



Postgres on Kubernetes Workshop Amsterdam

16 April | Amsterdam, The Netherlands

Borys Neselovskyi
Senior Sales Engineer, EDB

Niels van Noort
Sales Engineer, EDB

Agenda

Start	End	Session
13:00	13:30	Registration & Welcome
13:30	13:45	HCS Company Introduction
13:45	14:00	Red Hat OpenShift & EDB Partnership
14:00	14:15	Introduction to CloudNativePG and EDB
14:15	14:45	CNPG Operator Reference Architecture
14:45	15:15	Functionalities for CNPG
15:15	17:00	Interactive session & demo
17:00	18:00	Drinks and pizza



HCS Company Introduction





HCS company

TECH TRIBES

TECH TRIBES



Software Engineering



- Software Engineering
- Data
- Architecture
- Quality Assurance
- Infrastructure
- Cloud
- Devops
- PM & Analysis



Data Engineering



- Data Analytics
- Data Architecture
- Data Governance
- Data Literacy
- Machine Learning
- AI Governance
- AI Model Training
- AI Academy



End-to-end Projects and AI



- Project scoping
- Project sourcing
- Product development
- .NET Development
- Frontend development
- Agile
- Project-management
- Application Security



Hybrid Platform Engineering



- Developer Platforms
- Cloud Native Migration
- Platform Enablement
- DevOPS
- MLOps,
- Application Migration
- GitOPS
- Cyber Security



Managed Open Source



- Linux & Open Source Solutions
- 24/7 Support
- Automation
- Containerisation
- High Secure Environments
- Training



Cloud & Observability



- Reliability
- Automation
- Cost Management
- Monitoring
- Observability

A COLLECTIVE OF DIGITAL EXPERTS

c. €120m

Revenue ⁽¹⁾

c. €12,5m

EBITDA ⁽¹⁾

c. 11%

EBITDA % ⁽¹⁾

>750

Digital experts

72

Partner Distinguished
Engineers and
Accelerators

>700

Open Source
contributions
(Quarkus,
Kubernetes)

Great Place
to Work

Since 2016 ⁽³⁾

6

Tribes

5

Most Valuable
Partnership
Awards



ANVES

euricom

HCS
COMPANY

kingaroot
Linux & Open Source Solutions

XIFEO

Business
Data
Challengers

Red Hat Openshift and EDB Partnership



Red Hat OpenShift with EDB

David Szegedi
Chief Architect
Field CTO Organization

Stefan Van Oirschot
Global Solution Architect ING Bank

Red Hat is a Leader in the 2024 Gartner® Magic Quadrant™: Container Management

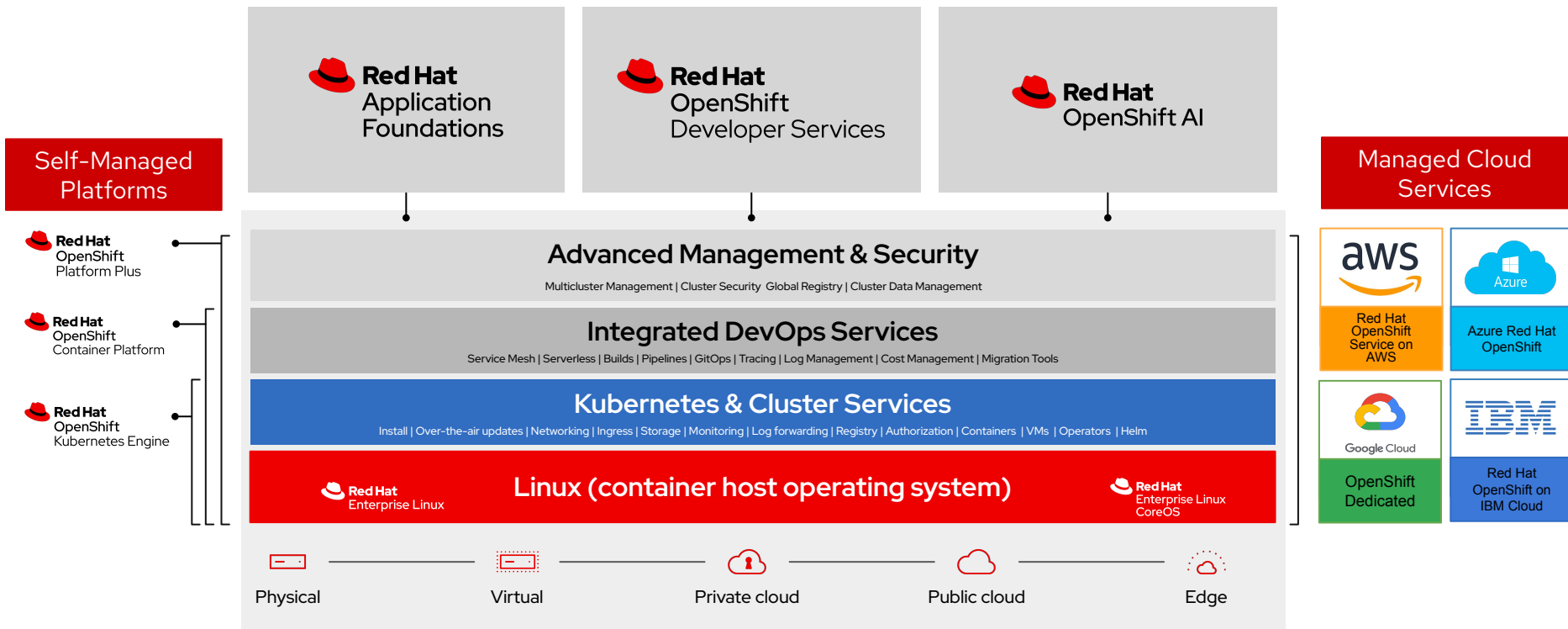
Figure 1: Magic Quadrant for Container Management



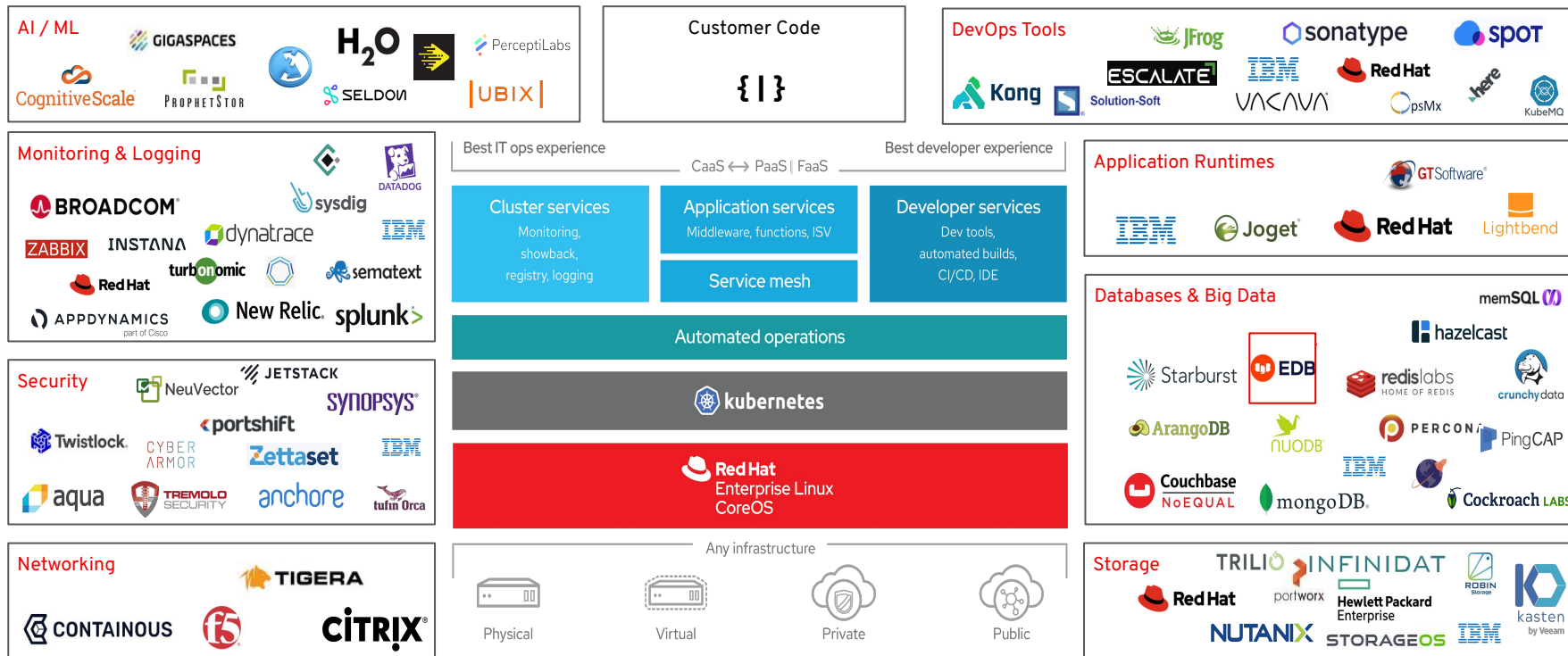
Gartner

Source: Gartner, "Magic Quadrant for Container Management," September 2024.

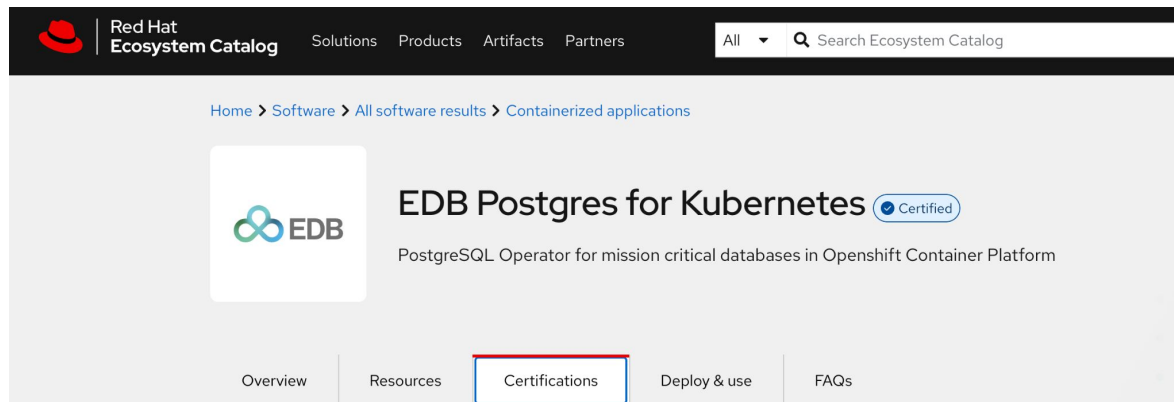
Hybrid Cloud Application Platform



Red Hat open hybrid cloud platform with ISV ecosystem



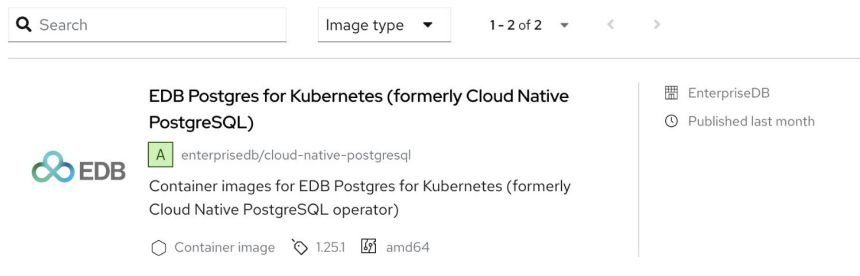
Why Red Hat OpenShift for EDB: operator certification



Certifications

Learn about Red Hat Certification and Partner Validation

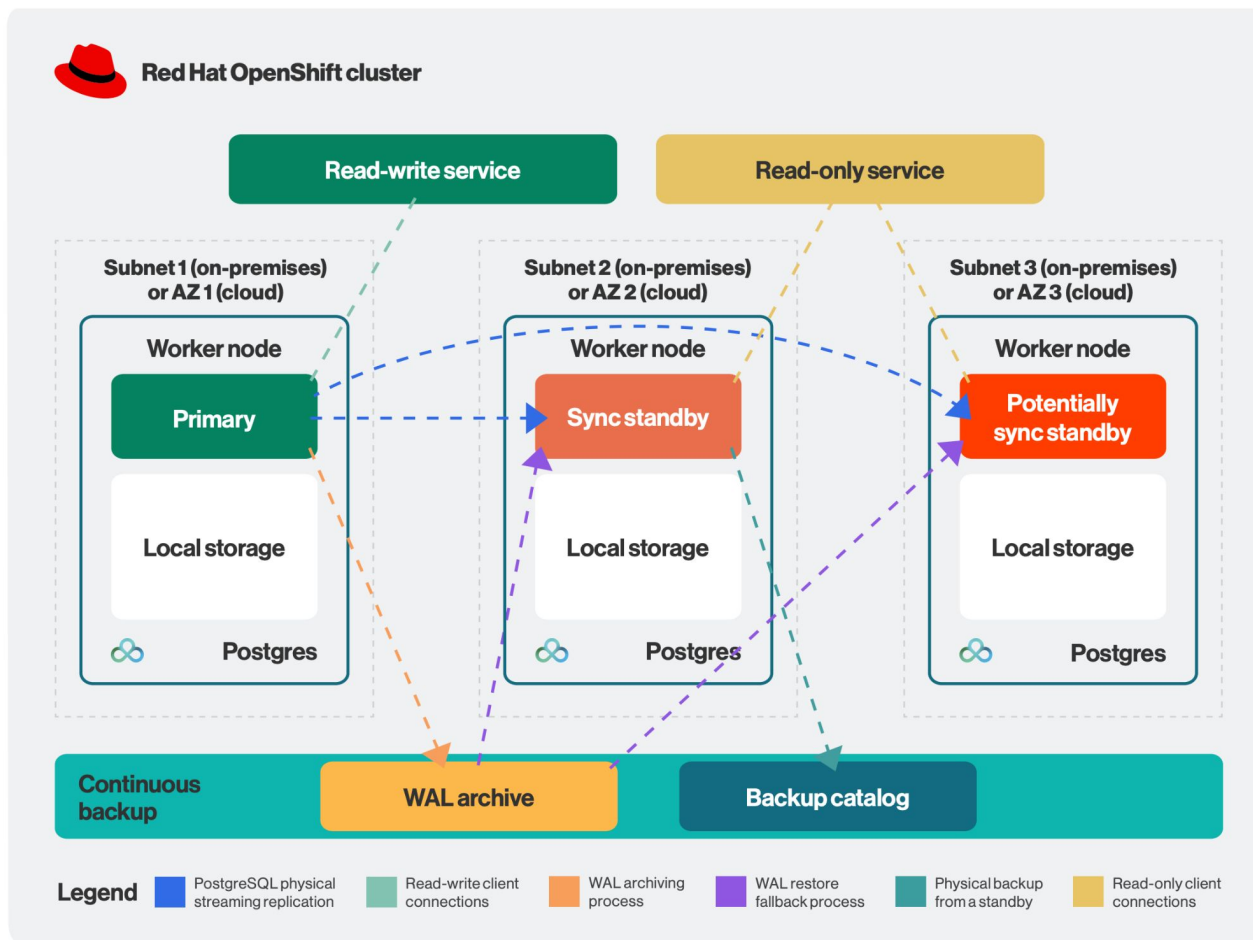
Certified components



EDB Postgres for Kubernetes is a certified Level 5 Operator for Red Hat OpenShift

- ▶ This is designed to streamline Day 2 operations of PostgreSQL databases
- ▶ Enhanced Database Management
- ▶ Supports point-in-time recovery (PITR)
- ▶ Ensures robust data protection and recovery options
- ▶ Integration with business continuity solutions such as Red Hat OpenShift API for Data Protection (OADP) and Veeam Kasten, Trilio, Portworx Backup, IBM Fusion, and others

Why Red Hat OpenShift for EDB : reference architecture



EDB on OpenShift use cases

- ▶ Cloud-Native Database Deployment
- ▶ Database as a Service (DBaaS)
- ▶ High Availability and Disaster Recovery (HA & DR)
- ▶ DevOps and Continuous Integration/Continuous Deployment (CI/CD)
- ▶ Microservices and Application Modernization
- ▶ Move from VMWare to OpenShift
- ▶ Data Security and Compliance (using **TDE** and Advanced Security provided by EPAS)
- ▶ Hybrid and Multi-Cloud Deployments
- ▶ Multi-Tenant Applications (isolation)

Euro Information

Company profile

Euro-Information is the fintech company of the Crédit Mutuel group. Euro-Information manages the IT systems of 16 federations of Crédit Mutuel as well as those of CIC and of all the financial, insurance, property, consumer credit, private banking, financing, telephony and technological subsidiaries.



- Red Hat OpenShift
- EDB Postgres for Kubernetes
- PostgreSQL
- EPAS



- EDB considerably reduces IT costs associated with database maintenance.
- 280 cores: Enterprise Plan + Production Support

Summary

Use Case

On prem DBaaS (**in Production**)

Workload

Transactional

Application Name

All internal Postgres applications

EDB Tools of Interest

PostgreSQL and EDB Postgres for Kubernetes

Problem

- Fast database deployment
- Adopt a supported and secure Open Source platform
- Onprem DBaaS
- Align to in-house RDBMS standardization

Solution

- Use Postgres capabilities to build and maintain local applications
- Use Red Hat OpenShift platform to accelerate the provisioning of databases and applications

Results

- Applications running with PostgreSQL databases in a centralized environment
- Massive reduction of TCO of database service operations



La Poste

Company profile

La Poste is a postal service company in France, operating in Metropolitan France, the five French overseas departments and regions and the overseas collectivity of Saint Pierre and Miquelon. Under bilateral agreements, La Poste also has responsibility for mail services in Monaco through La Poste Monaco and in Andorra alongside the Spanish company Correos.



- Red Hat OpenShift
- EDB Postgres for Kubernetes
- PostgreSQL



- EDB considerably reduces IT costs associated with database maintenance.
- 12 Cores: Standard Plan + Premium Support

Summary

Use Case

On prem DBaaS with HA and DR
(In Production)

Workload

Transactional

Application Name

Portail XaaS

EDB Tools of Interest

PostgreSQL and EDB Postgres for Kubernetes

Problem

- Provide a database HA solution for Ansible Automation Platform (AAP)
- Database must be in HA and DR

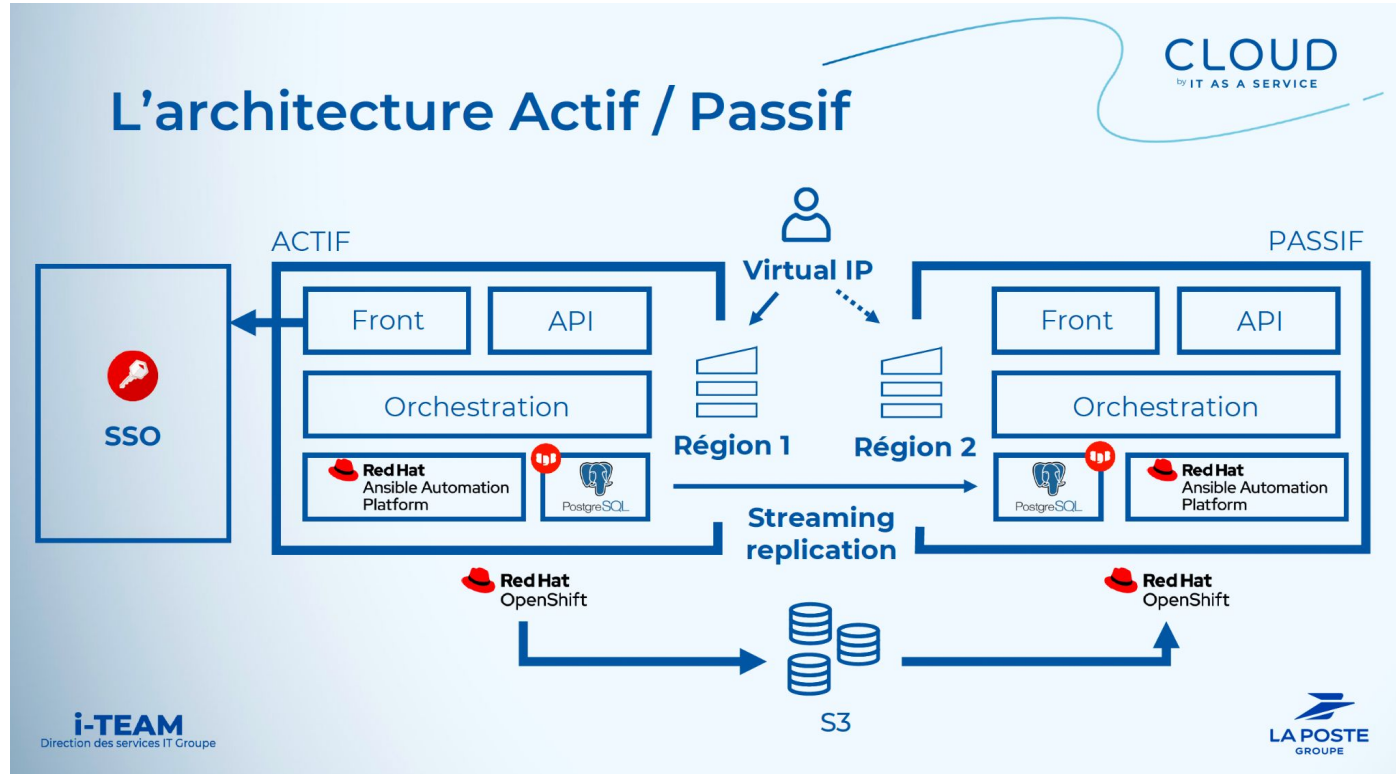
Solution

- Use EDB Postgres for Kubernetes to provide a HA and DR solution for PostgreSQL databases
- Deploy in 2 OpenShift clusters our operator

Results

- La Poste developer can use their internal 'La Post Service Portal' to provision more than 64 backends.
- Reduce risk deploying EDB solutions.

La Poste Architecture



Thank you

Red Hat is the world's leading provider of enterprise open source software solutions. Award-winning support, training, and consulting services make Red Hat a trusted adviser to the Fortune 500.



linkedin.com/company/red-hat



youtube.com/user/RedHatVideos



facebook.com/redhatinc



twitter.com/RedHat

Introduction to CloudNativePG and EDB





EDB

Postgres® for the AI Generation

20+ years of Postgres innovation & adoption

- Number one contributor to Postgres, fastest-growing and most loved Database in the world
 - 2 Core Team members, 7 Committers, 9 Major Contributors, 10 Contributors, #1 site for desktop downloads
- Over 700 employees in more than 30 countries
- EDB Postgres AI
 - The industry's first platform that can be deployed as cloud, software or physical appliance
 - Secure, compliant and enterprise grade performance guaranteed

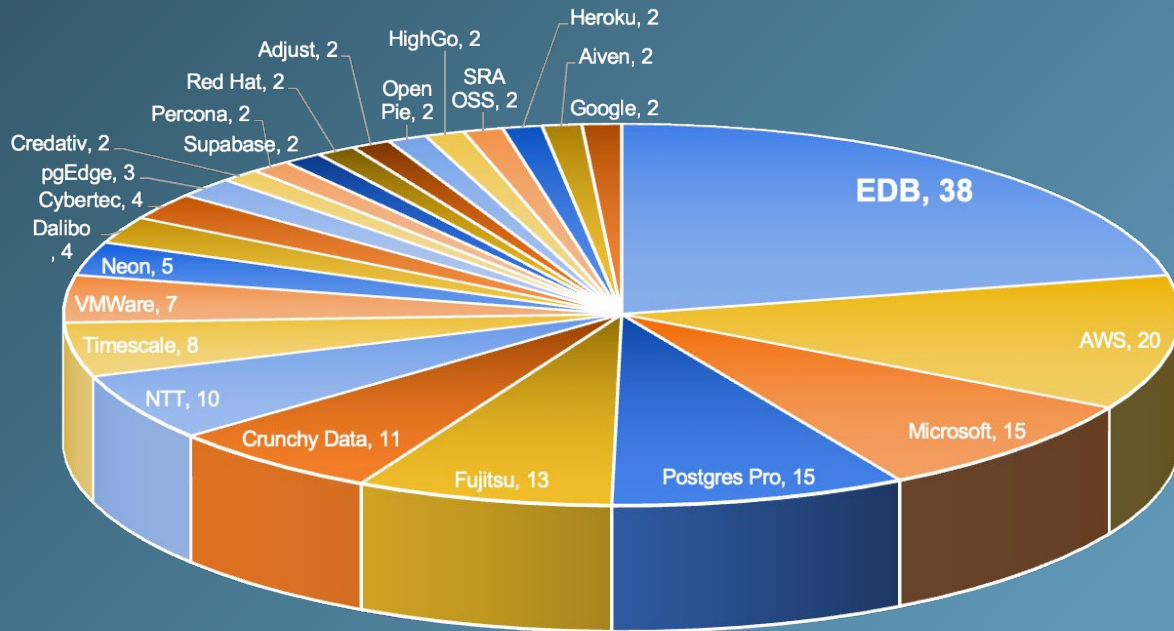


Large Developer Community

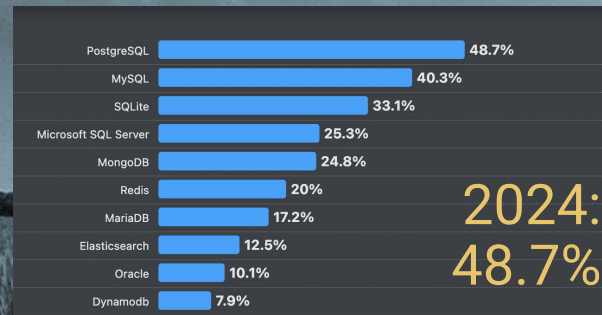
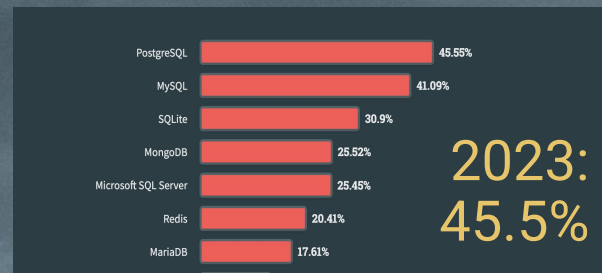
- **407 +** contributors to Postgres 16
- **111** companies actively contributing to Postgres 16

Contributors to Postgres 16 by Postgres Companies

Without individuals or unaffiliated contributors



Postgres has won the database race



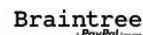
Stack Overflow Survey 2023/2024



BANKING FINANCIAL



TECHNOLOGY



TELCO



EDB POSTGRES AI PLATFORM

UNIFIED WORKLOAD MANAGEMENT

TRANSACTIONAL

ANALYTICAL

ARTIFICIAL INTELLIGENCE

SINGLE PANE OF GLASS ADMINISTRATION

HYBRID
CONTROL PLANE

DATABASE AUTOMATION
AND MANAGEMENT

INTELLIGENT
OBSERVABILITY

ENTERPRISE
SECURITY

MULTI-MODEL DATA MANGEMENT

RELATIONAL

JSON

TIME SERIES

VECTOR

COLUMNAR

HYBRID AND MULTI-CLOUD DEPLOYMENT

SOFTWARE
DEPLOYMENT

CLOUD
SERVICE

HARDWARE
INTEGRATED SOLUTION

PLATFORM TOOLS AND SERVICES

MIGRATION
PORTAL

CONTINUOUS HIGH
AVAILABILITY

BACKUP AND
RECOVERY

EXTENSIBILITY

CSP INTEGRATIONS

DEVOPS TOOLING

KUBERNETES TOOLING

GENAI & LLM INTEGRATIONS

LAKEHOUSE INTEGRATIONS

Crew

HCS team



Jan Kappert



Borys Neselovskyi



Niels van Noort

EDB team



Michael Willer



Janus Hägele



Kevin Li

Cloud Native Postgres

Kubernetes timeline

- 2014, June: Google open sources Kubernetes
- 2015, July: Version 1.0 is released
- 2015, July: Google and Linux Foundation start the CNCF
- 2016, November: The operator pattern is introduced in a blog post
- 2018, August: The Community takes the lead
- 2019, April: Version 1.14 introduces **Local Persistent Volumes**
- 2019, August: EDB team starts the Kubernetes initiative
- 2020, June: we publish this blog about benchmarking local PVs on bare metal
- 2020, June: Data on Kubernetes Community founded
- 2021, February: EDB Cloud Native Postgres (CNP) 1.0 released
- 2022, May: **EDB donates CNP** and open sources it under CloudNativePG
- 2025, January: CloudNativePG was recognized as an official **#CNCF** project



CloudNativePG/EDB Postgres for Kubernetes

CloudNativePG



- Kubernetes operator for PostgreSQL
- “Level 5”, Production ready
- Day 1 & 2 operations of a Postgres database
- Open source (May 2022)
 - Originally created by EDB
 - Apache License 2.0
 - Vendor neutral openly governed
- Extends the K8s controller
 - Status of the `Cluster`
 - “no Patroni, No statefulsets”
- Immutable application containers
- Fully declarative

EDB CloudNativePG

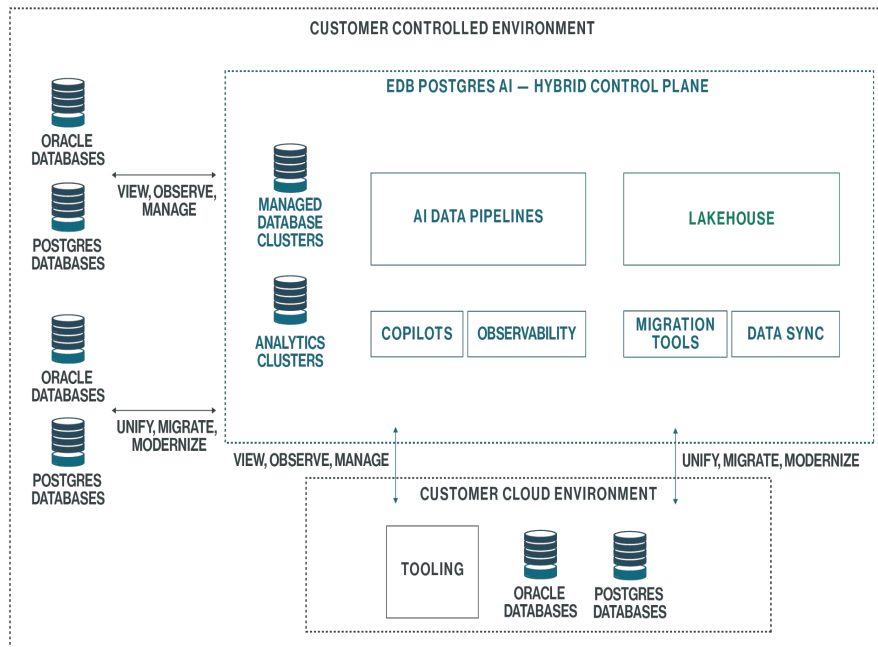


- All Features of CloudNativePG
- +
- Provides Long Term Support
- Access to EDB Postgres Advanced (TDE + Oracle Compatibility layer + Security features)
- Support on IBM Power and z/Linux through partnership with IBM
- Cold backup support with Kasten and Velero/OADP
- Generic adapter for third-party Kubernetes backup tools
- [Hybrid Control Plane \(Preview\)](#) - centralized management and automation solution, built on Kubernetes, for EDB Postgres AI

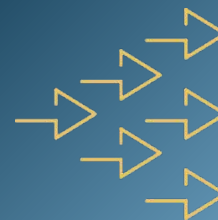


Hybrid Control Plane — Observability and automation

Centralized management, observability, and automation means customers can do more, with less, and innovate faster



Operate
Efficiently

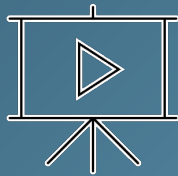


Innovate
Faster

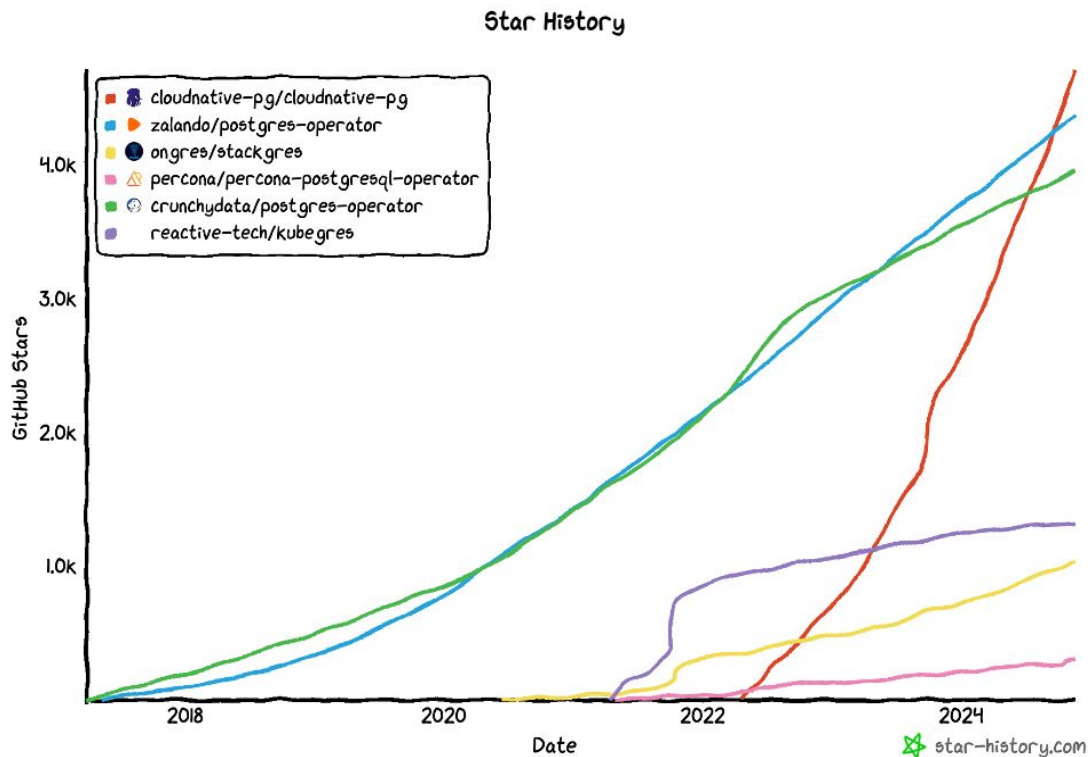


Hybrid Control Plane (HCP)

Live Demo



The world's most popular Postgres operator for Kubernetes



CLOUD NATIVE
COMPUTING FOUNDATION

SANDBOX PROJECTS



Run PostgreSQL.
The Kubernetes way.

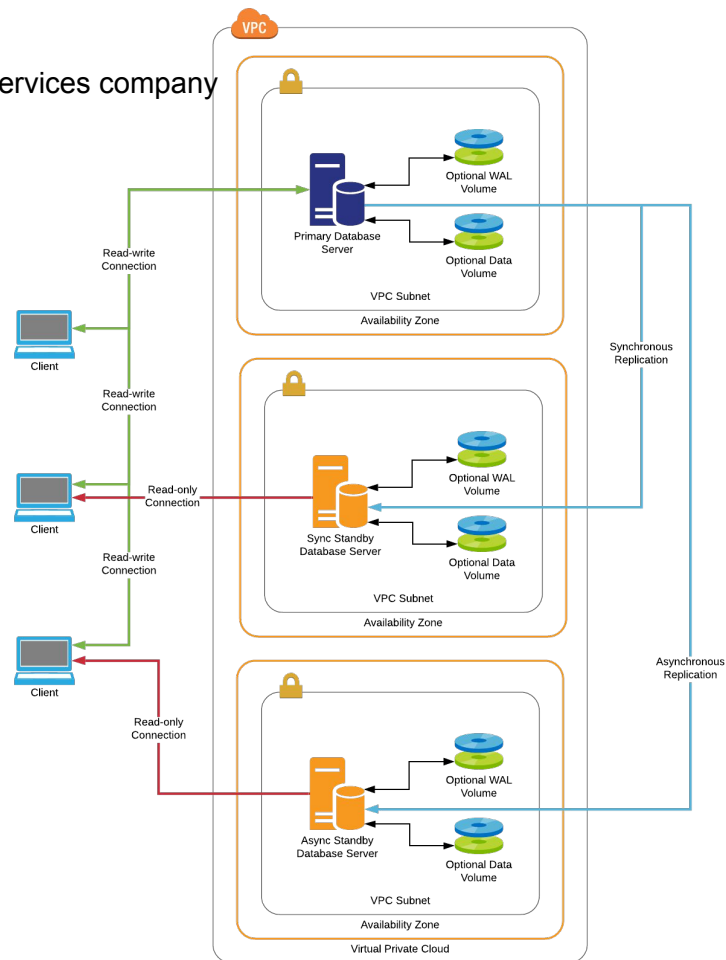
Gabriele Bartolini (He/Him) - 1st
VP of Cloud Native at EDB | DoK Ambassador | PostgreSQL

72 Million
downloads
and counting

Imperative vs Declarative

Built over 20 years, EDB is the world's largest software, support, and services company focused exclusively on PostgreSQL

- Create and configure VMs
- Create a PostgreSQL 17 instance
- Configure for replication
- Clone a second one
- Set it as a replica
- Clone a third one
- Set it as a replica
- Configure networking
- Configure security
- etc.



Convention over configuration

Declarative - simple to install, simple to maintain

There's a PostgreSQL 17 cluster with 2 replicas:

```
apiVersion: postgresql.k8s.enterisedb.io/v1
kind: Cluster
metadata:
  name: myapp-db
spec:
  instances: 3
  imageName: quay.io/enterisedb/postgresql:17.2

  storage:
    size: 10Gi
```



CNPG Operator Reference Architecture



“The same as running a
database on a VM”



*I would add: "... provided **you** ..."*

Know PostgreSQL

Know Kubernetes

Have a good operator like CloudNativePG

You = You organization, made up of one or more multidisciplinary teams



The right architecture for Kubernetes



Kubernetes architectural concepts

- A Kubernetes Cluster (**k-cluster**)
- Availability zones (**AZ**)- also known as failure zones or data centers
 - Connected by redundant, low-latency, private network connectivity
 - At least 3 per k-cluster
- Kubernetes control plane to be distributed across the AZ
- Kubernetes worker nodes in each AZ running applications (workloads)
- Normally:
 - **1 k-cluster = 1 region with 3+ AZ**

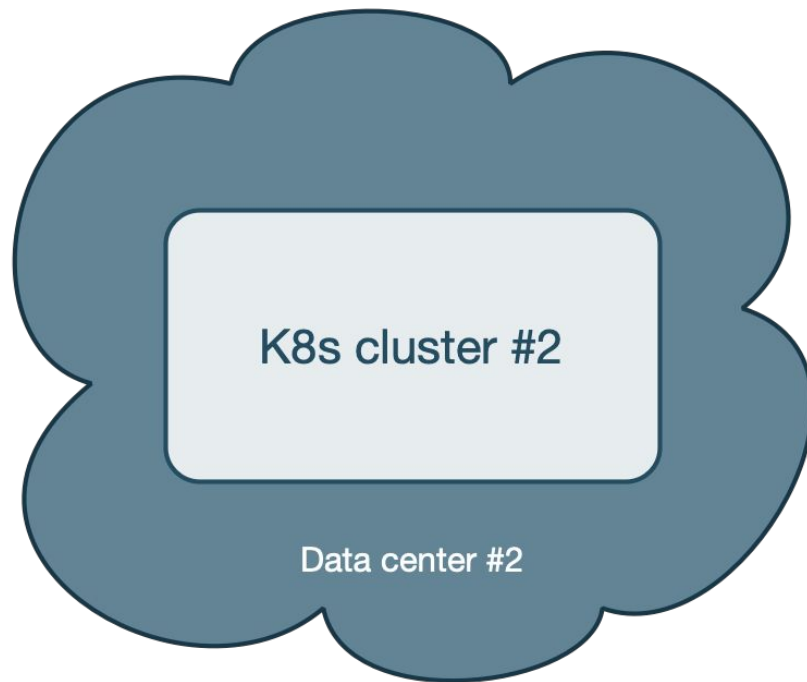
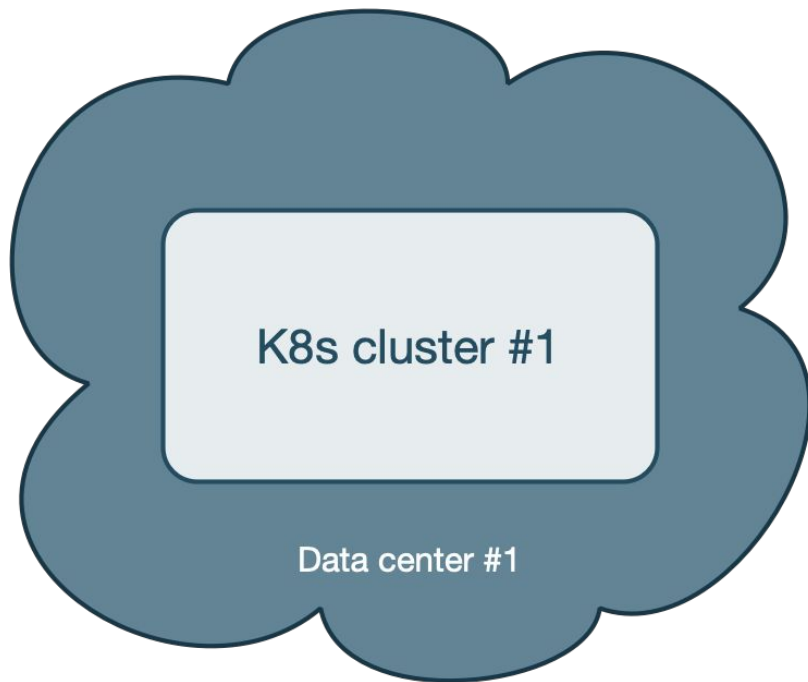


1 k-cluster = 1 region with 3+ AZ

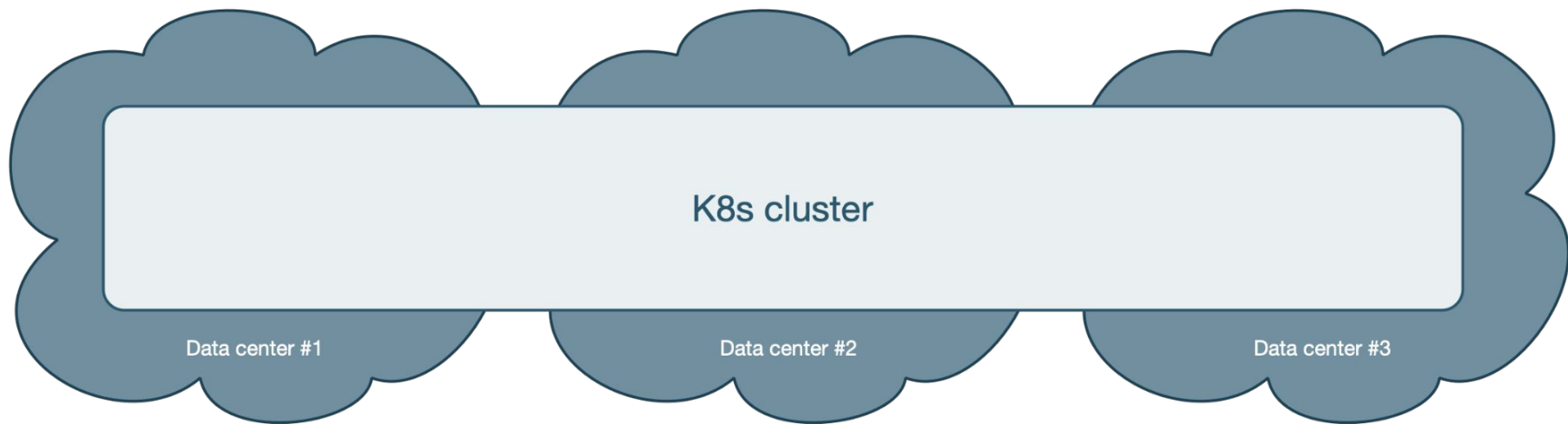
- Taken for granted if you know Kubernetes
- All major public cloud providers offering managed K8s services have 3+ AZ
- What about on-premise deployments?
 - You need to plan in advance
 - Stay away from the “2 data center in a region” setup typical of “Lift-and-Shift” exercises
 - Often results in 2 separate Kubernetes clusters
 - Severely impacts the benefits of Kubernetes, particularly self-healing
 - Shifts maintenance and procedural complexity up to the application level



No!



Yes!



Yes! Yes! Yes!



Synchronizing the state



Synchronizing the state of a Postgres database

- Being a DBMS, PostgreSQL is a stateful workload in Kubernetes
- Stateless workloads achieve HA and DR mainly through traffic redirection
- Stateful workloads require the state to be replicated in multiple locations:
 - **Storage-level** replication
 - **Application-level** replication (in our case, application = Postgres)
- Postgres has a very robust and powerful native replication system
 - We've built it
 - Founded on the Write Ahead Log
 - Read-only standby servers
 - Supports also synchronous replication controlled at the transaction level
- **We recommend application-level** over storage-level replication for Postgres



The right storage for you



Storage management

- Storage is the most critical component for a database
- Direct support for Persistent Volume Claims (PVC)
 - We deliberately do not use Statefulsets
- The PVC storing the PGDATA is central to CloudNativePG
 - Our motto is: “PGDATA is worth a 1000 pods”
- Storage agnostic
- Freedom of choice
 - Local storage
 - Network storage
- Automated generation of PVC
- Support for PVC templates
 - Storage classes



Scheduling Postgres instances with CloudNativePG

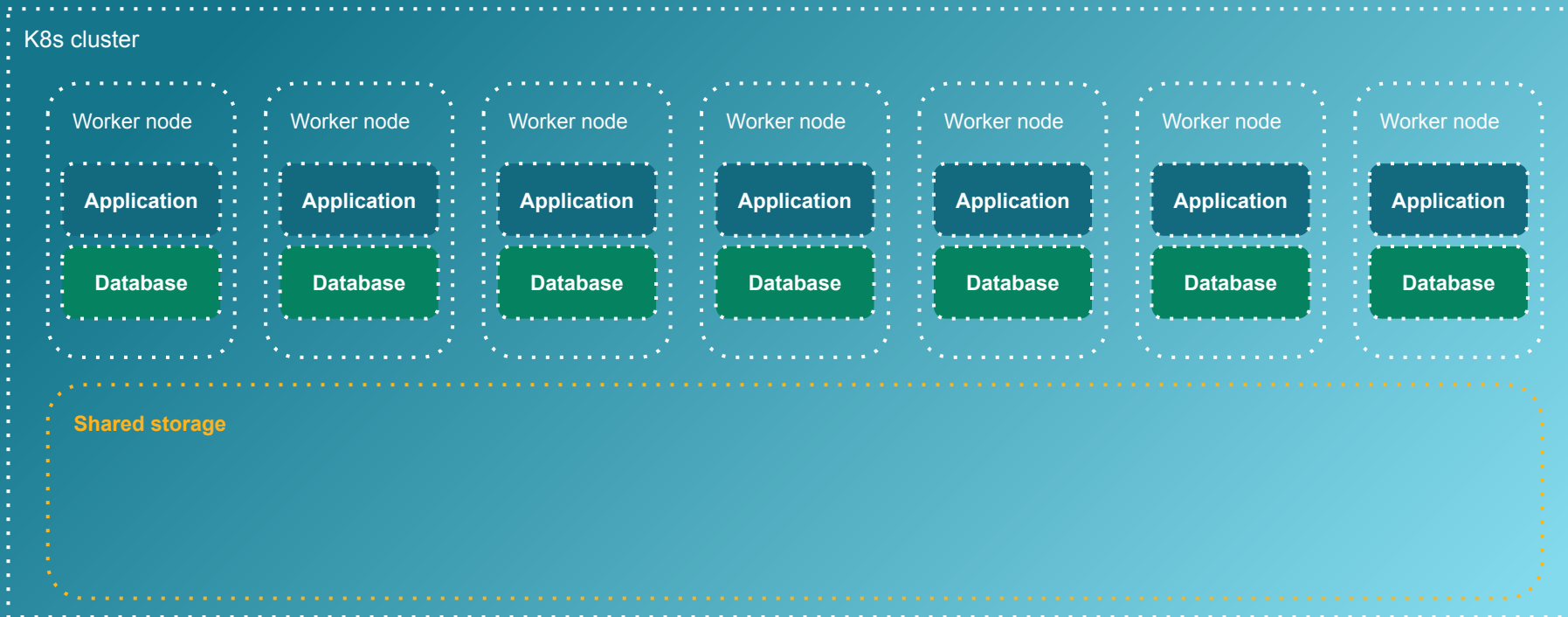
- Entirely declarative!
- Affinity section in the `Cluster` specification
 - pod affinity/anti-affinity
 - node selectors
 - tolerations against taints placed on nodes



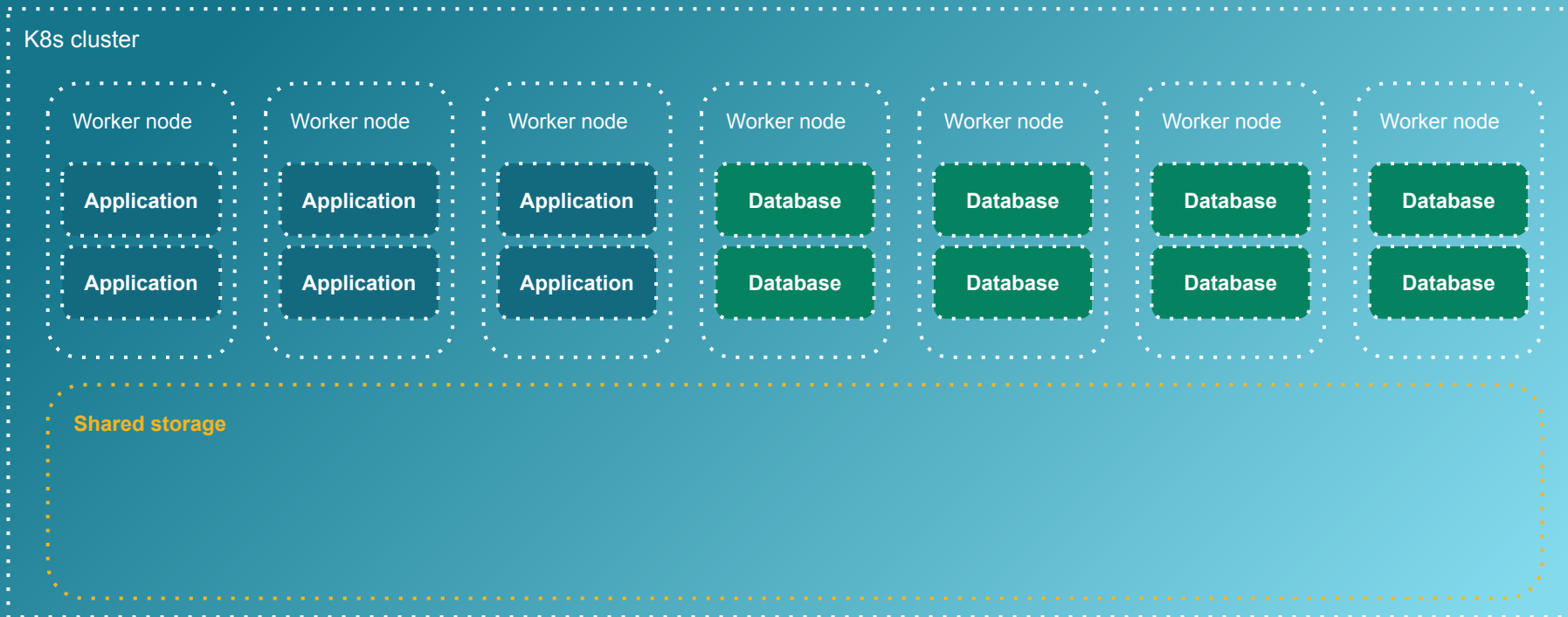
The right architecture for Database



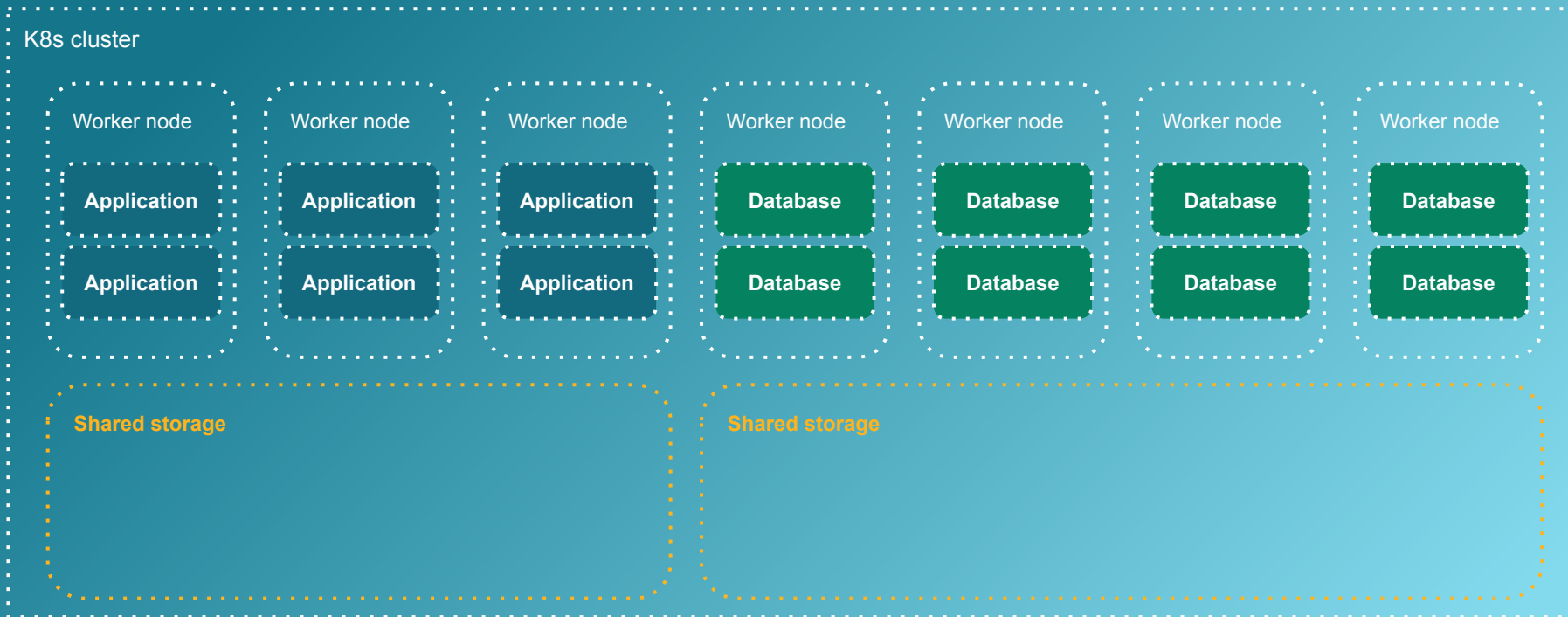
Shared workloads, shared storage #1



Shared workloads, shared storage #2



Shared workloads, shared storage #3

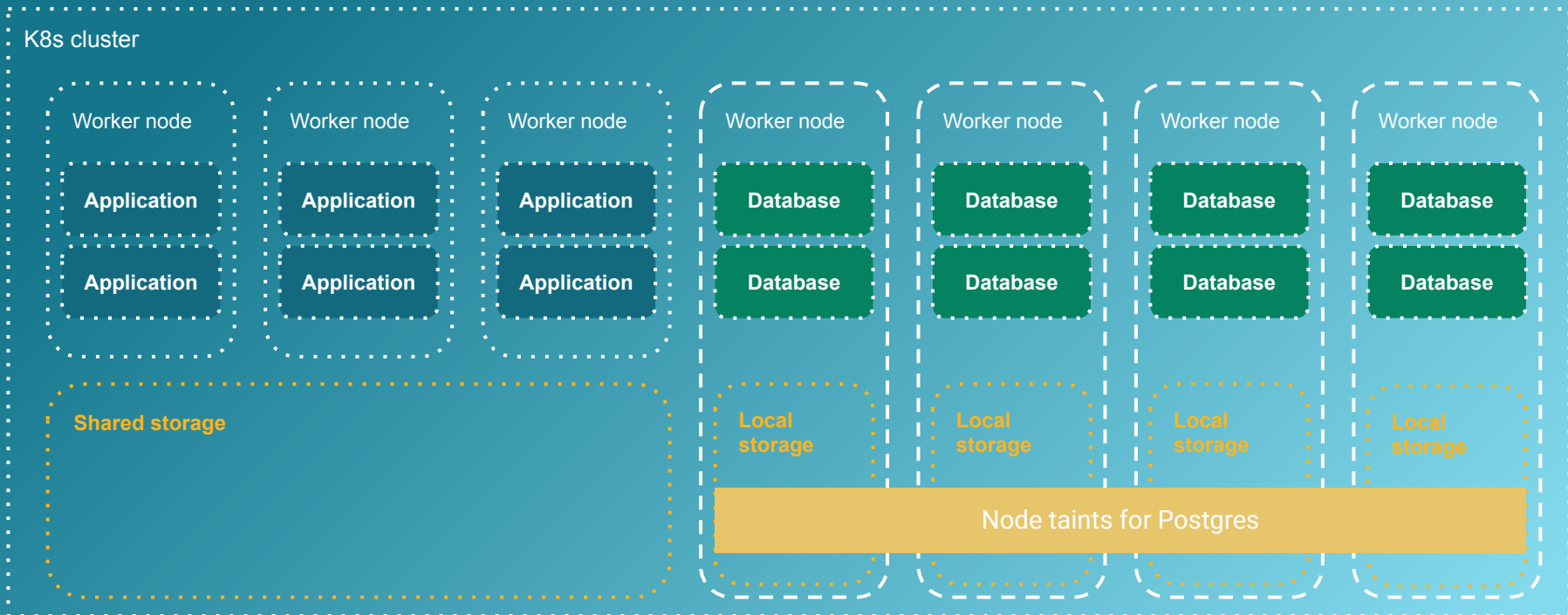


Shared workloads, local storage



Good value for money!

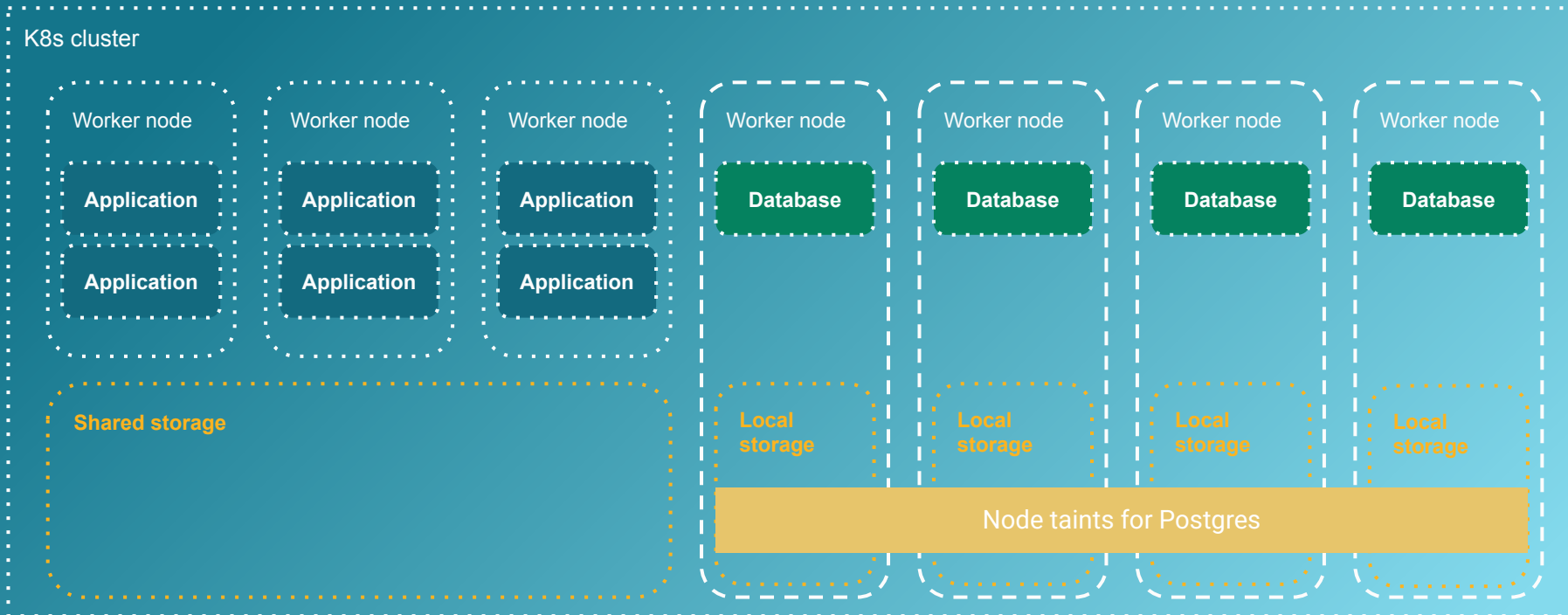
K8s cluster



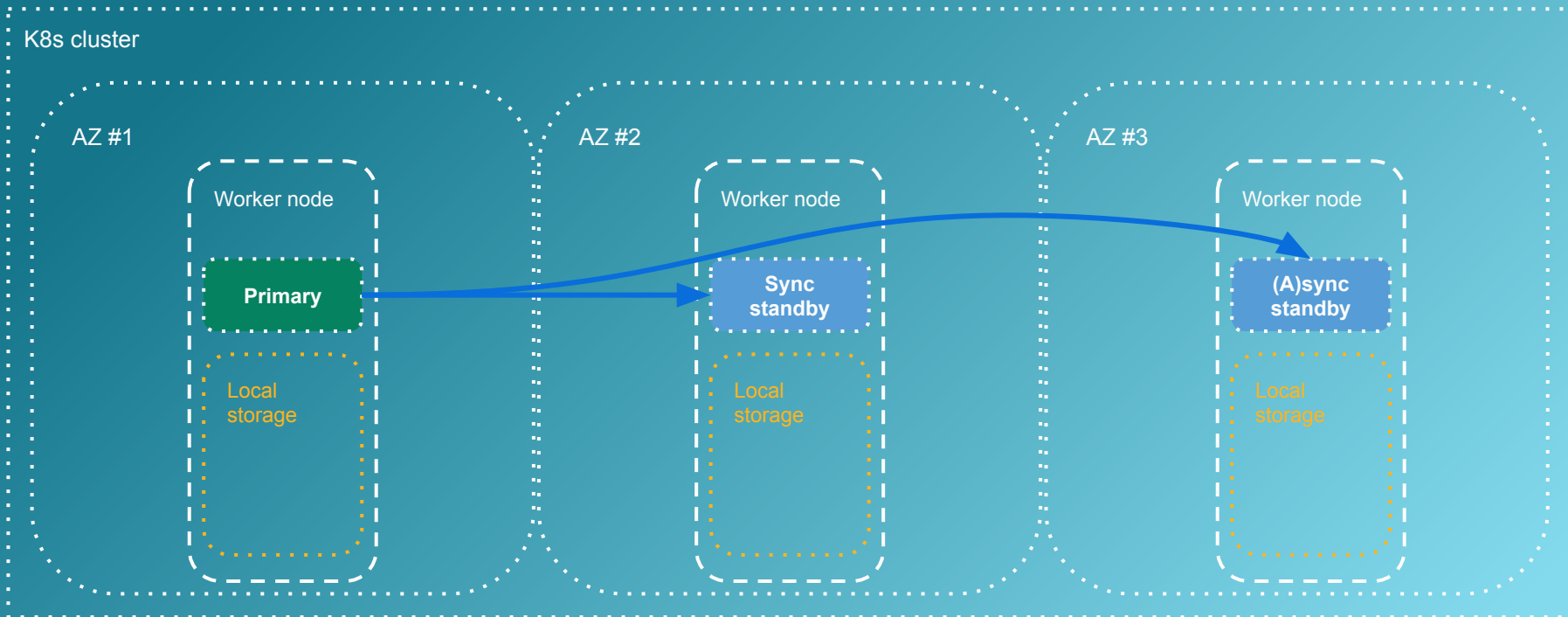
Dedicated workloads, local storage



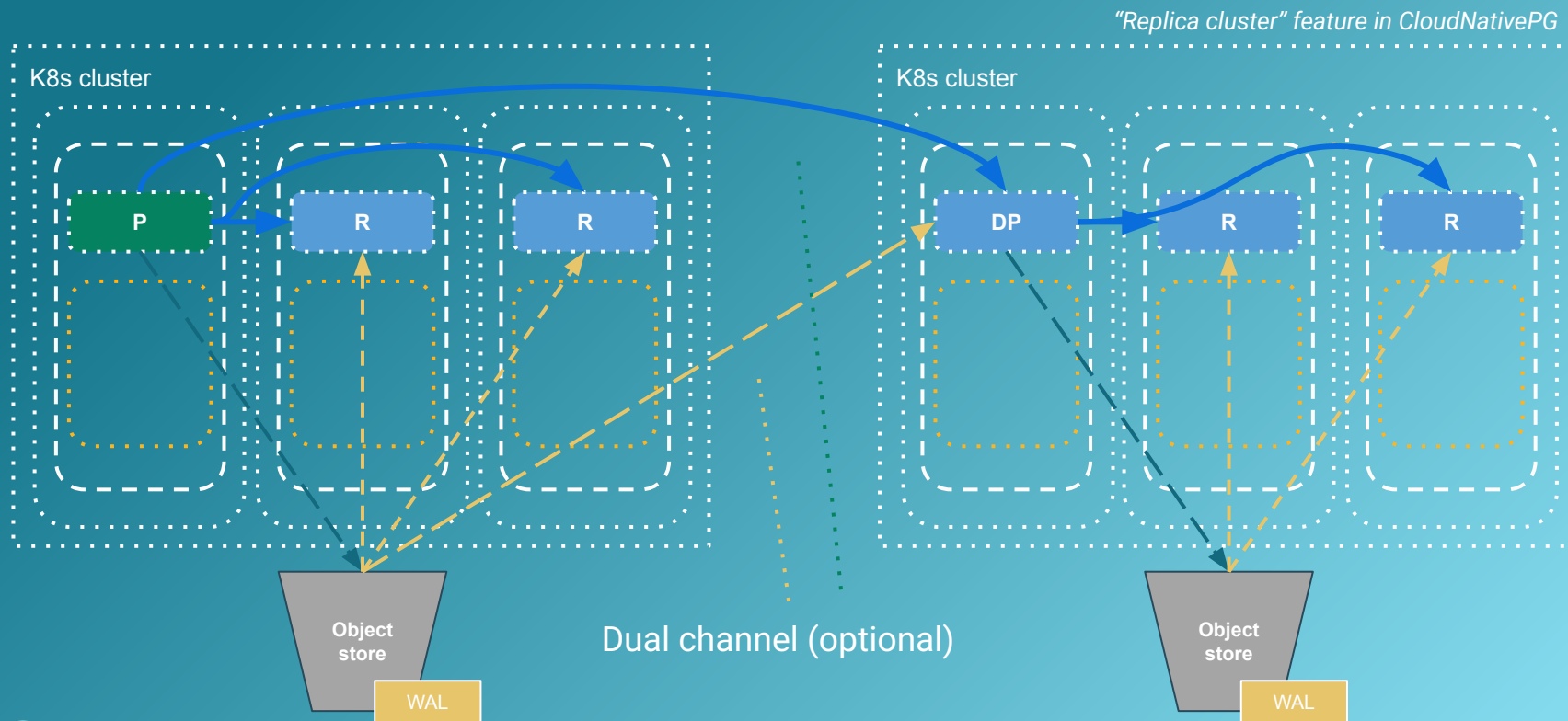
Best Postgres results!



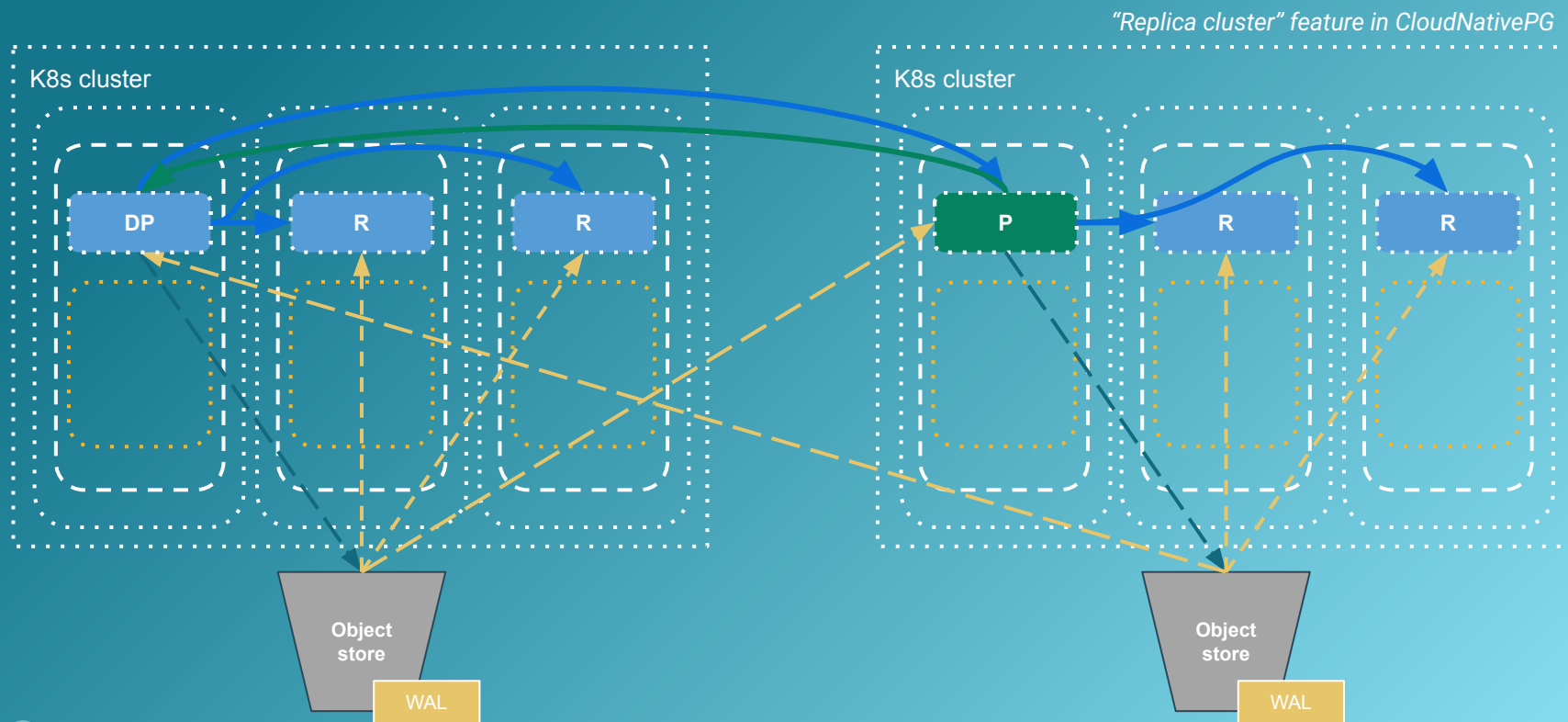
Shared nothing architecture



Shared nothing architecture (hybrid/multi)



Shared nothing architecture (hybrid/multi)



Functionalities for CNPG



Deploy a 3-node HA Postgres Cluster

- Install the latest PostgreSQL 17 minor release
- Create a 3-node PostgreSQL 17 cluster
 - A primary, two standby (one synchronous)
 - Use mTLS authentication for the replicas
 - Use replication slots
- Resources:
 - RAM: 4 GB
 - CPU: 8 cores
 - Storage: 40GB for PGDATA, 10GB for WALs
- A user and a database for the application
- A reliable and consistent way to access the primary via network





```
apiVersion: postgresql.cnpg.io/v1
```

```
kind: Cluster
```

```
metadata:
```

```
  name: myapp-db
```

```
spec:
```

```
  resources:
```

```
    requests:
```

```
      memory: "4Gi"
```

```
      cpu: 8
```

```
    limits:
```

```
      memory: "4Gi"
```

```
      cpu: 8
```

```
  ...
```

```
    ○
```



```
postgresql.conf
```

```
instances: 3
```

```
minSyncReplicas: 1
```

```
maxSyncReplicas: 1
```

```
replicationSlots:
```

```
  highAvailability:
```

```
    enabled: true
```

```
storage:
```

```
  size: 40Gi
```

```
walStorage:
```

```
  size: 10Gi
```

```
postgresql:
```

```
  parameters:
```

```
    shared_buffers: "1GB"
```



```
replication
```



```
storage
```

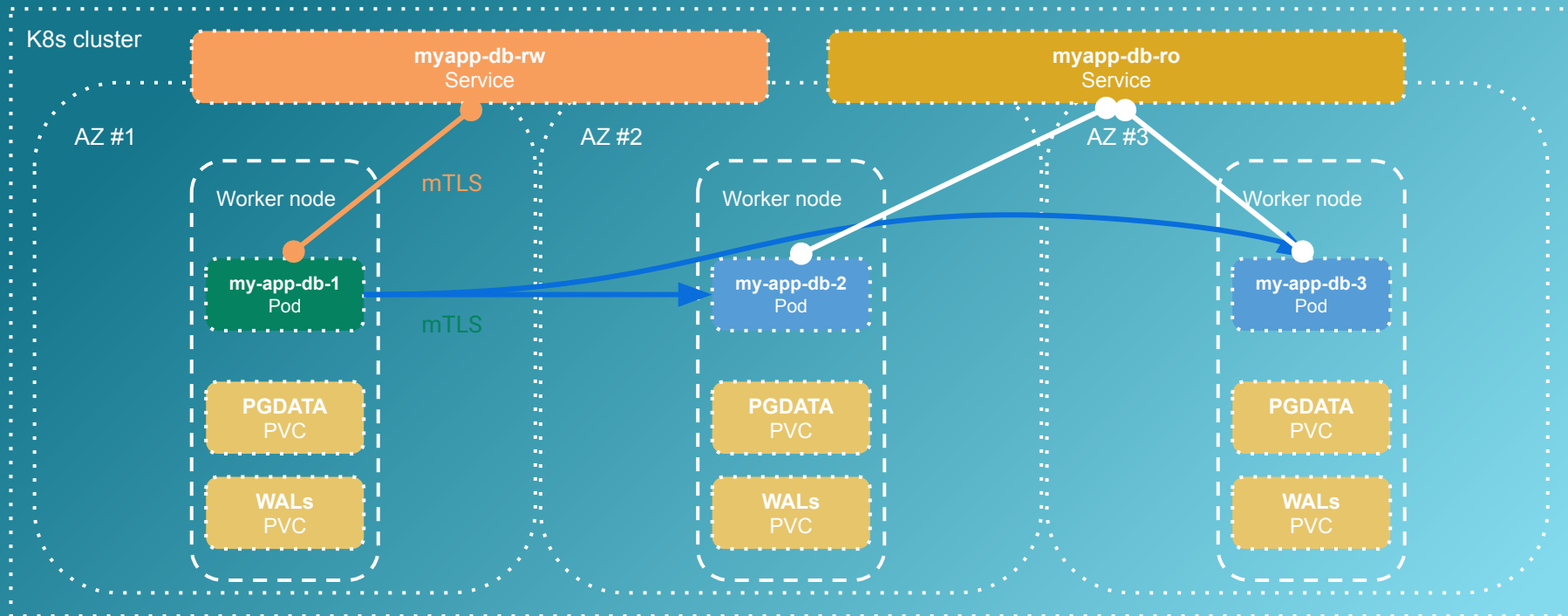


How to deploy the PostgreSQL Cluster

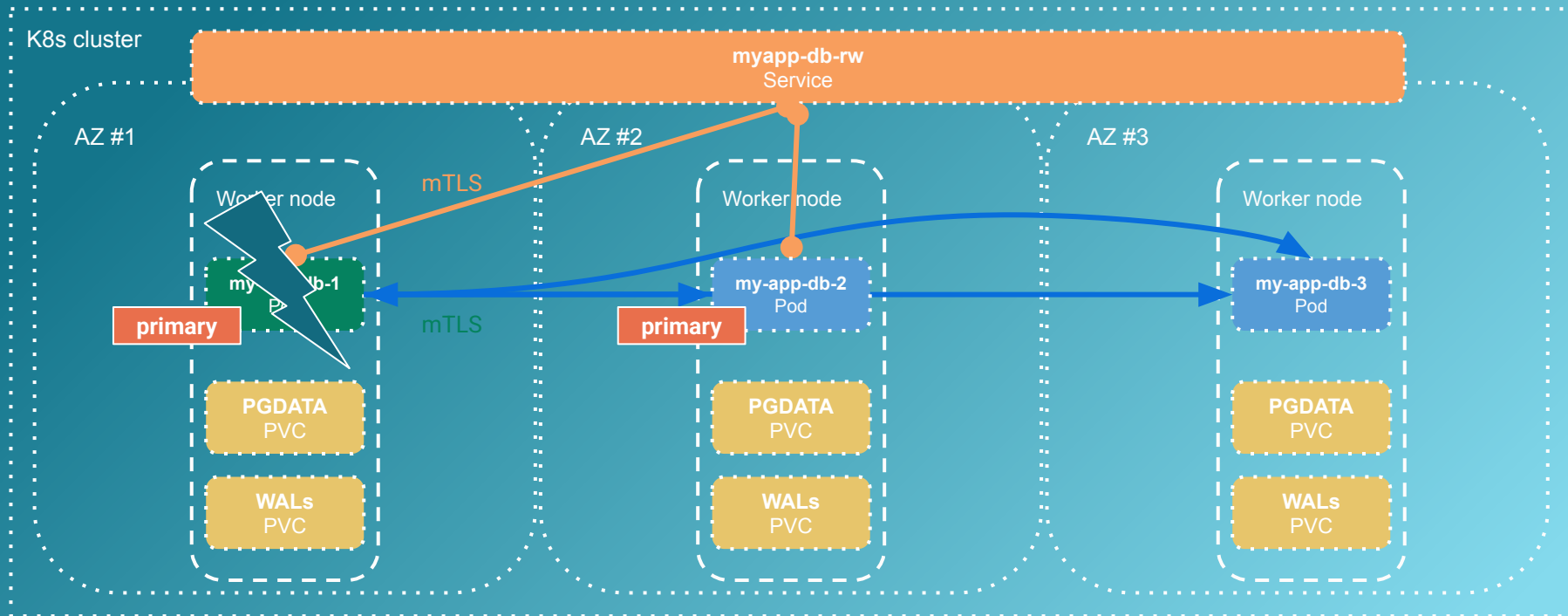
```
kubectl apply -f myapp-db.yaml
```



This is what happens under the hood



Automated failover



Advanced security



Advanced Security



Password policy management

DBA managed password profiles, compatible with Oracle profiles



Audit compliance

Track and analyze database activities and user connections



Virtual private databases

Fine grained access control limits user views



EDB/SQL protect

SQL firewall, screens queries for common attack profiles



Data redaction

Protect sensitive information for GDPR, PCI and HIPAA compliance



Code protection

Protects sensitive IP, algorithms or financial policies



Transparent Data Encryption (TDE)

- Transparent Data Encryption (TDE) is a feature of EDB Postgres Advanced Server and EDB Postgres Extended Server that prevents unauthorized viewing of data in operating system files on the database server and on backup storage
- Data encryption and decryption is managed by the database and does not require application changes or updated client drivers
- EDB Postgres Advanced Server and EDB Postgres Extended Server provide hooks to key management that is external to the database allowing for simple passphrase encrypt/decrypt or integration with enterprise key management solutions, with initial support for:
 - Amazon AWS Key Management Service (KMS)
 - Google Cloud - Cloud Key Management Service
 - Microsoft Azure Key Vault
 - HashiCorp Vault (KMIP Secrets Engine and Transit Secrets Engine)
 - Thales CipherTrust Manager
- Data will be unintelligible for unauthorized users if stolen or misplaced



Main capabilities



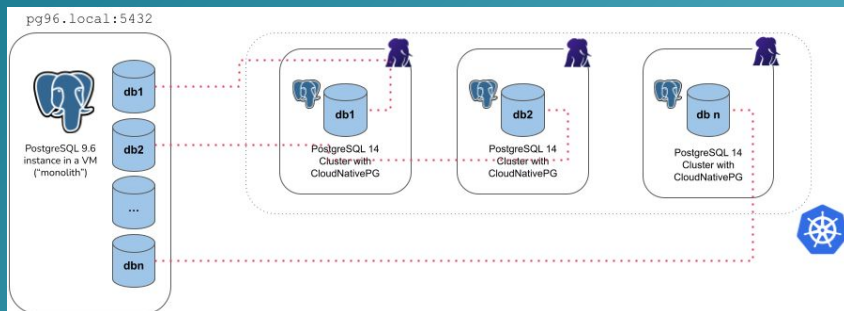
Bootstrap

- Create a new cluster from scratch
 - “initdb”: named after the standard “initdb” process in PostgreSQL that initializes an instance
 - This method can be used to migrate another database (“import”) or upgrade it
 - Uses pg_dump and pg_restore with some intelligence we’ve added
- Create a new cluster from an existing one:
 - Directly (“pg_basebackup”), using physical streaming replication
 - Indirectly (“recovery”), from an object store
 - To the end of the WAL
 - Can be used to start independent replica clusters in continuous recovery
 - Using PITR

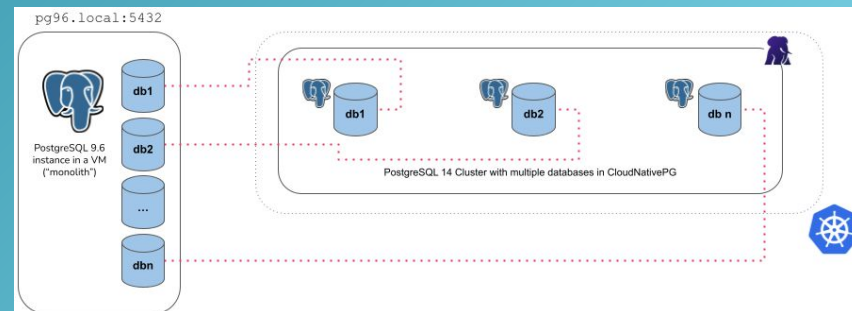


Migrate from external instance

Microservices



Monolith



Rolling updates

- Update of a deployment with ~zero downtime
 - Standby servers are updated first
 - Then the primary:
 - supervised / unsupervised
 - switchover / restart
- When they are triggered:
 - Security update of Postgres images
 - Minor update of PostgreSQL
 - Configuration changes when restart is required
 - Update of the operator
 - Unless in-place upgrade is enabled



Backup and Recovery - Part 1

- Continuous physical backup on “backup object stores”
 - Scheduled and on-demand base backups
 - Continuous WAL archiving (including parallel)
 - From primary or a standby
 - Support for recovery window retention policies (e.g. 30 days)
- Recovery means creating a new cluster starting from a “recovery object store”
 - Then pull WAL files (including in parallel) and replay them
 - Full (End of the WAL) or PITR
- Both rely on Barman Cloud technology
 - AWS S3
 - Azure Storage compatible
 - Google Cloud Storage
 - MinIO



Backup and Recovery - Part 2

- WAL management
 - Object store
- Physical Base backups
 - Object store
 - Kubernetes level backup integration (Velero/OADP, Veem Kasten K10, generic interface)
 - Kubernetes Volume Snapshots



Kubernetes Volume Snapshot: major advantages

- Transparent support for:
 - Incremental backup and recovery at block level
 - Differential backup and recovery at block level
 - Based on copy on write
- Leverage the storage class to manage the snapshots, including:
 - Data mobility across network (availability zones, Kubernetes clusters, regions)
 - Relay files on a secondary location in a different region, or any subsequent one
 - Encryption
- Enhances Very Large Databases (VLDB) adoption



Backup & Recovery via Snapshots: some numbers

Let's now talk about some initial benchmarks I have performed on volume snapshots using 3 `r5.4xlarge` nodes on AWS EKS with the `gp3` storage class. I have defined 4 different database size categories (tiny, small, medium, and large), as follows:

Cluster name	Database size	pgbench init scale	PGDATA volume size	WAL volume size	pgbench init duration
<i>tiny</i>	4.5 GB	300	8 GB	1 GB	67s
<i>small</i>	44 GB	3,000	80 GB	10 GB	10m 50s
<i>medium</i>	438 GB	3,0000	800 GB	100 GB	3h 15m 34s
<i>large</i>	4,381 GB	300,000	8,000 GB	200 GB	32h 47m 47s

The table below shows the results of both backup and recovery for each of them.

Cluster name	1st backup duration	2nd backup duration after 1hr of pgbench	Full recovery time
<i>tiny</i>	2m 43s	4m 16s	31s
<i>small</i>	20m 38s	16m 45s	27s
<i>medium</i>	2h 42m	2h 34m	48s
<i>large</i>	3h 54m 6s	2h 3s	2m 2s

<https://www.enterprisedb.com/postgresql-disaster-recovery-with-kubernetes-volume-snapshots-using-cloudnativepg>

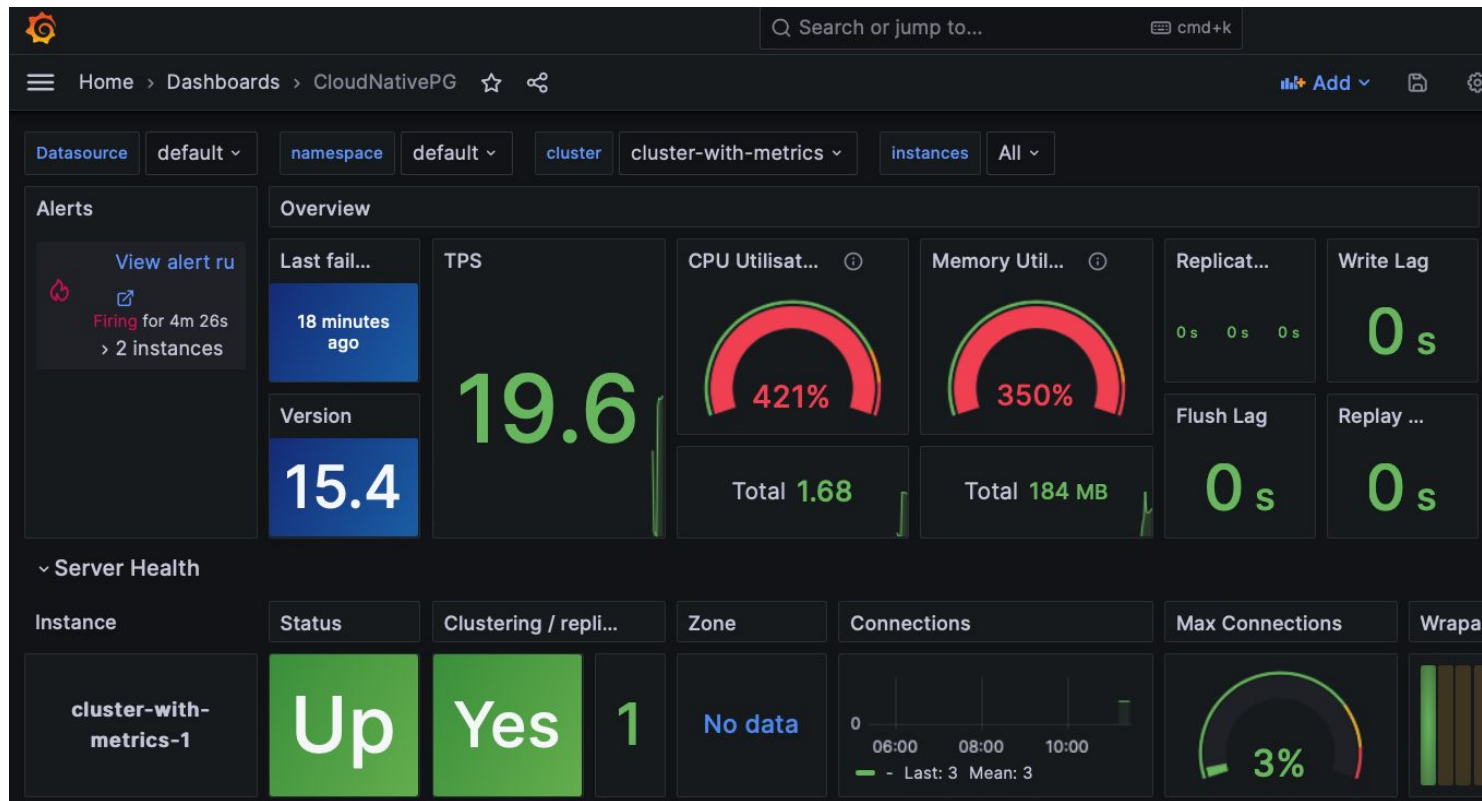


Native Prometheus exporter for monitoring

- Built-in metrics at the operator level
- Built-in metrics at the Postgres instance level
 - Customizable metrics at the Postgres instance level
 - Via ConfigMap(s) and/or Secret(s)
 - Syntax compatible with the PostgreSQL Prometheus Exporter
 - Auto-discovery of databases
 - Queries are:
 - transactionally atomic and read-only
 - executed with the `pg_monitor` role
 - executed with `application_name` set to `cnp_metrics_exporter`
- Support for `pg_stat_statements` and `auto_explain`



Grafana Dashboard



Logging

- All components directly log to standard output in JSON format
- Each entry has the following structure:
 - level: log level (info, notice, ...)
 - ts: the timestamp (epoch with microseconds)
 - logger: the type of the record (e.g. postgres or pg_controldata)
 - msg: the type of the record (e.g. postgres or pg_controldata)
 - record: the actual record (with structure that varies depending on the msg type)
- Seamless integration with many log management stacks in Kubernetes
- Support for PGAudit
 - EDB Audit as well for EDB Postgres for Kubernetes



The “cnpg” plugin for kubectl

- The official CLI for CloudNativePG
 - Available also as RPM or Deb package
- Extends the ‘kubectl’ command:
 - Customize the installation of the operator
 - Status of a cluster
 - Perform a manual switchover (promote a standby) or a restart of a node
 - Issue TLS certificates for client authentication
 - Declare start and stop of a Kubernetes node maintenance
 - Destroy a cluster and all its PVC
 - Fence a cluster or a set of the instances
 - Hibernate a cluster
 - Generate jobs for benchmarking via pgbench and fio
 - Issue a new backup
 - Start pgadmin



Name: cluster-example
Namespace: default
System ID: 7100921006673293335
PostgreSQL Image: ghcr.io/cloudnative-pg/postgresql:14.3
Primary instance: cluster-example-2
Status: Cluster in healthy state
Instances: 3
Ready instances: 3
Current Write LSN: 0/C000060 (Timeline: 4 - WAL File: 00000004000000000000000C)

Certificates Status

Certificate Name	Expiration Date	Days Left Until Expiration
cluster-example-replication	2022-08-21 13:15:00 +0000 UTC	89.95
cluster-example-server	2022-08-21 13:15:00 +0000 UTC	89.95
cluster-example-ca	2022-08-21 13:15:00 +0000 UTC	89.95

Continuous Backup status

First Point of Recoverability: 2022-05-23T13:37:08Z
Working WAL archiving: OK
WALs waiting to be archived: 0
Last Archived WAL: 00000004000000000000000B @ 2022-05-23T13:42:09.37537Z
Last Failed WAL: -

Streaming Replication status

Name	Sent LSN	Write LSN	Flush LSN	Replay LSN	Write Lag	Flush Lag	Replay Lag	State	Sync State	Sync Priority
cluster-example-3	0/C000060	0/C000060	0/C000060	0/C000060	00:00:00	00:00:00	00:00:00	streaming	async	0
cluster-example-1	0/C000060	0/C000060	0/C000060	0/C000060	00:00:00	00:00:00	00:00:00	streaming	async	0

Instances status

Name	Database Size	Current LSN	Replication role	Status	QoS	Manager Version
cluster-example-3	33 MB	0/C000060	Standby (async)	OK	BestEffort	1.15.0
cluster-example-2	33 MB	0/C000060	Primary	OK	BestEffort	1.15.0
cluster-example-1	33 MB	0/C000060	Standby (async)	OK	BestEffort	1.15.0



Connection pooling with PgBouncer

- Managed by the “Pooler” Custom Resource Definition
- Directly mapped to a service of a given Postgres cluster
- Deploys multiple instances of PgBouncer for High Availability
- Supports Pod templates, very important for Pod scheduling rules
- Transparent support for password authentication
- Connects to PostgreSQL via a standard user through a TLS certificate
- Supports configuration for most of PgBouncer options
- Automated integration with Prometheus
- JSON log in standard output





Thank you for participating in the
Postgres on Kubernetes Workshop

