PGECons
PostgreSQL Enterprise Consortium

PostgreSQL エンタープライズ・コンソーシアム 技術部会 WG#2

ＤＢ選定基準編

要件から採択するPostgreSQL

製作者
担当企業名
日本電子計算株式会社
株式会社富士通ソーシアルサイエンスラボラトリ
株式会社HTKエンジニアリング
NEC ソリューションイノベータ株式会社

改訂履歴

版	改訂日	変更内容
1.0	2016/4/22	新規作成
1.1	2016/5/26	表 11 の誤字修正

はじめに

■本資料の概要と目的

本資料は、任意の企業情報システムにおける一般的な DBMS に求められる要件を列挙し、PostgreSQL ではどのようにその要件に応えられる機能を有しているかを考察した文書です。

■資料内の記述について

本資料で記載されている要件および PostgreSQL の機能については、すべてを網羅しているわけではありません。
そのために読者の方々のプロジェクト要件と合致しない場合もあるかと思いますが、少しでもお役に立てれば幸いです。

目次

1. 技術要件（一般的な機能）

利用者がデータベースシステムに期待する一般的な技術要件には次のようなものがあります。

1.1. アーキテクチャ

PostgreSQL は DBMS として求められる一般的な機能についてはほぼ全て備えていると言えます。
DBMS に求められる一般的な機能とは例えば以下のようなものです。

・同時実行の制御（排他制御）
・トランザクション（ACID 属性）

この点では PostgreSQL も他の商用 DBMS も特に違いはありません。
しかし、アーキテクチャについては、各 DBMS ごとに異なっており、それぞれ特色があります。
移行先に PostgreSQL を選定するにあたっては、移行元の DBMS はどのようなアーキテクチャを採用しているか、移行先である PostgreSQL の側はどうか、それぞれ認識しておくと良いでしょう。

付録「【別紙】アーキテクチャ比較表」に、PostgreSQL と代表的な商用 DBMS がどのようなアーキテクチャを採用しているかについて、主なテーマに分類してまとめていますので参考にしてください。付録資料では商用 DBMS の側は Oracle、SQL Server、DB2 の 3 つを挙げています。

本節では、以降の 1.1.1～1.1.6 にて、付録資料で採り上げているテーマの中から主なものをピックアップし、補足しました。付録資料に目を通していただく際の導入となれば幸いです。
各テーマについて PostgreSQL、商用 DBMS の順に説明しています。商用 DBMS の例は Oracle としました。

1.1.1. プロセス構成

PostgreSQL はクライアント・サーバ型の DBMS であり、マルチプロセス構成です。(図 1)

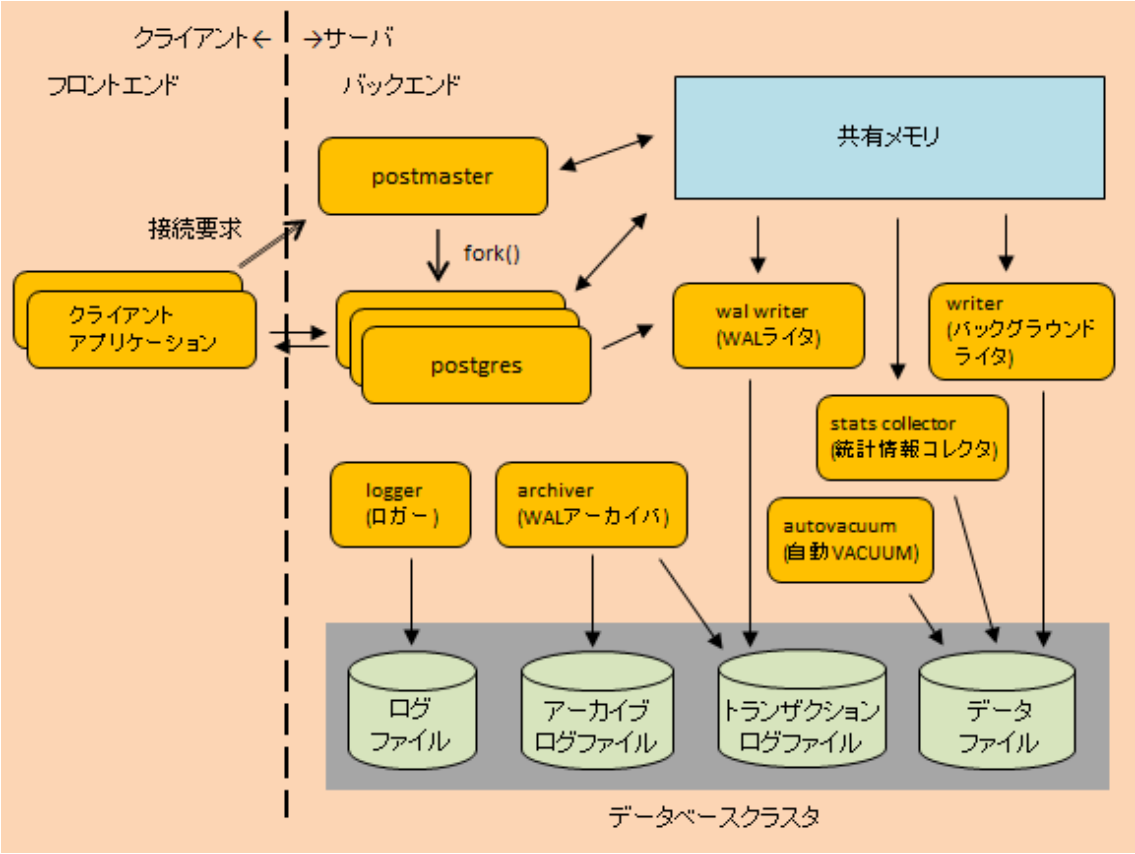


図 1: プロセスの構成(PostgreSQL)

PostgreSQL ではクライアント側を「フロントエンド」、サーバ側を「バックエンド」と呼びます。
バックエンドではマスタープロセス(postmaster)がフロントエンドからの接続要求を待ち受けます。
フロントエンドからの接続要求を受け入れると、マスタープロセスは postgres バックエンドプロセスを生成(fork)し、以降はフロントエンドのアプリケーションと postgres バックエンドプロセスが直接通信を行います。
postgres バックエンドプロセスはフロントエンドのアプリケーションと1対1で対応します。
フロントエンドとバックエンドの間のプロセス間通信には libpq プロトコルが使用されます。

PostgreSQL の起動は OS 上から pg_ctl コマンドを実行します。マスタープロセスが起動し、共有バッファが確保され、個別の各種ワーカープロセスが起動します。マスタープロセスは接続を受け付けるリスナーの役割も行うため、これでフロントエンドからの接続要求を待機している状態となります。

Oracle も同じくクライアント・サーバ型、マルチプロセス構成です。
プロセス構成は以下のようになります。(図 2)

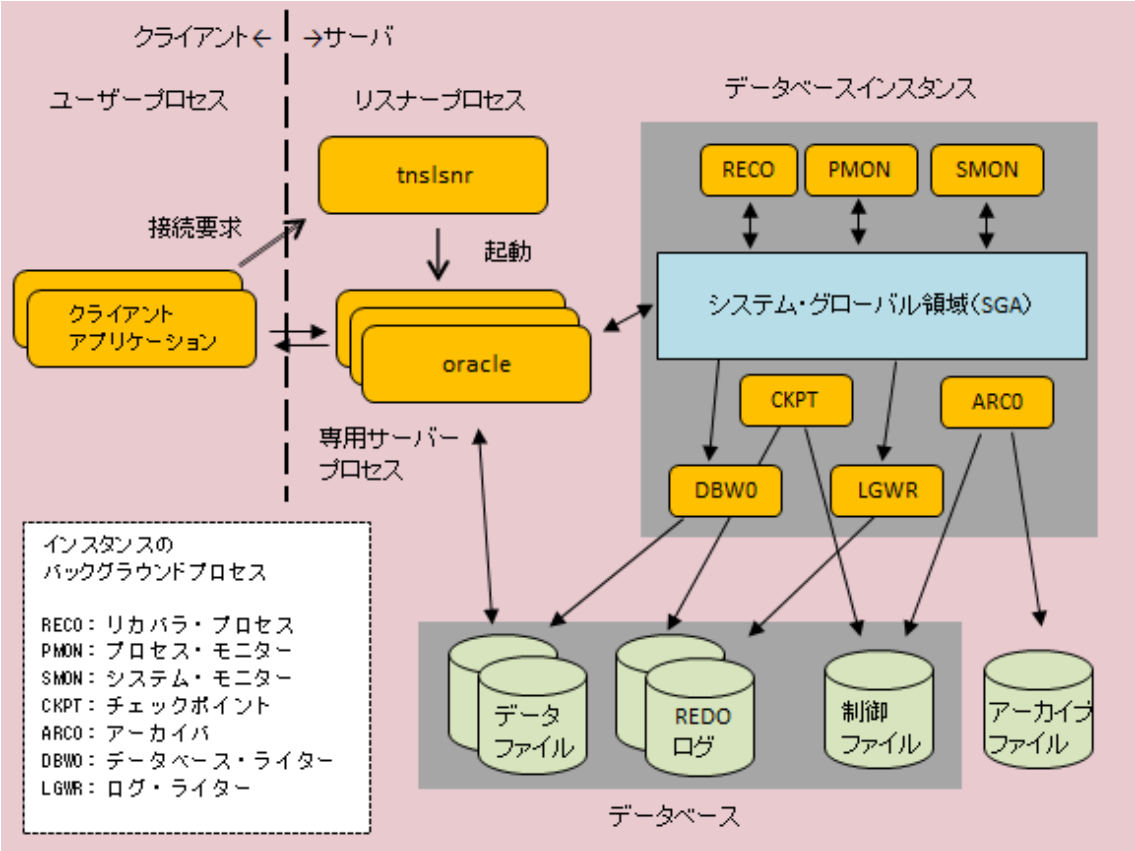


図 2: プロセスの構成(例:Oracle)

データベースインスタンス側は役割ごとの複数のプロセスに分けられており、分担して動作します。
クライアント側のユーザープロセスからの接続要求はリスナープロセスが受け付けます。
ユーザープロセスから接続の要求が行われると、リスナープロセスはインスタンスに対して認証を要求し、認証後にサーバプロセスを生成します。その後はユーザープロセスとサーバプロセスが直接通信を行います。

ユーザープロセスとサーバプロセスは1対1で対応しますが(専用サーバ構成)、同時接続ユーザ数が多い場合には、サーバ側で起動するプロセス数を抑えるような構成も選択できます(共有サーバ構成)。
ユーザープロセスとサーバプロセスの間の通信は Oracle Net プログラムが使用されます。

Oracle の起動は、SQL*Plus ユーティリティから startup コマンドによりデータベースインスタンスを起動します。共有メモリ領域が確保され、各プロセスが起動されます。インスタンスが起動しているのみではまだ接続は受け付けられない状態です。リスナープロセスを別個に起動する必要があります。

移行先の選定にあたってプロセスの構成の違いが影響することは特にありません。上記のように起動の手順が異なる点などについては認識しておく必要があります。

1.1.2. メモリ構成

PostgreSQL のメモリ構成は共有メモリ領域と、プロセス毎のメモリ領域に分けられます。(図 3)

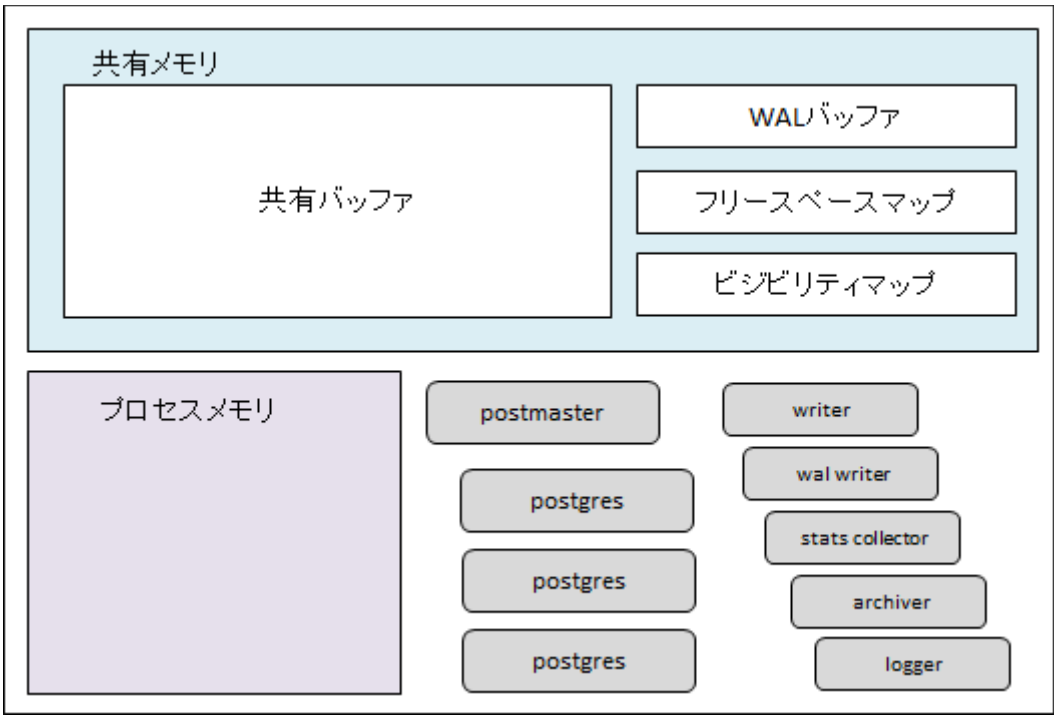


図 3: メモリの構成(PostgreSQL)

PostgreSQL は「追記型」のアーキテクチャを取っており、共有メモリ内にはこのために使用される「フリースペースマップ」「ビジビリティマップ」という領域があります。
フリースペースマップは、テーブル上で利用可能な空き領域を指し示す情報を扱う領域、またビジビリティマップはテーブルのデータが可視であるかを管理する領域になります。
メモリ管理は手動管理の形式で、データベースクラスタの制御ファイル postgresql.conf のパラメータにて指定した値が用いられます。(キャッシュメモリサイズ、チェックポイント実施間隔、WAL バッファサイズなどの項目があります)

Oracle のメモリ構成は、大きくはインスタンス内で共有されるメモリ領域(SGA)と、プロセスに固有のメモリ領域(PGA)に分けられます。(図 4)

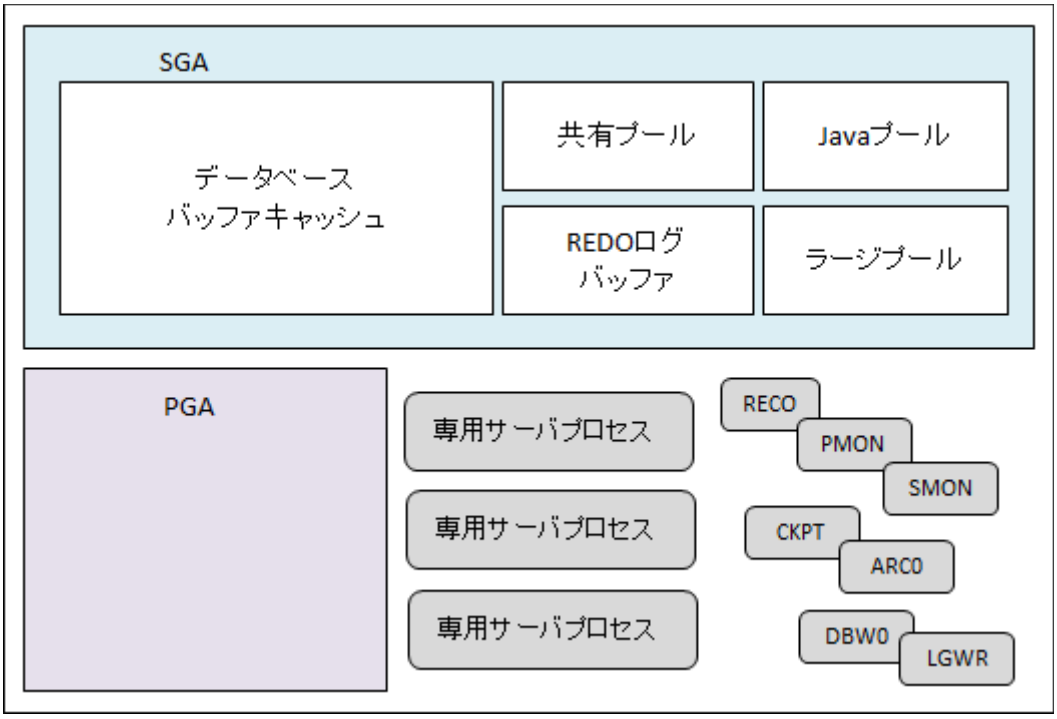


図 4: メモリの構成（例：Oracle）

共有プールと呼ばれる領域にライブラリキャッシュ領域があり、SQL の解析結果がキャッシュされ共有利用される仕組みとなっています。
メモリ管理は自動管理と手動管理の2種類があります。自動管理はメモリの初期化パラメータに SGA と PGA の合計メモリサイズを指定するのみで最適なサイズに管理されます。手動管理は Oracle の従来からの方式で、各メモリコンポーネント毎に個別にサイズを指定する方式です。

移行先の選定にあたってメモリ構成の違いが影響することは特にありません。各メモリ領域の名称と役割を認識しておくと良いでしょう。

1.1.3. データ構成

PostgreSQL は、データベースの集合を「データベースクラスタ」として管理します。「データベースクラスタ」は他の DBMS で「インスタンス」と呼ばれるものに相当します。

データベースクラスタ内に複数個のデータベースを持つことができるため、インスタンスとデータベースは1対Nの関係になります。(図 5)

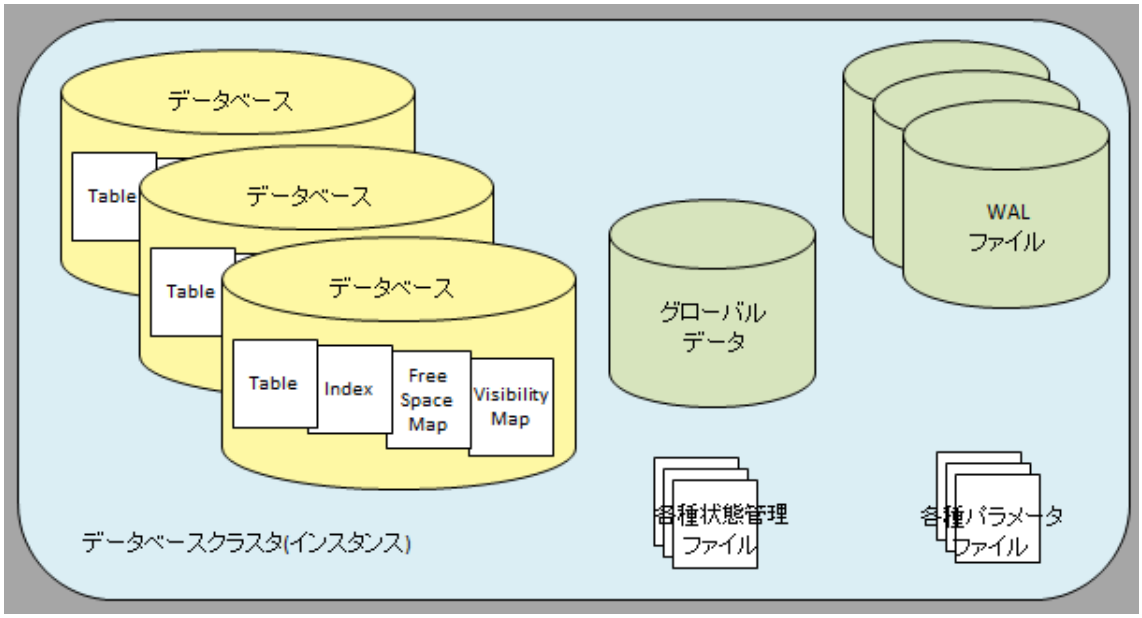


図 5: データの構成(PostgreSQL)

データベースクラスタは基本的には1つのディレクトリで管理され、データベースの情報はデータベース毎のサブディレクトリ内にファイルとして格納されます。

表データは基本的には 1 テーブルが 1 ファイルとなります。(図 6)

1 つの表データファイルに複数の表のデータを格納することはできませんが、テーブルスペース(表領域)機能により1つの表の表データを複数の物理ファイルに分割して配置することが可能です。

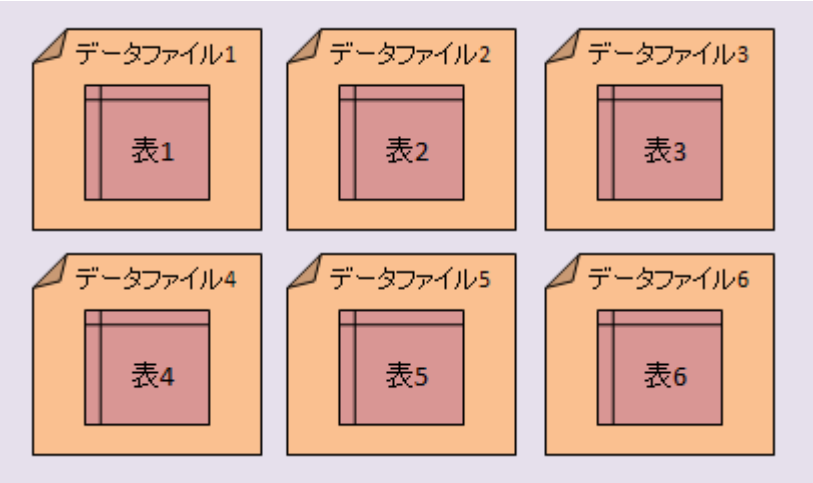


図 6: データファイルの構造(PostgreSQL)

Oracle では、インスタンスとデータベースは1対1の関係で、1 つのインスタンスにつき、インスタンス構成情報とデータベース構成情報を 1 つずつ持ちます。(図 7)

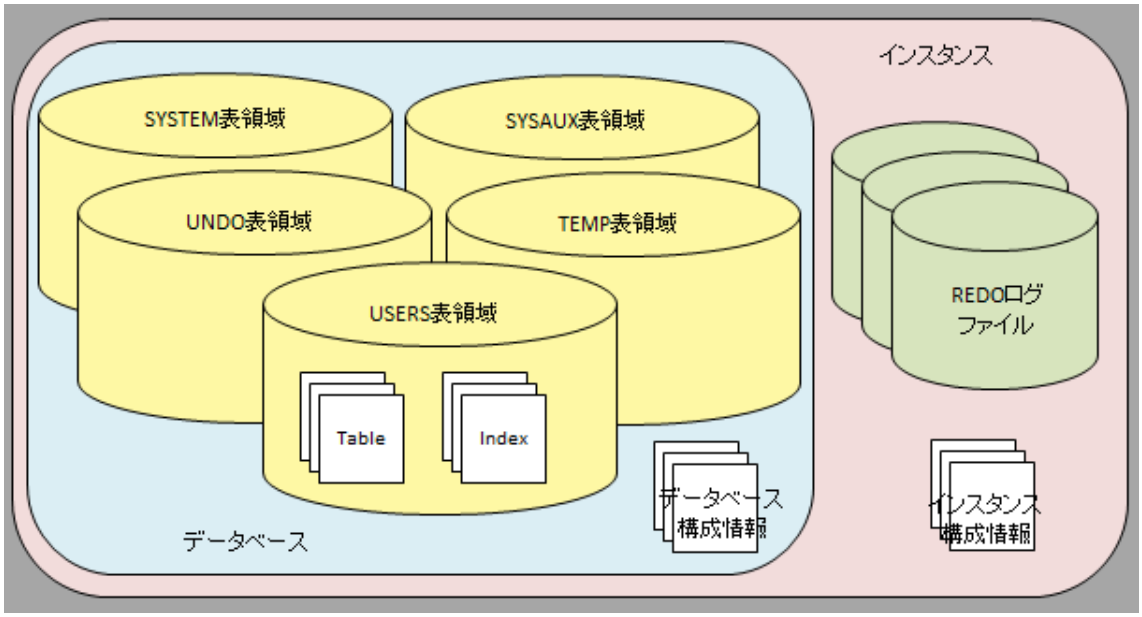


図 7: データの構成(例:Oracle)

表は「表領域」という論理的な領域に作成され、「表領域」に対して複数のデータファイルを割り当てることができます。
1 つのデータファイルに複数の表を格納することや、1 つの表のデータを複数のデータファイルに分割して格納することができます。(図 8)

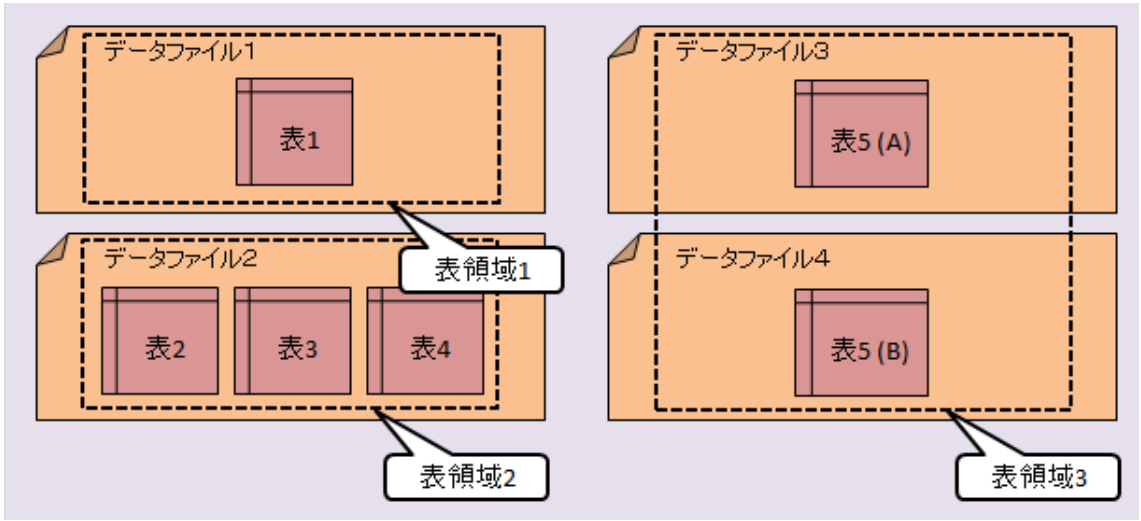


図 8: データファイルの構造(例:Oracle)

データ構成の違いは移行の難易度を大きく左右したりするものではありませんが、移行元 DBMS に構築されているデータを移行先の構成にどのように合わせるか、データ移行の設計に関わりますので、十分に理解しておく必要があります。

1.1.4. 同時実行性の実現方式

PostgreSQL はデータの複数のバージョンを持つ他版型同時実行制御の方式ですが、コミット前の情報を保存する専用の領域は持っていません。更新・削除でも以前の行データが物理的には削除されずに残される方式を取っています。(追記型アーキテクチャ)
あるセッションの変更がコミットされていないタイミングで、他のセッションが同じ情報を参照する場合は参照の時点で確定されている行データ情報が用いられます。

PostgreSQL の追記型では、その方式の都合上、表の領域が拡張していくため、不要になったデータを定期的に削除する必要があります(VACUUM 処理)。近年のバージョンでは VACUUM 処理の性能が向上し、自動実行(AUTOVACUUM)に任せて高いレベルの性能維持ができるようになっています。

Oracle では、DML(レコード操作)にて変更する前の情報を UNDO 表領域内の UNDO セグメント領域という専用の領域に保存しています。(他版型の同時実行制御)
あるセッションの変更がコミットされていないタイミングで、他のセッションが同じ情報を参照する場合は UNDO セグメントに保存された情報が用いられます。

Oracle の UNDO セグメントは循環方式で変更前の情報を保存する仕組みとなっていますが、この領域(以前の Oracle では「ロールバックセグメント」と呼ばれていた領域)のチューニングはかなり難しいものでした。近年のバージョンではメモリの自動管理の機能が設けられチューニングに関する手間は軽減されています。

同時実行性の実現方式について、他版型制御のほかにはロック法を採用している DBMS があります。実現方式自体は移行に直接の影響はありませんが、そこから派生する違いは影響があります。
例えばシステムの運用上で VACUUM を行う時間が確保できないというような場合は、移行先に PostgreSQL を選定することは難しくなります。

1.1.5. ロック

PostgreSQL では、テーブルの読み取りのみで変更を行わない問い合わせでは、共有ロック(読込みロック)が取得されます。
また、PostgreSQL では行レベルのロック量が増えることによって自動的にテーブルロックに変更を行うロック・エスカレーションはありません。

Oracle はテーブルの読み取りのみで変更を行わない問い合わせであれば共有ロック(読込みロック)を使用しません。
※更新を前提としたデータ読み取りの際は FOR UPDATE 句にて明示的に行ロックを取得します。
Oracle でも多数の行レベルロックを表レベルのロックにエスカレーションする機構はありません。

ロック機構の違いは移行の作業量にやや関わります。例えばロックエスカレーションの有無は並行して実行されるトランザクションの処理結果に影響します。アプリケーションで発行される SQL を調整したり、処理の順番を見直したりしなければならない可能性があります。

1.1.6. トランザクション分離レベルの実装の違い

PostgreSQL はトランザクション分離レベルは3種類をサポートしています。

・コミット読み取り(READ COMMITTED)
・繰り返し可能読み取り(REPEATABLE READ)
・直列可能(SERIALIZABLE)

追記型による実装により「繰り返し可能読み取り」についてはファントムリードの可能性がありません。
また、「未コミット読み取り(READ UNCOMMITTED)」も指定することは可能ですが、「コミット読み取り」として扱われます。(PostgreSQL では「コミット読み取り」と同じです。)
PostgreSQL の既定の分離レベルも「コミット読み取り」です。

Oracle では2つのトランザクション分離レベルをサポートしています。

・コミット読み取り(READ COMMITTED)
・直列可能(SERIALIZABLE)

それぞれの動作は UNDO セグメントを使用してどの時点のデータを生成するかの違いにより実現しています。
「コミット読み取り」では SQL 開始時点でコミットされたデータを使用し、「直列可能」では非リピータブル・リードが発生しないようにトランザクション開始時点でコミットされたデータを使用します。
Oracle の既定の分離レベルは「コミット読み取り」です。

トランザクション分離レベルについて実装の違いが移行に影響することはありません。移行元のシステムにて規定の分離レベル以外を用いて特殊な処理を行っている場合は、移行先で同等な処理ができるか、アプリケーションの改修が発生するかなど、確認が必要になります。

1.2. スキーマ・オブジェクト

PostgreSQL9.5 は SQL:2011 の主な機能のほとんどをサポートします[1]。一般的なテーブルやビュー、マテリアライズドビューをサポートし、多くの DBMS で採用されている列属性を提供します。また、性能を確保するためのインデックスやパーティショニング機能を提供します。

1.2.1. DBMS が提供するスキーマの差異

スキーマのオプションや固有な機能を定義するスキーマは製品ごとに異なりますが、商用 DBMS も OSS DBMS である PostgreSQL も基本的なスキーマはサポートしています。

表 1: 代表的なスキーマ一覧

スキーマの種別	PostgreSQL	Oracle	Microsoft SQL Server
データベース	データベース	データベース	データベース
スキーマ	スキーマ	スキーマ	データベースおよびデータベース所有者（DBO）
テーブルデータの物理的格納先	テーブルスペース	表領域	データベース
DB 接続ユーザ	ユーザ	ユーザ	ユーザ
DB へのアクセス権限定義	ロール／グループ	ロール	グループ／ロール
表・テーブル	表	表	表
一時表	一時表	一時表	一時表
チェック制約	チェック制約	チェック制約	チェック制約
順序・シーケンス	シーケンス	シーケンス	シーケンス
一意インデックス	一意インデックス	一意キー	一意キーまたは列の識別プロパティ
主キー	PRIMARY KEY	主キー	主キー
外部キー	外部キー制約	外部キー	外部キー
代理キー	インデックス	索引	非一意索引
ストアドプロシージャ	SQL 関数／PL/pgSQL など	PL/SQL プロシージャ	Transact SQL ストアド・プロシージャ
ファンクション	SQL 関数／PL/pgSQL など	PL/SQL ファンクション	Transact SQL ストアド・プロシージャ
パッケージ	なし	パッケージ	なし
トリガー定義	AFTER／BEFORE トリガー	AFTER／BEFORE トリガー	トリガー／複合ルール
シノニム・エイリアス	なし（VIEW で代替）	シノニム	なし
マテリアライズドビュー	マテリアライズドビュー	マテリアライズドビュー	インデックス付きビュー

1.2.2. データ型

PostgreSQL と商用 DBMS のデータ型の違いは、以下の資料をご覧ください。

- PostgreSQL と Oracle のデータ型の対応
- PostgreSQL と Microsoft SQL Serve のデータ型の対応

1 PostgreSQL マニュアル 付録 D SQL への準拠
http://www.postgresql.jp/document/9.5/html/features.html

1.3. 国際化対応

PostgreSQL の格納データやメッセージに関する国際化の対応状況についてご紹介します。

1.3.1. 格納データの国際化対応

PostgreSQL の文字セットサポートにより、シングルバイト文字や マルチバイトの各種文字セットでテキストを保存することができます。代表的な日本語の文字エンコーディングは以下の 2 つです。

- EUC_JP
- UTF8

PostgreSQL は SJIS の文字セットをサポートしていません。しかし、サーバ・クライアント間の文字セット変換により、SJIS のデータを扱うことができます。

名前など JIS2004 の日本語文字コードを使用する場合、UTF8 を選択します。EUC_JP に比べてデータサイズが大きくなるため、格納文字コードの変更には注意が必要です。

1.3.2. ソート順やメッセージの国際化対応

インスタンスの作成(initdb)時やデータベース作成時に、ソート順や格納データのエンコーディングを指定することができます。

表 2: ロケール

設定	設定内容	設定単位
LC_COLLATE	文字列のソート順	インスタンス／データベース単位
LC_CTYPE	格納データのエンコーディング	インスタンス／データベース単位
LC_MESSAGES	メッセージの言語	インスタンス単位
LC_MONETARY	通貨書式	インスタンス単位
LC_NUMERIC	数字の書式	インスタンス単位
LC_TIME	日付と時刻の書式	インスタンス単位

他データベースからの移行を行う際、レコードの返却順が変わるため LC_CTYPE や LC_COLLATE の設定には注意が必要です。この 2 つの設定はデータベース作成後に変更することはできません。また、ロケールを意識したソートを行うと意味のある順番にレコードが返却されますが、ソート性能が大幅に劣化するため注意が必要です。

LC_MESSAGES はログに出力されるメッセージの言語を設定します。PostgreSQL のビルド時に--enable-nls オプションを指定することで使用可能になります。

メッセージの翻訳については各言語ごとの対応状況[2]が公開されています。リリース直後のメジャーバージョンでは未翻訳の割合が大きくなりますが、日本語(ja)については概ね 8～9 割以上のメッセージが対応されています。

1.3.3. 日付や時刻の国際化対応

DateStyle パラメータにより、既定の日付の形式を指定することができます。また、タイムゾーンを指定し、各国の標準時でデータを扱うことができます。各国の標準時対応には夏時間の設定も含まれ、法改正に応じて夏時間の設定が変更されます。

1.3.4. ドキュメントの国際化対応

PostgreSQL のオンラインマニュアル[3]については、PostgreSQL のリリース初期の頃から日本語に翻訳されたものが公開されています。現在は国内コミュニティの貢献によりメンテナンスされており、新しいバージョン・リビジョンのリリースに対して、早期に翻訳対応がなされています。

2 NLS Status Tables https://babel.postgresql.org/

3 PostgreSQL 日本語ドキュメント http://www.postgresql.jp/document/current/index.html

1.4. 開発言語

DBMS に求められる要件として、利用できるプログラミング・インターフェイス（利用できる開発言語）が関心事のひとつとして挙げられます。PostgreSQL は、他の DBMS と同様に SQL を具備した上で、様々なプログラミング・インターフェイスを備えます。プログラマのスキルセットや要員確保のし易さや、ソースコードの保守性を考慮した上で、様々な開発言語を選択することが出来ます。定義した処理を実行するホストが DBMS（サーバ）か、PostgreSQL に接続するホストや別プロセスによる実行（クライアント）で選択できるインターフェイスが異なります。

表 3: プログラミング言語の種類（サーバ）

No.	インターフェイス	概要	利用可能な PostgreSQL バージョン（7.4 から）
1	SQL	SQL を用いて、関数や型の定義をすることができます。	7.4
2	C 言語	C 言語を用いて関数や型の定義をすることが出来ます。	7.4
3	PL/pgSQL（ぴーえる・ぴーじーえすきゅーえる）	上述の SQL や C 言語で定義した言語を利用して、独自の処理言語で関数の定義をすることが出来ます。他のプログラミング言語の実行環境等を導入したくないのであれば、選択はベターと言えるかもしれません。	7.4
4	PL/Tcl（ぴーえる・ていくる）	Tcl を持ちいて PL/pgSQL と同等の記述ができます。C 言語で関数を記述することに比べて安全に処理を記述することが出来ます。	7.4
5	PL/Perl（ぴーえる・ぱーる）	Perl を用いて PL/pgSQL と同等の記述ができます。Linux/Unix 利用者間において、広く扱われている Perl を選択することで、サーバサイドコードの保守が容易となるかもしれません。	7.4
6	PL/Python（ぴーえる・ぱいそん）	Python を用いて PL/pgSQL と同等の記述ができます。近年プログラマ人口が増加傾向と考えられる Python を選択することで将来的に保守を容易とできるかもしれません。	7.4

表 4: プログラミング言語の種類(クライアント)

No.	インターフェイス	概要	利用可能な PostgreSQL バージョン(7.4 から)
1	libpq (りぶぴーきゅー)	C 言語ライブラリです。PostgreSQL クライアント標準の関数群であるため、他の処理系に比して高速で動作します。様々なクライアントアプリケーションを実装することができます。	7.4
2	ECPG (いーしーぴーじー)	独自の構文(SQL 埋め込み C 言語記述式)を用いて、DBMS 操作を記述することが出来ます。SQL および C 言語を理解しているプログラマであれば記述のし易い形式でクライアントアプリケーションを実現できます。	7.4
3	Perl 言語	Perl で利用可能な PostgreSQL ドライバが CPAN 等で提供されています。DBI/DBD による制約のもと実装されているので、ある程度の DBMS 間の移植性はあります。Web アプリケーション等のクライアントアプリケーションが実現できます。	接続対応可能なバージョンは利用するドライバ実装に拠ります また、libpq を利用する場合は、libpq に依存します
4	PHP 言語	PHP では標準関数として libpq を利用する PostgreSQL 操作関数が提供されています。他のプログラム言語と異なり、追加ライブアリ等の利用をすることなくクライアントアプリケーションを実現できます。	利用する libpq に依存します
5	Java 言語	Java 言語のための PostgreSQL JDBC Driver が公開されています。ドライバは libpq を利用せずに利用できるため、別途 libpq を導入せずに利用できます。	7.4/JDBC2
6	Ruby 言語	Ruby のための PostgreSQL ライブラリが幾つか公開されています。libpq の導入が別途必要か否かはライブラリに拠りますので、導入対象のドキュメントを確認してください。	接続対応可能なバージョンは利用するドライバ実装に拠ります (サードパーティ製)
7	Python 言語	Pyhton のための PostgreSQL ライブラリが幾つか公開されています。libpq の導入が別途必要か否かはライブラリに拠りますので、導入対象のドキュメントを確認してください。	接続対応可能なバージョンは利用するドライバ実装に拠ります (サードパーティ製)
8	.Net Framework 実行環境言語 (VB.net、C#等)	.Net Framework 実行環境で利用できる Npgsql が公開されています。JDBC Driver と同様に libpq を必要とせずにクライアントアプリケーションを実現できます。	7.4
9	その他 (旧 Microsoft プログラミング言語等)	ODBC インタフェースを実現した psqlODBC が公開されています。VisualBasic 等のプログラミング言語で利用できます。	7.4

サーバのプログラミング・インターフェイス利用例は、別途 WG2 文書「ストアドプロシージャ移行調査編/組み込み関数移行調査編」を参照するとより具体的な知見が得られます。

1.5. 前提条件

PostgreSQL は POSIX、SYSV の UNIX 系 OS に対応しており、Windows についても PostgreSQL 8.0 から正式にサポートされています。

マニュアルにて動作が期待できると記載されている CPU および OS は以下の通りです[4]。

表 5: PostgreSQL が利用できる CPU アーキテクチャおよび OS の種類

CPU	x86, x86_64, IA64, PowerPC, PowerPC 64, S/390, S/390x, Sparc, Sparc 64, ARM, MIPS, MIPSEL, M68K,PA-RISC
OS	Linux, Windows (Win2000 SP4 以降), FreeBSD, OpenBSD, NetBSD, OS X, AIX, HP/UX, Solaris, UnixWare

コミュニティはボランティアで各環境におけるビルドと動作検証を行っています。動作検証が行われている環境は、ビルドファーム[5]で確認することができます。

4 15.6 サポートされるプラットフォーム
http://www.postgresql.jp/document/9.5/html/supported-platforms.html
5 PostgreSQL BuildFarm Status http://buildfarm.postgresql.org/cgi-bin/show_status.pl

2. 技術要件(特徴的な機能)

PostgreSQL には独特な機能や機能拡張の仕組みといった特徴的な機能が多数存在します。
ここでは機能の一部(表 6: 特徴的な機能一覧)を紹介しています。

表 6: 特徴的な機能一覧

カテゴリ	機能	概要	対応バージョン
データ型	範囲型	{ 下限, 上限 } をペアで扱うデータ型で、連続値だけでなく離散値も扱えます。要素としては、整数・数値・日付の範囲など、用途にあわせて型を選択できます。	9.2
	JSON データ型	JSON データを格納するために、json 型と jsonb 型という 2 種類のデータ型があります。json 型は格納効率、jsonb 型は検索効率の用途で使い分けができます。	9.2 (json 型) 9.4 (jsonb 型)
インデックス	部分インデックス	テーブルの一部のレコードだけに適用するもので、WHERE 句で指定した条件を満たすレコードに対してインデックスを作成します。そのため、インデックスのサイズが小さく、検索効率に優れています。	-
	式に対するインデックス	テーブルの1つ以上の列から計算される関数やスカラ式に対してインデックスを作成します。 ある演算結果に基づいたテーブルの高速アクセスが可能です。	-
セキュリティ	行レベルセキュリティ	テーブルに対して、行レベルのセキュリティを付与できます。これにより、テーブルに対して、あるユーザーが特定のデータを閲覧できるかどうかのポリシーを作ることができます。	9.5
オブジェクト指向	テーブルの継承	基本となる親テーブルからテーブル構造とデータを継承した子テーブルを作成できます。子テーブルを更新することで親テーブルに更新結果が反映されます。	-
地理空間情報	GIS(地理情報システム)	地理データ型、地理空間情報の演算、高速な検索が実装できます。PostgreSQL で地理空間情報を扱うための拡張である PostGIS が必要です。	7.2
外部データ	FDW(外部データラッパ)	PostgreSQL の外部にあるデータに対して標準的な SQL 文でアクセスできます。外部データへのアクセスはデータリソースへ接続する FDW を介して行います。外部データには、DBMS、NoSQL DB、フラットファイル、Web サービス等があります。	9.1(参照) 9.3(更新)
文書検索	全文検索	全文検索は、問合わせを満たす自然言語の文書を識別し、更には問合わせとの関連性の順に並び替えることができます。もっとも一般的な検索は、与えられた検索語を含む文書を探し、問合わせとの類似性の順に返すというものです。	-

また、次節以降では、企業情報システムの既存機能の代替案や新機能の追加案などに有効と思われるものを特徴的な機能を選定して、詳細に説明しています。

2.1. GIS（地理情報システム）

GIS(Geographic Information System/地理情報システム）とは地理情報(位置に関連づけられた情報)を要件に応じて管理、加工し、検索結果や分析結果を出力（可視化）するシステムを指します。
本節では、PostgreSQL を地理情報システムとして利用するための拡張機能である「PostGIS（ぽすとじす）[6]」について述べます。

2.1.1. 機能の説明

PostGIS は、GIS データベース（空間データベース）を PostgreSQL で取り扱うための拡張機能です。
空間データベースでは、他のデータ（文字列や数値等）と同じように地理情報データを SQL で処理することが可能です。下表に空間データベースが備える主な機能を記載します。

表 7: 空間データベースの主な機能

No.	カテゴリ	機能	概要
1	データ型	空間データ型	地理情報データを格納するための、データ型を提供します。
2	インデックス	空間インデックス	格納された地理情報データに高速にアクセスするためのインデックスを提供します。
3	関数	空間関数	地理情報データを処理するための関数群が提供されています。

空間データベースには、地図を表示する機能はないため、他のソフトウェアと組み合わせて利用する必要があります。下図は地図表示のサンプルです。下図では、国土交通省国土計画局が提供している、
「国土数値情報　ダウンロードサービス[7]」より、福岡県の「行政区域」のデータ、および、「学校」データを利用しています。本サンプルを作成するにあたり、PostGIS 導入を行った PostgreSQL に福岡県の「行政区域」のデータ、および、「学校」データを格納し、データの表示には、「QGIS（きゅーじあいえす）[8]」を利用しています。

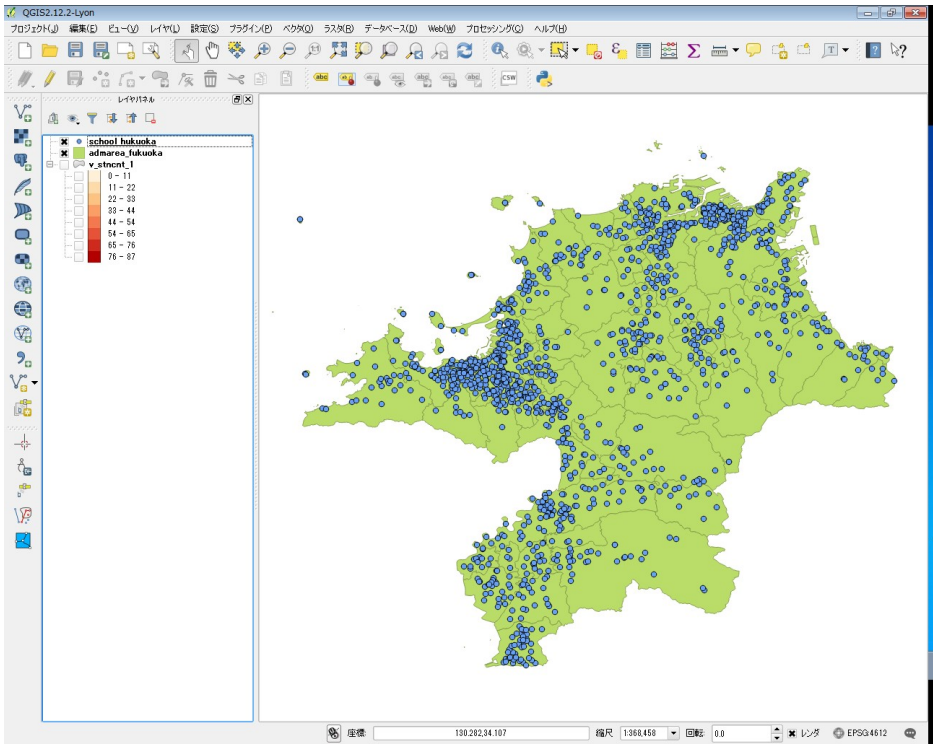


図 9: （PostGIS と QGIS の利用例）福岡県における学校所在地

6 PostGIS : http://postgis.net/
7 国土数値情報　ダウンロードサービス : http://nlftp.mlit.go.jp/ksj/
8 QGIS : http://www.qgis.org/ja/site/

地理情報を扱うための基礎知識を以下に記載します。

1　緯度経度(地理情報の単位)

地図では、地球上の任意の地点を表現する場合に緯度と経度を利用することがあります。緯度・経度の値は、地球上の任意の地点を下表の角度で表現します。

表 8: 緯度・経度

No.	項目	概要
1	緯度	ある地点の赤道面との角度で表現します。 赤道上を 0 度として、北極点、南極点はそれぞれ北緯 90 度, 南緯 90 度となります。
2	経度	ある地点のグリニッジ子午線からの角度で表現します。 子午線上を 0 度として、グリニッジから東を東経, 西を西経と呼びます。

2　測地系(地理情報の求め方)

緯度・経度を求めるために基準となるものが測地系です。測地系は様々な種類があり、測地系が違うと同じ位置でも緯度・経度が変わります。日本では下表 の 4 種類がよく利用されます。

表 9: 日本でよく利用される測地系

No.	項目	概要
1	Tokyo Datum	明治時代より古くから利用されてきた日本独自の測地系です。 他国で利用されている測地系と座標値が一致していないため、利用しづらいものとなり、日本では 2002 年 4 月 1 日から JGD2000 へ切り替えられました。
2	JGD2000	世界的な基準で決定された測地系です。 日本では、JGD2000 が 2002 年から 2011 年まで使われていました。
3	JGD2011	2011 年東北地方太平洋沖地震に伴う地殻変動が非常に大きかったため、JGD2000 から JGD2011 に切り替えられ、利用されています。
4	WGS84	米国が構築・維持している測地系です。GPS で利用されています。

測地系は、「準拠する楕円体(準拠楕円体)」、「地球と楕円体の重ね方」等の要素によって定まります。測地系の差異は構成する要素の違いが原因です。下記に「準拠楕円体」について記載します。
「地球と楕円体の重ね方」については、IERS(国際地球回転観測事業)という国際的な学術機関が構築している ITRF(International Terrestrial Reference Frame)が利用されています[9]。

2.1 準拠楕円体

実際の地球は、球体ではなく、自転による遠心力により赤道付近が膨らんだ形になっています。
緯度、経度などの情報を測量する際には、まず地球がどのような形になっているかを定義する必要があります。
そこで、地球の形を回転楕円体(どら焼き型)でモデル化し、近似する試みがなされています。このモデルは準拠楕円体と呼ばれます。準拠楕円体はいくつかの種類が存在します。
各測地系において、採用された準拠楕円体を下表に記載します

表 10: 測地系と準拠楕円体

No.	準拠楕円体	測地系	備考
1	ベッセル楕円体	Tokyo Datum	2002 年 3 月までの日本で使用されていた
2	GRS80 楕円体	JGD2000、JGD2011	アメリカ、ヨーロッパ、日本で使用されている
3	WGS84 楕円体	WGS84	GPS に使用されている

9 世界測地系移行に関する質問集 (問 1-7)ITRF 系とは何か : http://www.gsi.go.jp/LAW/G2000-g2000faq-1.htm#qa1-7

3　投影法と各種図法（地理情報の出力方法）

経度・緯度で位置を表現することはできますが、地球上のある物体を地図上（2 次元の平面）に表現するには、楕円である地球を平面上に変換する必要があります。楕円の地球を平面上に変換する手法を、「投影法」と言います。主なものをいくつか下表に記載します。

表 11: 投影法で表現される図法（一部）

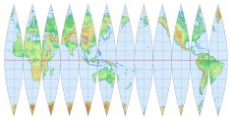
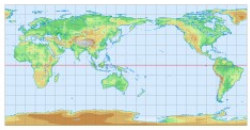
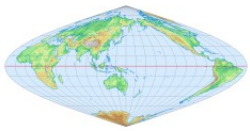
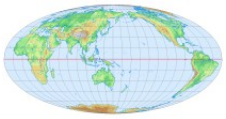
No.	名称	説明	図
1	舟形多円錐図法	地球の表面を経線を基準に切り開いた図。	
2	正距円筒図法	舟形多円錐図法の隙間がある部分を左右に伸ばして、隙間をなくした図。 高緯度ほど、伸びしろが大きく不正確な形になる。	
3	メルカトル図法	正距円筒図法を縦にも伸ばした図。 地形が正距円筒図法より正確。	
4	サンソン図法	舟形多円錐図法の隙間を中心に寄せることで、隙間をなくした図。 面積が正しく表されている。	
5	モルワイデ図法	サンソン図法をもとに緯度ごとの縦横比を変えて、形を整えた図。 面積が正しく表されている。	

表 11 は、佐藤崇徳先生の論文[10]を参考に作成しました。また図も引用させていただきました[11]。

4　投影座標系（地図上の位置）

投影法により、平面に投影された物体の位置は、平面上の原点からの座標で示すことが可能です。投影された座標は、「投影座標」と呼ばれます。投影座標は、同じ投影法を利用しても、原点や範囲の設定により異なります。投影法、原点、範囲を組み合わせたものを「投影座標系」と呼ばれています。また、緯度・経度から地図に投影する必要がない場合もあり、そのまま利用する場合もあります。この座標系を「地理座標系」といいます。

10 コンピューターを利用した地図投影法学習教材の作成および公開：
https://www.jstage.jst.go.jp/article/jgeography/124/1/124_124.137/_article/-char/ja/

11 地図投影法学習のための地図画像素材集: http://user.numazu-ct.ac.jp/~tsato/tsato/graphics/map_projection/index.html

5　空間参照系（位置の決定）

地球上の任意の位置の緯度・経度は、測地系により定まります。また、その緯度・経度の値を平面に投影した座標は、投影座標系によって定まります。従って、任意の場所を表す座標は、測地系と投影座標系を決定する必要があり、その組み合わせの一覧を「空間参照系（Spatial Reference System）」といいます。空間参照系は SRID（Spatial Reference Identifier）と呼ばれる識別番号（整数値）で管理されています[12]。SRID を利用し、任意の位置座標がどの測地系とどの投影座標系から定まっているのか、解決できます。日本でよく使用される空間参照系の一覧を下表に記載します。

表 12: 日本でよく利用される SRID

No.	SRID	測地系	投影座標系
1	4612	JGD2000	地理座標系
2	2443 から 2461	JGD2000	平面直角座標系（1 系から 19 系まで）
3	3097 から 3101	JGD2000	UTM 座標系（ゾーン 51 から 55 まで）
4	6668	JGD2011	地理座標系
5	6669 から 6687	JGD2011	平面直角座標系(1 系から 19 系まで）
6	6688 から 6692	JGD2011	UTM 座標系（ゾーン 51 から 55 まで）
7	4301	Tokyo Datum	地理座標系
8	3092 から 3096	Tokyo Datum	UTM 座標系（ゾーン 51 から 55 まで）
9	4326	WGS84	地理座標系

6　データ形式

地理情報データの形式はさまざま存在しますが、GIS で取り扱う際には、「ベクタデータ」と「ラスタデータ」の 2 種類に大別されます。

表 13: ベクタデータとラスタデータ

No.	データ形式	概要
1	ベクタデータ	地球上の物体を、ポイント（点）、ライン（線）、ポリゴン（多角形）等で表現する形式
2	ラスタデータ	地表をセルに分割し、分割したセル内に情報を入れることで、表現する形式

GIS では、上記の 2 つのデータ形式を用途によって組み合わせて用いることで、単純な地図を出力するだけに留まらず、用途に合わせた出力表現をとることが出来ます。

12 EPSG Geodetic Parameter Registry : http://www.epsg-registry.org/

2.1.2. 機能の特徴

PostGIS は、地理情報を扱うためのオブジェクトや関数を提供します。主な機能を下表に記載します。

表 14: PostGIS が備える主な機能

No.	機能	大分類	小分類	説明	PostGIS の機能
1	空間データ型	ベクタデータ	ポイント(点)	左記ベクタデータを格納するデータ型	GEOMETRY 型 ※ (ジオメトリ型)
2			ラインストリング (線)		
3			ポリゴン (多角形)		
4			マルチポイント (複数の点)		
5			マルチライトストリング (複数の線)		
6			マルチポリゴン (複数のポリゴン)		
7			ジオメトリ[13]コレクション (上記ジオメトリの集合)		
8		ラスタデータ	–	ラスタデータを格納するデータ型	PostGIS 2.0 以降[14]
9	空間インデックス	–	–	空間インデックスを付与することで、高速な検索を実現	GiST を利用した R-Tree インデックス
10	空間関数	計測関数	重心計測	ジオメトリの重心を返す関数	PostGIS 各種関数
11			面積計測	ジオメトリの面積を返す関数	
12			距離計測	2 点間の距離等のポリゴン同士の距離を、返す関数	
13		関係判定関数	同一判定	ジオメトリが空間上同一であるか否か判定する関数	
14			包含関係判定	2 つのジオメトリに包含関係があるか否かを判定する関数	
15			交差関係判定	2 つのジオメトリに交差する部分があるか否かを判定する関数	
16			距離判定	2 つのジオメトリが指定した距離内にあるか否かを判定する関数	
17		情報確認関数	座標値の取得	座標値を取得する関数	
18			SRID の取得	ジオメトリの元となる SRID を取得する関数	
19		データ形式変換関数	地理情報 テキスト形式	ジオメトリをテキスト形式 (Well-Known Text)で出力	
20			地理情報 バイナリ形式	ジオメトリをバイナリ形式 (Well-Known Binary)で出力	
20			GeoJSON	ジオメトリを GeoJSON 形式で出力	
21	その他	データ形式変換ツール	ESRI Shapefile	ESRI Shapefile を PostgreSQL ファイルに、SQL ファイルを ESRI Shapefile に変換	shp2pgsql、pgsql2shp

※ より厳密な距離を算出する場合等で利用する GEOGRAPHY 型も提供されています[15]

PostGIS には、地理情報をイメージとして表示する機能がないため、GUI を備えた他のソフトウェアと連携する必要があります。QGIS 等多くの GIS ソフトウェアと連携が可能です。また、Web へ配信するためのソフトウェアとも連

13 ジオメトリとは、表現したい対象を点・線・面等を用いて表した形状を指します。

14 PostGIS Raster Home Page : https://trac.osgeo.org/postgis/wiki/WKTRaster

15 PostGIS 2.2.0dev マニュアル日本語訳:
http://www.finds.jp/docs/pgisman/2.2.0/postgis-ja.html#PostGIS_Geography_AdvancedFAQ

携し、最終的に Web アプリケーションとして GIS を表現することも可能です。連携可能なソフトウェアの一部を、下表に記載します。

表 15: デスクトップ型 GIS ソフトウェア

No.	ソフトウェア名	概要	URI
1	QGIS	Open Source Geospatial Foundation (OSGeo) のオフィシャルプロジェクトで、Lunux, Unix, Mac OSX, Windows, Andoid 等の OS で動作し、地理情報を描画する。数多くのベクター・ラスターデータや、データベースフォーマットをサポートしている。	http://www.qgis.org/ja/site/
2	OpenJUMP	Java プログラミング言語で書かれたオープンソース地理情報システム。大量データの空間結合が速いという特徴がある。	http://www.openjump.org/
3	uDig	Eclipse Rich Client Platform(Eclipse RCP)フレームワークで開発されている。RCP プラグインとして Eclipse へ機能拡張という形で地理情報を参照することができる。	http://udig.refractions.net/
4	gvSIG Desktop	Java クライアントアプリケーションとして提供されている GIS クライアントソフトウェア。ベクター・ラスターデータの扱い、PostGIS 等のリモート GIS システムへ接続を行うことが出来る。	http://www.gvsig.com/en/products/gvsig-desktop
5	TileMill	編集や解析といった機能は持っておらず、地図の製図/カートグラフという点に特化している。作成した地図は、画像や MBTile などのファイル形式でエクスポートを行うことができる。	https://www.mapbox.com/tilemill/

表 16: サーバ型 GIS ソフトウェア

No.	ソフトウェア名	概要	URI
1	MapServer	ESRI SHAPE 形式・MapInfo TAB 形式など複数の GIS フォーマットに対応しているため、他の GIS ソフトとの連携も可能である。Windows・Linux 等の OS を問わずに利用できる。	http://mapserver.org/ja/index.html
2	GeoServer	REST (Representational State Transfer) API が組み込まれており、PostGIS/PostgreSQL などへの SQL 操作を隠蔽し、多様なアプリケーションと連携することができる。	http://geoserver.org/
3	Deegree	データアクセス、可視化などを含む地理空間データ管理のためのコンポーネントを提供している。Web Terrain Service を利用した三次元データの表示にも対応している。	http://www.deegree.org/
4	QGIS Server	QGIS と同じライブラリを使用してウェブ地図サービスを提供する。QGIS で作成した地図、印刷テンプレートをサーバにプロジェクトファイルをコピーするだけで公開できる。	http://www.qgis.org/ja/site/
5	MapGuide Open Source	Web-GIS アプリケーションおよび地理空間ウェブサービスの開発や配信を可能とするウェブベース基盤である。開発言語として PHP, Java, JavaScript APIs などが使える。	https://mapguide.osgeo.org/

また、PostGIS/PostgreSQL に関連した下表の拡張モジュールを併用することで、より便利に地理情報を扱うことが可能です。

表 17: PostGIS と関連した PostgreSQL の拡張機能

No.	ソフトウェア名	概要	URI
1	pgRouting	Postgis に経路探索機能を加える拡張モジュール 。 任意の地点 A から B までの最適経路を求める際等に利用する。	http://pgrouting.org/
2	ogrfdw	様々な地理情報ファイルを外部テーブル化するため拡張モジュール。	https://github.com/pramsey/pgsql-ogr-fdw
3	pgpointcloud	点群データ(Point Cloud Data) をサポートするための拡張モジュール。点群とは、3D 空間に存在する物や環境や人を、3D センサによってその表面形状を撮影した「3D 点の座標の集まり」のデータを指します 。	https://github.com/pgpointcloud/pointcloud

2.1.3. 実現方法

PostGIS は PostgreSQL における一般的な機能拡張の方法である、機能拡張モジュール(EXTENSION)として提供されます。

2.1.4. 制限・前提条件

PostGIS は、PostgreSQL の拡張モジュールであるため、動作させるためには PostgreSQL が必要です。
PostGIS の最新安定版（2016 年 3 月現在）リリースである PostGIS バージョン 2.2 を利用するためには、PostgreSQL9.1 以降が必要です。PostgreSQL と PostGIS の対応バージョン関係については「PostGIS Support Matrix[16]」を参考にしてください。

主な OS ついては、PostgreSQL とセットになった各種バイナリパッケージが提供されています[17]。
PostGIS をソースよりコンパイルとして利用可能とする場合（参考 PostGIS 2.2）は、地理情報の扱いや操作関数の実現のために、次のようなソフトウェアが必要になります[18]。

表 18: PosGIS のコンパイルに必要なソフトウェア（PostGIS 2.2 参考）

No.	ソフトウェア種類	実装	URI	バージョン指定
1	DBMS	PostgreSQL	http://www.postgresql.org/	9.1 以降
2	C コンパイラ	GNU C	https://gcc.gnu.org/	–
3	ビルドツール	GNU Make	https://www.gnu.org/software/make/	–
4	座標系投影変換	Proj4	https://trac.osgeo.org/proj/	4.6 以降
5	ジオメトリライブラリ	Geos	https://trac.osgeo.org/geos/	3.3 以降
6	XML 操作ライブラリ	LibXML2	http://xmlsoft.org/	2.5 以降
7	JSON 操作ライブラリ	JSON-C	https://github.com/json-c/json-c/wiki	0.9 以降
8	地理情報抽象化ライブラリ	GDAL	http://www.gdal.org/	1.8 以降

導入方法は「別紙 PostGIS インストール手順書」に記載しています。

2.1.5. 事例紹介

公開されている PostGIS 導入事例を下表に記載します。

表 19: PostGIS の導入事例

No.	導入事例	会社・団体	事例紹介 URI
1	スマポでの PostgreSQL 導入事例	株式会社スポットライト	https://www.pgecons.org/wp-content/uploads/2013/12/7c14ac1727a38c22295af840fc613321.pdf

また、日本における GIS の活用事例としては、「GIS ポータルサイト[19]」より日本政府や各省庁の取り組みや活用事例を参照することができます。
また本 WG の活動の一環として、実際の地理情報システム要件への PostGIS を用いたアプローチ例を、「別紙 GIS アセスメント編」を作成しました。一例としてご確認ください。

2.1.6. 考察

GIS 要件に応えるための PostgreSQL が提供する機能として PostGIS について本節では記述しました。PostGIS はインターネット上に資料も豊富に存在し、OSS で GIS を試すためのソフトウェアとして広く利用されています。また、事例紹介にあるように、商用利用にも耐えうる製品です。PostgreSQL を用いて情報システムを実装することに長けている技術者を擁す、企業や団体の場合は GIS を実現するためのソフトウェアとして、まずは PostGIS を試すことは十分に価値があると考えられます。

16 PostGIS Support Matrix: https://trac.osgeo.org/postgis/wiki/UsersWikiPostgreSQLPostGIS
17 PostGIS (Binary Installers): http://postgis.net/install/
18 PostGIS（インストールに必要なもの）: http://www.finds.jp/docs/pgisman/2.2.0/postgis-ja.html#install_requirements
19 GIS ポータルサイト（国における GIS の取り組み）: http://www.gis.go.jp/contents/government.html

2.2. FDW(外部データラッパ)

FDW(Foreign data wrapper/外部データラッパ)とは、標準 SQL 規格の SQL/MED 仕様を実現するための機能で、DBMS の外部にある様々なデータに対して SQL から管理、アクセスするための仕様です。PostgreSQL では、この中の外部 テーブル(Foreign Table)機能をサポートしており、本機能を利用して様々なデータリソースに対して 外部テーブルとしてアクセスすることを可能とする FDW モジュールが提供されています。

2.2.1. 機能の説明

FDW は、PostgreSQL の外部テーブル機能と FDW モジュールを組み合わせてデータ連携を行います。各機能の関係と役割や対応範囲について説明します。

① データ連携の仕組み

PostgreSQL では、外部に存在するデータリソースを外部テーブルとして登録し、あたかも内部のテーブルのように扱うことができます。この外部テーブルへの操作が行われると、FDW モジュールを介して、外部のデータリソースへの同様な操作が行われます。

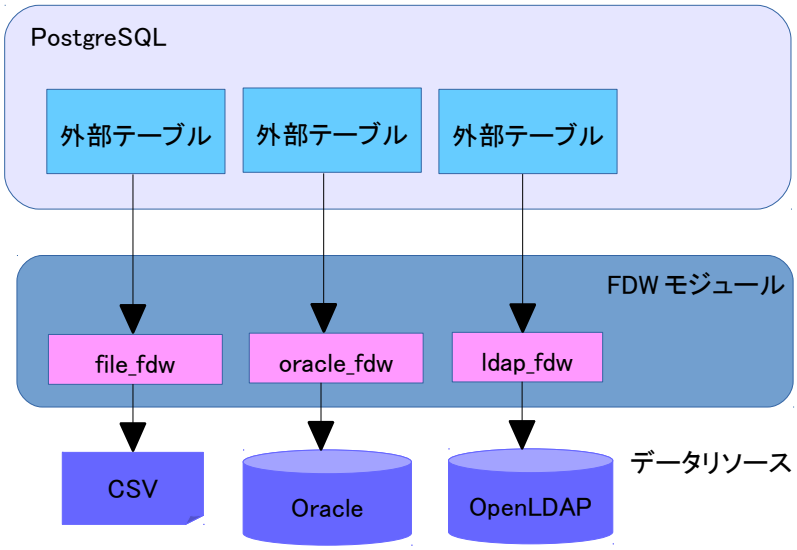


図 10: データ連携の仕組み

② 外部テーブル機能

外部テーブルの操作には、表(テーブル)と同様に参照や更新などの標準的な SQL 文で実行可能です。また、外部テーブと内部テーブルとの結合をすることや外部テーブルと内部テーブルの更新を同一トランザクションとして制御することも可能です。

③ FDW モジュール

FDW モジュールは、データリソースと通信できるライブラリです。外部テーブル機能と連携することで、データリソースへの接続やクエリの実行を行います。

④ 対応データリソース

FDW に対応したデータリソースの種類には、DBMS、NoSQL DB、フラットファイル、Web サービス、他多数がありますが、PostgreSQL 本体に含まれている FDW は、postgre_fdw と file_fdw だけです。それ以外については、様々なプロジェクトや企業等により提供されていますが、ライセンス形態、導入方法、使用方法、データの更新可能の有無などは、それぞれで異なっています。

以下に対応データリソースの一部を紹介します。 詳細は、PostgreSQL wiki[20]を参照してください。

20PostgreSQL wiki: https://wiki.postgresql.org/wiki/Foreign_data_wrappers

表 20: 対応データリソースの一部紹介

データリソース	FDW モジュール	ライセンス形態	更新対応	調査時のバージョン
PostgreSQL	postgre_fdw	PostgreSQL ライセンス配下	○	PostgreSQL 9.5 同梱
CSV	file_fdw	PostgreSQL ライセンス配下	参照のみ	PostgreSQL 9.5 同梱
ORACLE	oracle_fdw	PostgreSQL ライセンス配下	○	1.4.0
MySQL	mysql_fdw	EnterpriseDB Corporation	○	2.1.2
Firebird	firebird_fdw	PostgreSQL ライセンス配下	○	0.1.0
LDAP	ldap_fdw	PostgreSQL ライセンス配下	参照のみ	0.1.1
JSON	json_fdw	GPL3	参照のみ	1.3.0
Git	git_fdw	Franck Verrot	参照のみ	1.0.0
RSS	Multicorn.rss fdw	PostgreSQL ライセンス配下	参照のみ	1.3.2

⑤ 利用用途
FDW 機能を利用することで、様々なデータリソースを PostgreSQL データベース内で統合可能です。そのため、ETL ツール（データ連携）や分散データの 統合管理などのアプリケーションの代替機能として活用が期待できます。

2.2.2. 機能の特徴

FDW は、データリソース間の連携を実現する際に考慮すべき機能要件について、検討・実装されたものが提供されています。機能の実現は、個々の FDW モジュールの実装によって差異があります。

① 異種 DBMS との相互変換
異種 DBMS では元々仕様の相違があり、データ型の種類や範囲の違い、NULL と空文字の扱いの違い、選択列や条件式に指定できないデータ型の違い、SQL 関数の違いなどあります。FDW では PostgreSQL からの SQL を異種 DBMS の仕様に変換して処理を実行します。その処理結果は PostgreSQL 用に変換し処理結果を返却します。

② トランザクション管理
- トランザクションの制御
  ローカル（PostgreSQL）のトランザクションとリモート（外部の DBMS）のトランザクションを作成し、相互に連携することでトランザクションを制御します。ローカルのトランザクションがコミット、あるいはロールバックした時、リモートのトランザクションも同様にコミット、あるいはロールバックします。 セーブポイントも同様に管理され、リモート側に関連付けられたセーブポイントが作成されます。

- トランザクションの範囲
  基本動作としては、ローカルトランザクションでコミットされると、先にリモートのトランザクションがコミットされて正常終了すれば、続いてローカルのトランザクションがコミットされます。 反対に異常終了すれば、リモートとローカルのトランザクションは共にロールバックされます。ただし、2相コミットに対応していないために、ローカルのトランザクションのコミットが異常終了しても、リモートサーバは更新された状態になってしまいます。そのために、データを再検索して内容の確認するなどの工夫が必要です。

- トランザクション分離レベル
  トランザクション分離レベルは、PostgreSQL 側のローカル・トランザクションと FDW 側のリモート・トランザクションで扱いが異なります。FDW の種類による相違だけでなく、同一の FDW でも扱う分離レベルによる相違があります。

  例えば、postgre_fdw では、ローカルトランザクションが SERIALIZABLE 分離レベルの場合には、リモートトランザクションも SERIALIZABLE 分離レベルを使用します。 それ以外の場合には REPEATABLE READ 分離レベルを使用します。また、oracle_fdw では、リモートの Oracle 側では、SERIALIZABLE 分離レベル

が使用されます。

③ クエリの最適化

- 問い合わせ実行の最適化
  外部テーブルに指定された WHERE 句の条件をリモートサーバで評価することで、CPU リソースの分散使用、転送データ量の削減、SELECT 実行時間の高速化を行っています。

- 実行計画の最適化
  リモートテーブルに対する ANALYZE を実行すると、ローカルに統計情報の計算と保存を行います。ローカルな統計情報を保存する事で、クエリの度にリモートテーブルの実行計画を作成するオーバヘッドを削減する事ができます。しかし、リモートテーブルの更新頻度が高い場合には、実行計画作成時にリモートサーバの統計情報を取得することも可能です。

④ アクセス権限
外部テーブルは、ローカルのユーザ権限が適用され、リモートテーブルは、リモートサーバのユーザ権限が適用されます。また、外部テーブル定義のオプションで、更新可否を指定することでも可能です。

2.2.3. 実現方法

FDW を利用するための実現方法として、PostgreSQL オブジェクトと実装のカスタマイズについて説明します。

① オブジェクト構成一覧

データリソースにアクセスするためには PostgreSQL 上で以下のオブジェクトを作成する必要があります。

表 21: FDW 関連オブジェクト一覧

No	オブジェクト名	説明
1	EXTENSION （拡張）	拡張をインストールします。指定した FDW モジュールをデータベース内に読み込みます。外部データの種類ごとに作成が必要です。
2	SERVER （外部サーバ）	外部サーバを定義します。FDW がリモートのデータリソースにアクセスするために使用する接続情報等を指定します。
3	USER MAPPING （ユーザマッピング）	外部サーバのユーザマップを定義します。ローカルユーザとリモートユーザの対応付けを指定します。
4	FOREIGN TABLE （外部テーブル）	外部テーブルを定義します。名前はスキーマ内にあるオブジェクトと異なる必要があります。 また、リモートユーザは、サーバ接続権限とテーブルの使用権限を持たなければなりません。

② オブジェクト関連図

クライアン PC から「外部テーブル」へクエリが発行されると、PostgreSQL は「拡張」を介して「FDW」に処理を依頼します。その際に「FDW」が「データリソース」にアクセスするために必要な接続情報や認証情報を「外部サーバ」、「ユーザマッピング」から取得します。「FDW」は「外部リソース」に接続し処理を実行し、処理結果を返却します。

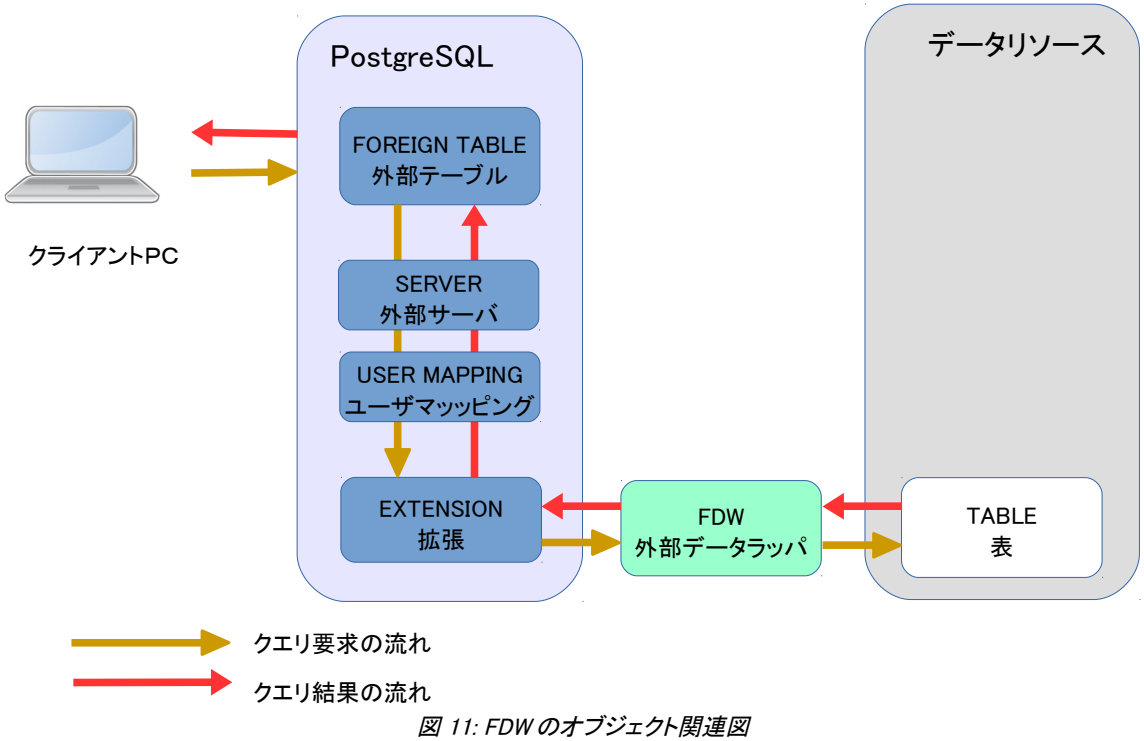


図 11: FDW のオブジェクト関連図

③ 実装のカスタマイズ
既存の FDW モジュールが用途に合わない場合や未対応のデータリソースに対処する場合には、実装のカスタマイズが可能です。FDW のインターフェイス仕様に従い、C 言語でハンドラ関数と検証関数を実装します。

表 22: FDW モジュール実装概要説明一覧

関数	概要
ハンドラ関数	PostgreSQL の問い合わせ経路の各内部処理（プランナやエグゼキュータなど）から呼び出される各ルーチンを作成します。
検証関数	FDW 関連オブジェクト作成時に設定されたオプションの妥当性を検証します。 実装は必須ではなく、関数が指定されない場合は検証しません。

上記のハンドラ関数は、下記の図のように PostgreSQL の問い合わせ経路（SQL受信から結果送信）上の内部処理から呼び出されます。

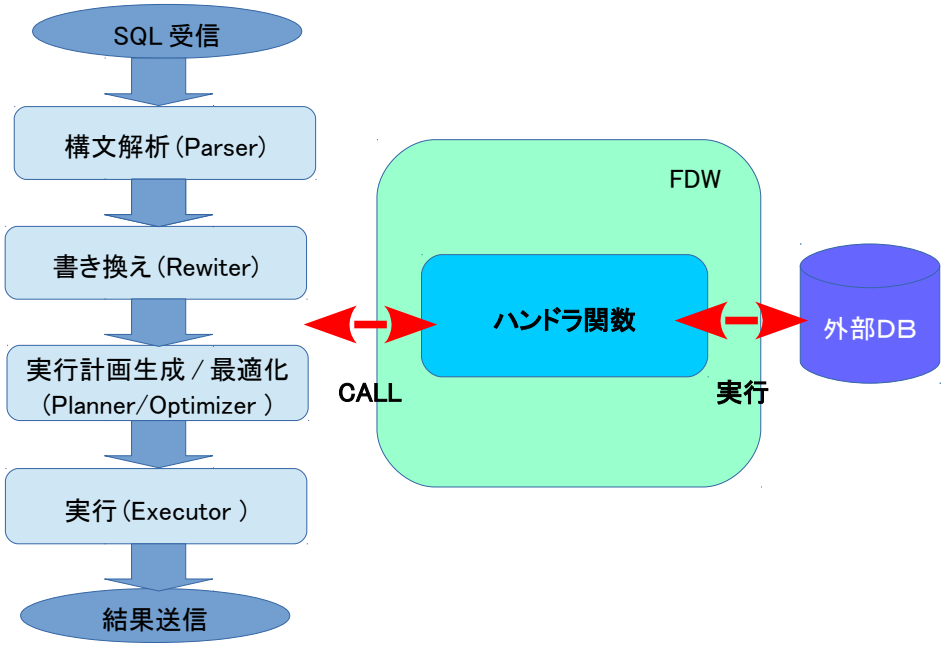


図 12: FDW の問い合わせ処理フローのイメージ図

2.2.4. 制限・前提条件

各データリソースに対応した FDW モジュールを導入するには、使用するデータリソースのクライアント用ライブラリを事前に導入し、C コンパイルが必要です。必要なライブラリは FDW モジュールのドキュメントで確認してください。

2.2.5. 事例紹介

導入事例や WG2 で検証した内容を記載します。

① 公開されている導入事例

表 23: 導入事例一覧

No.	導入事例	会社・団体	事例紹介 URI
1	PostgreSQL を活用した仮想データ統合基盤の実現	日本ユニシス(株)	http://www.unisys.co.jp/tec_info/tr111/11103.pdf
2	DataTanks のご紹介と外部テーブル&トリガを使った連携例	ー	http://qiita.com/qyaman/items/c4baa2c1ba81531b5193

② WG2 の検証

WG2 の活動として、PostgreSQL を使ったユーザ認証情報の統合管理という実現検証結果を紹介しています。詳細は別紙を参照してください。

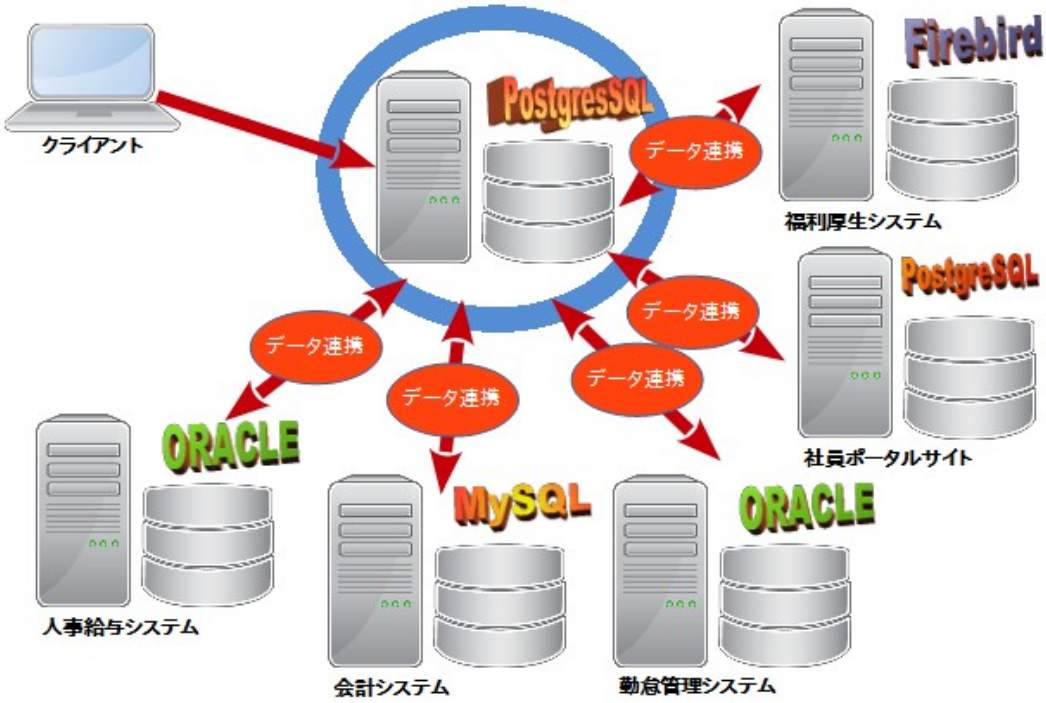


図 13: FDW によるユーザ認証情報の統合管理の検証

2.2.6. 他 DBMS との相違

PostgreSQL では、各種 DBMS との連携方法として、dblink(データベース・リンク)と外部テーブル(FDW)があります。PostgreSQL 以外の DBMS での連携方法の一部を以下に紹介します。

① Oracle Database
標準機能ではデータベース・リンクによる連携です。テーブルの検索と更新が可能です。

② Microsoft SQL Server
リンクサーバ（データベース・リンクと同等）による連携です。テーブルの検索と更新が可能です。

③ IBM DB2
DB2 の外部に連合システム（連合データベース）を置き、それが複数のデータリソースのクライアントとして動作します。SQL/MED に準拠した機能であり、2相コミットにも対応しています。

④ MySQL
MySQL はデータの物理的な保存場所を自身の DB 以外に、メモリや CSV ファイル、別のMySQLなどが選択できる「ストレートエンジン」という機能を持ちます。本機能を利用することで外部データと連携ができます。
また、MySQL からの派生である MariaDB では、異種 DBMS や各種ファイルを扱える CONNECT ストレージエンジンを搭載していて、データの検索や更新が可能です。

3. サポートツール

本節では、PostgreSQL のサポートツールを紹介します。

PostgreSQL 専用のツールや、ツールが対応している DBMS の中に PostgreSQL が含まれるもの、PostgreSQL の機能を拡張するモジュールを「サポートツール」とし、管理ツール、開発ツールに分けて主要なものを挙げました。

3.1. 管理ツール

PostgreSQL が稼動しているサーバに導入するツール、またサーバの監視・運用などをサポートするツールについて下表「管理ツール一覧」に示します。

各ツールの機能の概要については付録「【別紙】管理ツール一覧」にまとめていますので参考にしてください。

表 24: 管理ツール一覧

No.	目的	説明	ツール		
			名称	「説明」欄に挙げた使用目的との合致度 ◎:高、○:中、△:低	GUI 操作の有無
1	DBMS 移行	・データベースオブジェクトの移行	ESF Database Migration Toolkit	◎	○
2	データ投入	・大量データのロード	pg_bulkload	◎	×
3	バックアップ／リストア	・データのバックアップ ・バックアップデータのリストア	Barman for PostgreSQL	◎	×
			OmniPITR	◎	×
			pg_basebackup	◎	×
			pg_rman	◎	×
4	監視	・死活監視 ・CPU、メモリ、ディスク使用量監視 ・性能監視 ・閾値を超えたスロークエリ数の監視 ・データベース監視、SQL 監視	Hinemos	○	○
			OpManager	◎	○
			pg_monz	◎	×
			pgpool-II	○	×
			Slony-I	○	×
5	ログ	・ログ監視 ・ログ取得／解析 ・統計情報、活動状況の収集、蓄積 ・ログ フォーマットの指定	pg_monz	◎	×
			pgFouine	◎	×
			pgaudit	◎	×
			pg_statsinfo	◎	×
6	コネクションプール	・PostgreSQL サーバとのセッション プール、トランザクション プール、ステートメント プールを管理	pgbouncer	◎	×
			pgpool-II	◎	×
7	レプリケーション	・2 台以上の PostgreSQL サーバにリアルタイムでデータを保存	Slony-I	◎	×
			pgpool-II	◎	×
8	負荷分散	・多数のリクエストを分担して負荷を軽減 ・過大な接続数の制限	pgpool-II	◎	×
9	クラスタ	・共有ディスク型クラスタ ・ミラーディスク型クラスタ	Heartbeat	◎	×
			Pacemaker	◎	×
			Postgres-X2	◎	×

3.2. 開発ツール

ソフトウェア開発者がプログラム・アプリケーションを作成、デバッグ、保守する際に必要となるツールについて、下表「開発ツール一覧」に示します。

各ツールの機能の概要については付録「【別紙】開発ツール一覧」にまとめていますので参考にしてください。

表 25: 開発ツール一覧

No.	目的	説明	ツール		
			名称	「説明」欄に挙げた使用目的との合致度 ◎:高、○:中、△:低	GUI 操作の有無
1	SQL 製造/設計支援	・SQL 実行、データ編集 ・ストアドプログラムの開発 ・データのエクスポート／インポート	Aqua Data Studio for PostgreSQL	◎	○
			Common SQL Environment	◎	○
			dbwrench	◎	○
			EMS SQL Management Studio for PostgreSQL	◎	○
			Fluentd+各種 Fluentd 用プラグイン ■PostgreSQL 用プラグイン ・fluent-plugin-dbi： DBI を使って PostgreSQL への SQL 発行 ・fluent-plugin-heroku-postgres： PostgresSQL への SQL 発行 ・fluent-plugin-pgdist： PostgresSQL への SQL 発行 ・fluent-plugin-pghstore： PostgresSQL への SQL 発行 ・fluent-plugin-postgres： PostgreSQL への SQL 発行・Json 形式でのデータ取得	△	○
			Navicat for PostgreSQL	◎	○
			pgAdminⅢ	◎	○
			phpPgAdmin	◎	○
			RazorSQL	◎	○
			SI Object Browser for Postgres	◎	○
			SQL Image Viewer	◎	○
			Aqua Data Studio for PostgreSQL	◎	○
			DbVisualizer	◎	○
2	テーブル設計支援	・データベース オブジェクト生成／管理 ・定義書の Excel 出力 ・権限付与 ・データベース オブジェクトのモデリング	Aqua Data Studio for PostgreSQL	◎	○
			Datanamic DataDiff MultiDB	◎	○
			Datanamic SchemaDiff for PostgreSQL	◎	○
			Datanamic MultiRun for PostgreSQL	◎	○
			dbwrench	◎	○
			DeZign for Databases	◎	○
			EMS SQL Management Studio for PostgreSQL	◎	○

No.	目的	説明	ツール		
			名称	「説明」欄に挙げた使用目的との合致度 ◎:高、○:中、△:低	GUI 操作の有無
2	テーブル設計支援	・データベース オブジェクト生成／管理 ・定義書の Excel 出力 ・権限付与 ・データベース オブジェクトのモデリング	Navicat for PostgreSQL	◎	○
			Toad Data Modeler	○	○
			WWW SQL Designer	○	○
			phpPgAdmin	◎	○
3	テストデータ生成支援	・テスト データの生成	Data Generator for PostgreSQL	◎	○
4	試験支援	・PostgreSQL を用いた負荷テスト	JDBCRunner	◎	×
5	統合環境	・データベース管理、開発向け統合ソリューション	PostgreSQL Tools Family	◎	○

4. コスト面での考慮

DBMS の導入および運用には様々な費用がかかります。主な費用として、以下の費用がかかります。

- 導入費
- 設計、構築
- マイグレーション
- 機能拡張
- 保守サポート
- アップグレード

無償の OSS として公開されている PostgreSQL は、導入自体に費用がかかることはありません。しかしながら、商用製品とは異なる見えないコストがかかる場合がありますので、導入前に検討する必要があります。

表 26: 導入時／導入後のコスト

費用名目	商用 DBMS	PostgreSQL（OSS）	PostgreSQL ベースの商用 DBMS
導入費	導入時にライセンスの購入が求められる。 ライセンスの条件と費用は多様であり、規模で異なるほか VM やクラウド環境など様々な条件で費用が異なる。	導入時のライセンスは不要。	導入時に、サーバの規模や各種条件応じたライセンスもしくはサブスクリプションライセンスの契約費がかかる。
マイグレーション費用	他 DBMS からのマイグレーションを行う場合には、以降コストが必要。移行支援ツールが用意されている場合がある。	他 DBMS を使用する既存システムからのマイグレーションの場合、スキーマやアプリケーション、オプション機能の以降等のコストがかかる。	他 DBMS を使用する既存システムからのマイグレーションの場合、マイグレーションを支援するツールによりコストが抑制される場合がある。
機能拡張	オプション製品を追加する場合には、別途導入費とサポート費を求められる場合がある。	必要に応じて公開されている機能の追加が可能。無償公開されている機能も多く存在する。 PostgreSQL のバージョンアップにともなうアップグレードや機能強化は開発コミュニティの方針に依存する。	オプション製品を追加する場合には、別途導入費とサポート費を求められる場合がある。
保守サポート	DBMS 製品および追加オプションに対するサポートが用意されている。サポート契約の有無により、バグフィックス版の提供やアップグレード時にかかる費用が異なる。必要に応じて24時間サポートや延長サポート契約が用意されている。	各社が様々な条件で保守サポートを提供している。対応時間やパッチ提供、サポート対象コンポーネントによりサポート費用のバリエーションが用意されている場合がある。大規模に利用する場合、自社内で技術者を育成するケースがある。この場合、体制の維持に多大な費用が必要となる場合がある。	製品および追加オプションに対する有償サポートが用意されている。サブスクリプションライセンスの場合はライセンスにサポートが含まれる。サポートの提供時間帯や期間等により費用のバリエーションが用意されている。
アップグレード	保守サポート契約にアップグレードライセンスが含まれている場合、新たな導入費は不要。 非互換項目はあるが、旧バージョン互換機能が用意されている場合がある。アプリケーションの再評価が必要。	メジャーバージョンアップ時にデータ移行またはアップグレードツールによるデータの移行が必要。 一部機能に関しては、過去バージョンの動作をするオプションが用意されている。 アプリケーションの再評価が必要。 同一バージョンを使い続けることもできるが、リリースから 5 年でコミュニティによるアップデート提供が停止する。	メジャーバージョンアップ時にデータ移行またはアップグレードツールによるデータの移行が必要。 一部機能に関しては、過去バージョンの動作をするオプションが用意されている。 アプリケーションの再評価が必要。

5. PostgreSQL の市場価値

5.1. 製品の安定性

PostgreSQL の継続運用を妨げていたトランザクション ID の周回問題[21]も自動 VACUUM により解消に向かい、長期間にわたり連続運用される DB サーバも珍しくはありません。

サーバリソースの使用量や負荷状況、レプリケーションなど特定機能の運用など、運用環境や要件によりさまざまな要素が PostgreSQL の安定運用に関係しますが、本文書では利用環境や要件を問わず安定運用に影響するソフトウェア品質を通じて、PostgreSQL の安定性をご紹介します。

以下のグラフは、コミュニティによるサポートが終了した PostgreSQL8.4 のリビジョンごとのバグフィックス件数を一例としてご紹介しています。

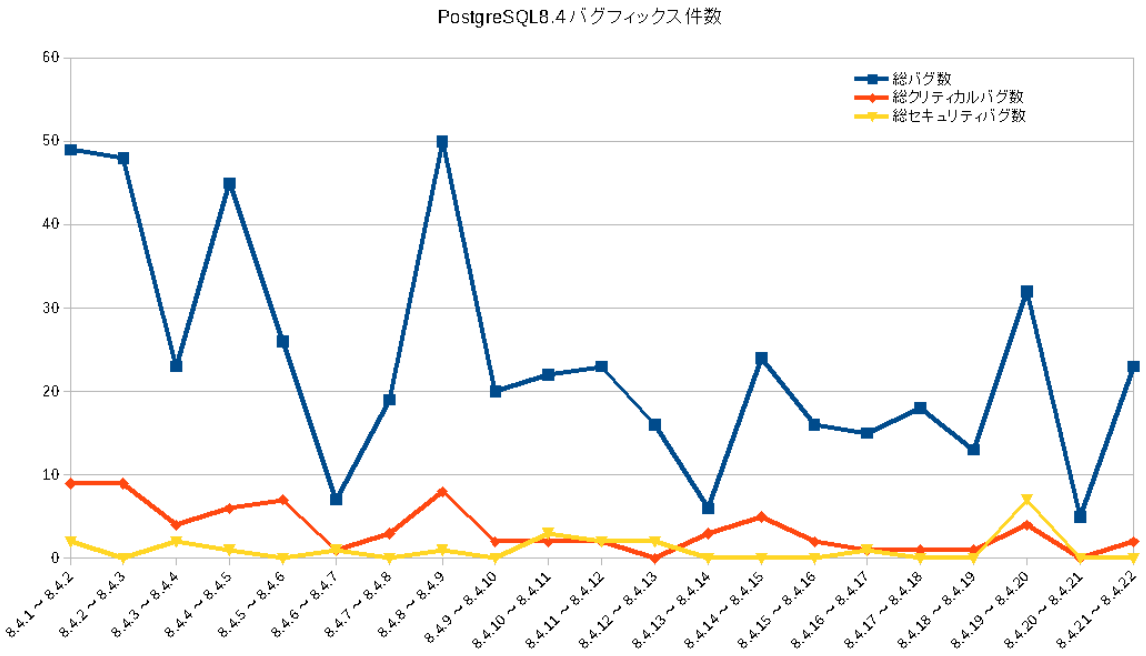


図 14: PostgreSQL8.4 バグフィックス件数

データ破壊や DB プロセスダウンなど、DB の継続運用に影響するバグをクリティカルバグ件数としてカウントしています。リビジョンごとに 10 件未満のクリティカルバグが摘出されていますが、データ破壊をともなうバグに限定した場合、リビジョンごとのクリティカルバグ摘出件数は 0 件～2 件でした。

バグ件数はリビジョンアップごとに減少する傾向にあります。メジャーバージョンアップで各種機能の実装が行われたタイミングでバグが組み込まれ、リビジョンアップごとにソフトウェア品質が改善していると考えられます。

特定の時期に極端にバグ摘出件数が増加しますが、これはメジャーバージョンアップによる機能強化のタイミングでコードが詳細に解析され、また様々な動作検証が行われることにより潜在バグの摘出が増加していると推測されます。

21 PostgreSQL のトランザクション ID の周回
http://www.postgresql.jp/document/9.5/html/routine-vacuuming.html#vacuum-for-wraparound

5.2. 市場ニーズ

PostgreSQL に対する市場ニーズとして、調査会社等の DBMS に関するレポートを紹介します。

5.2.1. OSS DBMS の利用状況

Gartner のレポートによると、2018 年までに社内向けアプリケーションの 70%はオープンソースの DBMS を対象として開発が行われ、商用 DBMS の 50%は OSS　DBMS に移行済みもしくは移行中であると予測されています[22]。

5.2.2. PostgreSQL の人気度

DB-Engines が公開している DBMS の人気度ランキング[23]によると、PostgreSQL の人気度は 2016 年 3 月の時点で 4 位につけています。この傾向は 2013 年ごろから変わらず、高い人気を誇っています。

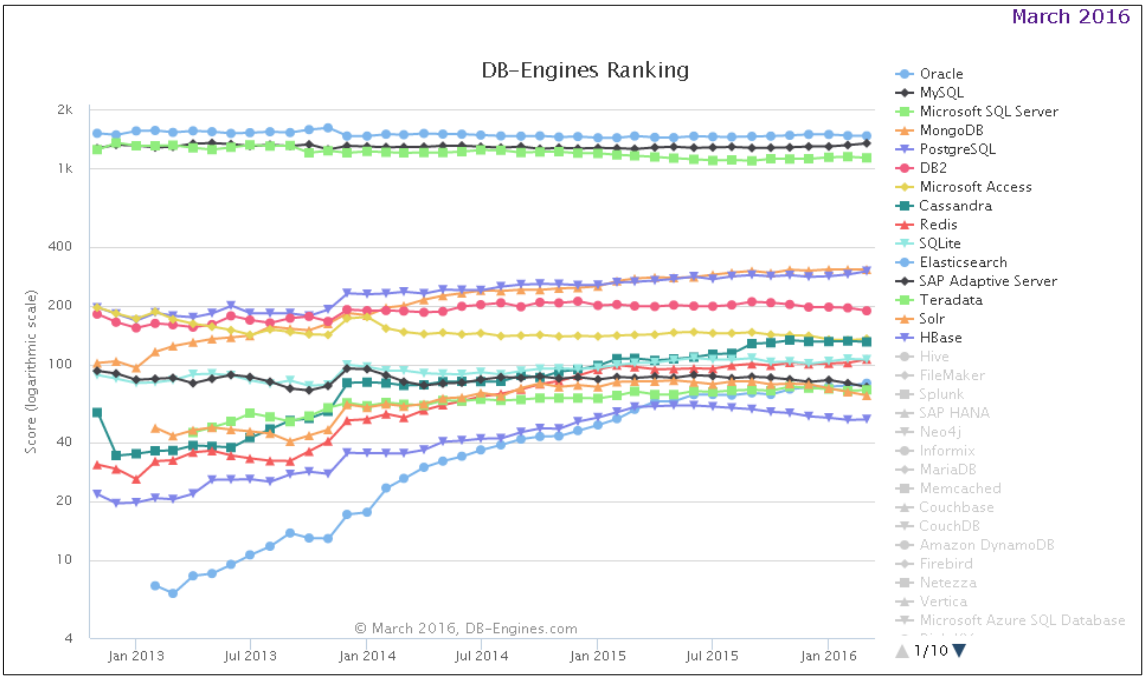


図 15: DBMS の人気ランキング

22 http://info.enterprisedb.com/20150423-gartner-osdbms-report.html?src=2015gosdbms-edb-home-slide

23 http://db-engines.com/en/ranking_trend

5.3. 技術情報の入手

PostgreSQL に関する書籍やユーザコミュニティの ML、セミナ等で情報を入手することができるほか、情報の多くは Web サイトで公開されています。

表 27: 公開されている技術情報

情報の種別	内容・サイト
マニュアル	PostgreSQL の一連の情報マニュアルに詳細にまとめられています。 国内コミュニティのメンバの貢献により、公開された新たなリビジョン・バージョンのマニュアルは早期に日本語マニュアルにも反映されます。 http://www.postgresql.jp/docs
コミュニティ活動	The PostgreSQL Global Development Group のサイトには、開発情報やリリース情報、ワールドワイドのイベント情報などが公開されています。 http://www.postgresql.org/
国内のコミュニティ活動	日本 PostgreSQL ユーザ会の活動状況はコミュニティのサイトに公開されています。 http://www.postgresql.jp/
テーマ別の技術情報	Let's Postgres にはテーマ別に整理された技術情報が公開されています。 http://lets.postgresql.jp/
PostgreSQL 開発者の Wiki	FAQ や開発中の機能などが PostgreSQL Wiki に掲載されています。 https://wiki.postgresql.org/wiki/Main Page
ブログ	PostgreSQL コミュニティメンバによるブログ。 http://planet.postgresql.org/
ビルド情報	各環境でのビルド確認状況は PostgreSQL build farm で公開されています。 http://buildfarm.postgresql.org/
拡張機能	PostgreSQL に組み込む各種エクステンションの情報が PGXN にまとめられています。 http://pgxn.org/ また、多くの拡張機能が SourceForge や各機能のサイトで紹介されています。

5.4. サポート

PostgreSQL に対し、様々な企業がサポートサービス、教育サービスおよび構築・コンサルティングサービスを提供しています。PGECons のサイトにはサービス提供を行っている企業[24]を紹介していますので、是非ご覧ください。

24 サービス一覧 https://www.pgecons.org/postgresql-info/services/

6. 別紙一覧

・別紙 01：アーキテクチャ比較表
・別紙 02：GIS アセスメント
・別紙 03：PostGIS インストール手順書
・別紙 04：pgRouting インストール手順書
・別紙 05：FDW（外部データラッパ）のアセスメント
・別紙 06：管理ツール一覧
・別紙 07：開発ツール一覧

著者

版	所属企業・団体名	部署名	氏名
DB選定基準編 第 1.0 版 （2015 年度 WG2）	日本電子計算株式会社	生産管理部	毛塚 賢一
			大久保 明彦
			中田 小映子
	株式会社富士通ソーシアルサイエンスラボラトリ	公共ビジネス本部第三システム部	小山田 政紀
			香田 紗希
			青木 俊彦
	(株) HTKエンジニアリング	開発本部 システム開発部	斎藤 靖光
			三藤 俊輔
			草山 雄太
			佐藤 和也
	NECソリューションイノベータ	第四PFソフトウェア事業部	岩浅 晃郎