# データベースゼミ2回目

指導 谷村吉田，若森，西村

# 第1章　psqlの操作

## 1.1　前回の復習

PostgreSQL はクライアント/サーバ構成です．つまりクライアントのアプリケーションは DB サーバに接続し，ネットワークを通じて問い合わせを発行し，クライアントはその結果を受け取ります．PostgreSQL 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．

PostgreSQL のデータベースをクライアントから用いる場合は psql という SQL インタプリンタを利用します．psql は起動されると SQL 文の入力待ちとなり，入力されるとそれをサーバ側に渡します．postmaster はデーモンプログラムで，psql から接続要求が来ると postgres と呼ばれるプロセスを作成し，その後は psql と postgres の間で問い合わせ処理が行われます．その関係を図 1.1 に示します．

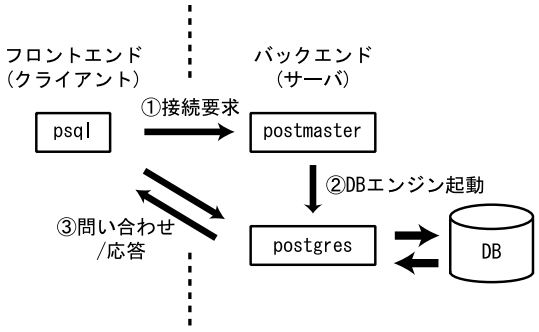


図 1.1: PostgreSQL を構成するプロセス

今回のゼミでは，この psql というインタプリンタを用いてデータベースを利用します．

## 1.2　psqlの基本的な使いかた

### 1.2.1　ユーザ登録＆ユーザ DB 作成

PostgreSQL を使う場合は，あらかじめユーザ登録を行っておく必要があります．ユーザ登録を行う場合には `createuser` コマンドを用います．

```
$ createuser ユーザ名
```

このコマンドを用いてユーザ登録を行えるのは，PostgreSQL のスーパユーザ[1]だけです．このユーザ登録で各ユーザに ID を発行し，DB 作成の権限や，スーパユーザの権限を与えることができます．

[1] UNIX のスーパユーザがシステムの管理を行うようにデータベースの管理者のことです．スーパユーザには，データベースに関する全てのセキュリティチェックが適用されません．

データベースを作成するには createdb コマンドを用います.

$ createdb データベース名

このようにして PostgreSQL では複数のデータベースを持つことができます.

### 1.2.2 psql の操作

createdb コマンドでデータベースを作成後，ユーザは次のようにしてデータベースを利用します.

$ psql データベース名

で，指定したデータベースに対しての psql インタプリンタが起動します．このとき，引数を指定しなかったとき，今まで作成したデータベースの中で一番最初のデータベースが指定されます．またどのようなデータベースがあるかわからないときには

$ psql -l

でデータベースの一覧表を表示できます.
また，日本語を使用する場合は,

$ psql -n データベース名

のように-n オプションを指定する必要があります.
psql を起動すると次のようなプロンプトになります[2].

```
test=>
```

これでtest というデータベースに対する SQL 文を発行できます.

[2] データベースはtest です

# 第2章　SQL文法

## 2.1　テーブルの作成

データベースを作成後，まずはデータを入れるテーブルを作成します．create table コマンドでテーブルを作成できます．

```
create table テーブル名（カラム名　データ型，カラム名　データ型);
```

SQL では"create"などの構文キーワードとテーブル名，カラム名は大文字と小文字が区別されません．例えば，テキスト型の"name"とint 型の"id"というカラムを持つstudent というテーブルを作成する場合，

```
test=> create table student (name text,id int);
```

とします．ここで，テキスト型の"name"とint 型の"id"と言うデータ型について説明します．text とは，可変長文字列のデータ型のことであり，int は整数を扱うデータ型です．その他のデータ型で頻繁に使う型は，Fig.2.1 に示す通りです．

テーブル作成時には，日本語のテーブル名やカラム名もつかえますが，プログラムを書く際に不便だったりするのでお勧めできません．また，SQL 文は複数行に分けて書くことができます．この場合，psql は";"が入力されるまでtest->というプロンプトを出して入力を促進します．

表 2.1: データ型について

| PostgreSQL のデータ型 | SQL92 のデータ型 | コメント |
|---|---|---|
| char | character または char | 文字 |
| char(n) | character(n) または char(n) | 固定長文字列 |
| int4 | smallint | 符号付き整数 |
| date | date | 日付（年月日） |

## 2.2　データの挿入

テーブルにデータを挿入するときは，次のようにinsert 文を用います．先ほどのstudent にデータを挿入する場合は以下のようにします．

```
test=> insert into student values (’よしだ’,0128);
```

とします．SQL では，文字列は’ ’で囲む必要があります．

## 2.3 データの検索

データを検索する場合はselect 文を用います．student というテーブルのデータすべてを検索したい場合は

```
test=> select * from student;
```

とします．また何かの条件を満たすタプル[1]を検索する場合にはwhere を用います．

```
test=> select name from student where id = 0128;
```

上の例ではid が 0128 の人の名前を表示するように問い合わせしています．

## 2.4 更新

データを更新することもできます．更新にはupdate を使います．id が 0128 の人の名前を’よしだ’から’わかもり’に更新するときは次のようにします．

```
test=> update student set name = ’わかもり’ where id = 0128;
```

とします．where 以下を省略すると，すべての名前が’わかもり’になります．

## 2.5 削除

データを削除するには，delete を使います．name が’よしだ’の行を削除するには次のようにします．

```
test=> delete from student where name = ’よしだ’;
```

とします．ここでもwhere 以降を省略すると，すべての行が削除されます．

## 2.6 コピー

PostgreSQL では，テキストファイルからデータベースへデータを取り組むことと，逆にデータベースのデータをテキストファイルに出力することができます．このためのコマンドが\copy コマンドです．

今，次のファイルが\data\input.dat という名前で置いてあるとします．これは，事前に data と言うフォルダ内の\verbinput.dat に次のファイルがある場合を示します．

```
わかもり 0119
にしむら 0070
おの 0075
```

このファイルには 1 行に 1 レコード分，各カラムのデータをそのまま書いておきます．カラムの間はタブ 1 個で区切ります．そして，このファイルからデータベースにデータを取り込むには，psql から以下のコマンドを実行します．

[1] テーブルにおける行です．

```
test=> \copy student from \data\input.dat
```

うまくいけば,

```
Successfully copied
```

と表示されます.
逆にテキストファイルにデータベースからデータを吐き出すのも\copy です

```
test=> \copy student to \data\input.dat
```

とすれば，\data\input.dat というファイルができます．これもうまくいけば,

```
Successfully copied
```

と表示されます.

## 2.7 テーブルごとのアクセス権設定

SQL 文の grant/revoke により，select/insert/update/delete の実行権をユーザ／グループに対して許可／不許可の設定ができます．SQL の標準では，テーブルを作成しただけでは，そのテーブルを作成したユーザ以外には一切アクセスが許されていません．そのため必要に応じて grant 文を使って，他のユーザがアクセスできるようにします.

### 2.7.1 grant

grant は，select/insert/update/delete の実行権を与えます．基本的な文法は次のようになります.

```
grant 権限の種類 on テーブル名 to 対象;
```

『権限の種類』は all,select,insert,update,delete があります．権限は, で区切って複数列挙することができます．all を指定すると，すべての権限が対象となります.
『テーブル名』は許可対象となるテーブルです．これも , で区切って複数のテーブルを列挙できます.
『対象』は許可の対象となるユーザなどです．public とすると，すべてのユーザが対象となります.

### 2.7.2 revoke

revoke はgrant の逆で，権限を取り上げるために使います.

```
revoke 権限の種類 on テーブル名 to 対象;
```

権限の種類，テーブル名，対象の仕方はgrant とまったく同じです.

### 2.7.3 アクセス権限の確認

現在そのテーブルに与えられているアクセス権限はpsql の\z コマンドで確認できます.

```
test=> \z
Database = test
+-----------+-------------------------+
| Relation  | Grant/Revoke Permissions |
+-----------+-------------------------+
| student   |                         |
+-----------+-------------------------+
```

初期状態ではこのように権限リストは空です．ここでユーザkaorun にselect 権限を与えてみます.

```
test=> grant select on student to kaorun
```

これでもう一度権限リストを見ると

```
test=> \z
Database = test
+-----------+-------------------------+
| Relation  | Grant/Revoke Permissions |
+-----------+-------------------------+
| student   | {"=","kaorun=r"}        |
+-----------+-------------------------+
```

この権限リストの見方は"="の左側がユーザで，右側が権限リストです．public の右側は権限リストです．記号の意味は表 2.2 に示します.

表 2.2: 権限リストの記号

| | |
|---|---|
| r | select |
| a | insert |
| w | update/delete |

## 2.8 ユーザ定義関数

PostgreSQL では．ユーザが自由に関数を定義することができます．ユーザ定義関数としてはSQL，C，Tcl を使うことができます．本ゼミではSQL によるユーザ定義関数についてのみ説明します.

ユーザ定義関数は create function 文で定義します．一般的な形を以下に示します.

```
create function 関数名 ([引数 1, 引数 2, 引数 3(8 個まで指定できます)])
returns [setof] 返り値の型名
as 'SQL 文'
language 'SQL';
```

例えばstudent テーブルのレコードにたいし，引数id に該当するタプルのname の値を返すgetname 関数を作ります．

```
test=> create function getname(int)
test-> returns setof text
test-> as 'select name from student where id = $1'
test-> language 'sql';
```

これでgetname 関数が作成できました．作成が成功すると

```
CREATE
```

と表示されます．この関数を利用する場合には

```
test=> select getname(0128);
```

とすることでid が 0128 の人の名前を検索できます．