

YugabyteDBの始め方

YugabyteDBの概要から、 OSS版・有償版の使い方まで徹底解説

2024/10/30

株式会社SRA OSS
彭 博 (ペン ボ)

ペン ボ

- 名前： 彭 博 (Bo Peng)
pengbo@sraoss.co.jp
- 所属：株式会社SRA OSS
技術部 基盤技術グループ
- 職務：
 - OSS技術サポート、ミドルウェア構築
 - クラスタリングソフトウェア：Pacemaker/Corosync
 - 監視ソフトウェア：Zabbix
 - 分散データベース：YugabyteDB
 - など
 - OSSコミュニティ活動



- YugabyteDBの概要・機能
- YugabyteDB製品紹介
- YugabyteDB OSS版の使い方
- YugabyteDB Aeon (Self-Managed) の使い方

RDBMS (SQL)

例: PostgreSQL、MySQL、Oracleなど

SELECT

SQL



テーブル

NoSQL

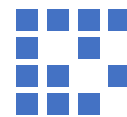
例: Cassandra、Redis、MongoDBなど



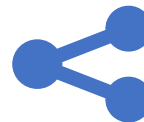
キー
バリュー



ドキュ
メント



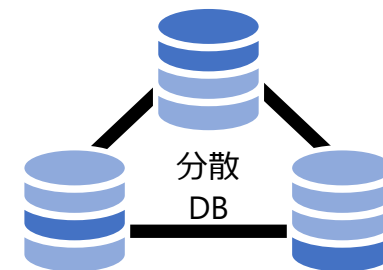
ワイド
カラム



グラフ

分散DB (NewSQL)

例: **YugabyteDB**、TiDBなど



高可用性



水平方向拡張性



ACIDトランザ
クション



SQL対応

RDBMS



NoSQL



分散DB (NewSQL)





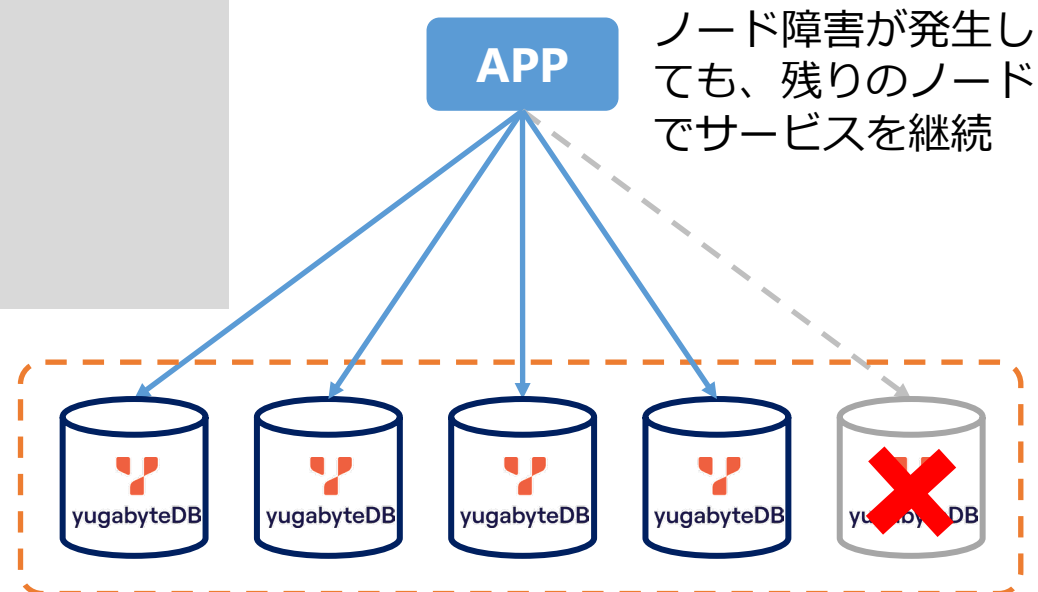
PostgreSQL・Cassandra互換

ACIDトランザクション

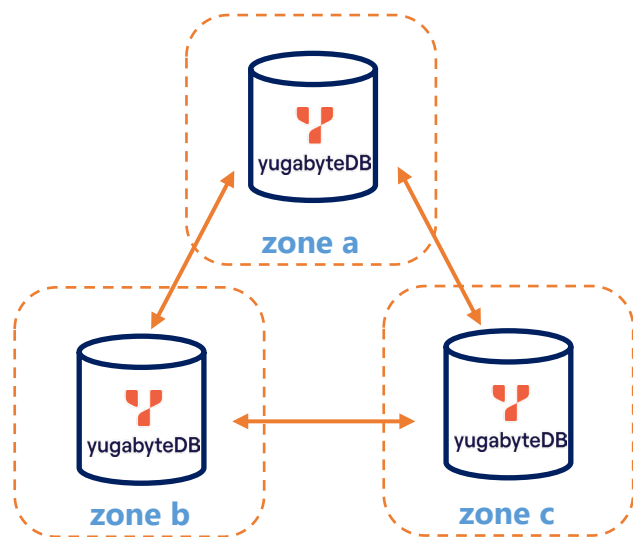
- RDBMSのPostgreSQLおよびNoSQLのCassandraと高い互換性を持っている
- ACIDトランザクションをサポートしており、強力なデータ一貫性を担保できる
- SQLを使って複雑な集計や検索が可能

高可用性

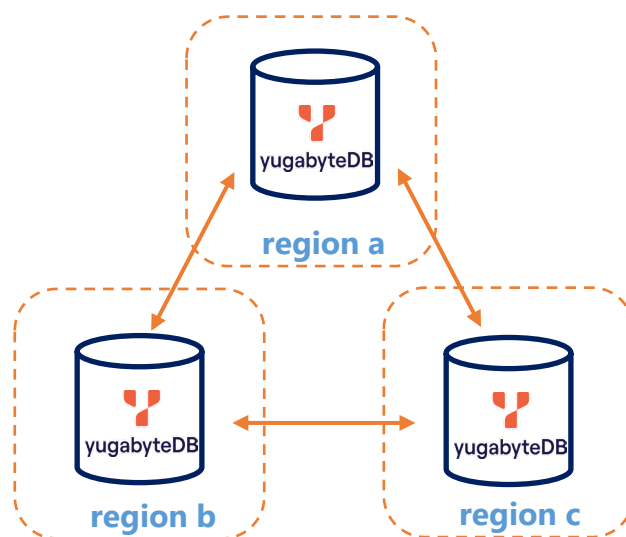
- 高可用性機能がビルトインされている
- ユーザにノード障害を意識させることなく運用
続行可能
 - 更新・参照処理は透過的にリトライされる
- 障害時データ損失なし
- ダウンタイムなしで
 - スケールアウト/スケールイン
 - バージョンアップなどのメンテナンス



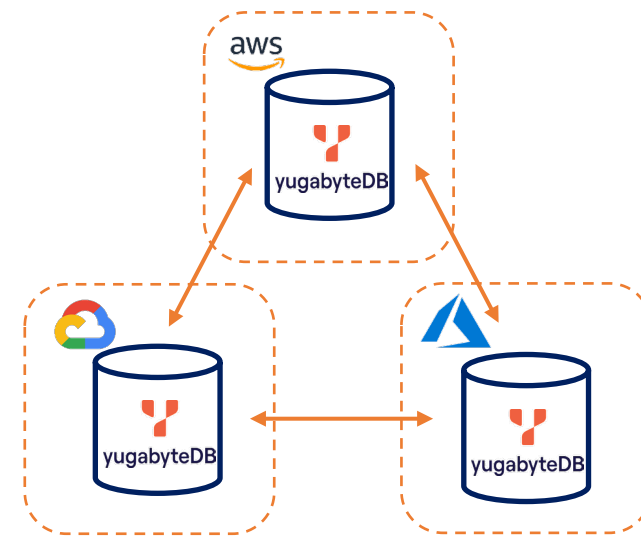
AZレベルからクラウドサービスレベルまでの耐障害性



マルチAZ
AZレベルの耐障害性

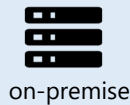


マルチリージョン
リージョンレベルの耐障害性



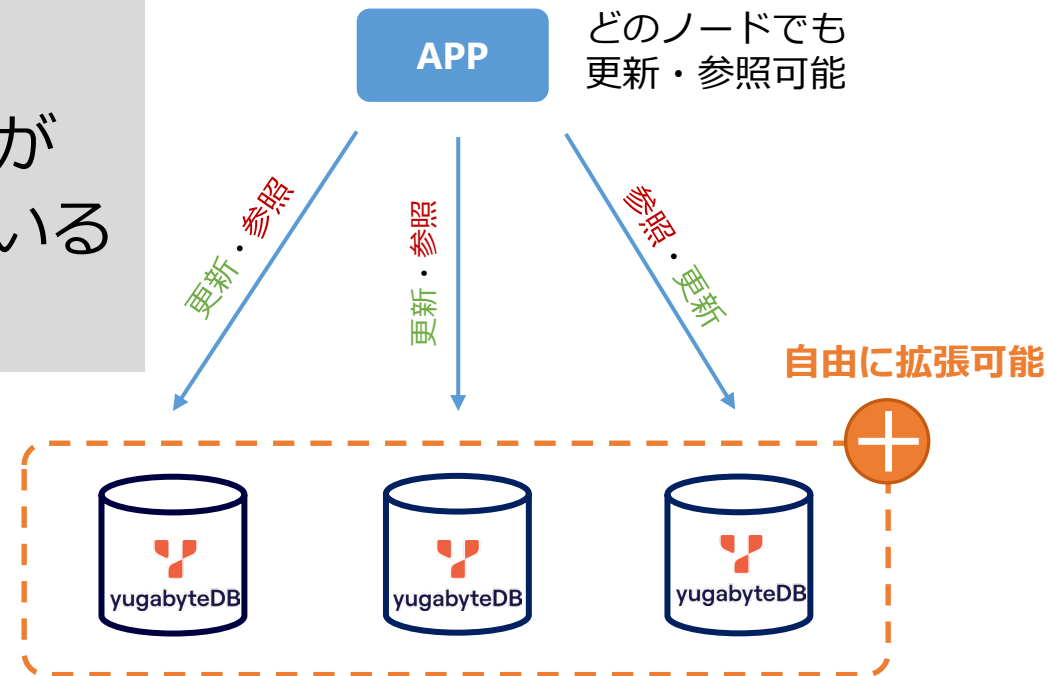
マルチクラウド
クラウドレベルの耐障害性

あらゆる環境で利用可能
(フルマネージドサービスの提供、マルチクラウド、ハイブリッドクラウドにも対応)



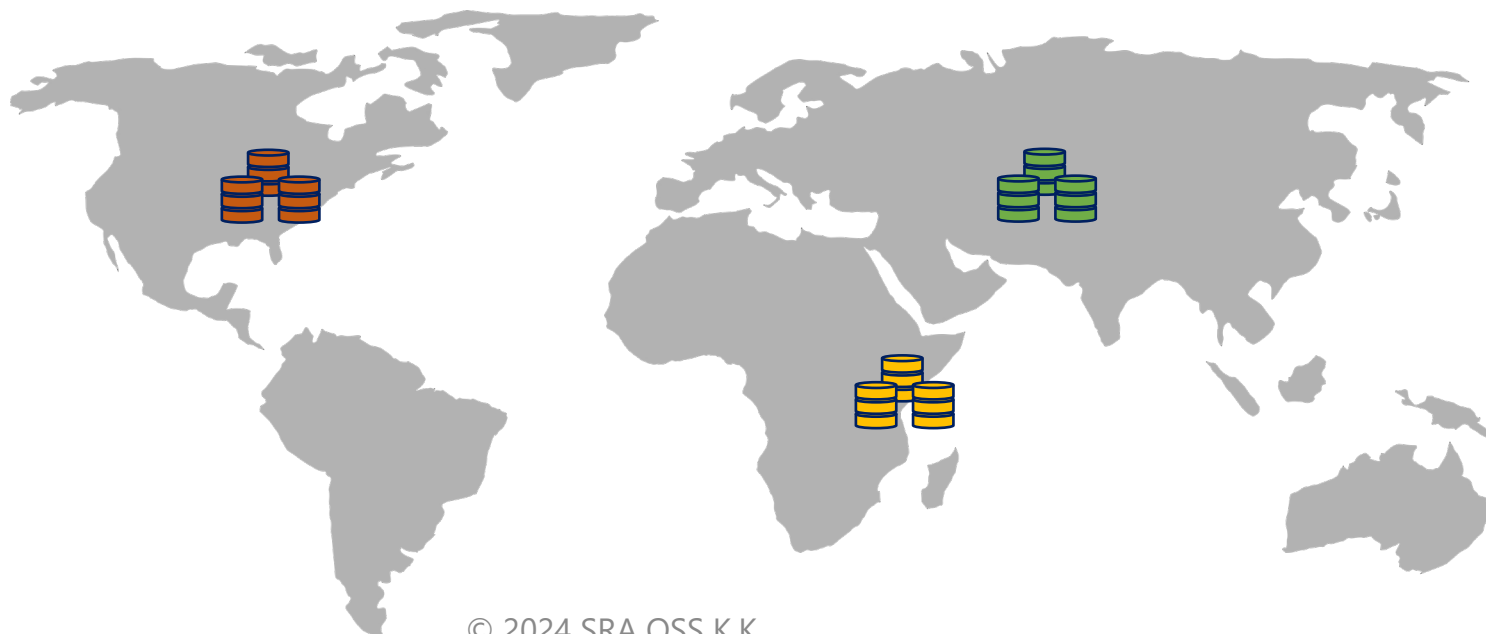
水平方向のスケラビリティ

- アクセス負荷の増減に応じて、ノードを追加/削除するだけで、簡単にスケールアウト/スケールインできる
- 更新、参照ともにスケールアウト可能
- 膨大なデータの処理、今後もデータ量の増加が予想されるようなアプリケーションに適している
- 分散トランザクション対応



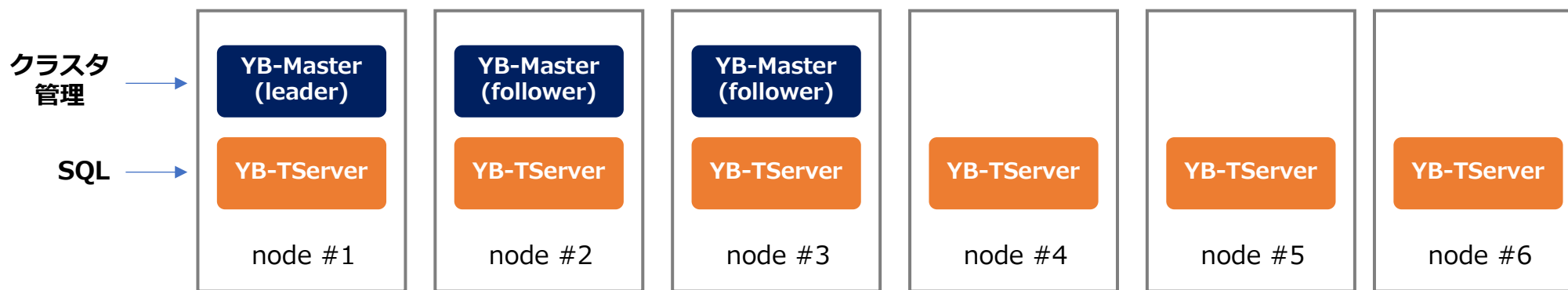
地理分散

- 地理的に分散した環境でデータベースをデプロイすることで、低レイテンシを実現できる
- 個人情報などの機密情報を特定の地域のデータセンターに保管することで、データの地理的な配置場所を制御できる



- **YB-Master**と**YB-TServer**の2つコンポーネントから構成されている
 - YB-Master
 - クラスタ全体のメタデータの管理、構成管理
 - 最大replication factorの数までしか作成されない
 - YB-TServer
 - アプリケーションからのクエリを処理して、データとして保持

replication factor = 3の場合



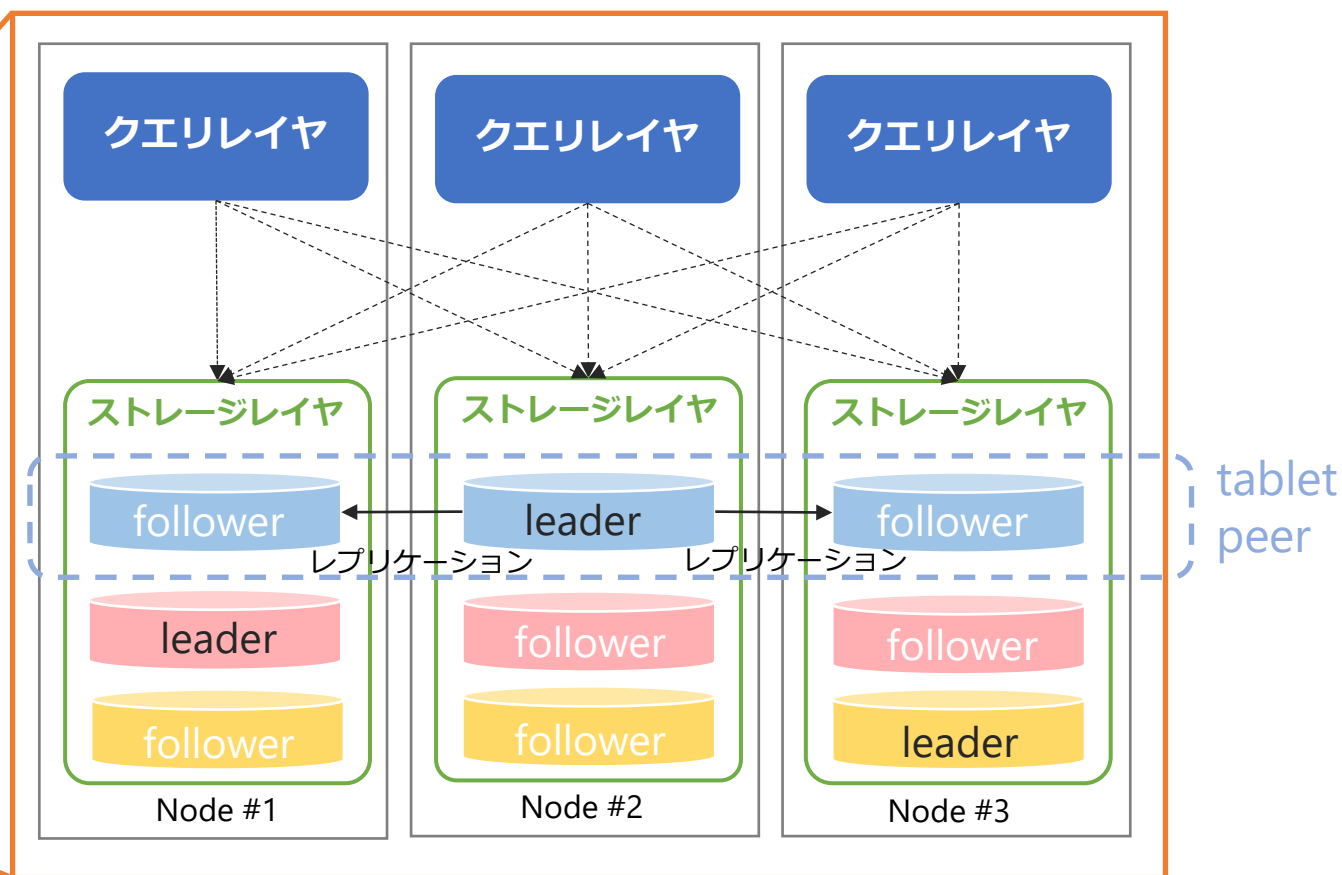
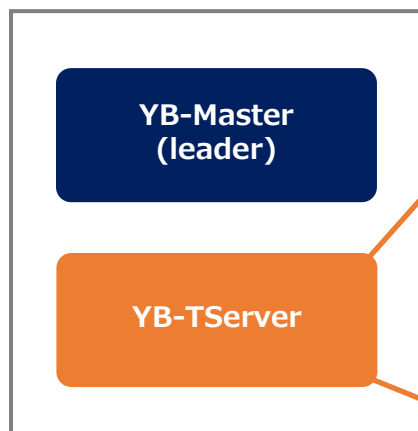
クエリレイヤ

- リクエストを受け付ける
- どのノードのストレージレイヤにも接続できる

ストレージレイヤ

- Tabletと呼ばれるシャーディング単位で分割
- 各tablet peerは1つのleaderと複数のfollowerから構成される
- tablet単位でレプリケーション
- 同一tablet peerの中でRaft合意形成が行われる

データベースノード



	OSS版	YugabyteDB Aeon		
		Self-Managed (旧 YugabyteDB Anywhere)	Bring Your Own Cloud (BYOC)	Full-Managed
概要	コアとなる基本的な機能	コア機能に加え、構築やDay2運用の自動化機能が追加されている ・ GUI管理コンソール ・ バックアップの自動化 ・ モニタリング・アラート ・ 無停止アップグレード ・ など	既存のクラウドインフラストラクチャを活用しつつ、YugabyteDBマネージドサービスの利点も享受できる	フルマネージドサービス ・ GUI管理コンソール ・ バックアップの自動化 ・ モニタリング・アラート ・ 無停止アップグレード ・ など
稼働環境	任意の環境 (オンプレミス、クラウド、Kubernetesなど)	任意の環境 (オンプレミス、クラウド、Kubernetesなど)	お客様のパブリッククラウドVPC	Yugabyteパブリッククラウド (AWS、GCP、Azure)
費用	無償	vCPUコアベースの年間サブスクリプション	vCPUコアベースの年間サブスクリプション	使用量課金
サポート	コミュニティサポート	24x365エンタープライズサポート	24x365エンタープライズサポート	24x365エンタープライズサポート

YugabyteDB OSS版の使い方

※ 2024年10月30日時点のYugabyteDBの対応OS

最新情報は <https://docs.yugabyte.com/preview/reference/configuration/operating-systems/> を参照

OS	x86	ARM	備考
AlmaLinux 8	✓	✓	- 本番環境や開発環境に推奨 - YugabyteDB AnywhereノードのデフォルトのOS
AlmaLinux 9	✓	✓	
Oracle Linux 8	✓		
Red Hat Enterprise Linux 8	✓		本番環境での使用が推奨される
Red Hat Enterprise Linux 8 CIS Hardened	✓		
Red Hat Enterprise Linux 9.3 and later	✓		v2.20.3以降でサポートされている (Early Access)
Red Hat Enterprise Linux 9 CIS Hardened	✓		v2.20.3以降でサポートされている (Early Access)
SUSE Linux Enterprise Server 15 SP5	✓		Early Access
Ubuntu 20	✓	✓	
Ubuntu 22	✓	✓	v2.18.5、v2.20.1でサポートされている

- NTPサーバのインストール
 - 分散データベースの特性上、各ノードの時刻が正確に同期されている必要がある

```
$ sudo yum install -y ntp または sudo yum install -y chrony
```

- ulimitsの推奨設定

```
#/etc/security/limits.conf
*          -          core           unlimited
*          -          data           unlimited
*          -          fsize          unlimited
*          -          sigpending     119934
*          -          memlock        64
*          -          rss            unlimited
*          -          nofile         1048576
*          -          msgqueue       819200
*          -          stack          8192
*          -          cpu            unlimited
*          -          nproc          12000
*          -          locks          unlimited
```

- python 3のインストール

```
$ python -version
```

Python 3.6.8

(pythonコマンドがPython 2に設定されている場合、Python 3を指すように設定が必要)

```
$ sudo alternatives --set python /usr/bin/python3
```

デフォルトポート一覧

Service	Type	Port
YB-Master	RPC	7100
YB-Master	Admin web server	7000
YB-TServer	RPC	9100
YB-TServer	Admin web server	9000
YCQL	RPC	9042
YCQL	Admin web server	12000
YSQL	RPC	5433
YSQL	Admin web server	13000

① YB-Masterのデプロイ



② YB-TServerのデプロイ

- YB-Masterは最大replication factorの数までしか作成されない
- ディスク障害に対応するために--fs_data_dirsはカンマ区切りで複数指定可

```
yb-master ¥  
  --master_addresses <ip1:7100,ip2:7100,ip3:7100> ¥  
  --rpc_bind_addresses <それぞれのノード自身のIP:7100> ¥  
  --fs_data_dirs "/mnt/yugabyte" ¥  
  --replication_factor=3 &
```

- YB-TServerの数はreplication factorと同じか、それ以上である必要がある

```
yb-tserver ¥  
  --tserver_master_addrs <ip1:7100,ip2:7100,ip3:7100> ¥  
  --rpc_bind_addresses <それぞれのノード自身のIP:9100> ¥  
  --enable_ysql ¥  
  --fs_data_dirs "/mnt/yugabyte" &
```

あるいは、設定フラグをファイルに書き込み、--flagfileを使用して
yb-masterとyb-tserverを実行することも可能

```
yb-master --flagfile master.conf &  
yb-tserver --flagfile tserver.conf &
```

YugabyteDBのクラスタ管理コマンド**yb-admin**を用いてYB-MasterとYB-TServerの状態を確認

- 1台のYB-MasterがLEADERに選出され、他の2台がFOLLOWERとして稼働している
- すべてのYB-TServerがALIVE状態で稼働している

YB-Masterの状態

```
yb-admin --master_addresses "172.31.71.100:7100,172.31.72.100:7100,172.31.73.100:7100" list_all_masters
Master UUID RPC Host/Port State Role Broadcast Host/Port
8709f57eb8e14433b9b2b0e6018d1b63 172.31.71.100:7100 ALIVE LEADER N/A
f50db74c5461467d8dc96fa88b4b89f4 172.31.72.100:7100 ALIVE FOLLOWER N/A
632005861c4146aa95dc031e72cdb67f 172.31.73.100:7100 ALIVE FOLLOWER N/A
```

YB-TServerの状態

```
yb-admin --master_addresses "172.31.71.100:7100,172.31.72.100:7100,172.31.73.100:7100" list_all_tablet_servers
Tablet Server UUID RPC Host/Port Heartbeat delay Status Reads/s Writes/s Uptime SST total size SST uncomp size
SST #files Memory Broadcast Host/Port
1b66c3cac2154d7fa8ea6f823dd346f6 172.31.73.100:9100 0.56s ALIVE 0.00 0.00 10 0 B 0 B 0 36.58 MB N/A
16a4004527284f4da35ac5187ca55089 172.31.72.100:9100 0.94s ALIVE 0.00 0.00 35 0 B 0 B 0 32.42 MB N/A
162f28708ff24259a68bc517c25cba3e 172.31.71.100:9100 0.94s ALIVE 0.00 0.00 65 0 B 0 B 0 32.87 MB N/A
```

ysqlshによる接続

- YSQL (YugabyteDBのPostgreSQL互換SQLインターフェース) に接続するためのコマンド
- 指定オプションはPostgreSQLのpsqlとほとんど同じ
- デフォルトのポートは5433

```
$ ysqlsh -h <いずれかのDBノード> -U yugabyte -d yugabyte
ysqlsh (11.2-YB-2024.1.2.0-b0)
Type "help" for help.
yugabyte=# CREATE DATABASE test;
yugabyte=# \c test
test=# CREATE TABLE t1 (c1 int primary key, c2 text);
CREATE TABLE
test=# \d t1
Table "public.t1"
Column | Type | Collation | Nullable | Default
-----+-----+-----+-----+-----
c1 | integer | | not null |
c2 | text | | |
Indexes:
"t1_pkey" PRIMARY KEY, lsm (c1 HASH)
```

アプリケーションからの接続

- 既存のPostgreSQLドライバー(例えば、Javaアプリケーション向けのPostgreSQL JDBC Driverなど)を使用してYugabyteDBに接続することは可能だが、スケールアウトやノード障害が発生した際、ノードのIPアドレスが動的に変わる
- この問題を解決するためには、以下のいずれかの対応が必要
 1. Smart Driverの利用
 - <https://docs.yugabyte.com/preview/drivers-orms/smart-drivers/>
 - Javaアプリケーション用のSmart Driverが用意されている
 - PostgreSQL JDBC Driverをベースに作られている
 2. YugabyteDBクラスタの前にLBを配置してルーティングさせる
 3. アプリ側で無効になった接続をリフレッシュしてリトライする処理を実装

YugabyteDBの管理

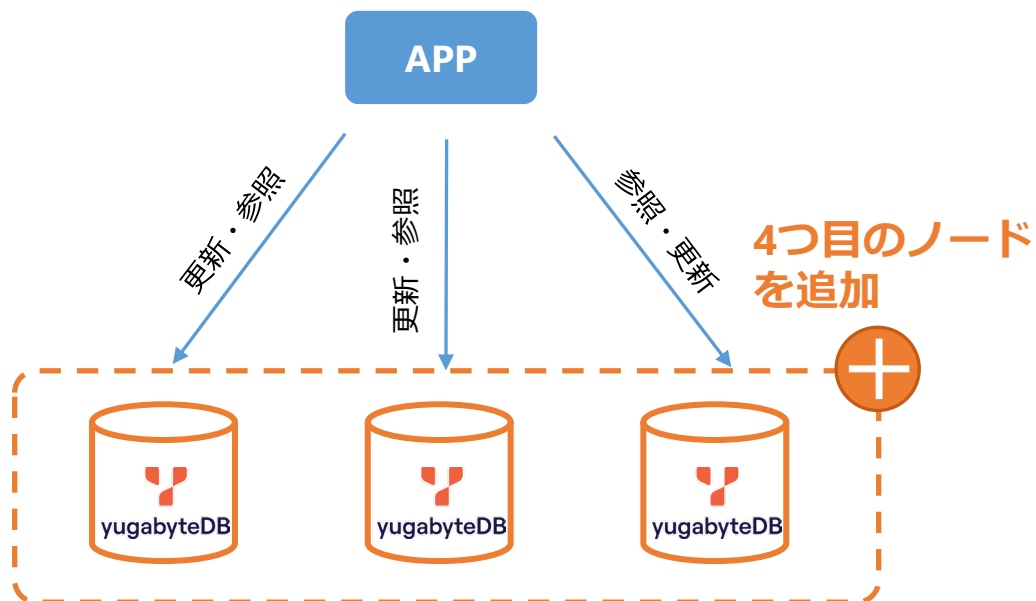
- YB-MasterとYB-TServerの設定変更
- スケールアウト
- バックアップ・リストア
- モニタリング

- 起動時にのみ設定可能なフラグは、YB-MasterやYB-TServerの**再起動**が必要
- 一部のYB-TServerのフラグ設定は、**yb-ts-cli ... set_flag**を使用して、稼働中のYugabyteDBに反映させることができる
 - 例えば、--ysql_log_statement (PostgreSQLのlog_statementに相当する)

```
yb-ts-cli --server_address="<YB-TServerのIP>:9100" set_flag ysql_log_statement all
```

```
# 上記コマンドを実行すると、該当YB-TServerのログに以下のようなメッセージが出力される  
2024-10-05 22:44:59.986 JST [150696] LOG:  received SIGHUP, reloading configuration files  
2024-10-05 22:44:59.987 JST [150696] LOG:  parameter "log_statement" changed to "all"
```

アクセス負荷の増減に応じて、ノードを追加/削除するだけで、簡単にスケールアウト/インできる



4つ目のノードをクラスタに追加

```
yb-tserver ¥  
--tserver_master_addrs <ip1:7100,ip2:7100,ip3:7100> ¥  
--rpc_bind_addresses <4つ目のノードのIP:9100> ¥  
--enable_ysql ¥  
--fs_data_dirs "/mnt/yugabyte" &
```

```
yb-admin --master_addresses <ip1:7100,ip2:7100,ip3:7100> list_all_tablet_servers
```

Tablet Server UUID	RPC Host/Port	Heartbeat delay	Status	Reads/s	Writes/s	Uptime	SST total size	SST uncomp size
1b66c3cac2154d7fa8ea6f823dd346f6	172.31.73.100:9100	0.21s	ALIVE	0.00	0.00	1118	0 B 0 B 0	46.16 MB N/A
dd21987f812b4a3b81e7c7b324321356	172.31.74.100:9100	0.21s	ALIVE	0.00	0.00	36	133.27 KB 133.33 KB 2	37.35 MB N/A
16a4004527284f4da35ac5187ca55089	172.31.72.100:9100	0.21s	ALIVE	0.00	0.00	1145	0 B 0 B 0	41.80 MB N/A
162f28708ff24259a68bc517c25cba3e	172.31.71.100:9100	0.22s	ALIVE	0.00	0.00	1172	66.65 KB 66.71 KB 1	51.26 MB N/A

バックアップ方式

- `mysql_dump`や`mysql_dumpall`を使った論理バックアップ
- 分散スナップショット

SRA OSS `mysql_dump`や`mysql_dumpall`を使った論理バックアップ

- データベースのバックアップをSQL形式で取得する
- PostgreSQLの`pg_dump`や`pg_dumpall`に相当する

コマンド	Type
<code>mysql_dump</code>	特定のデータベースやテーブルをバックアップ
<code>mysql_dumpall</code>	ユーザーやロールなどのグローバル設定を含めたクラスタ全体のバックアップ

バックアップ

```
mysql_dump -h <いずれかのDBノード> -U yugabyte -d mydb > mydb-dump.sql
```

リストア

```
mysqlsh -h <いずれかのDBノード> -U yugabyte -d new-db -f mydb-dump.sql
```

- バックアップする最も効率的な方法
- クラスタ全体にまたがってデータベースの**特定の時点での状態**をキャプチャする
- YugabyteDBクラスタ内のすべてのノードで同時に取得され、データの**一貫性が保証**される
- 関連するすべてのファイルに**ハードリンク**が作成される
- データが保存されている場所と**同じストレージに保存**される
- **定期スナップショット**機能も提供されている
 - PITRは、定期スナップショットを基にして過去の任意の時点にデータを戻すため、利用するには分散スナップショットが必要

スナップショットの作成

```
yb-admin --master_addresses <YB-Master-ip1:7100,YB-Master-ip2:7100,YB-Master-ip3:7100>  
create_database_snapshot mysql.<database_name>
```

スナップショットのリストア

```
yb-admin --master_addresses <YB-Master-ip1:7100,YB-Master-ip2:7100,YB-Master-ip3:7100>  
restore_snapshot <snapshot-UUID>
```

- ストレージの消費量の増加
- ファイルシステムの破損やハードウェア障害からの保護は提供されない

▶ スナップショットを外部ストレージに保存

- スナップショットを使ったリストアでは、データの変更のみが元に戻され、スキーマの変更は元に戻らない点に要注意

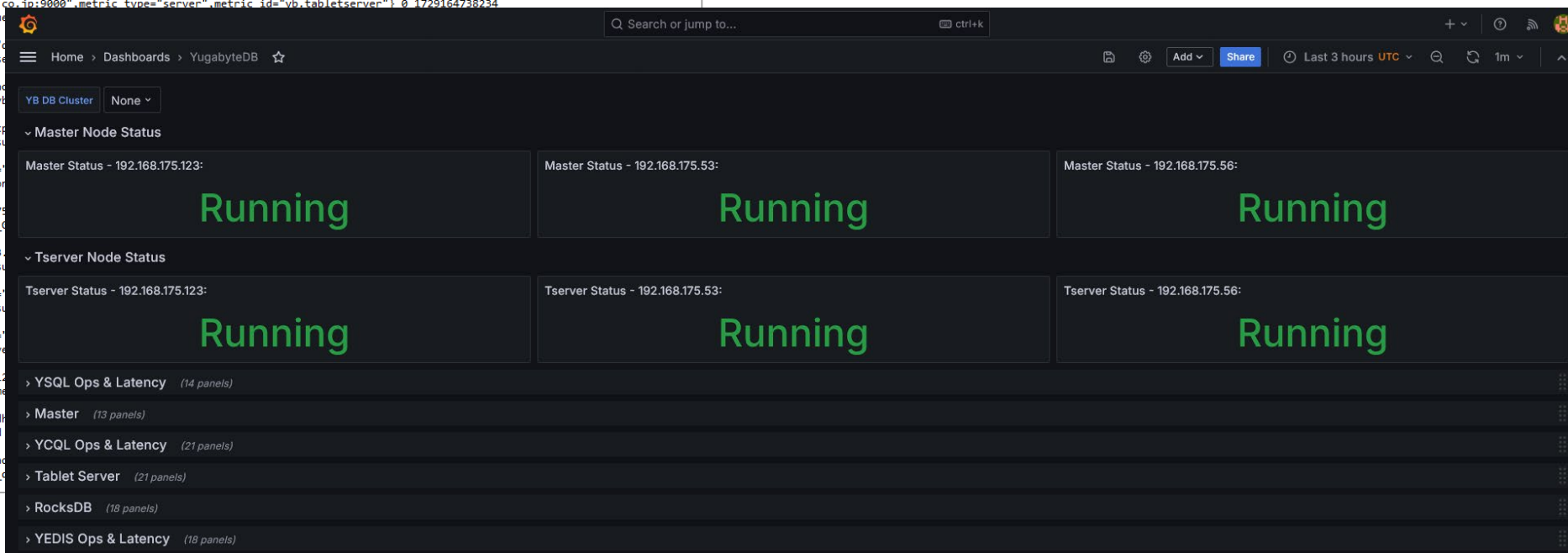
▶ スナップショットを外部ストレージに保存 または PITR (Point-In-Time Recovery) を使用

さまざまなメトリクスをJSONやPrometheus形式で出力

```
← → ↺ 🏠 192.168.175.123:9000/prometheus-metrics

# HELP mem_tracker_server_Read_Buffer_Inbound_RPC_Sending Memory consumed by Sending->Inbound RPC->Read Buffer->server->root
# TYPE mem_tracker_server_Read_Buffer_Inbound_RPC_Sending gauge
mem_tracker_server_Read_Buffer_Inbound_RPC_Sending{exported_instance="dhcp-175-123.sraoss.co.jp:9000",metric_type="server",metric_id="yb.tabletserver"} 0 1729164738234
# HELP mem_tracker_server_Read_Buffer_Inbound_RPC_Reading Memory consumed by Reading->Inbound RPC->Read Buffer->server->root
# TYPE mem_tracker_server_Read_Buffer_Inbound_RPC_Reading gauge
mem_tracker_server_Read_Buffer_Inbound_RPC_Reading{exported_instance="dhcp-175-123.sraoss.co.jp:9000",metric_type="server",metric_id="yb.tabletserver"} 0 1729164738234
# HELP mem_tracker_server_Tablets_overhead_Transactions Memory consumed by Transactions->Tablets_overhead->server->root
# TYPE mem_tracker_server_Tablets_overhead_Transactions gauge
mem_tracker_server_Tablets_overhead_Transactions{exported_instance="dhcp-175-123.sraoss.co.jp:9000",metric_type="server",metric_id="yb.tabletserver"} 0 1729164738234
# HELP mem_tracker_server_Read_Buffer_Inbound_RPC_Receive Memory consumed by Receive->Inbound RPC->Read Buffer->server->root
# TYPE mem_tracker_server_Read_Buffer_Inbound_RPC_Receive gauge
mem_tracker_server_Read_Buffer_Inbound_RPC_Receive{exported_instance="dhcp-175-123.sraoss.co.jp:9000",metric_type="server",metric_id="yb.tabletserver"} 6969968 1729164738234
# HELP mem_tracker_server_Read_Buffer_Memory consumed by Read Buffer->server->root
# TYPE mem_tracker_server_Read_Buffer_Memory gauge
mem_tracker_server_Read_Buffer_Memory{exported_instance="dhcp-175-123.sraoss.co.jp:9000",metric_type="server",metric_id="yb.tabletserver"} 0 1729164738234
# HELP rpcs_in_queue_yb_tserver_TabletServerBackupService rpcs_in_queue_yb_tserver_TabletServerBackupService gauge
# TYPE rpcs_in_queue_yb_tserver_TabletServerBackupService gauge
rpcs_in_queue_yb_tserver_TabletServerBackupService{exported_instance="dhcp-175-123.sraoss.co.jp:9000",metric_type="server",metric_id="yb.tabletserver"} 0 1729164738234
# HELP mem_tracker_server_PgResponseCache Memory consumed by PgResponseCache
# TYPE mem_tracker_server_PgResponseCache gauge
mem_tracker_server_PgResponseCache{exported_instance="dhcp-175-123.sraoss.co.jp:9000",metric_type="server",metric_id="yb.tabletserver"} 0 1729164738234
# HELP rpcs_in_queue_yb_tserver_RemoteBootstrapService rpcs_in_queue_yb_tserver_RemoteBootstrapService gauge
# TYPE rpcs_in_queue_yb_tserver_RemoteBootstrapService gauge
rpcs_in_queue_yb_tserver_RemoteBootstrapService{exported_instance="dhcp-175-123.sraoss.co.jp:9000",metric_type="server",metric_id="yb.tabletserver"} 0 1729164738234
# HELP mem_tracker_server_Read_Buffer_Outbound_RPC_Reading Memory consumed by Reading->Outbound RPC->Read Buffer->server->root
# TYPE mem_tracker_server_Read_Buffer_Outbound_RPC_Reading gauge
mem_tracker_server_Read_Buffer_Outbound_RPC_Reading{exported_instance="dhcp-175-123.sraoss.co.jp:9000",metric_type="server",metric_id="yb.tabletserver"} 0 1729164738234
# HELP rpcs_in_queue_yb_consensus_ConsensusService rpcs_in_queue_yb_consensus_ConsensusService gauge
# TYPE rpcs_in_queue_yb_consensus_ConsensusService gauge
rpcs_in_queue_yb_consensus_ConsensusService{exported_instance="dhcp-175-123.sraoss.co.jp:9000",metric_type="server",metric_id="yb.tabletserver"} 0 1729164738234
# HELP rpcs_in_queue_yb_server_GenericService rpcs_in_queue_yb_server_GenericService gauge
# TYPE rpcs_in_queue_yb_server_GenericService gauge
rpcs_in_queue_yb_server_GenericService{exported_instance="dhcp-175-123.sraoss.co.jp:9000",metric_type="server",metric_id="yb.tabletserver"} 0 1729164738234
# HELP mem_tracker_server_Read_Buffer_Outbound_RPC_Sending Memory consumed by Sending->Outbound RPC->Read Buffer->server->root
# TYPE mem_tracker_server_Read_Buffer_Outbound_RPC_Sending gauge
mem_tracker_server_Read_Buffer_Outbound_RPC_Sending{exported_instance="dhcp-175-123.sraoss.co.jp:9000",metric_type="server",metric_id="yb.tabletserver"} 0 1729164738234
# HELP mem_tracker_server_Read_Buffer_Outbound_RPC_Receive Memory consumed by Receive->Outbound RPC->Read Buffer->server->root
# TYPE mem_tracker_server_Read_Buffer_Outbound_RPC_Receive gauge
mem_tracker_server_Read_Buffer_Outbound_RPC_Receive{exported_instance="dhcp-175-123.sraoss.co.jp:9000",metric_type="server",metric_id="yb.tabletserver"} 0 1729164738234
# HELP rpcs_in_queue_yb_tserver_PgClientService rpcs_in_queue_yb_tserver_PgClientService gauge
# TYPE rpcs_in_queue_yb_tserver_PgClientService gauge
rpcs_in_queue_yb_tserver_PgClientService{exported_instance="dhcp-175-123.sraoss.co.jp:9000",metric_type="server",metric_id="yb.tabletserver"} 0 1729164738234
# HELP mem_tracker_server_Compressed_Read_Buffer_Receive Memory consumed by Receive->Compressed Read Buffer->server->root
# TYPE mem_tracker_server_Compressed_Read_Buffer_Receive gauge
mem_tracker_server_Compressed_Read_Buffer_Receive{exported_instance="dhcp-175-123.sraoss.co.jp:9000",metric_type="server",metric_id="yb.tabletserver"} 0 1729164738234
# HELP mem_tracker_server_Read_Buffer_CQL Memory consumed by CQL->Read Buffer->server->root
# TYPE mem_tracker_server_Read_Buffer_CQL gauge
mem_tracker_server_Read_Buffer_CQL{exported_instance="dhcp-175-123.sraoss.co.jp:9000",metric_type="server",metric_id="yb.tabletserver"} 0 1729164738234
# HELP rpcs_in_queue_yb_stateful_service_PgAutoAnalyzeService rpcs_in_queue_yb_stateful_service_PgAutoAnalyzeService gauge
# TYPE rpcs_in_queue_yb_stateful_service_PgAutoAnalyzeService gauge
rpcs_in_queue_yb_stateful_service_PgAutoAnalyzeService{exported_instance="dhcp-175-123.sraoss.co.jp:9000",metric_type="server",metric_id="yb.tabletserver"} 0 1729164738234
```

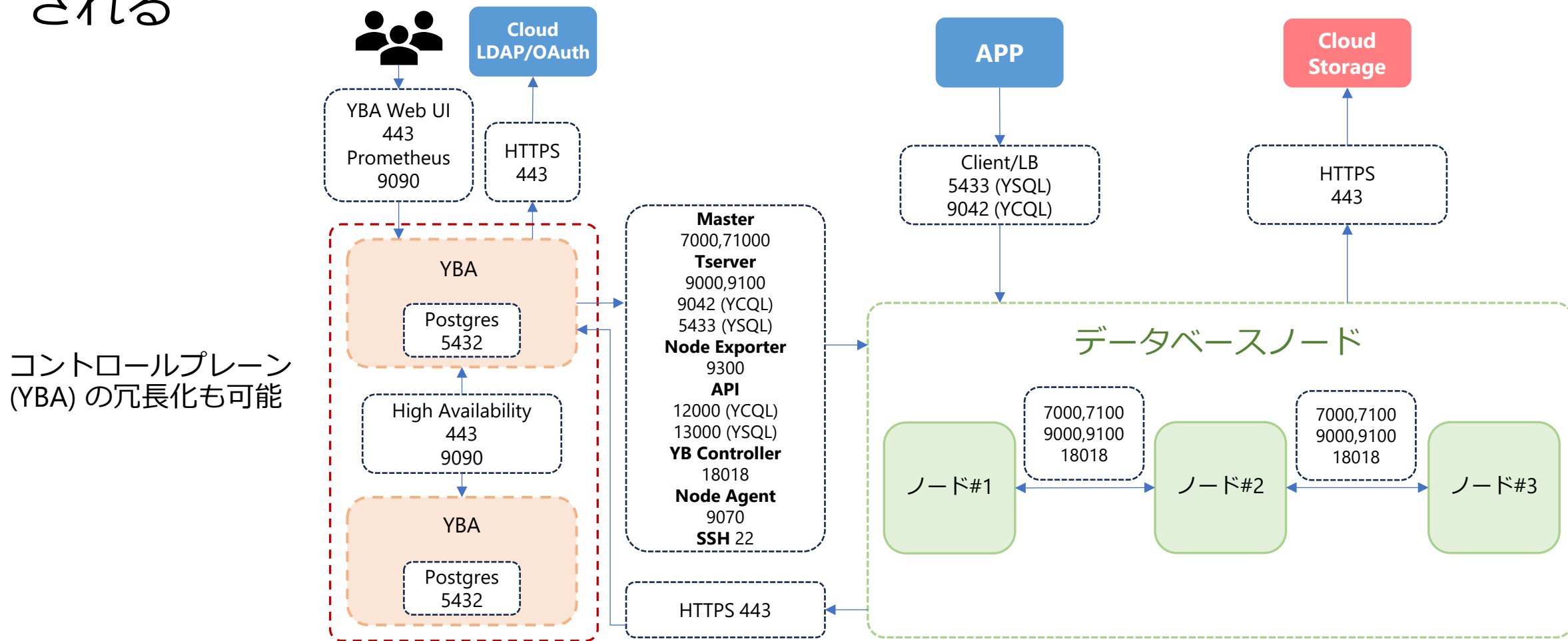
PrometheusやGrafanaと連携してデータを可視化することも可能だが、これらのツールは別途構築する必要がある



YugabyteDB Aeon (Self-Managed DBaaS) の使い方 (旧 YugabyteDB Anywhere)

YugabyteDB Anywhereの構成

YugabyteDB Anywhereはデータベースクラスタのデプロイと管理を行う**コントロールプレーン**(以下、**YBA**)と**データベースノード**で構成される

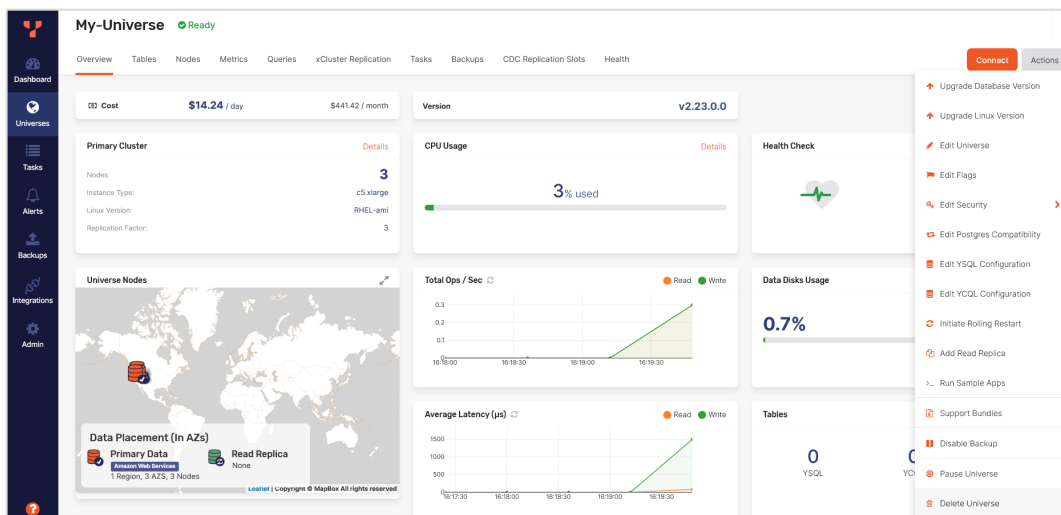




Web UIから行う



自動化ツールを利用する

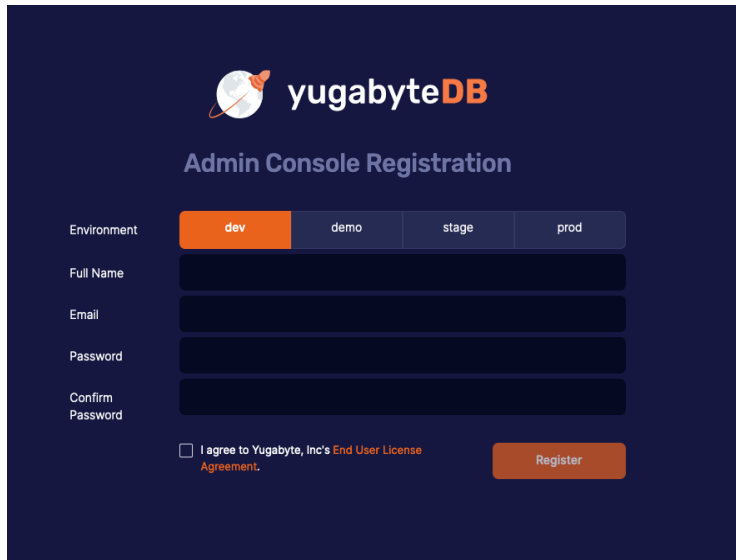


- REST API
- Terraform
- CLI
- Kubernetes Operator

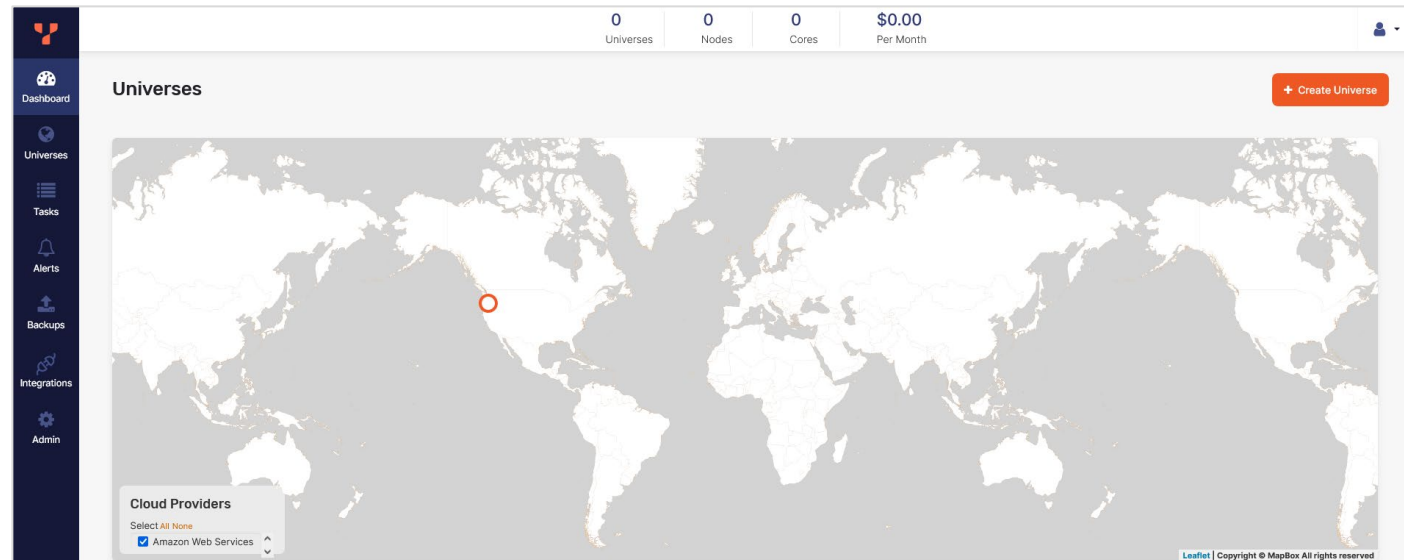
- YugabyteDB Anywhereのライセンス
- サーバの準備
 - YBAをインストールする**サーバ**の準備
 - データベースノード用の**サーバ**を準備する (オンプレミスの場合)
 - クラウド利用する場合は、自動的に作成される
- ネットワークの設定
 - クラウドを利用する場合、利用するVPCやサブネット、セキュリティグループをあらかじめ設定しておく
 - コントロールプレーンとデータベースノードの特定のポート (構成図参照) へのアクセスを許可しておく

① YBAのインストール

```
wget https://downloads.yugabyte.com/releases/2024.1.3.0/yba_installer_full-2024.1.3.0-b105-linux-x86_64.tar.gz
tar -xf yba_installer_full-2024.1.3.0-b105-linux-x86_64.tar.gz
cd yba_installer_full-2024.1.3.0-b105/
sudo ./yba-ctl preflight
sudo ./yba-ctl install -l /path/to/license
```

② ブラウザからYBAの管理画面 <https://<yugabytedbanywhere-host-ip>/register> にアクセスし、Adminユーザを作成し、ログインする

The registration form for the YugabyteDB Admin Console. It features the YugabyteDB logo at the top. Below the logo, the title "Admin Console Registration" is displayed. The form includes several input fields: "Environment" with tabs for "dev", "demo", "stage", and "prod"; "Full Name"; "Email"; "Password"; and "Confirm Password". At the bottom, there is a checkbox for "I agree to Yugabyte, Inc's End User License Agreement." and a "Register" button.



Create AWS Provider Configuration

Provider Name

Cloud Info ⓘ

Credential Type ⓘ ☒ Special Access Key

Access Key ID

Secret Access Key

Use AWS Route 53 DNS Server ☐

+ Add Region

Region

VPC ID

Security Group ID

+ Add Zone

ap-northeast-1a	subnet-066
ap-northeast-1c	subnet-0313
ap-northeast-1d	subnet-035

Cancel

+ Add Linux Version

1. Give this Linux version an easy-to-recognize name

2. Choose CPU architecture

☒ x86 (Intel) ☐ AArch64 (ARM)

3. Enter the Amazon Machine Image (AMI) ID for each region

REGION	AMI ID
Asia Pacific (Tokyo)	ami-04d3ba818c434b384

4. Provide SSH user and SSH port details

SSH User

SSH Port

5. Other Configurations

Use IMDSv2 ⓘ ☐

Cancel Add Linux Version

UniverseはYugabyteDBクラスタ全体を管理する単位

Create universe 1. Primary Cluster 2. Read Replica

Cloud Configuration

Name: My-Universe

Provider: AWS

Regions: Asia Pacific (Tokyo)

Master Placement: ☒ Place Masters on the same nodes as T-Servers ☐ Place Masters on dedicated nodes

Total Nodes: 3

Replication Factor: 1 3 5 7

Availability Zones [Reset Config](#)

Name	Nodes	Preferred
ap-northeast-1a	1	☒
ap-northeast-1c	1	☒
ap-northeast-1d	1	☒

Primary data placement is geo-redundant, universe can survive at least 1 availability zone failure

Instance Configuration

CPU Architecture: x86 (Intel) AArch64 (ARM)

Linux Version: RHEL-ami Default

Instance Type: m6i.2xlarge (8 cores, 32GB RAM)

Volume Info: 1 x 250 GB

EBS Type: GP3

Provisioned IOPS: 3000

Provisioned Throughput: 125 MB/s

My-Universe Pending

Overview Tables Nodes Metrics Queries xCluster Replication **Tasks** Backups CDC Health

Task Progress

- 1 VALIDATING CONFIGURATIONS
- 2 PROVISIONING
- 3 **INSTALLING SOFTWARE**
- 4 UPDATING GFLAGS
- 5 STARTING MASTER PROCESS
- 6 STARTING NODE PROCESSES
- 7 CONFIGURING THE UNIVERSE
- 8 DONE

Current Step: Installing software

Configuring mount points, setting up the various directories and installing the YugaByte software on the newly provisioned nodes.

Task History

TYPE	NAME	STATUS	USER	START TIME	END TIME	NOTES
Create Universe	My-Universe	Pending (34%)	pengbo@sraoss.co.jp	Oct-25-2024 15:31:02 UTC+0900	-	Abort Task

Cloud Configuration

Name: My-Universe

Provider: AWS

Regions: US West (Oregon)

Master Placement:
☒ Place Masters on the same nodes as T-Servers
☐ Place Masters on dedicated nodes [When to use this setting?](#)

Total Nodes: 6

Replication Factor: 1 3 5 7

- Upgrade Database Version
- Upgrade Linux Version
- Edit Universe
- Edit Flags
- Edit Security
- Edit Postgres Compatibility
- Edit YSQL Configuration
- Edit YCQL Configuration
- Initiate Rolling Restart
- Add Read Replica
- Run Sample Apps
- Support Bundles
- Disable Backup
- Pause Universe
- Delete Universe

3ノードから6ノードにスケールアウト

Upgrade Database

1. Choose target version

Current Version: 2.23.0.0-b710

Target Version: Please select

2. Configure upgrade details

☒ Rolling upgrade (upgrade nodes one at a time)

Upgrade delay between Nodes: 180 Seconds

ローリングアップグレード

Primary Cluster / TSERVER / Add Flag

ysql_max_connections

Most used All Flags

ysql_max_connections

Flag Details

Name: ysql_max_connections

Description: Overrides the maximum number of concurrent ysql connections. If set to 0, Postgres will dynamically determine a platform-specific value

Default Value: 0

[More about this flag](#)

Flag Value: T-Server

0

フラグ設定変更

The screenshot displays the 'My-Universe' management interface. The 'Nodes' tab is selected, showing a table of three nodes in a 'Primary Cluster'. All nodes are 'Live' and running on AWS in the 'ap-northeast-1' region. An 'Actions' dropdown menu is open for the first node, listing various management tasks with corresponding red Japanese labels.

NAME	STATUS	CLOUD INFO	RAM USED	SST SIZE	UNCOMPRESSED SST SIZE	READ WRITE OPS/SEC	PROCESSES	ACTION
yb-prod-My-Universe-n1 172.31.71.130	Live 14 mins 12 secs	aws ap-northeast-1 / ap-northeast-1a	67.0 MB	0.0 B	0.0 B	0.00 0.00	Master TServer	Actions
yb-prod-My-Universe-n2 172.31.72.190	Live 14 mins 15 secs	aws ap-northeast-1 / ap-northeast-1c	68.0 MB	0.0 B	0.0 B	0.40 0.00		
yb-prod-My-Universe-n3 172.31.73.69	Live 14 mins 15 secs	aws ap-northeast-1 / ap-northeast-1d	64.1 MB	0.0 B	0.0 B	0.00 0.00		

- Connect ssh接続コマンド
- Show Live Queries ライブクエリ
- Show Slow Queries スロークエリ
- Stop Processes 停止
- Reboot Node 再起動
- Replace Node リプレイス
- Download Logs ログダウンロード
- Advanced >

Backups

Scheduled Backup Policies

Point-in-time Recovery

Restore History

オンデマンドバックアップ

Status:

All

More Filters

☐ Backup Type	Incremental Backups	API Type	Created At ▼	Expiration	Size
☐ On Demand	Not Present	YSQL	Oct-14-2024 16:54:17 UTC+0900	Oct-15-2024 16:54:26 UTC+0900	30.75 MB

Backups

Scheduled Backup Policies

Point-in-time Recovery

Restore History

10 ▼

定期バックアップ

バックアップを外部ストレージ（Amazon S3など）に保存可能

Backups

Scheduled Backup Policies

Point-in-time Recovery

Restore History

daily backup YSQL

SCOPE

Full Backup

BACKUP CONFIG

[my-s3](#)

PITR

Databases/Keyspaces with Point-In-Time Recovery Enabled

Database/Keyspace Name ▲	API Type ▼▲	Retention Period ▼▲
yugabyte	YSQL	7 Days

10 ▼

モニタリングやデータの可視化機能が標準搭載されているため、追加ツールの構築は不要



Live Queries

Slow Queries

Performance Advisor

Live Queries

実行中のクエリの表示
クエリ、経過時間、クライアント情報など

Node Name: yb-dev-My-Universe-n3 Filter by column or query text

NODE NAME ▾	PRIVATE IP ▾	DB NAME ▾	QUERY ▾	ELAPSED TIME (MS) ▾	CLIENT NAME ▾	CLIENT HOST ▾
☐ yb-dev-My-Universe-n3	172.31.30.16	yugabyte	SELECT abalance FROM ysql_bench_accounts WHERE ai...	0 ms	ysql_bench	54.245.78.252
☐ yb-dev-My-Universe-n3	172.31.30.16	yugabyte	UPDATE ysql_bench_tellers SET tbalance = tbalance ...	0 ms	ysql_bench	54.245.78.252

Live Queries

Slow Queries

Performance Advisor

Slow Queries

スロークエリの表示

Column Display

- ☒ Query
- ☒ Database
- ☒ User
- ☒ Count
- ☐ Total Time
- ☒ Rows
- ☒ Avg Exec Time (ms)
- ☐ Min Exec Time (ms)
- ☐ Max Exec Time (ms)
- ☐ Std Dev Time
- ☐ Temp Tables RAM

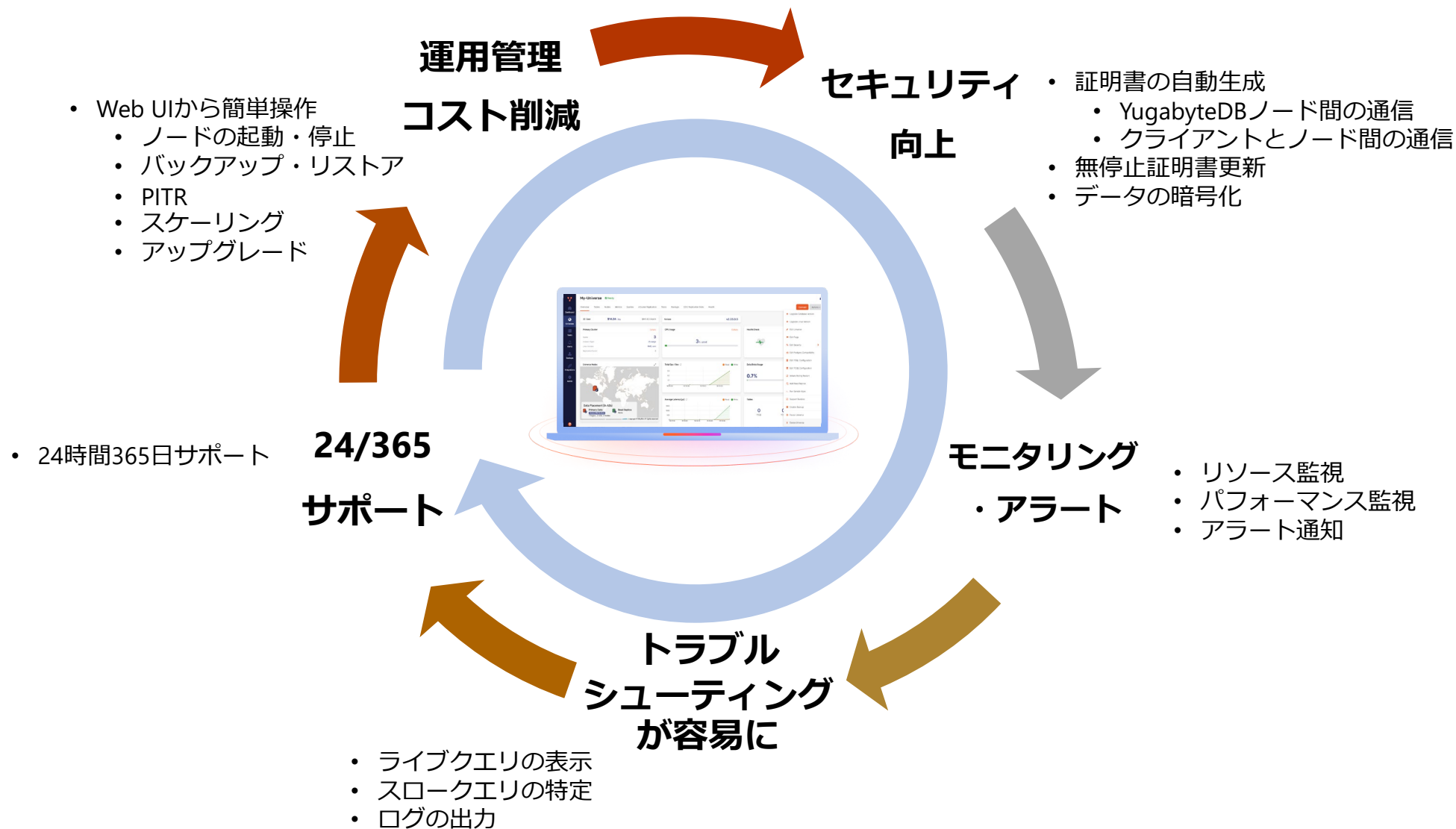
Filter by query text

QUERY ▾	DATABASE ▾	USER ▾	COUNT ▾	ROWS ▾	AVG EXEC
☐ SELECT jsonb_agg(x) FROM (SELECT pg_stat_statements_reset()) as x	postgres	yugabyte	3	3	0.847
☐ select from write_read_test where id = \$1	system_platform	yugabyte	6	6	0.287
☐ UPDATE ysql_bench_branches SET bbalance = bbalance + \$1 WHERE bid = ...	yugabyte	yugabyte	2275	2275	2.177

Query Monitoring

Reset Stats





詳細な手順や設定につきましては、弊社Techブログで公開しております。

<https://www.sraoss.co.jp/tech-blog/category/yugabytedb/>



YugabyteDB OSS版の使い方

👍 いいね! 0

分散データベース YugabyteDB は、機能やエンタープライズ向けのサポートがない代わりに、無償で利用でき

以前の文章では、[YugabyteDB の概](#)した。

今回は、実際に YugabyteDB を使

- デプロイ
- 接続方法
- 設定変更

YugabyteDB Aeon Self-Managed (旧 YugabyteDB Anywhere) の使い方

👍 いいね! 0

分散データベース YugabyteDB は、運用・管理の自動化機能やエンタープライズ向けのサポートが追加された有償版「YugabyteDB Aeon」以外にも、自動化機能やサポー



前回の文章では、[YugabyteDB OSS版の使い方](#)について紹介しました。

今回は有償版 [YugabyteDB Aeon Self-Managed \(旧 YugabyteDB Anywhere\)](#) の使い方について紹介

[YugabyteDB Aeon Self-Managed](#) (以下、YugabyteDB Anywhere) のデプロイおよび管理は、Web UI から行うか、以下の自動化ツールを利用して行います。本記事では、Web UIからの操作方法について紹介

ご清聴ありがとうございました。



製品・サービスに関するお問い合わせ:



sales@sraoss.co.jp



03-5979-2701