Troubleshooting Toolkit for PostgreSQL DBA’s

Lalit choudhary

Why DBA needs testing and monitoring tools ?

- Can't test on production.
- Experiments (new version/feature/extension test/POC).
- Analysing impact/change (before implementing to Prod).
- Historic data analysis and patterns to optimize postgresql configuration.
- Real time data analysis sessions , locks, load, resource utilization.

© Copyright 2024 Percona® LLC. All rights reserved

Obstacles for testing setup

- Time consuming process get new test servers.
- Approvals for certain software installation.
- Access issue and approvals for security.
- Cost of VM/licencing (cloud/on-premises) $$$
- Time consuming setups, like pg install, replication , patroni cluster setup etc

Available platforms

- Utilizing open source tools, Containerization/virtualization tools.
- Virtualbox : Time consuming (VMWare platform, Oracle virtual box).
- LXD, docker : Less time consuming but need extra installation steps.
- Docker could be also used with systemd (build own image).

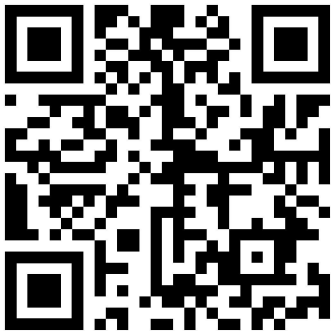
© Copyright 2024 Percona® LLC. All rights reserved

Anydbver Tool [https://github.com/ihanick/anydbver]

- Configuring MySQL, Percona MySQL/Postgresql/Mongo, MongoDB with ansible scripts. Running multi-node replication clusters in Docker, LXD and Kubernetes.
- Open Source tool , Easy to use

  PostgreSQL setup options:

  - Standalone PostgreSQL server
  - Postgresql Replication (Streaming Replication)
  - Patroni Cluster (**standby cluster support**)
  - K8 postgresql operator



© Copyright 2024 Percona® LLC. All rights reserved

Anydbver [setup]

```
cd ~/.local/bin

wget -O -
https://github.com/ihanick/anydbver/releases/download/v0.1.21/anydbver_Li
nux_x86_64.tar.gz | tar xz anydbver

export PATH=~/.local/bin:$PATH
anydbver update
```

List of commands:

```
anydbver test list | grep -i pg
```

© Copyright 2024 Percona® LLC. All rights reserved

Anydbver [Standalone PostgreSQL server setup]

```
anydbver  deploy os:el7 pg:15.5
anydbver  deploy os:el8 pg:14
anydbver  deploy node0 pg:latest
```

$ anydbver deploy os:el8 pg:17 -n pg_single

$ anydbver exec node0 -n pg_single
[root@node0 /]# psql
psql (17.4)
Type "help" for help.

postgres=#

© Copyright 2024 Percona® LLC. All rights reserved

Anydbver [Postgresql Replication (Streaming Replication)]

```
$ anydbver deploy node0 pg:latest,wal=logical node1 pg:latest,primary=node0,wal=logical
```

```
$ anydbver deploy node0 ppg:latest  node1 ppg:latest,primary=node0 -n pg_replication
```

```
TASK [postgresql : Start database with systemd] ********************************
changed: [pg_replication-mgogo-node1]

PLAY RECAP *********************************************************************
pg_replication-mgogo-node0 : ok=32   changed=16   unreachable=0    failed=0    skipped=65   rescued=0    ignored=0
pg_replication-mgogo-node1 : ok=32   changed=16   unreachable=0    failed=0    skipped=65   rescued=0    ignored=0
```

```
$ anydbver exec node1 -n pg_replication
[root@node1 /]# psql
psql (16.8 - Percona Distribution)
Type "help" for help.

postgres=# select pg_is_in_recovery();
 pg_is_in_recovery
-------------------
 t
(1 row)
```

Anydbver [[Patroni Cluster]]

[Patroni Cluster]

$ anydbver deploy pg:17 patroni node1 pg:17,master=node0 patroni:17,master=node0 node2 pg:17,master=node0 patroni:master=node0 -n pg_ha

```
TASK [patroni : Start etcd with systemd] ************************************
changed: [pg_ha-mgogo-node1]

TASK [patroni : Setup patroni] **********************************************
changed: [pg_ha-mgogo-node1]

PLAY RECAP ******************************************************************
pg_ha-mgogo-node0          : ok=39   changed=21   unreachable=0    failed=0    skipped=65   rescued=0    ignored=0
pg_ha-mgogo-node1          : ok=39   changed=21   unreachable=0    failed=0    skipped=65   rescued=0    ignored=0
pg_ha-mgogo-node2          : ok=39   changed=21   unreachable=0    failed=0    skipped=65   rescued=0    ignored=0
```

© Copyright 2024 Percona® LLC. All rights reserved

Anydbver [[Patroni Standby cluster]]

$ anydbver deploy ppg:16 patroni:cluster=cluster11 node1 ppg:16,master=node0 patroni:master=node0,cluster=cluster11 node2 ppg:16,master=node0 patroni:master=node0,cluster=cluster11 node3 ppg:16 patroni:standby=node0,cluster=cluster12 node4 ppg:16,master=node3 patroni:master=node3,cluster=cluster12

- sub-command: patroni:standby
- We can mention the custom cluster name

© Copyright 2024 Percona® LLC. All rights reserved

Anydbver [K8 operator]

https://operatorhub.io/?category=Database

```
$ anydbver deploy k3d:latest
$ anydbver deploy k3d:latest k8s-pg:2.6.0
```

```
$ kubectl get pods -n pgo
NAME                                        READY   STATUS      RESTARTS   AGE
percona-postgresql-operator-657f794b5b-z9rcb   1/1     Running     0          16m
cluster1-pgbouncer-847fdc6487-f4qv8         2/2     Running     0          16m
cluster1-pgbouncer-847fdc6487-kh5lw         2/2     Running     0          16m
cluster1-pgbouncer-847fdc6487-z5fwz         2/2     Running     0          16m
cluster1-repo-host-0                        2/2     Running     0          16m
cluster1-backup-c6nc-tg68w                  0/1     Completed   0          7m13s
cluster1-instance1-qv9z-0                   4/4     Running     0          16m
cluster1-instance1-zjdq-0                   4/4     Running     0          16m
cluster1-instance1-6hxd-0                   4/4     Running     0          16m
```

© Copyright 2024 Percona® LLC. All rights reserved

Anydbver [other options]

`namespace` usage for multiple setups:

$ ./anydbver --namespace pgrep_setup deploy pg:15,docker-image

$ ./anydbver --namespace pgrep_setup destroy

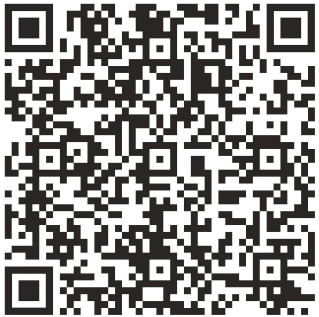
$ anydbver destroy

Using existing k8 setup:

PROVIDER= kubectl

© Copyright 2024 Percona® LLC. All rights reserved

PMM Monitoring [Graphs]



Real time and Historic monitoring capabilities.

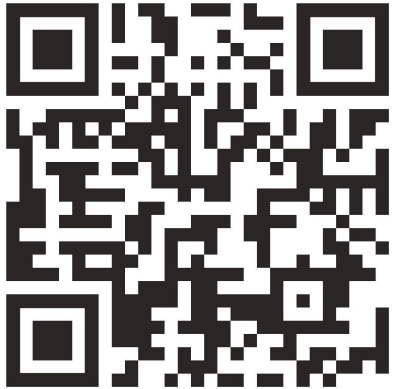
- PostgreSQL and OS monitoring
- Query Analytics: QAN (pg_stat_statements, pg_stat_monitor extension)
- Peak hours vs non-peak hours
- Resource usage over period of the time
- Other features like Alerting, custom graphs , etc

https://pmmdemo.percona.com/

© Copyright 2024 Percona® LLC. All rights reserved

Pg_gather [https://github.com/jobinau/pg_gather]

- Scan and collect the minimal amount of data needed to identify potential problems in your PostgreSQL database, and then generate an analysis report using that data.
- (Developed and Maintained by Jobin Augustine)



  - Everything is SQL-only, leveraging the built-in features of psql, the command-line utility of PostgreSQL.
  - Supported PostgreSQL Versions : 10, 11, 12, 13, 14, 15, 16, & 17

© Copyright 2024 Percona® LLC. All rights reserved

Pg_gather

This project provides three SQL scripts for users:

- **gather.sql**: Gathers performance and configuration data from PostgreSQL databases.
- **gather_schema.sql:** Importing collected data
- **gather_report.sql**: Analyzes the collected data and generates detailed HTML reports.

An alternative is to use the `generate_report.sh` script, which can spin up a PostgreSQL Docker container and automate the entire process.

© Copyright 2024 Percona® LLC. All rights reserved

Pg_gather [Collect data , Generate report]

Collecting data:

```
psql -d postgres -X -f gather.sql >
out.tsv
```

Generating report:

```
psql -f gather_schema.sql -f out.tsv ## Creates unlogged tables pg_*
psql -X -f gather_report.sql > GatherReport.html ##Generates HTML report
```

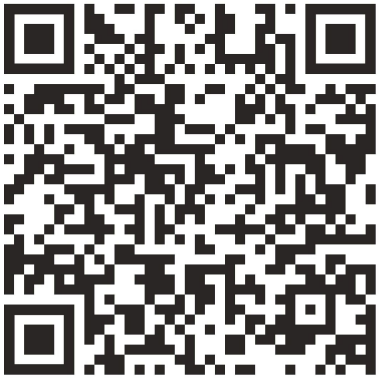
Docker method:

```
./generate_report.sh out.tsv
```

© Copyright 2024 Percona® LLC. All rights reserved

Pg_gather [Examples and Use cases]

1. Table/index issues.
2. Vacuum related issues.
3. PostgreSQL Parameters review.
4. Sessions: Idle connections/transaction , Wait Events.
5. Sessions: Chain blockers.
6. Buffers/Flushing and summary.



Full Examples: https://github.com/lalitvc/pg_conf_2024_talk_ref/tree/main/pg_gather_use_cases_tests

© Copyright 2024 Percona® LLC. All rights reserved

Thank You