

障害時回復と同時実行制御を 体験する (PostgreSQL編)

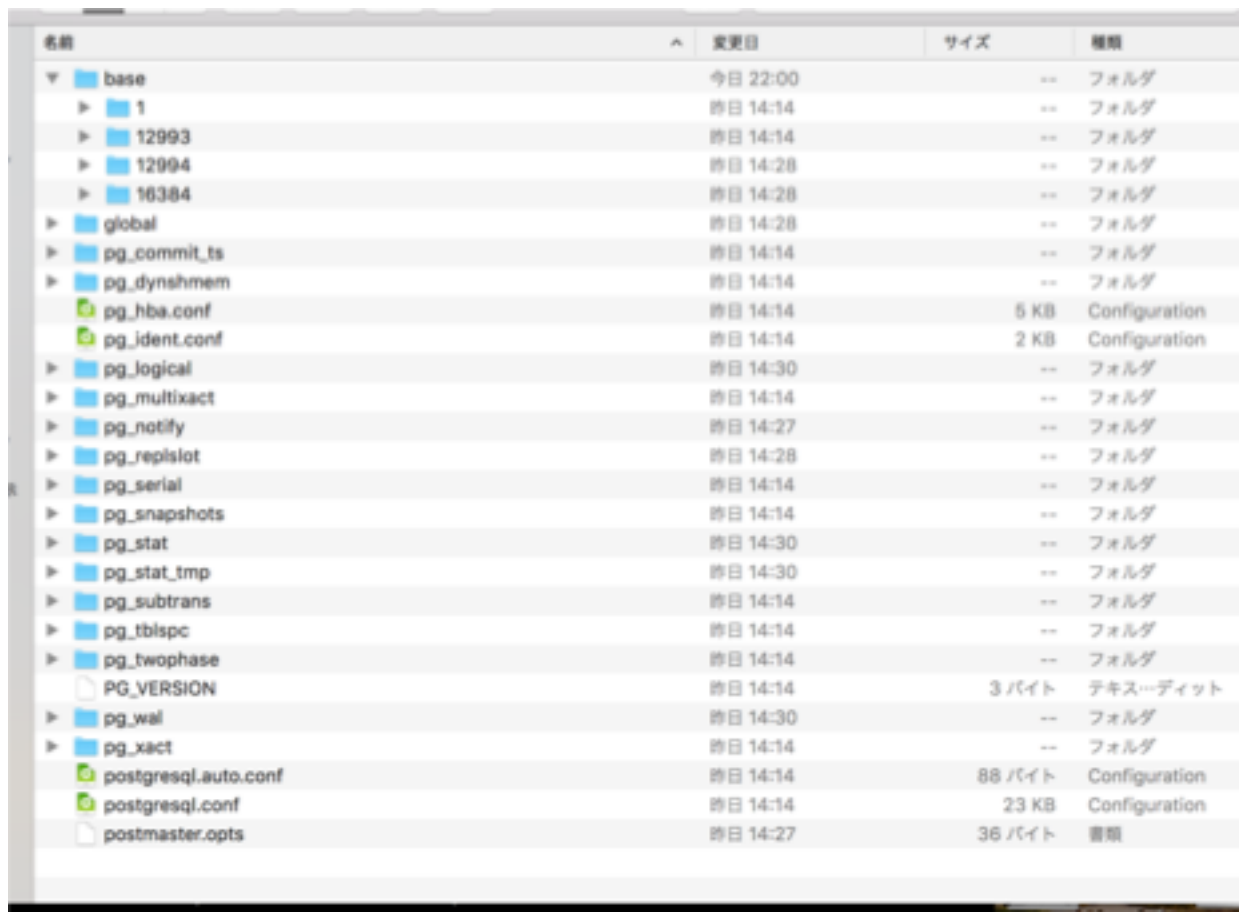
障害時回復の体験

- チェックポイントとWAL(Write Ahead Log)を使って、急にシステムを落とした時の復帰を実現するよ
- customerテーブルのユーザ名を帰る作業の途中でサーバを落としてデータベースファイルを削除
 1. id=1のuserをAliceにする
 2. ログをフラッシュする (チェックポイント)
 3. id=2のuserをBobにする
 4. コミットする
 5. id=3のuserをCharlieにする
 6. コミット前にサーバを落とす
 7. データベースファイルを消す

回復後にはどのような状態に復帰させるべき？

PostgreSQLのデータベースファイル

- PostgreSQLのデータベースは、環境変数PGDATAとして設定されている場所にファイルとして保存されている



The screenshot shows a file explorer window displaying the contents of a PostgreSQL data directory. The files and folders are listed in a table with columns for Name, Last Modified, Size, and Type.

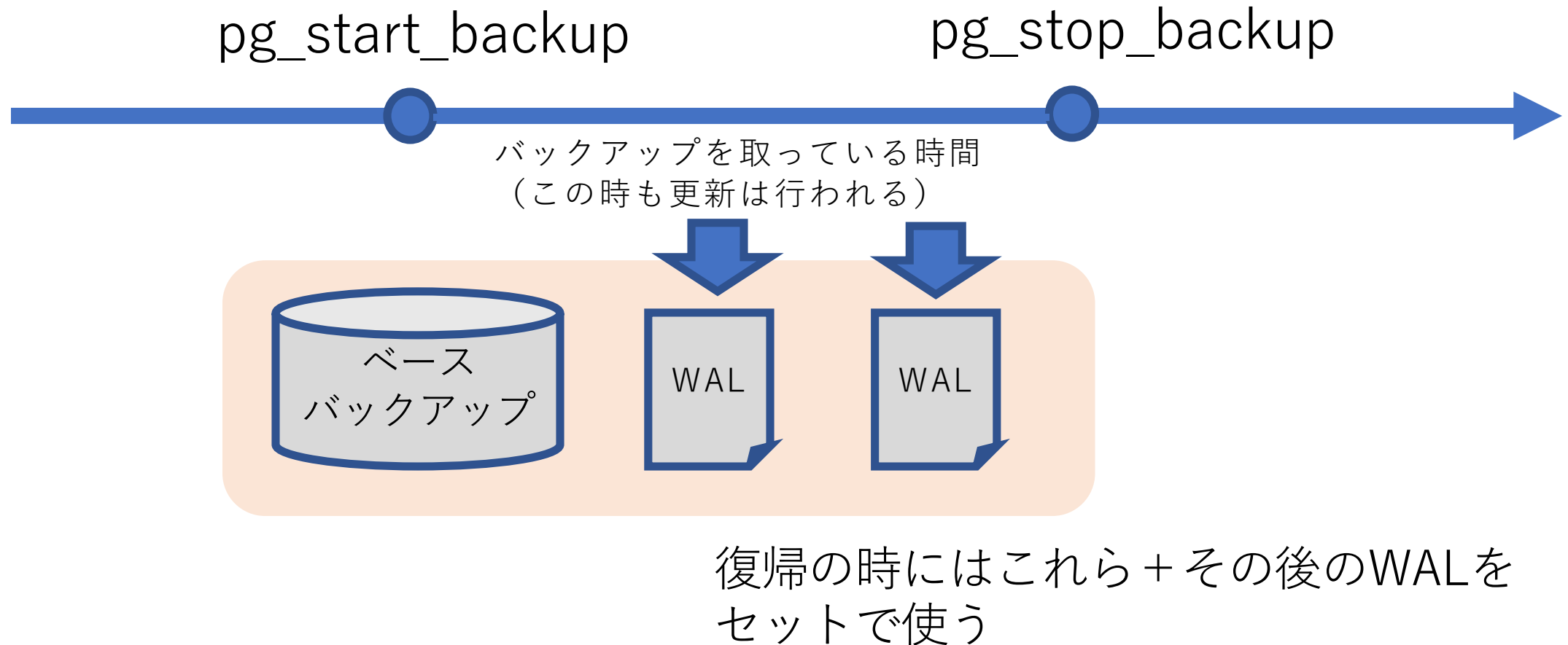
名前	変更日	サイズ	種類
base	今日 22:00	--	フォルダ
1	昨日 14:14	--	フォルダ
12993	昨日 14:14	--	フォルダ
12994	昨日 14:28	--	フォルダ
16384	昨日 14:28	--	フォルダ
global	昨日 14:28	--	フォルダ
pg_commit_ts	昨日 14:14	--	フォルダ
pg_dynshmem	昨日 14:14	--	フォルダ
pg_hba.conf	昨日 14:14	5 KB	Configuration
pg_ident.conf	昨日 14:14	2 KB	Configuration
pg_logical	昨日 14:30	--	フォルダ
pg_multixact	昨日 14:14	--	フォルダ
pg_notify	昨日 14:27	--	フォルダ
pg_replicat	昨日 14:28	--	フォルダ
pg_serial	昨日 14:14	--	フォルダ
pg_snapshots	昨日 14:14	--	フォルダ
pg_stat	昨日 14:30	--	フォルダ
pg_stat_tmp	昨日 14:30	--	フォルダ
pg_subtrans	昨日 14:14	--	フォルダ
pg_tblspc	昨日 14:14	--	フォルダ
pg_twophase	昨日 14:14	--	フォルダ
PG_VERSION	昨日 14:14	3 バイト	テキストディレクトリ
pg_wal	昨日 14:30	--	フォルダ
pg_xact	昨日 14:14	--	フォルダ
postgresql.auto.conf	昨日 14:14	88 バイト	Configuration
postgresql.conf	昨日 14:14	23 KB	Configuration
postmaster.opts	昨日 14:27	36 バイト	書類

base: データページ
pg_wal: WALの
バイナリログ

PostgreSQLのバックアップ・リカバリ

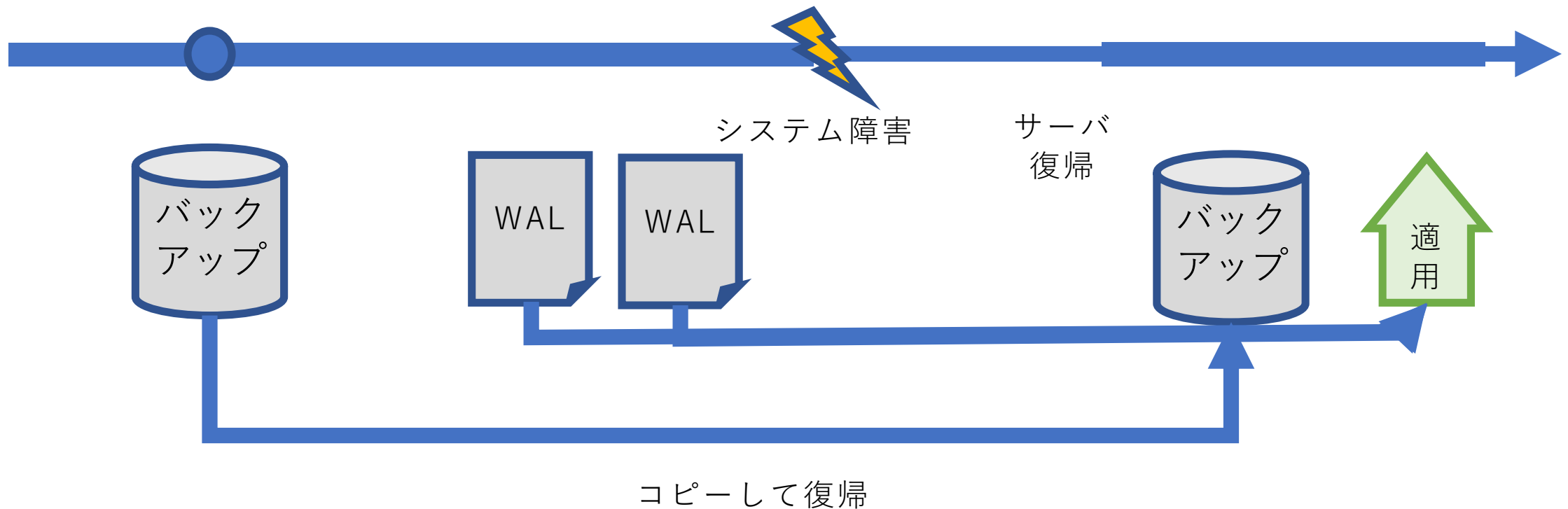
- SQL文によるバックアップ (pg_dump)
 - 「論理バックアップ」とも言う
- PGDATAディレクトリを丸ごとコピー
 - 「物理バックアップ」
 - サーバを一回止めて丸コピー (オフラインバックアップ)
- Point In Time Recovery (PITR)
 - 物理バックアップ+WALを使って、サーバを動かしながらバックアップをとる (オンラインバックアップ)

PITR: Point In Time Recovery



PostgreSQLのリカバリ

- チェックポイントまで物理バックアップファイルで戻して、その後WALをつかってredoのみを行う



トランザクションの実行例：

```
sample=# BEGIN; トランザクション開始
BEGIN
sample=# update customer set name = 'Alice' where id = 1;
UPDATE 1
sample=# select * from customer where id = 1;
 id |          name          | purchase
----+-----+-----
  1 | Alice                  |    53324
(1 row)
```

```
sample=# ROLLBACK; ロールバック
ROLLBACK
sample=# select * from customer where id = 1;
 id |          name          | purchase
----+-----+-----
  1 | 4kifujt5              |    53324
```

変更が反映
されていない

さて：更新途中で落ちました。どうなる？

```
sample=# BEGIN;  
BEGIN  
sample=# update customer set name = 'Alice' where id = 1;  
UPDATE 1  
sample=# COMMIT;  
COMMIT  
sample=# BEGIN;  
BEGIN  
sample=# update customer set name = 'Bob' where id = 2;  
UPDATE 1  
sample=# COMMIT;  
COMMIT  
sample=# BEGIN;  
BEGIN  
sample=# update customer set name = 'Charlie' where id = 3;  
UPDATE 1  
sample=# camabook:tmp chiemi2$
```

Aliceの更新はコミット

この時点で物理バックアップを取りました

Bobの更新もコミット

Charlieの更新をコミットする前にサーバが落ちました

さらにデータベースファイルを破壊します

```
camabook:postgresql chiemi2$ cp -r tmp/db/pg_wal .
camabook:postgresql chiemi2$ ls
docker-compose.yml      pg_wal                  tmp
camabook:postgresql chiemi2$ rm -rf tmp/db/*
camabook:postgresql chiemi2$ docker-compose up
Starting postgresql_db_1 ... done
Attaching to postgresql_db_1
db_1 | The files belonging to this database system will be owned by user "postgres".
db_1 | This user must also own the server process.
db_1 |
db_1 | The database cluster will be initialized with locale "en_US.utf8".
db_1 | The default database encoding has accordingly been set to "UTF8".
db_1 | The default text search configuration will be set to "english".
db_1 |
db_1 | Data page checksums are disabled.
db_1 |
db_1 | initdb: directory "/var/lib/postgresql/data" exists but is not empty
db_1 | It contains a dot-prefixed/invisible file, perhaps due to it being a mount po
int.
db_1 | Using a mount point directly as the data directory is not recommended.
db_1 | Create a subdirectory under the mount point.
postgresql_db_1 exited with code 1
camabook:postgresql chiemi2$
```

WALファイルを退避しておいて...

データベースファイルを消しちゃった

案の定、データがないから起動できないよと怒ってる

さて復帰させます

- 物理バックアップをコピーして戻す

```
camabook:postgresql chiemi2$ cp -r tmp/backup/* tmp/db/.
```

- 起動はできるんですが...

```
root@a240c2156771:/# psql -U postgres sample
psql (10.4 (Debian 10.4-2.pgdg90+1))
Type "help" for help.
```

```
sample=# select * from customer where id < 4;
```

id	name	purchase
3	b8j4v830	83446
2	wkjr8dj3	61914
1	Alice	53324

(3 rows)

あれ？id=2の名前がBobになってない！

WALファイルを使って復帰

- 退避させていたpg_walディレクトリの中身をコピーして戻し、データベースサーバを立ち上げると...

```
db_1 | 2018-07-16 14:50:54.294 UTC [22] LOG:  database system was not properly shut  
down; automatic recovery in progress  
db_1 | 2018-07-16 14:50:54.409 UTC [22] LOG:  redo starts at 0/A000028  
db_1 | 2018-07-16 14:50:54.428 UTC [22] LOG:  invalid record length at 0/B004118: wa  
nted 24, got 0  
db_1 | 2018-07-16 14:50:54.428 UTC [22] LOG:  redo done at 0/B0040E0  
db_1 | 2018-07-16 14:50:54.430 UTC [22] LOG:  last completed transaction was at log  
time 2018-07-16 14:36:03.090179+00  
db_1 | 2018-07-16 14:50:54.693 UTC [1] LOG:  database system is ready to accept conn  
ections
```

redoしてます

redo完了しました

データ復帰できてる？

```
root@a240c2156771:/# psql -U postgres sample
psql (10.4 (Debian 10.4-2.pgdg90+1))
Type "help" for help.
```

```
sample=# select * from customer where id < 4;
```

id	name	purchase
3	b8j4v830	83446
1	Alice	53324
2	Bob	61914

```
(3 rows)
```

```
sample=# █
```

Charlieの更新は復元されずにそのまま。

ちゃんとBobの更新ができています！