第3章

PostgreSQL

3.1 インストール

PostgreSQL 完全攻略ガイド [1] に付属している CD-ROM や PostgreSQL ユーザ会のホームページ [2] で配布しているバイナリやソースを利用するとよい．ただし，Linux のような PC-UNIX 用のバイナリではインストールディレクトリがデフォルトとは異なっていることがあるので注意されたい．なお，本ドキュメントはソースから Linux へインストールすることを想定しているので，他の OS にインストールする場合は，適宜読み替えていただきたい．

ソースからコンパイルしてインストールする場合はインストール時のオプションとしてエンコーディングを指定する必要がある．具体的には以下の手順でインストールする．

```
# ./configure --enable-multibyte=EUC_JP
# make
# make test /* ここで tests Passed と表示されれば問題なし */
# make install
```

ここで指定した日本語コードは，DandD インスタンスから参照する際のプロトコルの指定，<Protocol> の属性 Encoding に Encoding="EUC_JP" のように反映する必要がある．

なお，技術的なことになるが 7.3 以降の PostgreSQL に付属する JDBC

ドライバでは，データベースのエンコーディング属性を利用することで適切な文字コード変換を行うことができるようになった．

3.1.1 環境変数

/usr/loca/pqsql の下に bin，doc，include，lib，man，share というサブディレクトリが作成されるので， PATH に /usr/local/pgsql/bin を，MANPATH に /usr/local/pgsql/man を追加する必要がある． さらに， 環境変数 PGLIB を新たに定義する必要がある．PGLIB は Postgres 関係のプログラムが必要に応じてロードするライブラリの存在場所を示す環境変数で /usr/local/pgsql/lib を指定する． サーバを立ち上げたりするのでない限り PATH と PGLIB さえ適切に設定されていれば十分である．

PGDATA はデータベースクラスタつまりデータベースが置かれた場所を示す環境変数で，データベースクラスタを作成するときや，サーバを起動するときに設定されていれば，便利である．しかし，引数で与えることもできるので設定されていなくてもよい..

3.1.2 データベースクラスタの作成

PostgreSQL では，ひとつはサーバが立ち上がっていないと, なにもできないが，サーバには必ず対象となるデータベースクラスタが必要なので，まずデータベースクラスタを作成する．

initdb --pgdata=作成するデータベースクラスタのディレクトリ名

--pgdata 以下を省略すれば，環境変数 PGDATA の値がデータベースクラスタのディレクトリ名となる． しかし， --pgdata=/home/postgres/data のように指定すれば，/home/postgres/data が一つのデータベースクラスタとして作成される． このクラスタディレクトリにはサブディレクトリ base，global，pg-xlog のほか， postgresql.conf をはじめとする様々な設定ファイルが作成される． たとえば， この postgresql.conf でサーバの動きを制御する設定ができる（初期状態ではすべてコメントアウト）．デー

タベースクラスタは誰でも任意の場所に作成できるが，それを対象とするサーバはそのデータベースクラスタの所有者しか立ち上げられない．

なお，PostgreSQL 7.3 以降では，initdb に--Encoding=エンコーディング名 をオプションとして渡すことで，データベースのエンコーディングを明示的に指定することが可能になっている．

3.2 PostgreSQL サーバ postmaster

PostgreSQL のサーバは postmaster と呼ばれるプログラムで，これを起動するには

postmaster -D 参照するデータベースクラスタ

が基本で， 必要に応じて -S，-i オプションも付け加えた方がよい． 前者は Silent mode, 後者は Internet Connection を許す指定である．

なお，DandD サーバから PostgreSQL を利用する場合は後者である．その際，/usr/local/pgsql/data ディレクトリの中にある postgresql.conf の一部を次のように書き換えておく必要がある．

```
tcpip_socket = true
```

異なるデータベースクラスタに対してそれぞれ独立したサーバを立ち上げる時にはポート番号を変えることもできる．省略時は 5432 である．

3.2.1 ユーザ登録

クライエントプログラム psql などを介して，サーバにアクセスするためにはユーザ登録が必要であるが，この登録はサーバ，つまりデータベースクラスタ単位である． 登録には createuser, 削除には dropuser を用いるが，これらのプログラムもクライエントプログラムの一つであり，相手のサーバに対して登録，削除を要求する． サーバは対象のデータベースクラスタのサブディレクトリ global にその登録情報を書き込む．データベースクラスタの所有者はすべての権限を持つユーザであるが，新たに登録したユーザにも登録時に同じ権限を与えることもできる． 登録しただけでは，パスワードまで

設定しないのでパスワードを設定する必要がある時は，コマンド pg_passwd を用いる．データクラスタに移ったあと

pg_passwd ファイル名

によって対話型で指定されたファイルにパスワードの設定情報が書き込まれる．ただし，これが実際パスワードファイルとして有効となるためには，pg_hba.conf でこのファイルをパスワードファイルとして設定する必要がある．この pg_hba.conf はセキュリティー管理のためのファイルであり，外部ネットワークからのアクセス制限などを設定できる．

以上のようなユーザ管理をするには，権限が必要であるが標準的なデータベースクラスタは postgres というユーザの所有となっているので，このユーザに su してから careateuser, dropuser, pg_passwd などのコマンドを起動すればよい．あるいは，そのような権限付きで登録されたユーザならユーザ postgres になる必要もない．

3.2.2 データベースの新規作成や削除

データクラスタが作成されたとき， template0 あるいは template1 といった名前のデータベースは作成されているが，これらは雛形で削除はできない．そこに新たにデータベースを作成したり，削除したりもサーバを介して行う．作成するには

createdb 作成したいデータベース名

であり，削除するには

dropdb 削除したいデータベース名

である．

3.3 クライエントプログラム psql

クライエントプログラム psql で現在立ち上がっているサーバを介してデータベースにアクセスできる．

psql データベース名

データベース名を省略したとき，このコマンドを起動したユーザの名前が省略時のデータベース名となり，またそのユーザ名でサーバにアクセスすることになるが，異なるユーザ名でアクセスしたい場合は引数 -U ユーザ名 を加えればよい．ポート番号を明示的に指定したいは， -p ポート番号 を加える．データクラスタにあるデータベース名がわからないときは

```
psql -l
```

で，データベース名のリストが得られる． psql が起動すると対話型で仕事が進められる．そのときの入力は大きく分けて，\（円記号）で始まるバックスラッシュコマンドと，SQL 文からなる． バックスラッシュコマンドの主なものとしては

\q 抜ける

\h ヘルプの表示

\c データベースの切り替え

\c データベース名

\l 存在するデータベースのリスト表示

\d 現在扱っているデータベースに存在するテーブルのリスト表示. 引数としてテーブル名を与えれば，そのカラム名と型などを表示する．**\d** テーブル名

また，SQL 文の主なものとしては

create table テーブルの作成

create table テーブル名 (列名 **text**, ..., 列名 **text**)

DandD Project では数字であっても原則として text 型として保存しておくことを推奨している．これは数値であっても，欠損値 NA が混在する可能性があるからである．

drop table テーブルの削除

drop table テーブル名

copy 外部ファイルから読み込んで代入したり，外部ファイルへ書き出す

copy テーブル名 **from** ’外部ファイルへの絶対パス ’

copy テーブル名 **to** ’外部ファイルへの絶対パス’
ただし，外部ファイルから読み込む場合には，**create table** であらかじめテーブルを定義しておく必要がある．なお，外部ファイルはローカルになければならず，ネットワークを介してマウントされているファイルシステム上のファイルはアクセスできない．特に指定しない限り，区切り記号はタブであり，改行記号で一記録の終わりとみなす．区切り記号を明示的に指定するには
copy テーブル名 **from** ’外部ファイルへの絶対パス’ **delimiters** ’区切り記号’
のように指定する．

各 SQL 文のあとには； をつけること． この入力で初めて入力された SQL 文が実行される．忘れて改行したときはこれだけを入力すればよい．

なお，DandD サーバから PostgreSQL を利用する場合，データベースを参照するための権限を **grant** コマンドで与えておく必要があるので注意されたい．たとえば

grant all on テーブル名 **to public**

は，テーブルに関する一切の権限を，全てのユーザに与えることになる．他の SQL 文については参考書あるいはオンラインマニュアルを参照してほしい．

3.3.1 大文字と小文字の区別

すくなくとも Linux 版の PostgreSQL ではデータベースの名前については大文字，小文字の区別をするが，テーブル名，列名に関しては区別しない．これは SQL が区別しないからのようである．従って，DandD インスタンスに記述されるクエリーに大文字，小文字が混在していても意味を持たない

3.4 データベースのダンプ

データベースを他のマシンの PostgreSQL のもとに移植したりするときや，バックアップを取っておきたいときには， データベース単位のダンプは

```
pg_dump データベース名 > ダンプファイル名
```

であり，これを元へ戻したり，取り込んだりするには

```
createdb データベース名
psql -e データベース名 < ダンプファイル名
```

参考文献

[1] 石井達夫,『PostgreSQL 完全攻略ガイド 改訂第 3 版』, 技術評論社 (2001).

[2] 日本 PostgreSQL ユーザ会, 『日本 PostgreSQL ユーザ会ホームページ』, http://www.postgresql.jp/