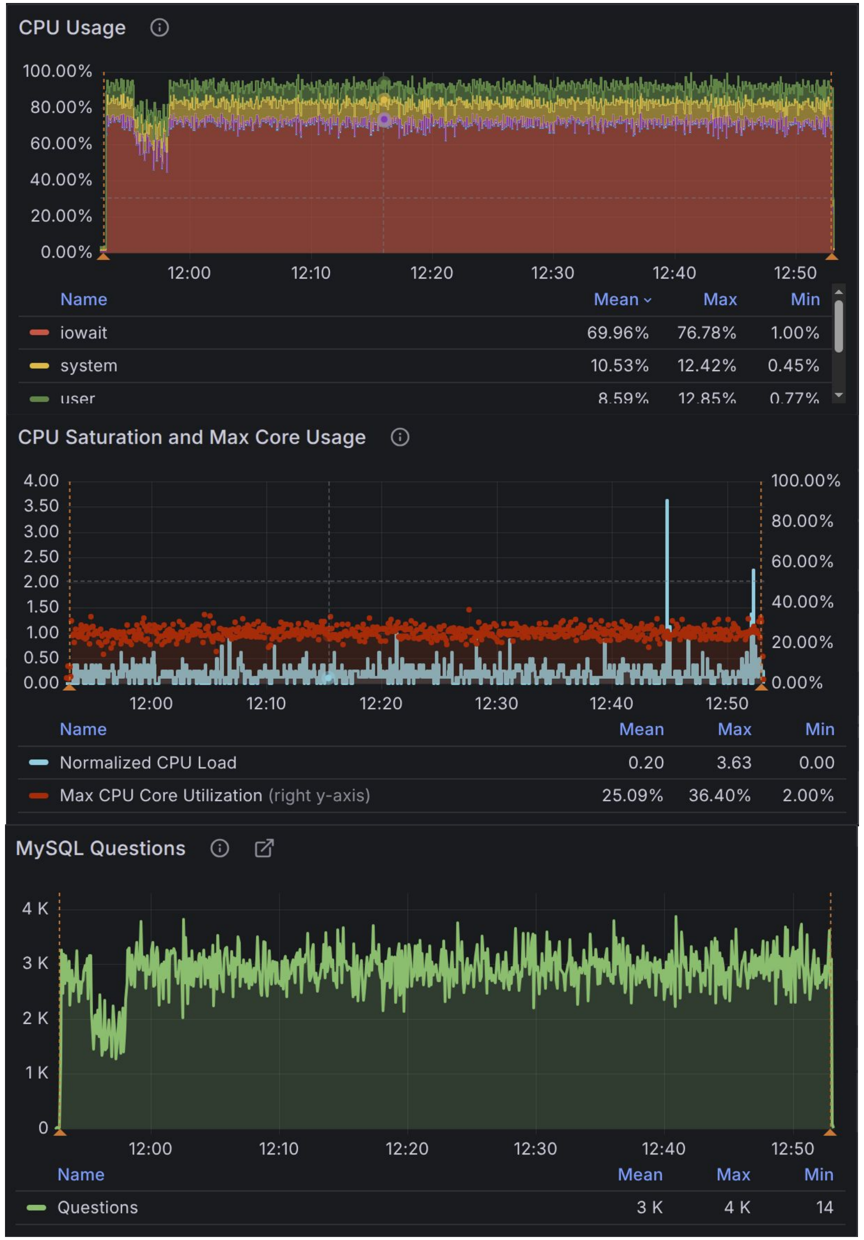
Name	Mean	Max	Min
Normalized CPU Load	1.02	4.13	0.06
Max CPU Core Utilization (right y-axis)	44.59%	51.60%	2.75%

MySQL Questions

Name	Mean	Max	Min
Questions	8 K	9 K	5

Part 2 – TPCC, big dataset: AWS

EKS



CPU Usage

Name	Mean	Max	Min
iowait	69.96%	76.78%	1.00%
system	10.53%	12.42%	0.45%
user	8.59%	12.85%	0.77%

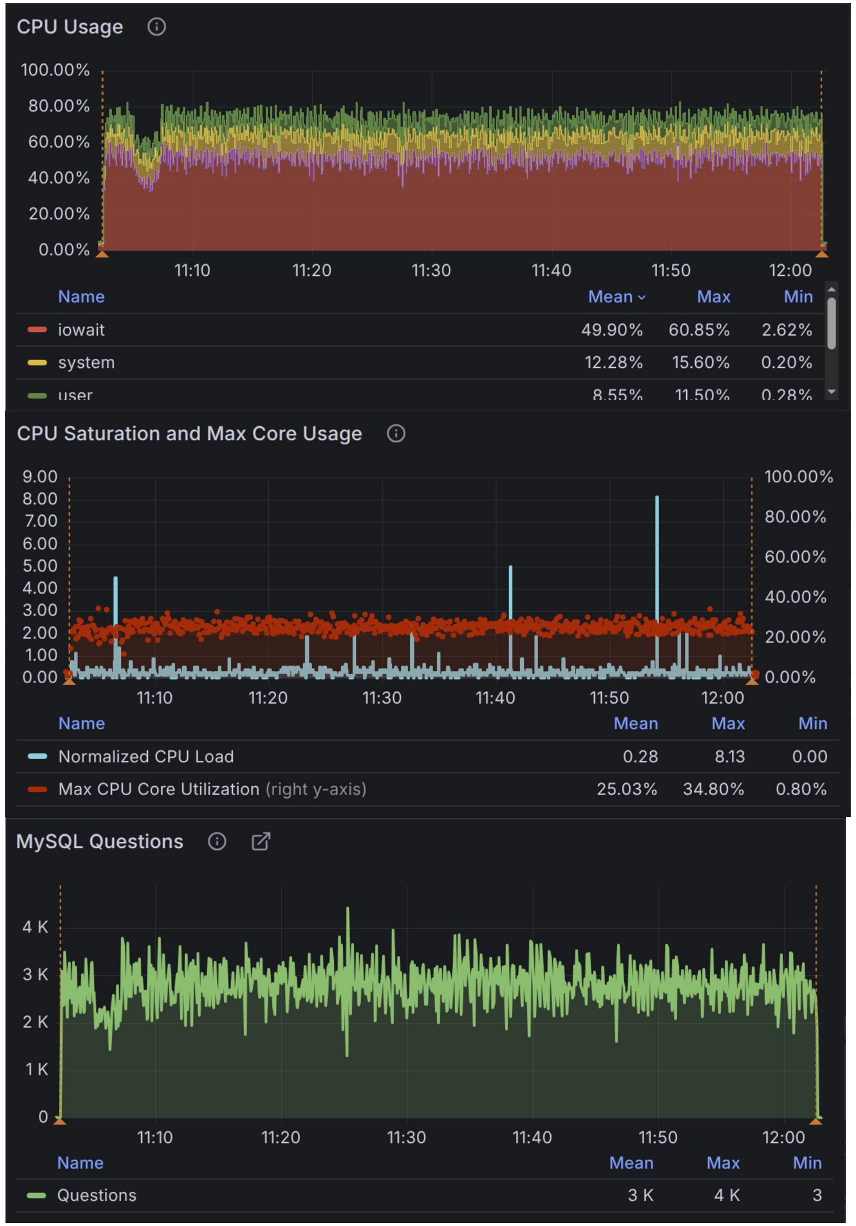
CPU Saturation and Max Core Usage

Name	Mean	Max	Min
Normalized CPU Load	0.20	3.63	0.00
Max CPU Core Utilization (right y-axis)	25.09%	36.40%	2.00%

MySQL Questions

Name	Mean	Max	Min
Questions	3 K	4 K	14

EC2



CPU Usage

Name	Mean	Max	Min
iowait	49.90%	60.85%	2.62%
system	12.28%	15.60%	0.20%
user	8.55%	11.50%	0.28%

CPU Saturation and Max Core Usage

Name	Mean	Max	Min
Normalized CPU Load	0.28	8.13	0.00
Max CPU Core Utilization (right y-axis)	25.03%	34.80%	0.80%

MySQL Questions

Name	Mean	Max	Min
Questions	3 K	4 K	3

K8s: ressources requests & limits

percona-xtradb-cluster-operator/deploy/cr.yaml:

```
spec:
  pxc:
   resources:
      requests:
        memory: 1G
        cpu: 600m
#        ephemeral-storage: 1G
#      limits:
#        memory: 1G
#        cpu: "1"
#        ephemeral-storage: 1G
```

Thank You!

fernando.laudares@percona.com

EKS: deploying a new cluster with the operator

```
eksctl create cluster --name nando-1 --region us-west-2 --without-nodegroup

aws eks update-kubeconfig --name nando-1 --region us-west-2

eksctl utils associate-iam-oidc-provider --region=us-west-2 --cluster=nando-1 --approve

eksctl create iamserviceaccount --name ebs-csi-controller-sa --cluster nando-1 --role-name NandoAmazonEKS_EBS_CSI_DriverRole
--role-only --attach-policy-arn arn:aws:iam::aws:policy/service-role/AmazonEBSCSIDriverPolicy --approve --region us-west-2
--namespace kube-system

eksctl create addon --name aws-ebs-csi-driver --cluster nando-1 --service-account-role-arn
arn:aws:iam::686800432451:role/NandoAmazonEKS_EBS_CSI_DriverRole --force --region us-west-2

eksctl create nodegroup --cluster nando-1 --region us-west-2 --name nando-1-data --node-type m7i.2xlarge --nodes 3 --nodes-min 2
--nodes-max4 --node-volume-size 100 --node-volume-type gp3

kubectl create namespace pxc

kubectl apply -n pxc --server-side -f deploy/bundle.yaml

# Make the necessary changes to cr.yaml
kubectl apply -n pxc -f deploy/cr.yam

# Sysbench
eksctl create nodegroup --cluster nando-1 --region us-west-2 --name nando-1-extra --node-type m7i.2xlarge --nodes 1
--nodes-min 1 --nodes-max 2 --node-volume-size 100 --node-volume-type gp3
```