初心者歓迎！
PostgreSQLビルド手順のご紹介

株式会社SRA OSS 三和陽菜

会社紹介

株式会社SRA OSS

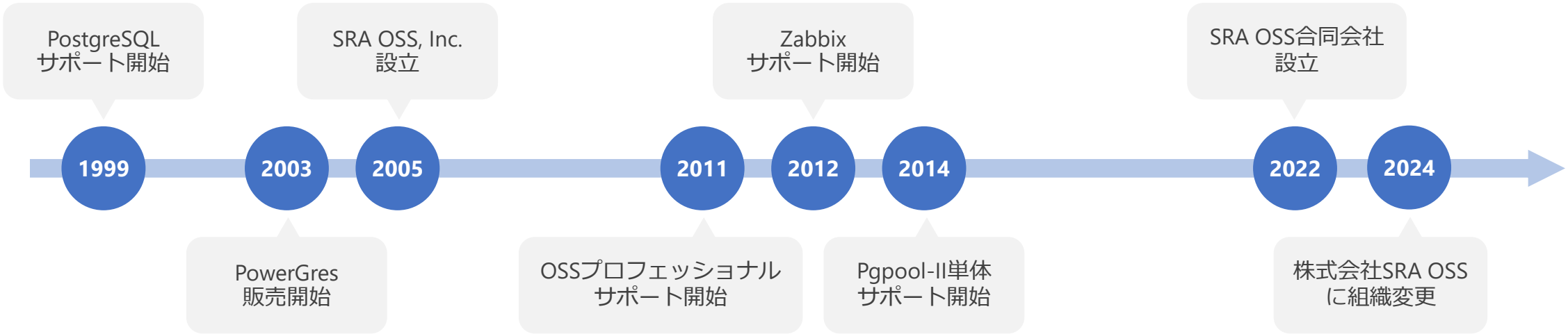
所 在 地: 東京都豊島区南池袋2-32-8
設 立 日: 2022年6月17日
株　　主: 株式会社SRA
　　株式会社NTTデータ
資 本 金: 7,000万円
社　　長: 稲葉 香理

事業内容

- オープンソースソフトウェア (OSS) 関連のサポート、製品開発・販売、構築・コンサル
- OSSの教育、開発、コミュニティ運営支援
- ソフトウェアの研究開発

顧　　問: 石井達夫

技術顧問:増永 良文 (お茶の水女子大学名誉教授)



自己紹介

- 名前: 三和陽菜
- 所属: OSS事業本部 技術部 データベース技術グループ
- 業務内容: PostgreSQLのサポート、構築支援、トレーニング講師

- PostgreSQL歴: 5年(PostgreSQL 13)

はじめに

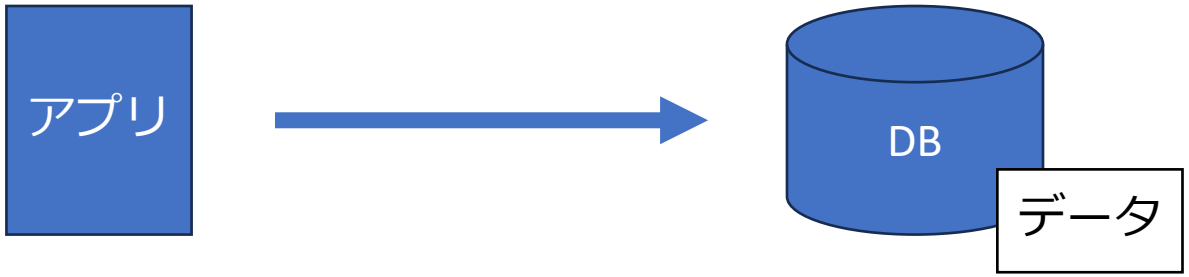
- OSS製品(PostgreSQLに限らず)について調べると、このような記述をみることがあるかと思います
  - **ソースコードが公開されているので、自分でビルドすることも可能です**
- とはいえ……
  - 「可能」といっても難しいのでは？
  - なんかハードル高そう
  - ソースコードからビルドするメリットは？
- この講演では、PostgreSQLを例にとり、ソースコードからビルドする方法を紹介します
  - これで少なくともPostgreSQLについては自分でビルドできるようになります

PostgreSQLとは？

前提: データベース

- データベース
  - 狭い意味では：データの集合
  - 広い意味では：狭義のデータベースを管理するシステム(DBMS)を指すことも

- アプリケーションを作る際、一般的にデータの管理についてはDBMS製品を使用します
  - データの整合性担保や同時実行制御など、複雑な部分をDBMSに任せます
  - アプリケーション固有な処理の開発に専念できるようになります



前提: リレーショナルデータベース

- データを行と列で構成される表に格納するデータベース
- 1970年代にE.F.Coddという人が提唱しました
- 物理的な格納方式をユーザが考慮しなくて済むというメリットがあります
- データの整合性を保つことができます

社員番号	氏名	部署
1001	佐藤 太郎	総務
1002	鈴木 花子	開発
1003	田中 一郎	営業

表形式でデータを管理するデータベース

PostgreSQLとは

- OSSのRDBMS
- 幅広い用途で使え、大規模なシステムにも採用実績多数
- オンプレの他各種クラウドサービスでも使用可能
- 高い拡張性を持つ
  - PostgreSQLの拡張として時系列データベース、ベクトルデータベース
  - 分散データベースがPostgreSQL互換であることも
- 安定したリリースサイクル
  - 1年に1度のメジャーリリース(機能追加)
  - 1年に数回のマイナーリリース(バグ修正)

PostgreSQLのインストール

- インストール方法は大きく分けて2種類
  - ビルド済みパッケージを使用
  - ソースコードから自分でビルドする

- ビルド済みパッケージからインストール
  - OSが提供しているパッケージと、PostgreSQL公式開発コミュニティ(PGDG)が提供しているパッケージがある
  - PGDG提供のパッケージであれば、リリース当日からパッケージを入手できる
  - 手軽にインストールすることが可能
  - 実運用では(特殊な要件がない限り)このビルド済みパッケージを使用

PostgreSQLのインストール

- ソースコードからインストール
  - 細かなオプションを柔軟に指定することができる
    - ブロックサイズ(データを読み書きするときの単位)をデフォルトから変更する、など
  - ビルドした環境に合わせて最適化されるメリットもある
  - 一方で、自分でビルドに必要なパッケージを集めなければいけない・作業手順が複雑であるといったデメリットも
  - PostgreSQL本体の開発を行いたいときなどはソースコードからビルドする

- 実際どんなパッケージが必要なのか、どんな手順で行えばいいかを解説します

ビルドとは？

コンパイラ言語とインタプリタ言語

- ソースコードはただ書くだけでは動かず、コンピュータがわかる命令(機械語)に翻訳してあげる必要があります
- 機械語へどのように翻訳するかで、プログラミング言語は「コンパイラ言語」と「インタプリタ言語」に分けることができます

- **コンパイラ言語:** プログラム実行前にあらかじめまとめて翻訳します
  - C, Java, Go, Rust...
  - PostgreSQLもコンパイラ言語であるC言語で書かれています。
- **インタプリタ言語:** プログラム実行時に1行ずつ翻訳します
  - Python, Ruby, PHP...

コンパイルとビルド

- **「ビルド」と「コンパイル」との違いは？**

- **コンパイル:** ソースコードを機械語に翻訳します
- **ビルド:** コンパイルしたものを組み合わせて、実際に実行可能な形式にまとめる作業です

- ソースコードのファイルが分かれているので、それぞれコンパイルした後に組み合わせてまとめる作業が必要になります。

ビルドに必要なツール

GNUビルドシステム

- 一口にLinux(Unix系システム)といっても、コンパイラのバージョンなど環境の差があります
- ユーザが環境の差を自力でなんとかしようとすると大変です
- そのため、ビルド自動化システムが使用されています
- GNU ビルドシステム(Autotools)が一般に広く使われています
  - プラットフォームの差分を吸収してくれます
  - ユーザは、$ ./configure && make && make install の3つのコマンドを実行するだけでビルド可能です
  - ただし、ビルドを成功させるのに必要なパッケージを揃えておく必要があります

GNUビルドシステム

- ./configureは、ビルド環境の構築を行うためのスクリプトです
  - 必要なパッケージがそろっているかのチェックなどを行います
- makeは、configureで作成されたMakefileに従って実際にビルド作業を行います
- make installは、ビルドした成果物を所定のディレクトリに配置します

makeビルド

環境について

- Windows 11/WSL2/AlmaLinux 9を使用します
  - 環境構築方法については本講演で解説を省略します
- RHEL系のOS(RHEL 9/Rocky Linux 9/AlmaLinux 9)であれば、dockerコンテナや仮想マシンなど、別の環境でも実行可能であるはずです
  - 追加でさらに必要なパッケージがある可能性はあります
- Debian/Ubuntsuを使用する場合、コマンドを読み替える必要があったり(dnf -> apt)、パッケージ名が違ったりするかもしれません
- sudo権限が与えられた一般ユーザで実行しています
  - miwaユーザで実行しているところは自分のユーザに読み替えてください
  - sudo権限を持つ一般ユーザがいない環境の場合、sudoコマンドのところはrootユーザで実行してください

パッケージインストール

- まずは必要なパッケージをインストールします

```
[miwa@localhost ~]$ sudo dnf install -y gcc flex bison perl
libicu-devel readline-devel zlib-devel
```

  - dnf install パッケージをインストールするコマンド
  - デフォルトでインストールするときに必要なパッケージは以下の通り

パッケージ名	なぜ必要か？
gcc	コンパイラ
flex/bison	字句解析・構文解析に必要
perl	ビルドする際に使用
libicu-devel	icuロケールプロバイダに必要
readline-devel	psql(SQL発行ツール)のコマンド履歴機能に必要
zlib-devel	バックアップを取るときの圧縮に必要

  - オプションによっては他にも必要です

ソースコードの入手

- ソースコードをダウンロードし、展開します

```
[miwa@localhost ~]$ wget https://ftp.postgresql.org/pub/source/v18.1/postgresql-
18.1.tar.bz2
[miwa@localhost ~]$ tar xvf postgresql-18.1.tar.bz2
[miwa@localhost ~]$ ls
postgresql-18.1  postgresql-18.1.tar.bz2
```

- 今回はPostgreSQL 18.1をダウンロードしてきました
- https://www.postgresql.org/ftp/source/ にアクセスして、好きなバージョンをダウンロードしてもよいです(その場合は、wgetコマンドの後ろのURLを置き換えてください)

./configureの実行

- ./configureを実行します

```
[miwa@localhost ~]$ cd postgresql-18.1/

[miwa@localhost postgresql-18.1]$ ./configure
```

- 必要なライブラリがそろっているか、そのライブラリはどこにあるかを確認します
- ビルド(make)に必要なファイルを作成します

makeの実行

- makeを実行します

```
[miwa@localhost postgresql-18.1]$ make
```

- makeを実行すると、次々にコンパイルされていき、実行可能ファイルが作成されます
- 完了するまでにある程度時間を要します

make install の実行

- make install を実行します

```
[miwa@localhost postgresql-18.1]$ sudo make install
```

- makeで作成されたファイル群が、/usr/local/pgsql以下に配置されます

```
[miwa@localhost postgresql-18.1]$ ls /usr/local/pgsql/
bin  data  include  lib  share
[miwa@localhost postgresql-18.1]$ls /usr/local/pgsql/bin/
clusterdb   pg_amcheck          pg_controldata        pg_recvlogical  pg_verifybackup
createdb    pg_archivecleanup   pg_createsubscriber   pg_resetwal     pg_waldump
createuser  pg_basebackup       pg_ctl                pg_restore      pg_walsummary
dropdb      pgbench             pg_dump               pg_rewind       postgres
dropuser    pg_checksums        pg_dumpall            pg_test_fsync   psql
ecpg        pg_combinebackup    pg_isready            pg_test_timing  reindexdb
initdb      pg_config           pg_receivewal         pg_upgrade      vacuumdb
```

ユーザ・ディレクトリの作成

- PostgreSQLを扱うための専用のユーザを作成します

```
[miwa@localhost postgresql-18.1]$ sudo useradd postgres
[miwa@localhost postgresql-18.1]$ sudo passwd postgres
[miwa@localhost postgresql-18.1]$ sudo usermod -u 26 postgres
[miwa@localhost postgresql-18.1]$ sudo groupmod -g 26 postgres
```

  - 慣習的に、PostgreSQLの管理用コマンド(起動するコマンドなど)は、Linuxの場合postgresユーザで実行します
  - uid, gidを明示的に変更しています

- データベースクラスタ(PostgreSQLがデータを保管・管理するために使用するディレクトリ)に使用するディレクトリを作成します

```
[miwa@localhost postgresql-18.1]$ sudo mkdir -p /usr/local/pgsql/data
[miwa@localhost postgresql-18.1]$ sudo chown postgres:postgres /usr/local/pgsql/data
```

データベースの起動

- postgresユーザに切り替えます

```
[miwa@localhost postgresql-18.1]$ su - postgres
[postgres@localhost ~]$
```

- データベースクラスタの初期化を行います

```
[postgres@localhost ~]$ /usr/local/pgsql/bin/initdb -D /usr/local/pgsql/data
```

  - initdbという専用のコマンドがあり、それを実行するだけでOKです
  - オプションの-Dは、データベースクラスタを作成するディレクトリを指定します

- PostgreSQLを起動します

```
[postgres@localhost ~]$ /usr/local/pgsql/bin/pg_ctl start -D /usr/local/pgsql/data
server started
```

  - pg_ctl startで起動できます
  - オプションの-Dは、データベースクラスタを指定します

データベースに接続

- PostgreSQLに接続してみましょう

```
[postgres@localhost ~]$ /usr/local/pgsql/bin/psql
psql (18.1)
Type "help" for help.

postgres=#
```

  - psqlは、データベースに接続し、SQLをサーバに送信したり(SQLを発行する、といいます)、その結果を受け取ったりするクライアントアプリケーションです
  - PostgreSQLはクライアント・サーバアーキテクチャを採用しています
  - 今回はサーバもクライアントも同じマシンにある状況です

SQLの実行

- SQLを実行してみましょう

```
[postgres@localhost ~]$ /usr/local/pgsql/bin/psql
psql (18.1)
Type "help" for help.

postgres=# CREATE TABLE employee (id INT, name TEXT, department TEXT);
CREATE TABLE

postgres=# INSERT INTO employee VALUES (1001, '佐藤 太郎', '総務'),(1002, '鈴木 花子', '
開発'),(1003, '田中 一郎', '営業');
INSERT 0 3

postgres=# SELECT * FROM employee ;
  id  |   name    | department
------+-----------+------------
 1001 | 佐藤 太郎 | 総務
 1002 | 鈴木 花子 | 開発
 1003 | 田中 一郎 | 営業
(3 rows)
postgres=# ¥q
```

mesonビルド

mesonビルド

- ここまでのスライドで紹介したGNUビルドシステムは、歴史があり普及しているビルドシステムです
- これからも使用していくにあたり、問題点があります
  - 新しいビルドシステムが登場したため、時代遅れなGNUビルドシステムを扱える開発者が少なくなりつつある
  - 依存関係に関する問題が多い
  - 増分ビルドが遅い

- PostgreSQLプロジェクトでは、他のOSSプロジェクトでの採用状況を考慮して、新しいビルドシステムであるmesonにも対応することにしました
- mesonビルドについても紹介します

クリーンアップ

- 今回はmakeビルドを実行した環境で引き続きmesonビルドを実行します
- まずは環境をクリーンアップしておきます

```
[postgres@localhost ~]$ /usr/local/pgsql/bin/pg_ctl -D /usr/local/pgsql/data stop
[postgres@localhost ~]$ rm -rf /usr/local/pgsql/data/*
[postgres@localhost ~]$ su - miwa
[miwa@localhost ~]$ cd postgresql-18.1
[miwa@localhost postgresql-18.1]$ sudo make uninstall
[miwa@localhost postgresql-18.1]$ make distclean
```

パッケージインストール

- mesonビルドに必要な追加パッケージをインストールしていきます

```
[miwa@localhost ~]$ sudo dnf install dnf-plugins-core python3-pip
[miwa@localhost ~]$ sudo pip install meson
[miwa@localhost ~]$ sudo dnf config-manager --set-enabled crb
[miwa@localhost ~]$ sudo dnf install ninja-build
```

  - mesonはpython製のツールで、pip install によりインストールします
  - mesonビルドはninjaというビルドツールと組み合わせて使用するため、ninjaについてもインストールしています

meson setup の実行

- meson setupを実行します

```
[miwa@localhost ~]$ cd postgresql-18.1/
[miwa@localhost ~]$ meson setup build
```

  - ビルド用設定ファイルがロードされ、ビルドディレクトリが設定される
  - オプションを指定することもできる

ninja の実行

- ninjaコマンドを実行し、ビルドします

```
[miwa@localhost ~]$ cd build/
[miwa@localhost ~]$ ninja
```

  - 完了するまでにある程度時間を要します
  - ninjaは、コンピュータのCPU数を自動で検出し、並列化をしてくれます

ninja install の実行

- ninja installを実行します

```
[miwa@localhost ~]$ sudo ninja install
```

  - ビルドされたものが/usr/local/pgsql以下に配置されます

- インストール後はmakeビルド時と同様に、データベースを初期化し、起動させます

```
[miwa@localhost ~]$ mkdir -p /usr/local/pgsql/data
[miwa@localhost ~]$ chown postgres /usr/local/pgsql/data
[miwa@localhost ~]$ su - postgres
[miwa@localhost ~]$ /usr/local/pgsql/bin/initdb -D /usr/local/pgsql/data
[miwa@localhost ~]$ /usr/local/pgsql/bin/pg_ctl -D /usr/local/pgsql/data start
```

makeとmesonの比較

make vs meson

- make と meson ビルドについて、どちらが速いか比較
- 5回ビルドを繰り返し、かかった時間の平均を比較
  - makeビルドでは、make コマンドの実行時間を計測
  - meson ビルドでは、ninja コマンドの実行時間を計測
- 並列オプションを明示的に指定しない場合では、mesonビルドの方が速い
  - mesonでは自動で並列化されるため
- 並列オプションを明示的に指定した場合では、makeの方が速かった

	make	meson
並列オプション無し	317.2 s	142.8 s
並列オプション有り(-j 12)	79.8 s	112.1 s

ビルド時に指定できるオプション

configureオプション	meson setupオプション	説明
--prefix	--prefix	ファイルのインストール先を、デフォルトの/usr/local/pgsqlから変更
--with-perl --with-python --with-tcl	-Dplperl -Dplpytho -Dpltcl	サーバ側手続き言語(plperl/plpytho/pltcl)を追加 ※使用するにはこのオプション指定の他、CREATE EXTENSIONによる登録が必要
--with-lz4 --with-zstd	-Dlz4={ auto | enabled | disabled } -Dzstd={ auto | enabled | disabled }	lz4圧縮、zstd圧縮を追加
--with-ssl=openssl	-Dssl={ auto | openssl }	SSL接続を有効化
--with-blocksize=*BLOCKSIZE*	-Dblocksize=*BLOCKSIZE*	ブロックサイズ(ストレージとIOの単位)を変更
--enable-debug	--debug	デバッグシンボル付きでコンパイル ※合わせてCFLAGS='-O0'をつけ、ビルド時の最適化を無効にすることを推奨

まとめ

まとめ

- PostgreSQLは、オープンソースのRDBMSです
- ソースコードが公開されているので、ビルド済みパッケージを使う他、自分でビルドすることも可能です
- 自分でビルドすることで、細かなオプション指定ができたり、環境に合わせて最適化させたりできます
- PostgreSQLのビルド方法は大きく分けて2つあります
  - Makeによるビルド
  - Mesonによるビルド

PostgreSQLの操作について

- 今回の講演ではPostgreSQL自体の操作方法については深堀しませんでしたが、詳しく知りたい方はこちらもご参照ください！
- **「データベース入門 ～ PostgreSQLをインストールして、SQLの学習を始めよう！ ～」**

- YouTubeもあります
  - https://youtu.be/SA9-CnQYRb8?si=07JMlD4n9KkBdzt4

- Tech Blogでは、PostgreSQLのほぼ全バージョンのリリースノートの日本語訳を掲載中
  - https://www.sraoss.co.jp/tech-blog/pgsql/

– RECRUIT –

【 PostgreSQLエキスパート】
【 インフラ系OSSミドルウェアエンジニア 】

OSSのように、オープンで情熱的であれ。

全世界のエンジニア・OSSコミュニティの揺るぎない信念とたゆまぬ努力に敬意を。
私たちも情熱を持ち、会社の枠を超えてオープンであり続けます。

Our Value

- ■ OSS貢献
- ■ カスタマーファースト
- ■ マルチジョブ/マルチキャリア

株式会社SRA OSS 人事担当
✉ personnel@sraoss.co.jp

業務内容、待遇、カジュアル面談などお気軽にお問合せください。

SRA OSS