KS Solutions

いざ出陣！
DB 盤石化に向けた PostgreSQL 設計 / 運用

関電システムソリューションズ株式会社

松添 隆康
今井 大嘉

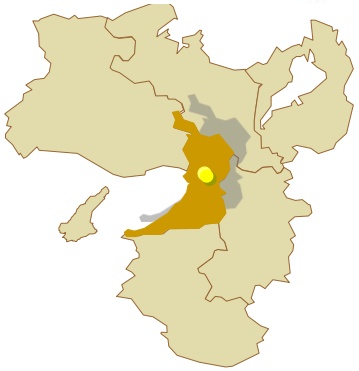
当社のご紹介

経営 ビジョン	当社は関西電力グループの総合情報サービス企業として、長年培ってきた信頼と技術力を活かしながら、自己変革によって進化し続け、ITソリューションサービスにおける、お客さま満足No.1企業を目指します。
主な 営業品目	・情報通信システムのコンサルティング、情報化戦略の立案 ・情報通信システムの計画、設計、構築、保守、運用管理 ・情報通信アプリケーションサービスの開発、提供 ・情報通信システム設備管理・運用のアウトソーシング

関電システムソリューションズ株式会社

あしたをつくる革新を、ともに。

Innovate with you



大阪第1データセンター
（大阪市北区　2001年10月～）

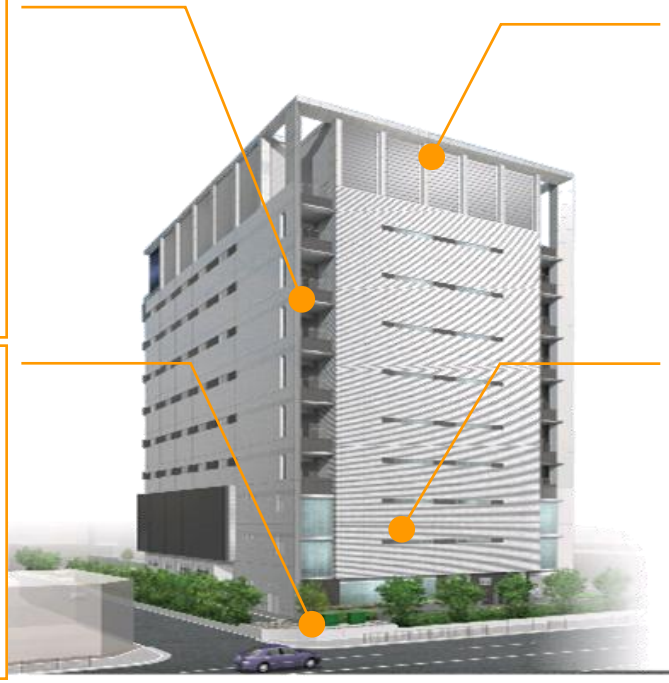
↓

大阪第2データセンター
（大阪市福島区　2002年10月～）

大阪第3データセンター
（大阪市北区　2011年12月～）

監視ルーム
24時間365日お預かりしたシステムの稼働状況を有人監視。万が一のトラブルにも技術者が即座に対応いたします。

免震システム
通常の耐震基準を超える大地震でも、基準内の揺れに抑制するビル免震システムを採用。



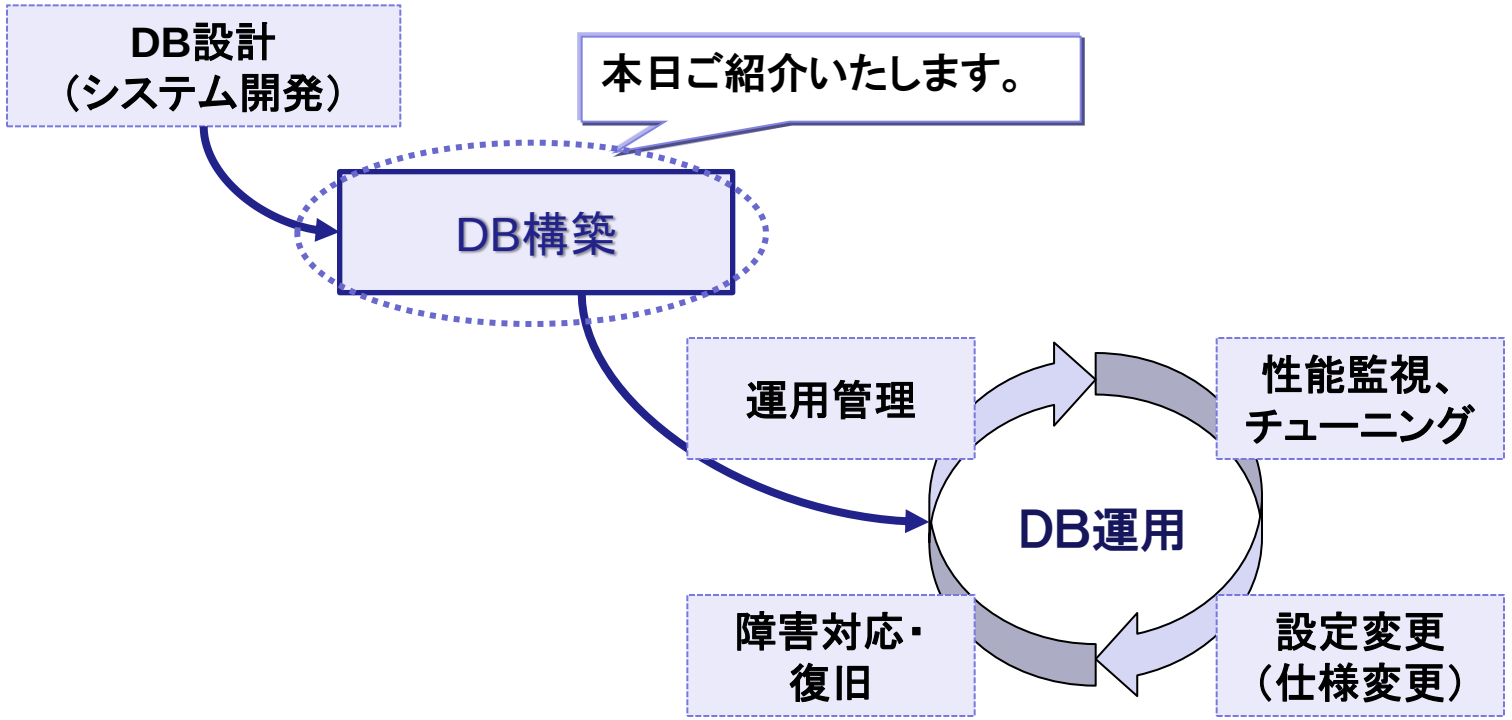
グリーンIT設備
太陽光発電や外気冷却による自然エネルギーを利用。高効率空調システムで国内最高レベルのPUEを実現。

電源設備
3スポット給電、非常用発電機(EG)、無停電電源装置(CVCF)の冗長化による止まらない受電設備。

はじめに

Oracle Database の技術サポートやガイドラインの制作に取り組んできた当社が、
昨年から、新技術への取り組みの一環として、
PostgreSQL 9.0 の設計 / 運用ガイドラインの制作に取り組んでおります。
本日は PostgreSQL 構築時に最低限抑えておきたいポイントと、
Oracle Database との違いについてご紹介いたします。

データベースのライフサイクル



アジェンダ

Ⅰ. データファイルの物理配置設計
- データベースクラスタの物理配置
- WALファイル
- アーカイブログ
- テーブルスペースの利用(補足)

Ⅱ. メモリの設計
- 共有メモリバッファ
- WALバッファ
- ワークメモリ
- メンテナンスワークメモリ
- 共有メモリについて(補足)

Ⅲ. 運用設計
- ログ
- ディスク使用量監視
- 自動VACUUMの実行監視
- 統計情報
- データベースクラスタの計画停止

Ⅰ. データファイルの物理配置設計
- データベースクラスタの物理配置
- WALファイル
- アーカイブログ
- テーブルスペースの利用(補足)

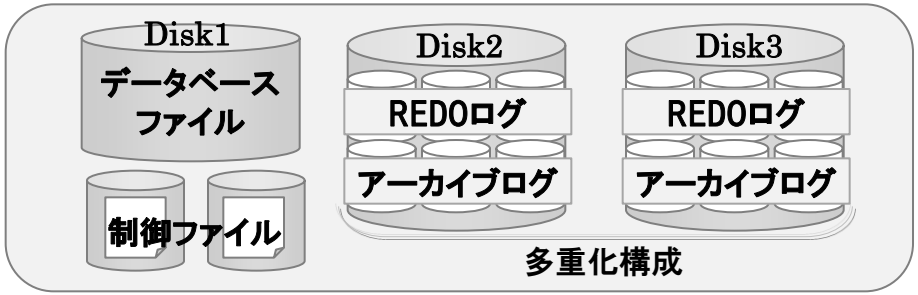
Ⅱ. メモリの設計
- 共有メモリバッファ
- WALバッファ
- ワークメモリ
- メンテナンスワークメモリ
- 共有メモリについて(補足)

Ⅲ. 運用設計
- ログ
- ディスク使用量監視
- 自動VACUUMの実行監視
- 統計情報
- データベースクラスタの計画停止

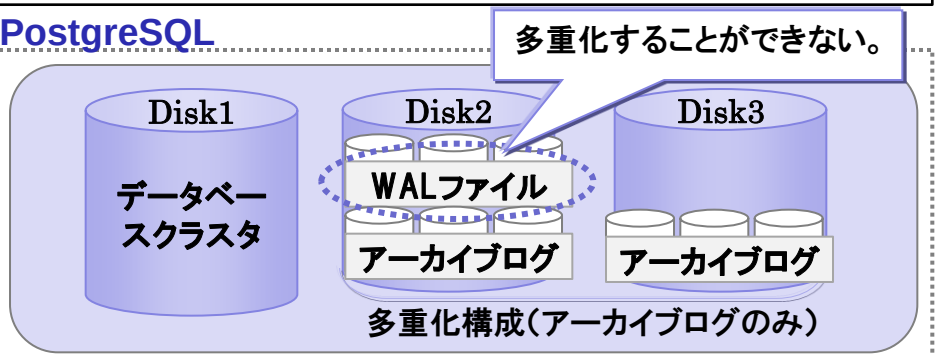
データベースクラスタの物理配置(障害対策)

概要	PostgreSQL も Oracle と同様にDB停止後、障害直前の状態まで復旧することができる。

Oracle Database



PostgreSQL



※定期的に Disk1 のバックアップ(ベースバックアップ)を取得。

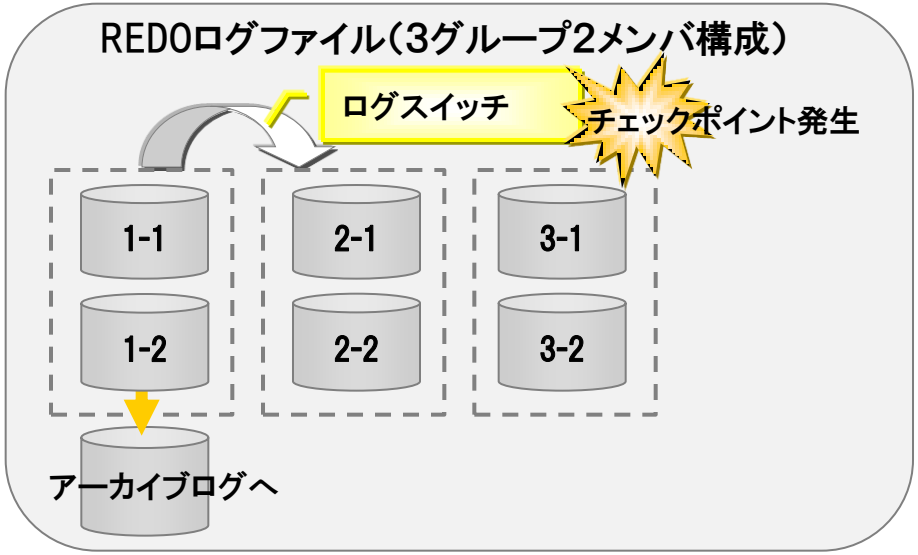
現象	DB状態(停止/起動)			DB復旧		
	相違	Oracle	PostgreSQL	相違	Oracle	PostgreSQL
Disk1 破損		停止	停止		障害直前まで復旧可能 (ﾍﾞｰｽﾊﾞｯｸｱｯﾌﾟからのﾘｽﾄｱ後、ｱｰｶｲﾌﾞﾛｸﾞ、REDOﾛｸﾞの順でﾘｶﾊﾞﾘを実施)	障害直前まで復旧可能 (ﾍﾞｰｽﾊﾞｯｸｱｯﾌﾟからのﾘｽﾄｱ後、ｱｰｶｲﾌﾞﾛｸﾞ、WALﾌｧｲﾙの順でﾘｶﾊﾞﾘを実施)
Disk2 破損	✓	起動	停止	✓	DB復旧作業不要 (DBは停止しないため)	アーカイブログ(Disk3)適用時点まで復旧可能 (ﾍﾞｰｽﾊﾞｯｸｱｯﾌﾟからのﾘｽﾄｱ後、ｱｰｶｲﾌﾞﾛｸﾞを使用しリカバリを実施)
Disk3 破損		起動	起動		DB復旧作業不要 (DBは停止しないため)	左記同様

PostgreSQLには、WALファイルを多重化する機能が搭載されていないため、RAID等の信頼性の高いディスク装置に格納することを推奨する。

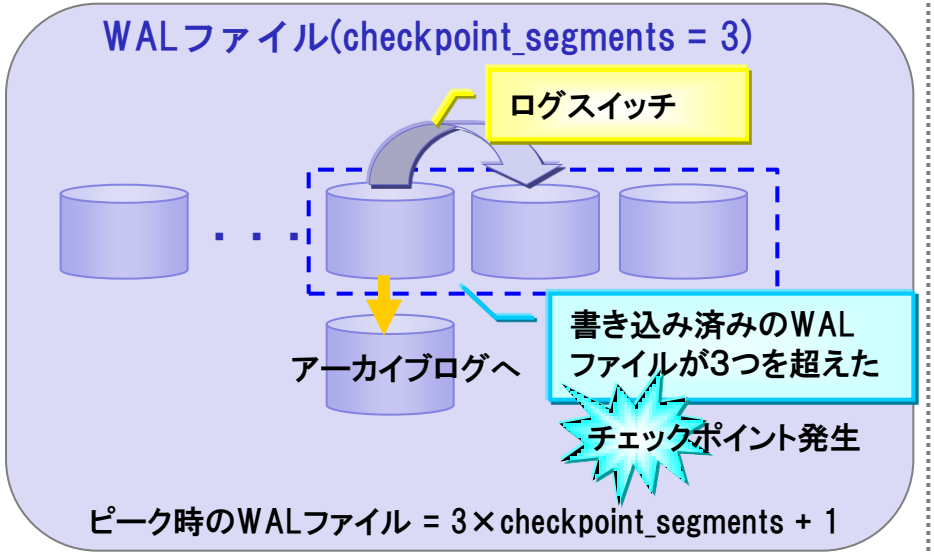
WALファイル

概要	WALファイルはインスタンス障害直前の状態まで　復旧するためには不可欠なファイルであるため、基本的な仕様をきちんと理解し、構築・運用を行うことが重要である。

Oracle Database



PostgreSQL



相違	項目	Oracle Database	PostgreSQL
✓	ログスイッチ時の動作	以下の処理が行われる。 ・チェックポイント （キャッシュとデータファイル間の同期処理） ・アーカイブログファイル出力	以下の処理が行われる。 ・アーカイブログファイル出力 ※チェックポイントは、書き込みWALファイルが3つ(デフォルト)を超えた時点で発生する。

チェックポイントはその他にも一定の時間が経過したとき(checkpoint_timeout：デフォルト5min(5分))や SQL文で CHECKPOINT を発行したとき、および、DBクラスタを停止したときに実行される。

WALファイルー 機能比較ー

概要	WALファイルはインスタンス障害直前の状態まで　復旧するためには不可欠なファイルであるため、基本仕様をきちんと理解し、DB構築を行うことが重要である。

相違	項目	Oracle Database	PostgreSQL
✓	多重化	REDOログファイルを複数メンバーで多重化することができる。	WALファイルを多重化する機能は搭載されていないため、耐障害性を高めるためには、RAID等の信頼性の高いディスク装置に格納する。
	ログスイッチ	REDOログファイルがサイズー杯まで書き込まれるとグループ単位でログスイッチが発生する。アーカイブログ運用をしている場合は、ログスイッチのタイミングでREDOログファイルがアーカイブされる。	WALファイルがサイズー杯まで書き込まれるとログスイッチが発生する。アーカイブログ運用をしている場合は、ログスイッチのタイミングでWALファイルがアーカイブされる。
✓	チェックポイント	ログスイッチが発生したタイミングで発生する(他にも発生タイミングあり)。	書き込み済みのWALファイルが3つ(checkpoint_segments = 3)を超えた時点で発生する(他にも発生タイミングあり)。
✓	ファイル名	定義した REDOログファイル名 は運用中に変わらない。	WALファイル名はログスイッチの都度、変わっていく (24桁の16進数で出力:タイムラインID(8桁)+WALログID(16桁))
	ファイルの追加	DB起動状態でREDOログファイルを追加することができる(SQLコマンドで追加)。	DBクラスタ起動状態でWALファイルを追加することができる(checkpoint_segmentsの値の変更後、pg_ctl reload コマンドで反映)。

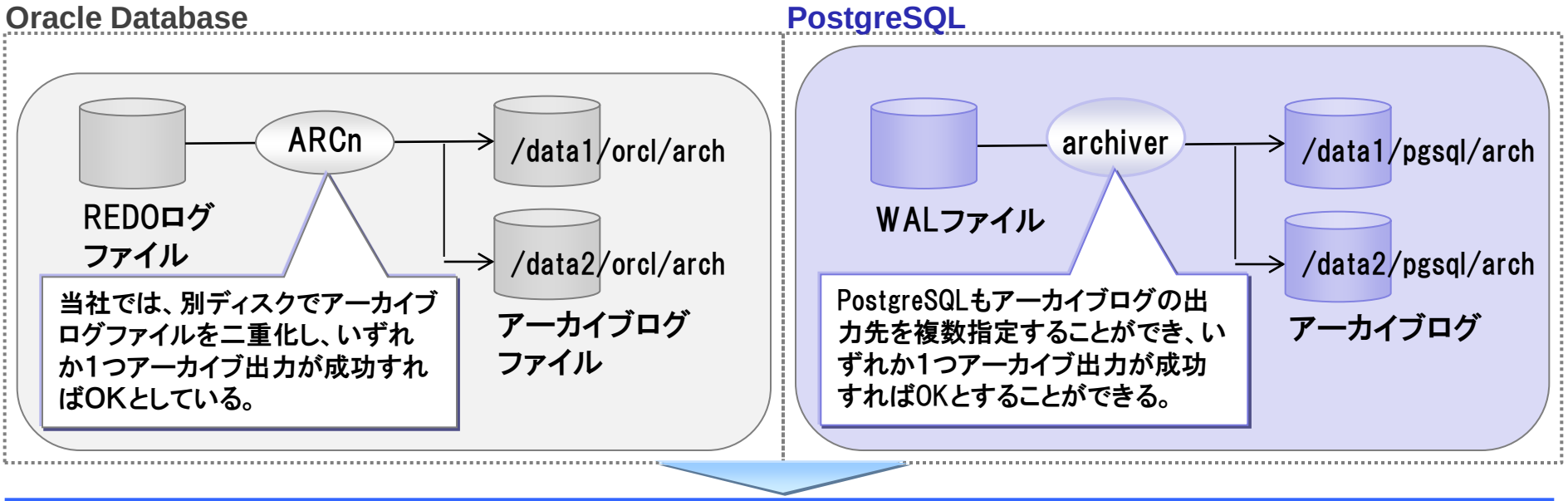
WALファイルー 機能比較ー ＜続き＞

相違	項目	Oracle Database	PostgreSQL
✓	ファイルの物理配置変更	DB起動状態でREDOログファイルの物理配置を変更することができる(SQLコマンドで変更)。	DBクラスタを停止すればWALファイルの物理配置を変更することができる(シンボリックリンクを使用して、WALファイルの物理配置を変更)。
✓	ファイルのサイズ変更	DB起動状態でREDOログファイルのサイズを変更することができる(SQLコマンドで変更)。	既にDBクラスタを作成している場合は、再インストールが必要となる(configureスクリプト実行時、--with-wal-segsize オプションを指定)。
	その他	アーカイブログ運用時、アーカイブログファイルは増え続けるため、定期的に削除する必要がある。	左記同様。

下記3つの課題に関しては、DB構築の「設計フェーズ」で対応することを推奨する。
1.WALファイルを多重化する機能を備えていない。
　→WALファイルをRAID等の信頼性の高いディスク装置に配置することを検討する。
2.DBクラスタ構築後、WALファイルの物理配置を変更する場合は、DBクラスタの停止が生じる。
　→WALファイルの物理配置について検討する。
3.DBクラスタ構築後、WALファイルのサイズを変更する場合は、再インストール作業が生じる。
　→WALファイルのサイズについて検討する。

アーカイブログ

概要	PostgreSQL も Oracle と同様なアーカイブログ出力機能を搭載している。



アーカイブログの出力制御については、OSの機能と組み合わせることで、柔軟な設定が可能となる。

（例）

1.両方のディレクトリへのアーカイブ出力が成功した場合に OK とする設定

```
archive_command = 'cp %p /data1/pgsql/arch/%f && cp %p /data2/pgsql/arch/%f '
```

2.いずれかのディレクトリへのアーカイブ出力が成功した場合に OK とする設定

```
archive_command = '$PGDATA/script/arc.sh %p %f '
```

（$PGDATA/script/arc.sh の具体的なシェル内容については割愛させて頂きます）

アーカイブログ－ 機能比較－

概要	アーカイブログもWALファイル同様、障害復旧するためには不可欠なファイルであるため、基本仕様をきちんと理解する必要がある。

相違	項目	Oracle Database	PostgreSQL
	多重化	アーカイブログファイルの出力先を複数指定することができる。	左記同様(archive_command で設定)。
	アーカイブ成功数の設定	複数の出力先のうち、いくつ成功したらOK とみなすかを設定することができる。	左記同様(archive_commandで設定) ※いずれかのディレクトリへのアーカイブが成功した場合にアーカイブ成功とみなす場合は、シェル作成後、archive_command を設定する。
✓	ファイル名	アーカイブログファイル名は柔軟に設定することができる。 (参考) log_archive_format 設定書式 %s: ログ順序番号 %S: 0を埋め込んだログ順序番号 %t: スレッド番号 %T: 0を埋め込んだスレッド番号 %a: アクティブ化ID %d: データベースID %r: リセットログID	アーカイブログ名を柔軟に設定することができない(WALファイル名と同様の名前で生成される)。 (出力例) $ ls /data1/pgsql/arch 000000010000000000000001 000000010000000000000002 000000010000000000000003 ※24桁の16進数で出力 タイムラインID(8桁)+WALログID(16桁)

アーカイブログの複数の出力先のうち、全て成功しなくても OK とみなす場合は、いずれか1つアーカイブ出力が成功したときに リターンコード　0(ゼロ) を返すシェルスクリプトを作成する必要がある。

補足）テーブルスペースの利用

概要	テーブルスペースを利用することで、データベースクラスタ($PGDATA)以外の場所にオブジェクトファイル(テーブル、インデックス)を配置することができる。

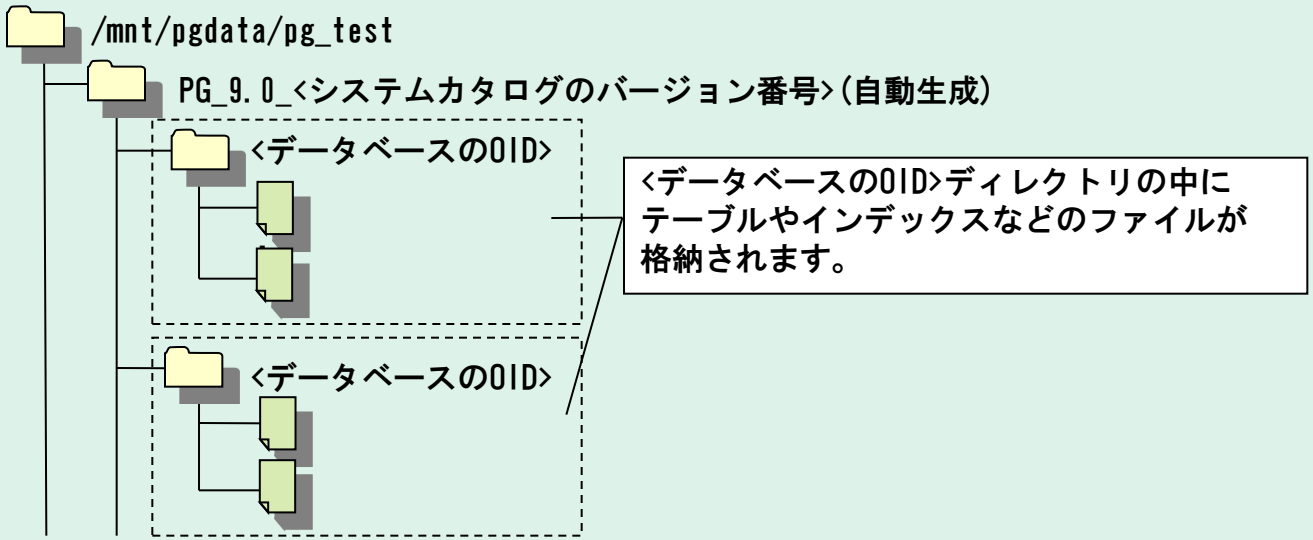
相違	項目	Oracle Database	PostgreSQL
✓	テーブルスペース(表領域)	表領域の実態は物理ファイル(データファイル)である。	テーブルスペースの実態はディレクトリである。

テーブルスペースの構成

（SQL例）

```
--テーブルスペースを作成
=# CREATE TABLESPACE pg_test OWNER testuser LOCATION '/mnt/pgdata/pg_test';
```

（構成）



※ $PGDATA/pg_tblspc ディレクトリ配下にテーブルスペースへのシンボリックリンク情報が格納されます。

Ⅰ. データファイルの物理配置設計
- データベースクラスタの物理配置
- WALファイル
- アーカイブログ
- テーブルスペースの利用(補足)

Ⅱ. メモリの設計
- **共有メモリバッファ**
- **WALバッファ**
- **ワークメモリ**
- **メンテナンスワークメモリ**
- **共有メモリについて(補足)**

Ⅲ. 運用設計
- ログ
- ディスク使用量監視
- 自動VACUUMの実行監視
- 統計情報
- データベースクラスタの計画停止

共有メモリバッファ

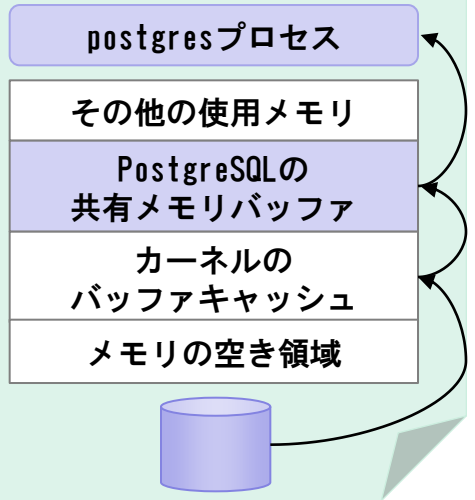
概要	本領域は、テーブルやインデックスのデータを格納するデータキャッシュとして利用される。 データを共有メモリバッファ上にキャッシュすることにより、物理読み込み回数が減り、 性能向上を図ることができる。

Oracle Database	PostgreSQL
db_cache_size(db_nk_cache_size)	shared_buffers

Oracleとの違い と 共有メモリバッファの設定

(Oracleとの違い)
Oracleでは db_nk_cache_size で複数の非標準ブロックをキャッシュする
領域を設定できますが、PostgreSQLはインストール時に設定したブロックサイズを
持つブロックのみキャッシュします。

(共有メモリバッファの設定)
一般的に物理メモリが１ＧＢ以上を持つシステムであれば、共有メモリバッファの
最初の見積もりサイズは物理メモリの１／４くらいを割り当てます。
基本的には shared_buffers を大きな値に設定するほど読み取り性能を
向上させることができますが、PostgreSQLはオペレーティングシステムの
キャッシュにも依存しているため、当社では目安として、
物理メモリ量の ２５%～４０% 程度を割り当てることを推奨しています。



共有メモリバッファのサイズの開始値は 25%(1／4)とし、DB構築後の性能テストで、
vmstat(*1) や pg_stat_database (*2)、pg_statio_user_tables(*3)を監視し、必要に応じて、
物理メモリの40%程度までサイズを大きくすることを推奨する。

*1 スワップの利用有無を確認
*2 DB単位でキャッシュヒット率を確認(計算式: キャッシュヒット率 = blks_hit / (blks_hit + blks_read))
*3 テーブル単位でキャッシュヒット率を確認(計算式: キャッシュヒット率 = heap_blks_hit / (heap_blks_hit + heap_blks_read))

WALバッファ

概要	本領域は、データ変更内容(トランザクションデータ)を格納するキャッシュとして使用される。 WALバッファ上のトランザクションデータはコミット時にWALファイルに書き出される。

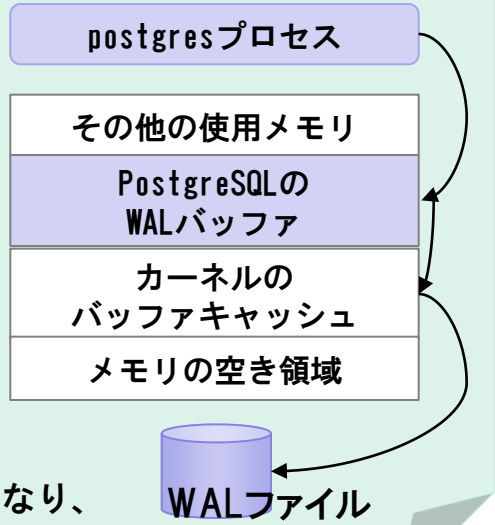
Oracle Database	PostgreSQL
log_buffer	wal_buffers

WALバッファの設定

一般的に最初の見積もりサイズは １ＭＢ 程度割り当てます。

(参考)
PostgreSQL Wiki　<抜粋>
<http://wiki.postgresql.org/wiki/Tuning_Your_PostgreSQL_Server/ja>

ベンチマークでは通常、多少大規模なシステムでは1MBまで増やすだけで
十分であると提案します。またWALセグメント全体（16MB、ここでは有用な上限）
を割り当てる最近のサーバでメモリを提供することが合理的です。



当社では目安として、１ＭＢ ～ １６ＭＢ 程度を割り当てることを推奨しています。
(PostgreSQL9.1から、wal_buffers はデフォルトで shared_buffers の容量の1/32になり、
上限は16MB（shared_buffers が 512MB の時に達する値）になりました)

WALバッファのサイズの開始値は 1MB としているが、物理メモリ搭載量が少ないマシンでなければ、
WALセグメント(WALファイル)のサイズ(デフォルト16MB)に合わせることを推奨する。

ワークメモリ

概要	本領域は、SQL実行時のソートやハッシュ操作などのために postgresプロセス毎に確保される。ワークメモリのサイズを増やすことにより、メモリ内においてソートやハッシュ操作などを効率的に処理できるようになるため、SQLの性能を向上させることができる。

Oracle Database	PostgreSQL
pga_aggregate_target ※総PGAメモリー・サイズの上限値(目標値) ワークメモリ容量（*_area_size）自動調整機能（自動PGA管理機能）で使用される。	work_mem ※postgresプロセス毎に確保される。

ワークメモリの設定

PostgreSQLは Oracleの自動PGA管理機能 のようにワークメモリを自動調整する
機能は備えていないため、PostgreSQLで使用されるワークメモリの総量を
下記の計算式で見積もる必要があります。

ワークメモリ使用総量 = work_mem（デフォルト1MB） × max_connections（最大接続数）

(参考)
プランナでは実行計画の作成時にワークメモリの大きさも考慮するため、ワークメモリが
少ない場合にはハッシュ結合やハッシュ集約のような高速な計画タイプが選択されないこともあります。

postgresql.conf の work_mem に大きなサイズを設定すると、メモリが不足してしまう可能性がある。
そのため、バッチ処理などで大量のワークメモリを必要する場合は、個別に SET を使用し、
セッションやトランザクション毎に work_mem を一時的に増やすことを推奨する。

メンテナンスワークメモリ

概要	本領域は、VACUUM や CREATE INDEX などのメンテナンス作業時に postgresプロセス毎に確保される。メンテナンスワークメモリのサイズを増やすことにより、メンテナンス処理を効率的に行うことができるようになるため、性能を向上を狙うことができる。

Oracle Database	PostgreSQL
該当するメモリ領域がない。	maintenance_work_mem ※postgresプロセス毎に確保される。

メンテナンスワークメモリの設定

通常、メンテナンスを並行して大量に実行することはないため、
ワークメモリと比べて大きなサイズを設定しても問題ありません。
しかし、自動VACUUMの autovacuum worker プロセス（デフォルトで最大３プロセス起動）も
メンテナンスワークメモリを使用するため、留意する必要があります。
PostgreSQLで使用されるメンテナンスワークメモリの総量は下記の計算式で
見積もることができます。

メンテナンスワークメモリ使用総量 =
maintenance_work_mem(デフォルト16MB)×(autovacuum_max_workers ＋ メンテナンス作業による接続セッション)

maintenance_work_mem を大きくすると、自動VACUUMの複数のプロセス(デフォルト3)が
メンテナンスワークメモリ領域を確保するため、値を大きくしすぎないように注意する必要がある。
大量データ処理を伴うメンテナンスコマンド(VACUUM FULL、CREATE INDEX など)を実行する
場合は、個別に SET を使用し、 maintenance_work_mem を一時的に増やすことを推奨する。

補足）共有メモリについて

概要	PostgreSQL には Oracle のように 実行計画を複数セッションにて共有できないため、 それは制約条件のため留意する。

調査結果

PostgreSQLでは、Oracle のように解析済みSQL（実行計画）を共有する領域（共有プール）が
搭載されていないため、各接続セッションで実行計画を共有することはできません。

そのため、新規接続セッションで、同一のSQLを発行する場合は、毎回SQLの解析処理が行われます。

しかし、JDBCドライバを用いて全く同じSQLを同一接続セッションで発行する場合は、
実行計画を再利用することが可能となるため、検索条件にバインド変数を用いる対応は効果的です。
※この機能は、JDBC接続パラメータの prepareThreshold （デフォルト 5）の設定で実現可能となり、
同一接続セッションで5回（デフォルト）まで、実行計画を再利用することができます。
なお、psql からのSQLを実行する場合は、同一セッションにおいても実行計画が再利用されることは
ありません。

Ⅰ. データファイルの物理配置設計
- データベースクラスタの物理配置
- WALファイル
- アーカイブログ
- テーブルスペースの利用(補足)

Ⅱ. メモリの設計
- 共有メモリバッファ
- WALバッファ
- ワークメモリ
- メンテナンスワークメモリ
- 共有メモリについて(補足)

Ⅲ. 運用設計
- **ログ**
- **ディスク使用量監視**
- **自動VACUUMの実行監視**
- **統計情報**
- **データベースクラスタの計画停止**

ログ

概要	PostgreSQL も Oracle と同様に様々な情報をログに出力することができるため、基本機能を整理し、活用することが重要である。

相違	項目	Oracle Database	PostgreSQL
	出力場所	任意に指定した場所にログファイルを出力することができる。 (参考)初期化パラメータ ～10g background_dump_dest ～11g diagnostic_dest	左記に加え、syslog(Windowsであれば、eventlog) と csvlog(CSV形式で出力)にも出力することができる。 (設定例) llog_destination = 'stderr' logging_collector = on log_directory = 'pg_log'
✓	ファイル名	alert_<SID>.log （固定）	パラメータを用いてファイル名を柔軟に設定することができる。 スライド２０参照
✓	ログローテーション	ログローテーション機能は搭載されていない(別途、仕組みを作成する必要がある)。	パラメータを用いてログローテーションを行うことができる。 スライド２０参照
✓	接頭情報(出力内容)	年月日時分秒 がデフォルト(変更不可)で出力される。	パラメータを用いて接頭情報を柔軟に設定することができる(デフォルトは何も出力されないため、注意が必要)。 スライド２１参照

PostgreSQL は Oracle に比べ、ログ関連の設定情報を柔軟に変更することができる。
当社では、ログ出力先、ログファイル名、ログローテーション、接頭情報を設定することを推奨する。

（補足）ログの設定 ー ファイル名、ローテーション ー

概要	PostgreSQLでは、パラメータを用いてファイル名を柔軟に設定することができる。 またログローテーションも設定することができる。

弊社ガイドライン（postgresql.conf）

パラメータ	推奨値	説明
log_filename	‘postgresql-%Y%m%d.log’ 上記設定で使用される %から始まる引数は、以下の値に置換される。 %Y…年(西暦4桁) %m…月(01-12) %d…日(01-31)	年月日をファイル名に持つことにより、DB運用保守作業または障害対応の際、該当日のファイルを即座に把握することができる。また、1日分のログ内容しか格納されていないため、軽量となり、ファイルサイズが大きすぎてオープンできないという事象も防ぐことができる。
log_rotation_age	1d	1日単位にログがローテートするよう設定（夜間24時になった時点で、新規ログが作成される）。
log_truncate_on_rotation	off	log_rotation_age に設定した期間が経過すると、ログ出力先が新しいファイルに切り替わる。このとき、同名のファイルがすでに存在する場合、ファイルの内容を切り捨てず追記を行う。 log_filename の設定で（年月日単位でログ出力）、同一ファイル名のログが作成されないように設定しているが、不測の事態に備えて、本設定を off に設定することを推奨する。

※ Oracleと同様に不要ログを削除する機能は搭載されていないため、別に作成する必要がある。

（補足）ログの設定 ー接頭情報(出力内容）ー

概要	PostgreSQLでは、パラメータを用いてログに出力される接頭情報を柔軟に設定することができる。

弊社ガイドライン(postgresql.conf)

パラメータ	推奨値	説明
log_line_prefix	‘[%m][%d][%u][%e][%p]’ 上記設定で使用される %から始まる引数は、以下の値に置換される。 %m・・・タイムスタンプ(ミリ秒付き) %d・・・データベース名 %u・・・ユーザ名 %e・・・SQLSTATEエラーコード %p・・・プロセスID	ログ内容の可読性も考慮し、必要最低限の情報を出力する。 ログにSQLを出力させる設定を施した場合(例:log_statement、log_min_duration_statement)、同一プロセスで一連にて実行される処理(SQL)は別々の行にログ出力されるため、プロセスID(%p)の設定をしなければ、処理の流れを追跡することができない。 そのため、log_line_prefix パラメータに %p を含めることで、処理実行におけるトレーサビリティを担保することができる。

他にも当社では、チェックポイントが適切な間隔で発生していることを確認するため、
log_checkpoints = on にし、チェックポイント発生状況をログに記録することを推奨しています。

ディスク使用量監視

概要	PostgreSQLもOracle と同様に様々な視点でディスク容量を監視するための機能が用意されている。

相違	項目	Oracle Database	PostgreSQL
	ディスク使用量監視	データファイル単位に監視可能。	データベース単位、テーブルスペース単位、テーブルと付随するインデックスの合計サイズの単位、 テーブル単位/インデックス単位にて監視可能。

データベースオブジェクト容量関数の使用

大 → 容量 → 小

```
=> SELECT pg_size_pretty(pg_database_size( 'データベース名' ));
 pg_size_pretty
----------------
 966 MB

=> SELECT pg_size_pretty(pg_tablespace_size('テーブルスペース名'));
 pg_size_pretty
----------------
 945 MB

=> SELECT pg_size_pretty(pg_total_relation_size('テーブル名'));
 pg_size_pretty
----------------
 272 MB

=> SELECT pg_size_pretty(pg_relation_size( 'テーブル名/インデックス名'));
 pg_size_pretty
----------------
 96 MB
```

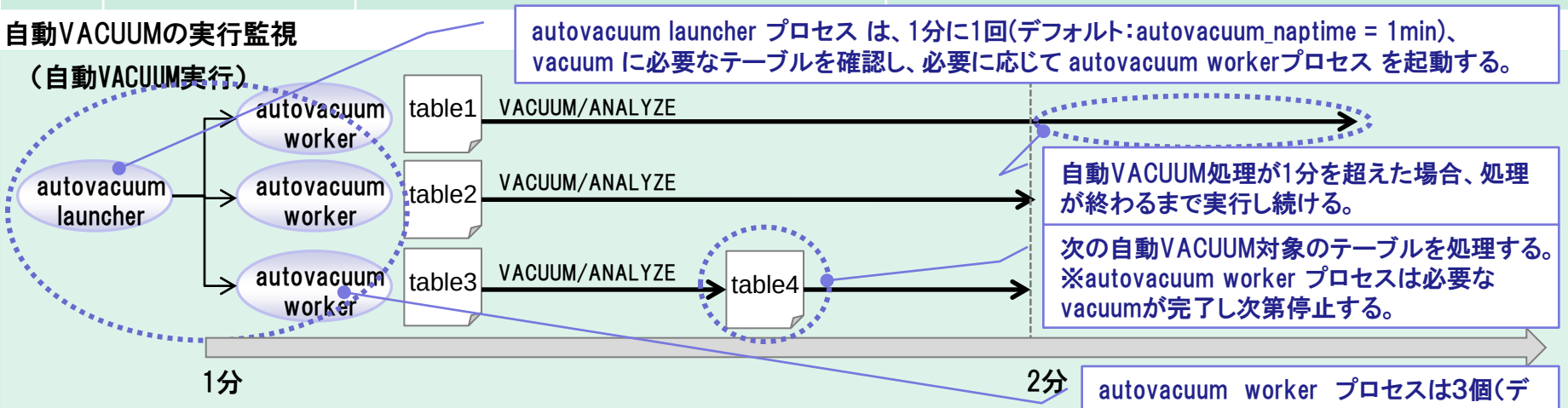
大きな視点から小さな視点まで、ディスク使用量を監視することができるため、効果的に運用管理を行うことが可能となる。

自動VACUUMの実行監視

概要	PostgreSQLでは、追記型アーキテクチャが採用されており、データの変更処理を繰り返すとファイルサイズの肥大化に繋がる可能性があるため、自動VACUUMを有効(デフォルト)にし、VACUUM/ANALYZE を定期的に実行させる必要がある。

相違	項目	Oracle Database	PostgreSQL
✔	VACUUM	該当する機能がない(Oracleは追記型アーキテクチャではないため不要)。	自動VACUUM で対応可能。 (他にも、VACUUM FULL、CLUSTER がある)

自動VACUUMの実行監視



自動VACUUMで VACUUM/ANALYZE が実行されたかどうかは以下のSQL文にて確認します。

```
=# select relname, last_autovacuum, last_autoanalyze from pg_stat_all_tables;
         relname          |        last_autovacuum        |       last_autoanalyze
--------------------------+-------------------------------+-------------------------------
 customer                 |                               | 2011-11-10 14:42:08.349025+09
 sales                    | 2011-11-10 14:42:08.349025+09 | 2011-11-10 14:42:08.349025+09
```

自動VACUUMを有効(デフォルト)にすることで、ファイルの肥大化とパフォーマンス低下およびトランザクションID周回問題を解決することができる。

統計情報

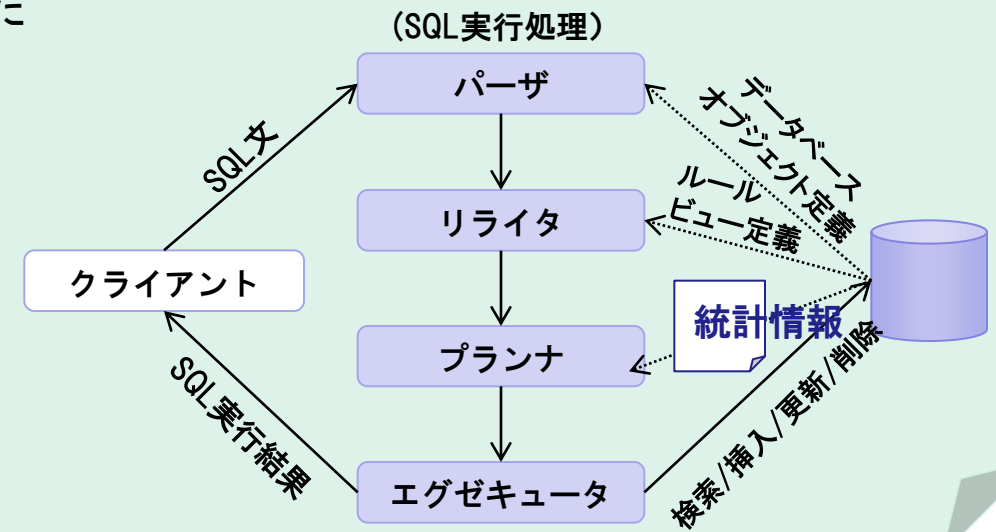
概要	PostgreSQLには、統計情報を収集する機能として 自動VACUUM と ANALYZEコマンドが搭載されている。

相違	項目	Oracle Database	PostgreSQL
	統計情報の収集	DBMS_STATS パッケージで収集することができる。	自動VACUUM、ANALYZE で収集することができる。

PostgreSQL 統計情報の利用

SQL実行処理にて、プランナがコスト計算を行う際に統計情報が参照されます。

統計情報は SQL実行処理のパフォーマンスに大きく影響するため、
当社では大量データ投入後やテーブルおよびインデックスの再構築後にANALYZEコマンドを実行することを推奨しています。



PostgreSQLには1分間隔(デフォルト)で自動VACUUMが実行されていますが、大量データ更新後はANALYZEコマンドで統計情報を収集することを推奨する。

データベースクラスタの計画停止

概要	PostgreSQL も Oracle と同様にいくつかの停止方法(停止モード)が用意されている。

停止モード

相違	Oracle Database	PostgreSQL	説明
	NORMAL	smart	接続中のクライアントが切断するのを待ってから停止する(デフォルト)。
✓	TRANSACTIONAL	ー	アクティブ・トランザクション実行完了後、接続中のクライアントを強制的に切断して停止する。
	IMMEDIATE	fast	接続中のクライアントを強制的に切断して停止する。実行中のSQLはキャンセルされる。
	ABORT	immediate	クリーンアップ処理を行わずにサーバープロセスを停止する。再起動時にはインスタンス・リカバリ(復旧処理)が行われる。

DBサーバのメンテナンス作業時など、直ちに PostgreSQL を停止したい場合は、
停止モードを「fast」に指定することを推奨する。
また、停止モード「immediate」は、Oracle 停止モード「ABORT」に該当するため注意する。

(コマンド例)
$ pg_ctl stop –m fast

ご清聴、ありがとうございました。