

Zabbix Conference Japan 2024

Zabbixのデータベース監視 トラブル検知から予防へアップグレード！ ～PostgreSQL編～

2024年11月22日

日本電気株式会社 湯村昇平

本日本話すること

- ◆ 本日は、Zabbixのデータベース監視について、
 - **Zabbix + PostgreSQLにフォーカスしたデータベース監視**
 - **Zabbixの標準テンプレートで「検知」できるトラブル**
 - **検知から予防にはもう一步踏み込んだカスタマイズが必要**というお話をします

※Zabbixは6.0LTS、OSはRHEL8、PostgreSQLは16でとりあげます



講師紹介

日本電気株式会社

プラットフォーム・テクノロジーサービス事業部門

湯村 昇平

- PostgreSQLサポート・関連サービス/製品担当
 - PostgreSQL 関連製品の開発から保守サポート、技術支援業務に従事
 - Zabbixと連携しDB監視を強化する製品を開発しています(DB Monitor for PostgreSQL)

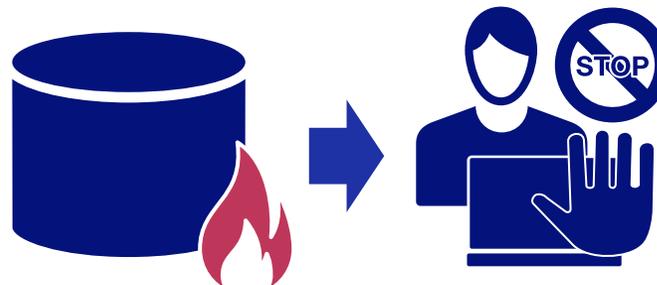


Zabbixでデータベースの監視ができます

- ◆ データベース用の監視テンプレートが各種用意されています

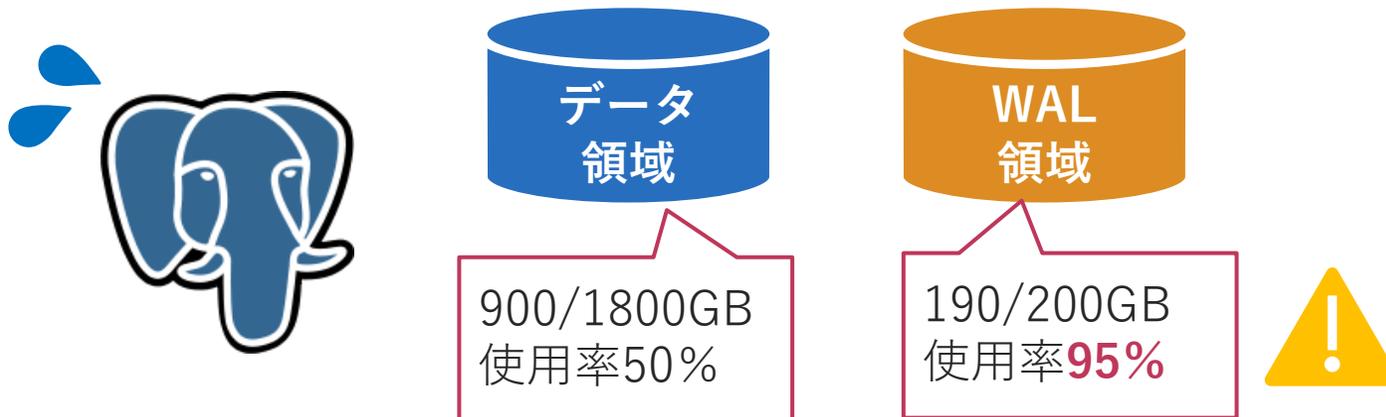


- ◆ データベースも、監視しないとある日トラブルが爆発します
 - NECのPostgreSQLサポートで対応したトラブル事例をもとに、監視のポイントを紹介します



【Zabbixですぐできる】ディスク領域を監視しよう

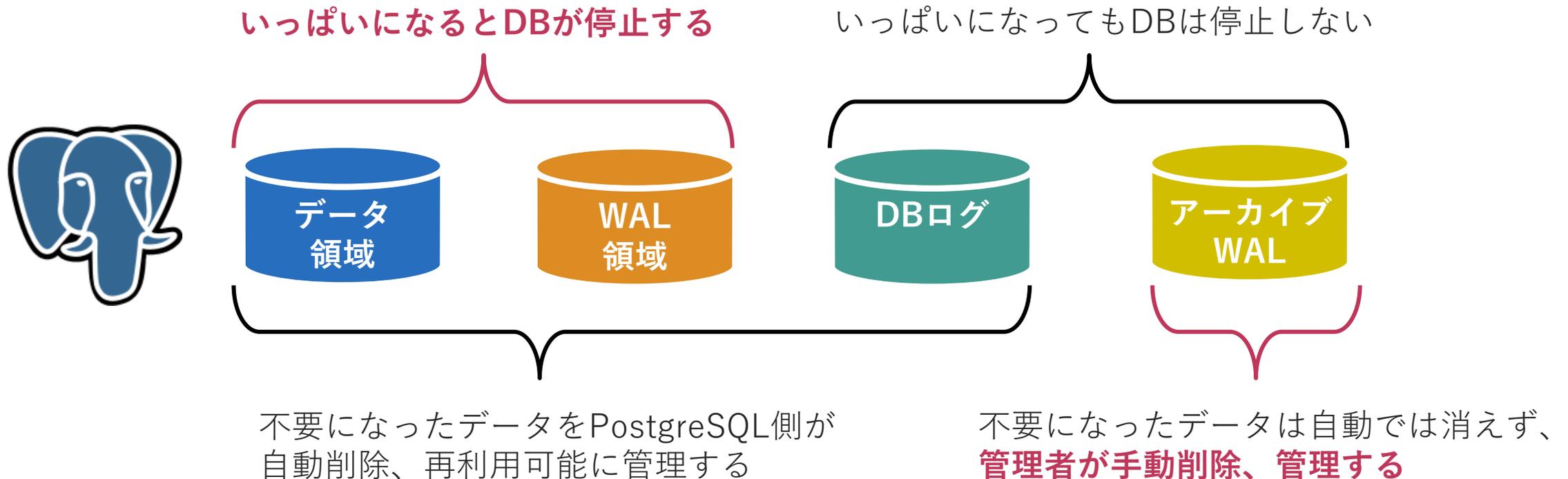
- ◆ WAL(更新ログ)の保存領域が逼迫。DBが停止する寸前！
 - PostgreSQLは保存領域がいっぱいになって新たに書き込めなくなるとPANICエラーでDBが停止してしまう
 - WAL領域は大きく割り当てることが少ないが、トラブルで急にいっぱいになるケースがある



参考知識 PostgreSQLの保存領域

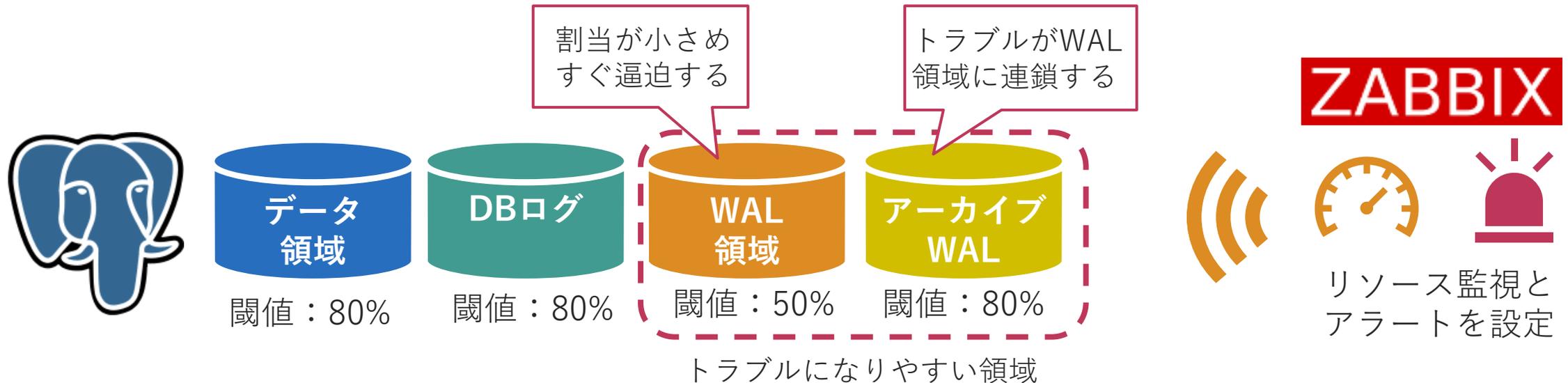
◆ PostgreSQLの代表的な保存領域は主に4つ

- 「いっぱいになるとDBが止まる/止まらない」と、「PostgreSQLが自動管理してくれる/手動で管理する」の観点でトラブル因子を整理できる



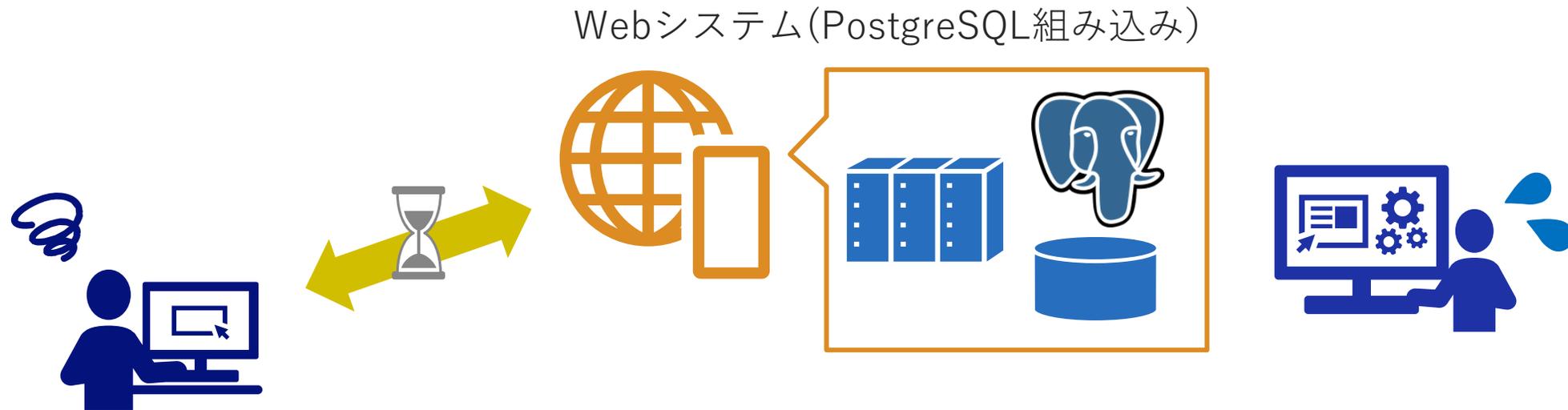
【Zabbixですぐできる】ディスク領域を監視しよう

- ◆各保存領域のディスク使用率を、Zabbixで監視！
 - ZabbixはHWリソース情報の取得と、閾値設定でアラートを挙げる事が可能
 - 単純だが効果のある監視。DB停止に至る前に問題を検知できる
検知後の対処と対策にはPostgreSQLの専門知識が必要



【Zabbixですぐできる】スロークエリを監視しよう

- ◆利用者から、普段すぐに返る処理が、10秒以上固まる！との報告が
 - NW回線の混雑？クラウドで何か起きた？DBの処理に問題が？
 - 調査の結果、DB側の処理で時々スロークエリが発生していると判明

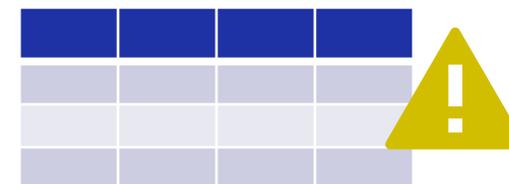
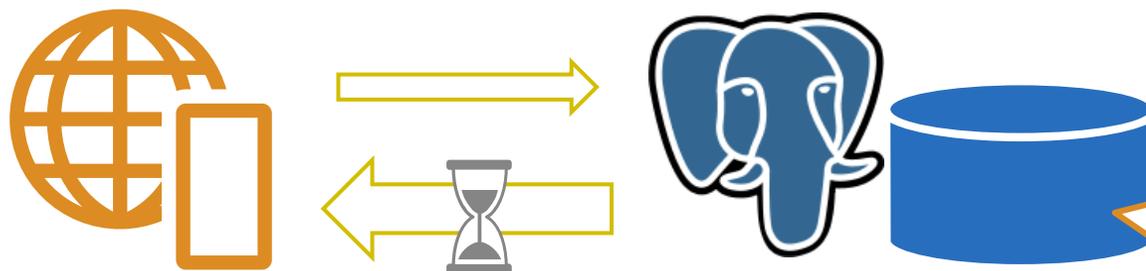


参考知識 PostgreSQLのスロークエリ

◆ 事前評価では問題がなかったシステムで、運用開始後にある日スロークエリが発生した場合、以下の原因が考えられる

■ VACUUM(自動メンテナンス)が動かずゴミが多く溜まってしまっている

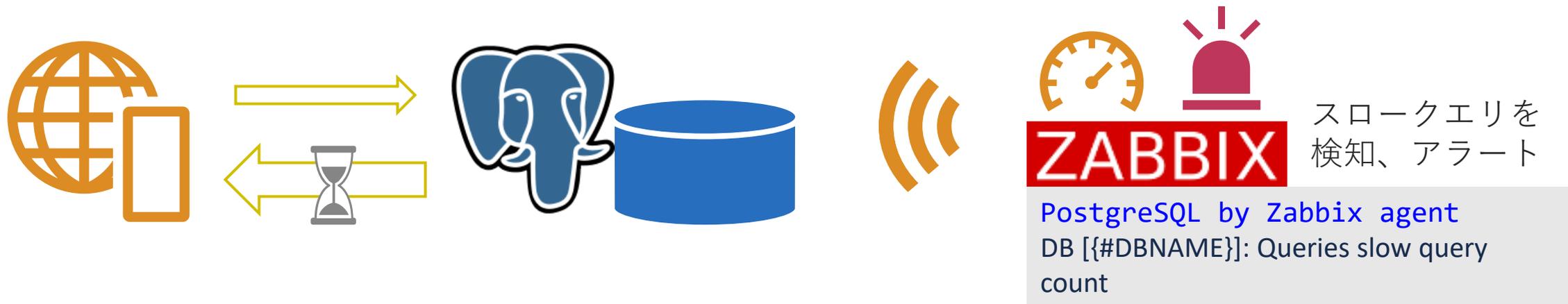
■ インデックスが肥大化し機能しなくなっている
などなど**メンテナンスの失敗や不調**に起因しがち



PostgreSQLは追記型アーキテクチャであり、UPDATEやDELETEの際に古いデータはその場で消えない。VACUUMというガベージコレクション処理が必要。これが動かないと、レコード数1000のテーブルでも実データが数百MBということがある

【Zabbixですぐできる】スロークエリを監視しよう

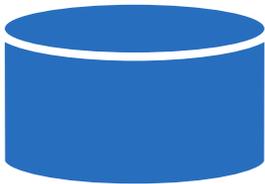
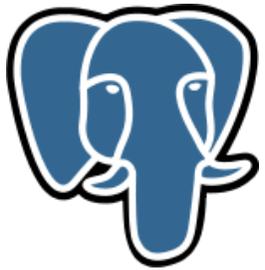
- ◆ スロークエリの発生をZabbixで監視して、DBの問題と特定する
 - DB用テンプレート「PostgreSQL by Zabbix agent」で
 - 秒以上かかっているSQLを監視可能
 - テンプレートでスロークエリが起きたことを「検知」することは可能だが、スロークエリ自体は、発生させないことが望ましい



検知から予防のために

- ◆ 予兆を捉えて発生前に手を打つならば、スロークエリの**原因となりうる要素を監視**していくことが必要
 - スロークエリの原因要素の一つ、PostgreSQLのメンテナンス状況の調子はDBが内部に持つ「統計情報」をSQLで取得し、読み解くことで判断できる
 - ZabbixのPostgreSQL監視では実際にPostgreSQLにSQLを発行しているため、テンプレートにはない独自のSQLを追加すれば予防のための監視を実現可能
※別途、PostgreSQLの専門知識も必要

[Zabbix/templates/db/postgresql/postgresql/pgsql.query.time.sql](#)



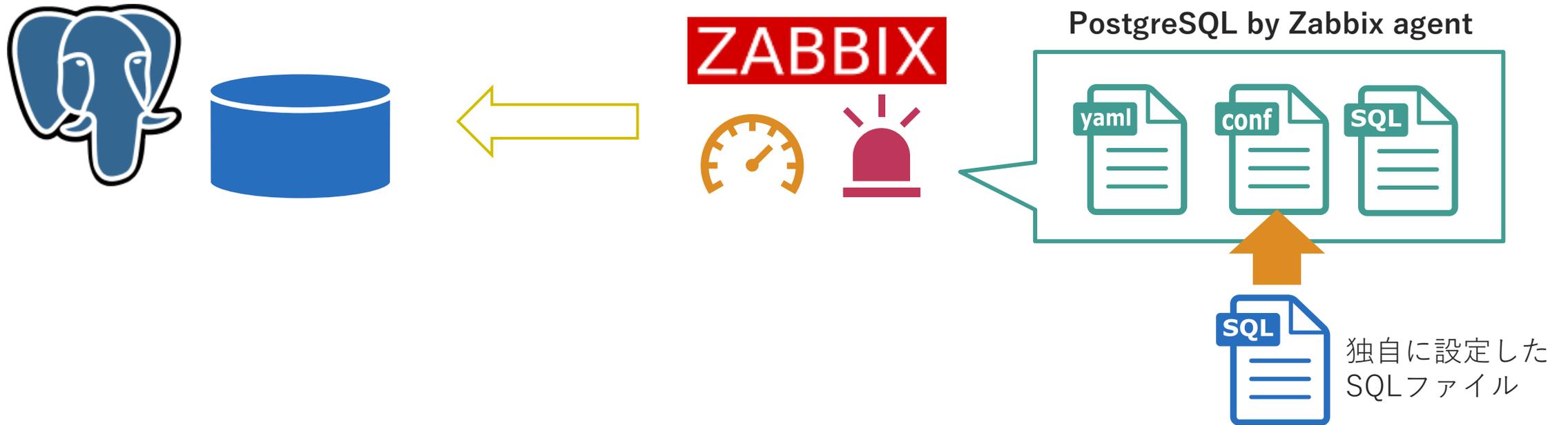
ZABBIX



```
(SELECT db.datname,  
  coalesce(T.query_time_max, 0) query_time_max,  
  coalesce(T.tx_time_max, 0) tx_time_max,  
  coalesce(T.mro_time_max, 0) mro_time_max,  
  coalesce(T.query_time_sum, 0) query_time_sum,  
  coalesce(T.tx_time_sum, 0) tx_time_sum,  
  coalesce(T.mro_time_sum, 0) mro_time_sum,  
  coalesce(T.query_slow_count, 0) query_slow_count,  
  coalesce(T.tx_slow_count, 0) tx_slow_count,  
  coalesce(T.mro_slow_count, 0) mro_slow_count  
FROM pg_database db NATURAL  
LEFT JOIN (
```

検知から予防のためにテンプレートをカスタマイズ

- ◆ 既存のテンプレート「PostgreSQL by Zabbix agent」を改修して任意のSQLを実行させる手順を解説します



【テンプレの改修】 予兆を掴むためのSQLを設定

- ◆以下のSQLで、各テーブルのゴミ(不要領域)の割合を取得可能
VACUUMがうまく動いていないテーブルを把握できる

```
SELECT
  relname,
  n_live_tup,
  n_dead_tup,
  CASE n_dead_tup
    WHEN 0 THEN 0
    ELSE round(n_dead_tup*100/(n_live_tup+n_dead_tup),2)
  END AS ratio
FROM
  pg_stat_user_tables;
```

pg_stat_user_tables

テーブル内の

- ・有効な行数
(**n_live_tup**)
- ・削除フラグはついた
が未回収な行数
(**n_dead_tup**)

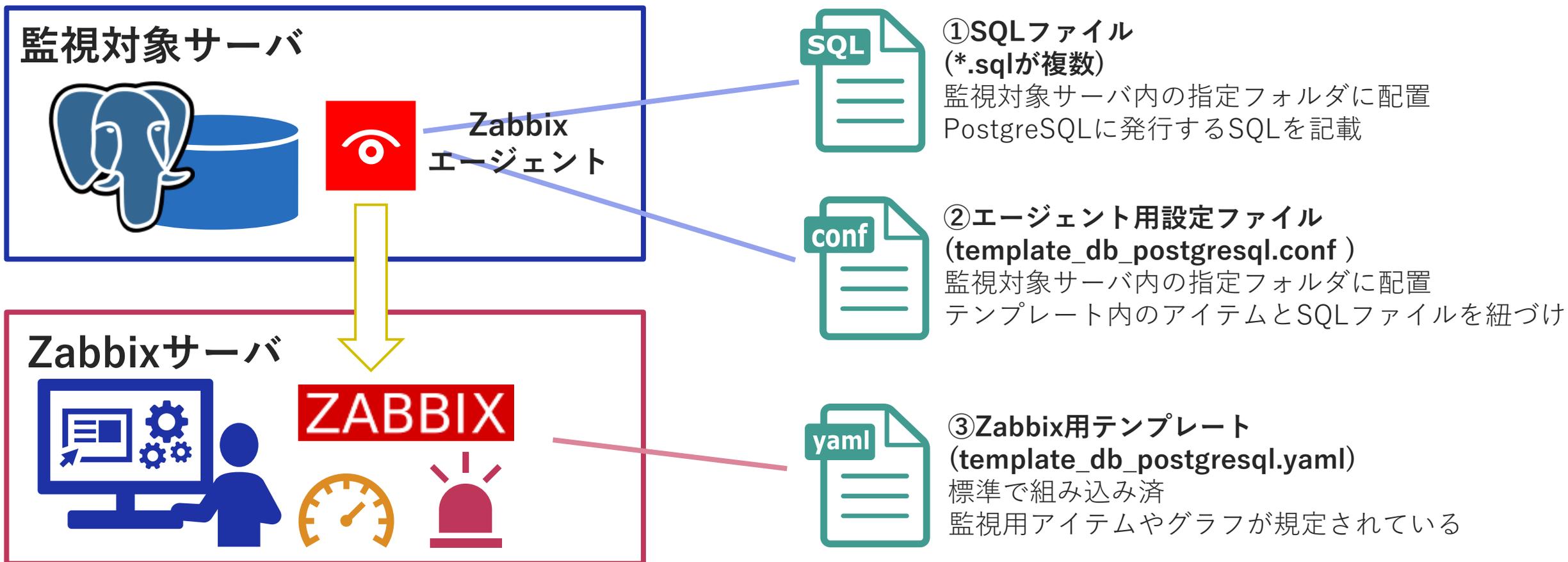
を確認出来るビュー

```
relname | n_live_tup | n_dead_tup | ratio
-----+-----+-----+-----
hogehoge |      1000 |    984767 | 99.00
```

有効行1000に対し
未回収が約100万、
不要領域99%！

【テンプレの改修】 テンプレートの構成要素

- ◆ 標準テンプレートである「 PostgreSQL by Zabbix agent 」をREADMEに沿って構築すると以下の3つの要素で構成される



【テンプレの改修】SQL実行ファイルを作成

◆以下の3つの要素を順番にカスタマイズし、反映させる

1. SQLファイルの作成

- 事前にSQLの実行をテストした上で、*.sqlファイルを作成、配置する
※例として不要領域の割合を取得するSQLを記載



pgsql.yumura.sql

```
SELECT
    CASE n_dead_tup
        WHEN 0 THEN 0
        ELSE
            round(n_dead_tup*100/(n_live_tup+n_dead_tup),2)
    END AS ratio
FROM
    pg_stat_user_tables;
```

【テンプレの改修】 エージェント側設定ファイルを改修

2. エージェント用設定ファイルの編集

- template_db_postgresql.confの末尾に、
独自に作成した*.sqlファイルのパスを他のパラメータに倣って追記する
その後エージェントの再起動で1.と2.のカスタマイズが反映される



template_db_postgresql.conf

```
UserParameter=pgsql.replication.status[*]....
```

```
...
```

追記

```
UserParameter=pgsql.yumura[*], psql -qtAX  
postgresql://" $3 ":" $4 "@ " $1 ":" $2 "/" $5 " -f  
"/var/lib/zabbix/postgresql/pgsql.yumura.sql"
```

【テンプレの改修】 Zabbix側テンプレートを改修

3. Zabbix用テンプレートファイルの編集

- template_db_postgresql.yaml内に、既にある他のアイテム記述を参考に独自に作成したSQLで値を取得するテンプレートを追記する
その後テンプレートをインポートすると上書きされ追加したものが利用可能に



template_db_postgresql.yaml

```
...
- uuid: d17c6c08314d40d1afa5b1eafb2ed93e
  name: 'Dead Ratio by Yumura'
  type: DEPENDENT
  key: 'pgsql.yumura.test["{$PG.HOST}","{$PG.PORT}","{$PG.U
SER}","{$PG.PASSWORD}","{$PG.DATABASE}"]'
  delay: '0'
  history: 7d
  value_type: FLOAT
  units: '%'
  description: '独自に追加した、不要領域の割合です'
  master_item:
    key: 'pgsql.yumura["{$PG.HOST}","{$PG.PORT}","{$PG.USER
}","{$PG.PASSWORD}","{$PG.DATABASE}"]'
...

```

追記

カスタマイズ後の実行イメージ

◆先述の1.~3.の手順実行後、アイテムとして選択可能に

The screenshot displays the Zabbix web interface for configuring a new item. The left sidebar shows the navigation menu with '設定' (Settings) expanded to 'ホスト' (Hosts). The main content area is titled 'アイテム' (Item) and shows the configuration for a parent item 'PostgreSQL by Zabbix agent'. The new item is named 'Dead Ratio by Yumura' and is of type '依存アイテム' (Dependent Item). The key is 'pgsql.yumura.test["\${PG.HOST}","\${PG.PORT}","\${PG.USER}","\${PG.PASSWORD}']', the data type is '数値 (浮動小数)' (Numeric (Floating Point)), and the unit is '%'. The master item is 'burano-yumura-rhel8: PostgreSQL: yumura'. The history and trend retention periods are set to '保存期間' (Retention Period) of 7d and 365d, respectively. The value mapping is empty, and the host inventory field is set to '-なし-' (None). The description is '独自に追加した、不要領域の割合です' (A ratio of unnecessary areas added independently).

ZABBIX << アイテム

bunora-yumura-rhel8

すべてのホスト / burano-yumura-rhel8 有効 **ZBX** アイテム 43 トリガー 11 グラフ 9 ディスカバリールール 3 Webシナリオ

アイテム タグ 1 保存前処理

親アイテム PostgreSQL by Zabbix agent

* 名前

タイプ 依存アイテム

* キー

データ型 数値 (浮動小数)

* マスターアイテム

単位 %

* ヒストリの保存期間

* トレンドの保存期間

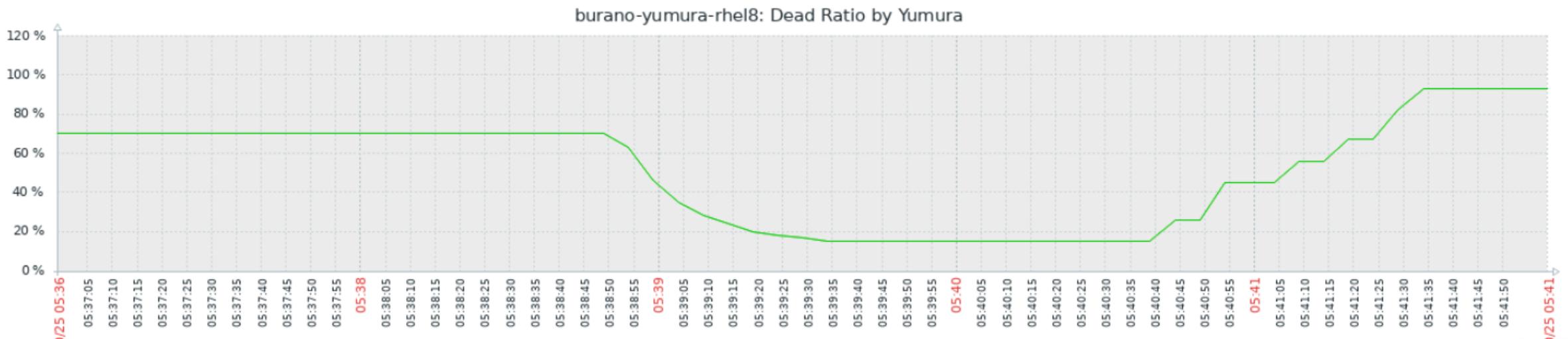
値のマッピング

ホストインベントリフィールドの自動設定

説明

カスタマイズ後の実行イメージ

- ◆ グラフでの推移の確認や、不要領域が70%以上になったテーブルはアラートを設定するなどの予防監視を実現できる
- 実際の事例では、APの不備でトランザクションの閉じ忘れがありVACUUMを阻害、不要領域が溜まっているという原因が判明したため、一定時間idleとなっているトランザクションを自動で閉じる設定を追加
スロークエリ発生を予防できた



検知から予防のための高度な監視を実現するには

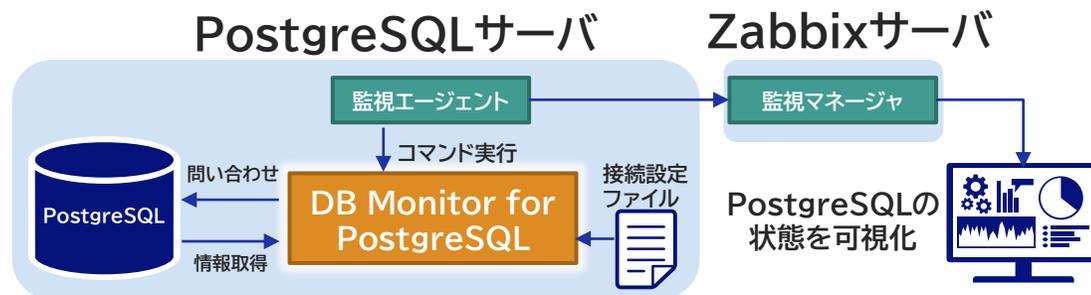
商用製品やサポートベンダに頼るのも有効です

◆ NECの商用製品 DB Monitor for PostgreSQL

- Zabbixと連携して、PostgreSQLの監視機能を強化できる製品です！
- NECのPostgreSQLサポートの強力なノウハウ(20年、500PJ超)をもとにしており、Zabbixテンプレートにはない、様々な採取用クエリ、監視項目を簡単に扱えます！

◆ NECのPostgreSQLサポート 運用サービス

- NECのPostgreSQLプロフェッショナルが、DB監視の導入を支援します！
- 監視設計、Zabbixの設定、監視レポートの見方、改善施策などなどアドバイスいたします



【製品】 DB Monitorで監視できる統計情報

スロークエリの原因となる「VACUUMの不調」を監視、スロークエリを発生前に事前察知し、対処することも可能となる

監視内容の例

- データベースの起動状態、再起動検知
- 接続成功/失敗数、同時接続数
- ロックウエイト状況、デッドロック検知
- コミット・ロールバック回数
- テーブルへのアクセス方法・回数
- ブロックアクセス回数、バッファヒット率
- クエリタイムアウト発生回数
- テーブルの更新状況
- **有効・不要タプルの件数**
- テーブルスペースごとの空き容量
- **autovacuumの処理時間・回数**
- エラーログの出力回数
- チェックポイントの発生状況
など

エラーを**確実に検出**、運用に関わる**統計情報の傾向を把握**し、サービス影響が出ないよう**システムを改善**することができます。

【製品】 DB Monitor監視項目 Zabbix比較

- ◆ PostgreSQLの安定運用に不可欠な重要項目の監視を追加・サポート
 - DBMはZabbixではサポートしない様々な項目の監視に対応しています

監視製品 監視項目	Zabbix +DB Monitor	Zabbix (5.0, 6.0)
ステータス	○	○
DDL検知	○	×
コネクション	○	○
ロック	○	○
アクセス統計	○	×
トランザク ション	○	△
I/O	○	○
スロークエリ	○	△
サイズ・容量	○	△
件数	○	○
エラー	○	×

DDLの発行（オブジェクト定義の変更）を検知します。

データベースアクセスの状況を監視します。

検知したスロークエリの詳細をログから取得、参照できます。

エラーメッセージの出力を監視します。

【製品】 DB MonitorのZabbixUI上での使用イメージ

◆ Zabbix UI上で監視項目を視覚的に監視が可能

■ DB Monitor用テンプレートを用意しておりインポートするだけで連携可能

Zabbixユーザにとっては導入、運用共に容易です



The screenshot displays the Zabbix web interface. On the left is a dark blue sidebar with the 'ZABBIX' logo and navigation menu items: '監視データ', 'ダッシュボード', '障害', 'ホスト', '最新データ', 'マップ', 'ディスカバリ', and 'サービス'. The main content area shows a table of monitoring items for host 'node1'. A red box highlights a section of the table containing PostgreSQL session-related metrics.

ホスト	名前	最新のチェック時刻	最新の値
node1	PostgreSQL(5432):Session.testdb(Database) (3アイテムのうち2から3まで表示しています)		
<input type="checkbox"/>	Established Connection	2022/07/21 11:03:34	-1
<input type="checkbox"/>	% Established Connection Rate	2022/07/21 11:03:33	-1
node1	PostgreSQL(5432):Session.postgres(User) (3アイテム)		
<input type="checkbox"/>	Unexpected Disconnection	2022/07/21 11:03:23	0
<input type="checkbox"/>	Established Connection	2022/07/21 11:03:22	1
<input type="checkbox"/>	% Established Connection Rate	2022/07/21 11:03:21	-1
node1	PostgreSQL(5432):Session.postgres(Database) (3アイテム)		
<input type="checkbox"/>	Unexpected Disconnection	2022/07/21 11:03:04	0

おわりに

- ◆ データベース(PostgreSQL)の専門知識をもとにZabbixのテンプレートをカスタマイズすれば、取得する情報を増やし、トラブル発生前の予兆を掴んで早めの対策を取ることができます
- PostgreSQLの専門知識はサポートベンダに頼って補いましょう
- 商用製品を頼って、カスタマイズの手間を省くことも選択肢です
- よいテンプレートができたなら皆さんでアップグレードしていきましょう！

本セッションへのご質問、ご感想をアンケートにて募集しております
よろしければぜひご回答ください！
オンラインの方は下記リンクをどうぞ

<https://forms.office.com/r/AqLDcTXVjJ>



\Orchestrating a brighter world

NECは、安全・安心・公平・効率という社会価値を創造し、
誰もが人間性を十分に発揮できる持続可能な社会の実現を目指します。

\Orchestrating a brighter world

NEC