

# PostgreSQL の周辺ツール

## pg\_rman でバックアップ・リカバリーを管理する技術を知る

- |                                   |                             |                                |                                 |  |
|-----------------------------------|-----------------------------|--------------------------------|---------------------------------|--|
| <input type="checkbox"/> 導入／環境設定  | <input type="checkbox"/> 移行 | <input type="checkbox"/> 性能    | <input type="checkbox"/> チューニング | <input checked="" type="checkbox"/> バックアップ／リカバリー |
| <input type="checkbox"/> 冗長化／負荷分散 | <input type="checkbox"/> 監視 | <input type="checkbox"/> データ連携 | <input type="checkbox"/> 災害対策   | <input type="checkbox"/> 豆知識                     |

データベース運用において、バックアップ・リカバリーは日常的に行う重要な作業の1つです。「PostgreSQL インサイド」の“技術を知る：PostgreSQL のバックアップとリカバリー”で説明していますが、お客様の重要なデータを扱うシステムでは最新状態まで迅速に復旧できることが求められ、そのようなシステムには物理バックアップのオンラインバックアップ・リカバリーが利用されます。

ここでは、PostgreSQL の周辺ツールの1つである、pg\_rman を使った PostgreSQL のバックアップ・リカバリーについて解説します。

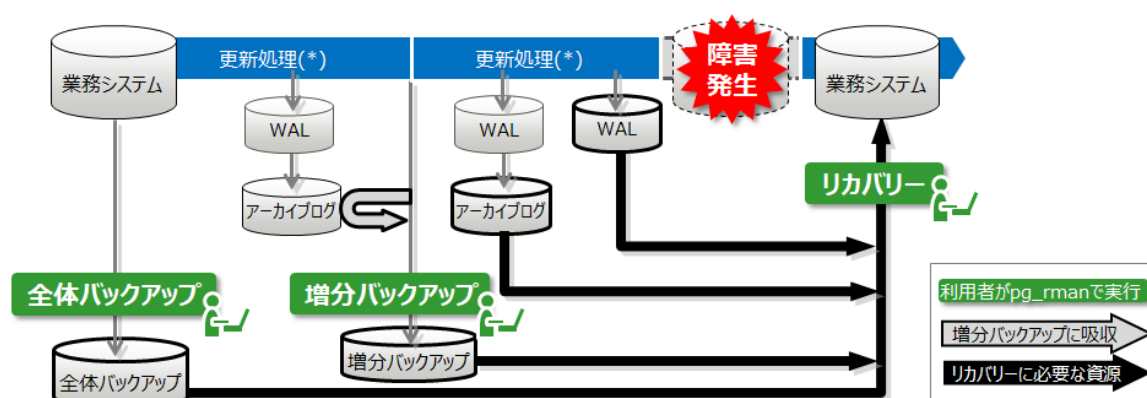
### 1. pg\_rman とは

pg\_rman は、PostgreSQL のバックアップとリカバリーを管理するツールです。物理バックアップであるオンラインバックアップの1つで、Linux 上で動作します。pg\_rman は、PostgreSQL のポイントインタイムリカバリー（以降、PITR と略します）に対応しています。PITR は、バックアップした時点から障害発生直前までの任意の時点で復旧できる機能ですが、復旧時には非常に煩雑な手順を必要とします。pg\_rman を使用することにより、PITR の煩雑な手順を簡略化し、より少ない手順で実施できます。

pg\_rman には以下の機能があります。

- 1 コマンドによるオンラインバックアップ・リカバリー
- 増分バックアップやバックアップの圧縮
- バックアップ世代ごとの管理と一覧表示
- ストレージ・スナップショットによるバックアップ取得時間の短縮

pg\_rman によるオンラインバックアップ・リカバリーの流れを以下に示します。



\*) 更新処理はトランザクションログ(以降、WALと略します)として保存されます。  
WALは必要に応じてアーカイブされ、アーカイブログとして保存されます。

## 参照

- pg\_rman の仕様については、オープンソース・ソフトウェアのウェブページを参照してください。
- 物理バックアップの詳細については、「PostgreSQL インサイド」の“技術を知る：物理バックアップ”を参照してください。

## 2. pg\_rman の使い方

では、pg\_rman を使ってバックアップ・リカバリーを体験してみましょう。PostgreSQL データベースで構築したデータベースクラスタ：人事情報システムを想定しました。

PostgreSQL のバージョン		PostgreSQL 10.3
pg_rman のバージョン		1.3.6
データベース名		mydb
テーブル名		人事テーブル 1 (jtbl1)
データ格納先（ディスク 1）	データベースクラスタのパス	/data
	WAL のパス	/tran
バックアップデータ格納先（ディスク 2）	アーカイブログのパス	/arch
	バックアップカタログのパス	/back

なお、pg\_rman のインストールおよびバックアップカタログの初期化は完了しているものとします。

### 2.1 バックアップ

pg\_rman では、全体バックアップおよび増分バックアップを取得できます。人事情報システムで、毎日 1 時に全体バックアップを行い、6 時間おきに増分バックアップを行う例を説明します。バックアップ開始前の、人事テーブル 1 (jtbl1) のデータは以下とします。

```
$ psql -c "SELECT * FROM jtbl1;" mydb
 id | name | code
-----+-----+-----
 0001 | Kato  |   21
 0002 | Goto  |   22
 0003 | Sato  |   31
 0004 | Takao |   50
 0005 | Yamao |   21
```

- 人事情報システム全体のバックアップを実行します。検証して結果を確認します。

```
$ pg_rman backup --backup-mode=full --with-serverlog --progress -d mydb      #全体バックアップ
$ pg_rman validate                                                            #検証
$ pg_rman show                                                                #結果表示
```

```
=====
StartTime      EndTime      Mode   Size  TLI  Status
=====
2019-01-22 01:00:01 2019-01-22 01:00:06 FULL 4036kB 2 OK
```

2. 定時バックアップとして増分バックアップを実行します。検証して結果を確認します。

```
$ pg_rman backup --backup-mode=incremental --with-serverlog --progress -d mydb #増分バックアップ
$ pg_rman validate #検証
$ pg_rman show #結果表示
```

```
=====
StartTime      EndTime      Mode   Size  TLI  Status
=====
2019-01-22 07:00:01 2019-01-22 07:00:03 INCR 55kB 2 OK
2019-01-22 01:00:01 2019-01-22 01:00:06 FULL 4036kB 2 OK
```

3. 組織変更に伴って人事テーブル1(jtbl1)における部署コード(code)の値を変更します。

```
$ psql -c "UPDATE jtbl1 SET code = code + 1000;" mydb
$ psql -c "SELECT * FROM jtbl1;" mydb
 id | name | code
-----+-----+-----
0001 | Kato | 1021
0002 | Goto | 1022
0003 | Sato | 1031
0004 | Takao | 1050
0005 | Yamao | 1021
```

4. 順次、定時バックアップとして増分バックアップを実行し、検証して結果を確認します。

```
$ pg_rman backup --backup-mode=incremental --with-serverlog --progress -d mydb #増分バックアップ
$ pg_rman validate #検証
$ pg_rman show #結果表示
```

```
=====
StartTime      EndTime      Mode   Size  TLI  Status
=====
2019-01-22 13:00:01 2019-01-22 13:00:03 INCR 72kB 2 OK
2019-01-22 07:00:01 2019-01-22 07:00:03 INCR 55kB 2 OK
2019-01-22 01:00:01 2019-01-22 01:00:06 FULL 4036kB 2 OK
```

## ポイント

pg\_rman show コマンドの Status に 'DONE' が表示された場合は、バックアップのあとに検証が実行されていないことを意味します。

\$ pg_rman show						
StartTime	EndTime	Mode	Size	TLI	Status	
2019-01-22 07:00:01	2019-01-22 07:00:03	INCR	55kB	2	DONE	バックアップ完了したが、検証未は'DONE'
2019-01-22 01:00:01	2019-01-22 01:00:06	FULL	4036kB	2	OK	
\$ pg_rman validate						
\$ pg_rman show						
StartTime	EndTime	Mode	Size	TLI	Status	
2019-01-22 07:00:01	2019-01-22 07:00:03	INCR	55kB	2	OK	pg_rman validateを実行すると'OK'に変更
2019-01-22 01:00:01	2019-01-22 01:00:06	FULL	4036kB	2	OK	

バックアップデータとして利用するためには、Status を'OK'（検証済）にする必要があります。そのため、pg\_rman のバックアップでは、以下をセットで実行することをお勧めします。

1. pg\_rman backup
2. pg\_rman validate

## 2.2 最新状態にリカバリー

システム運用中にデータベースに異常が発生した場合、障害が発生する直前の状態に戻す必要があります。pg\_rman では、全体バックアップ、増分バックアップ、WAL およびアーカイブログを利用して簡単にリカバリーできます。人事情報システムでデータ更新後にデータベース（/data ディレクトリー）に異常が発生したため、システムを最新状態にリカバリーする例を説明します。

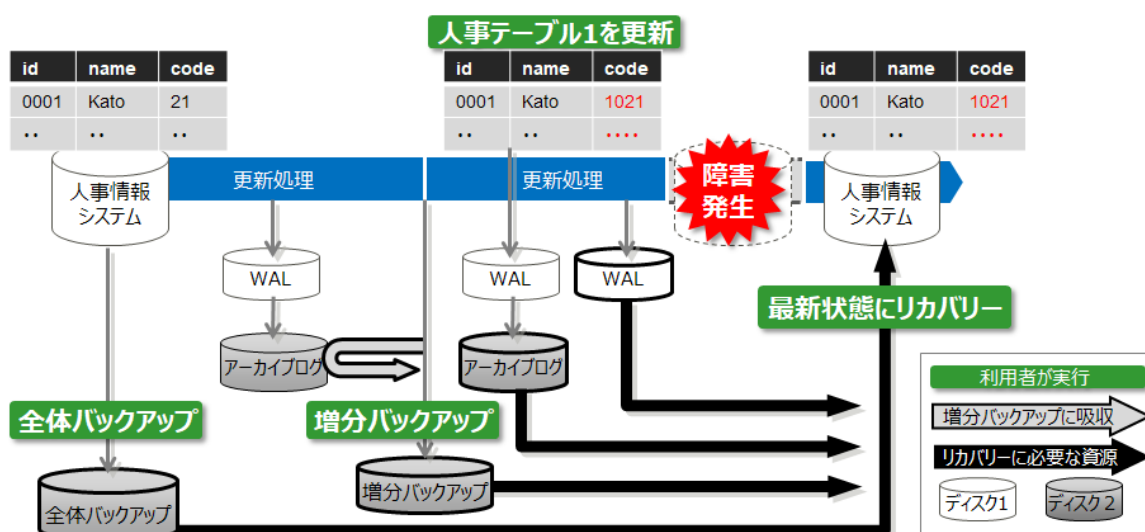


図 1：pg\_rman によるオンラインバックアップと最新状態へのリカバリー

1. データベース異常が発生！人事情報システムを緊急停止して、データベースクラスタを退避します。

```
$ pg_ctl stop -m immediate
$ cp -rf /data /data_crash
```

2. バックアップデータをリストアします（recovery.conf が生成されます）。

```
$ pg_rman restore
```

3. 人事情報システムを起動すると、最新状態にリカバリーされます。

```
$ pg_ctl start
```

4. 最新状態にリカバリーできたかを確認します。

```
$ psql -c "SELECT * FROM jtbl1;" mydb
 id | name | code
-----+-----+-----
 0001 | Kato  | 1021
 0002 | Goto  | 1022
 0003 | Sato  | 1031
 0004 | Takao | 1050
 0005 | Yamao | 1021
```

## 2.3 指定した日時にリカバリー

ディスク内のデータが論理的に破壊され、データベースが正常に動作しない場合は、バックアップを取得した正常な状態にデータベースを戻す必要があります。pg\_rman では、復旧する日時を指定して簡単にリカバリーできます。人事情報システムでデータ更新中にデータベース（/data ディレクトリー）に異常が発生したため、システムをデータ更新前の状態にリカバリーする例を説明します。

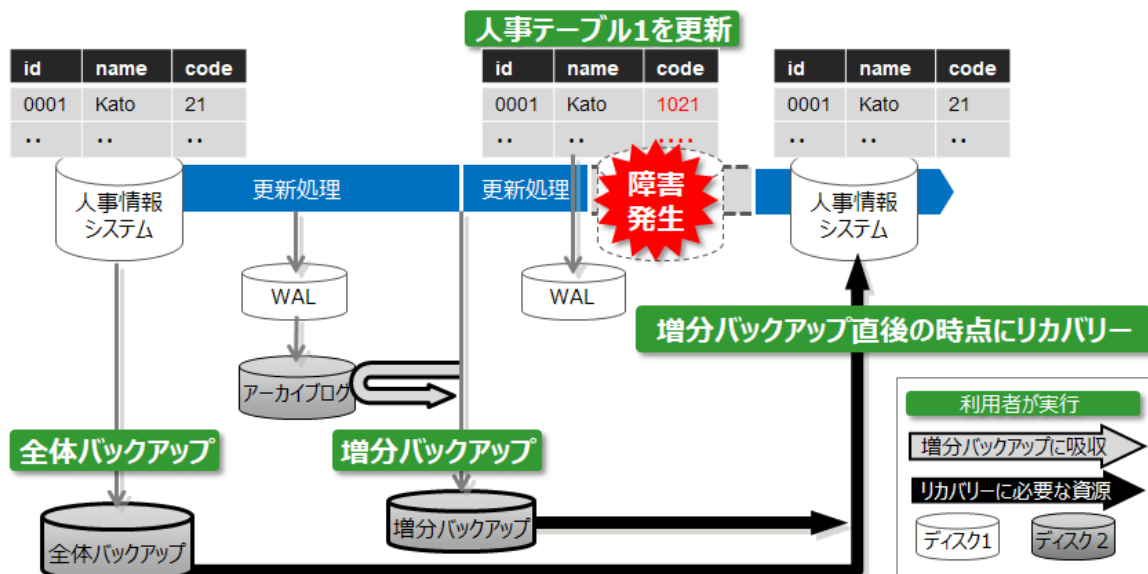


図 2：pg\_rman によるオンラインバックアップと指定日時へのリカバリー

1. データベース異常が発生！人事情報システムを緊急停止して、データベースクラスタを退避します。

```
$ pg_ctl stop -m immediate
$ cp -rf /data /data_crash
```

2. 増分バックアップの完了時点に戻したいため、バックアップデータの一覧を表示してバックアップ終了日時と Status を確認します。

```
$ pg_rman show
```

StartTime	EndTime	Mode	Size	TLI	Status
=====					
2019-01-22 07:00:01	2019-01-22 07:00:03	INCR	55kB	2	OK
2019-01-22 01:00:01	2019-01-22 01:00:06	FULL	4036kB	2	OK

3. 増分バックアップの完了日時を指定してバックアップデータをリストアします（recovery.conf が生成されます）。

```
$ pg_rman restore --recovery-target-time '2019-01-22 07:00:03'
```

4. 人事情報システムを起動すると、指定した日時にリカバリーされます。

```
$ pg_ctl start
```

5. 増分バックアップの直後の状態にリカバリーできたかを確認します。

```
$ psql -c "SELECT * FROM jtbl1;" mydb
 id | name | code
-----+-----+-----
 0001 | Kato  |   21
 0002 | Goto  |   22
 0003 | Sato  |   31
 0004 | Takao |   50
 0005 | Yamao |   21
```

### 3. pg\_rman のメリット

#### 3.1 PostgreSQL と pg\_rman の比較

PostgreSQL 標準機能によるオンラインバックアップ・リカバリーと、pg\_rman によるオンラインバックアップ・リカバリーの手順を比較してみましょう。なお、リカバリーはデータベース異常時に最新状態にリカバリーする場合を想定しています。

工程	PostgreSQL 標準機能	pg_rman
バックアップ	<ol style="list-style-type: none"> <li>pg_basebackup コマンドでベースバックアップ</li> <li>ベースバックアップデータが生成されたかを確認</li> <li>WAL が生成されたかを確認</li> </ol>	<ol style="list-style-type: none"> <li>pg_rman コマンドで全体（増分）バックアップと検証</li> <li>pg_rman コマンドでバックアップ状態を確認</li> </ol>
リカバリー	<ol style="list-style-type: none"> <li>ベースバックアップデータをリストア</li> <li>リストアした古い WAL を削除</li> <li>未アーカイブの WAL を pg_wal にコピー</li> <li>復旧コマンドファイル（recovery.conf）を作成</li> <li>データベースサーバーを起動</li> <li>最新状態かを確認</li> </ol>	<ol style="list-style-type: none"> <li>pg_rman コマンドでリストア（recovery.conf を自動生成）</li> <li>データベースサーバーを起動</li> <li>最新状態かを確認</li> </ol>

pg\_rman は、PostgreSQL 標準機能と比べて多くのメリットがあります。1 コマンドでバックアップとリカバリーの操作が実行できるため、複数のコマンドを使分ける必要が無く、操作ミスを防ぎます。特にリカバリー（PITR）は PostgreSQL 標準機能の手順のような古い WAL の削除や復旧コマンドファイルの作成など事前準備が不要なため、格段に手軽です。また、増分バックアップ機能およびバックアップの圧縮機能があるため、定時バックアップの時間とサイズが少なく済みます。全体バックアップと増分バックアップで取得したバックアップデータは世代管理されて一覧表示して確認できるため、バックアップデータの取得状況や障害発生時に戻りたいバックアップ時点をすぐに把握できます。

## 参考

メリットの多い周辺ツールの pg\_rman ですが、バグ修正の必要性が生じてコミュニティの開発ペースに依存します。そこで、富士通が提供する FUJITSU Software Enterprise Postgres（以降、Enterprise Postgres と略します）に同梱している pg\_rman をご紹介します。Enterprise Postgres では、pg\_rman をプリインストールしていますので、現在 PostgreSQL データベースで pg\_rman を使って業務運用している場合、Enterprise Postgres に移行したあとも、そのまま継続利用できます。富士通の 24 時間 365 日保守サポートにより、PostgreSQL 本体および pg\_rman を含む周辺ツールのご質問、トラブル対応およびバグ修正にも迅速に対応します。また、Enterprise Postgres では、PostgreSQL のバックアップ・リカバリーを独自強化しており、pg\_rman とはまた違ったメリットがあります。PostgreSQL の継続的アーカイブによるバックアップを 1 コマンドまたは GUI ツールで実施でき、リカバリー（PITR）も 1 コマンドまたは GUI ツールで実施できます。さらに、WAL を配置したディスクの障害に備えて WAL を二重化できるため、ディスク障害が発生した場合も常に最新状態に復旧できます。例えば pg\_rman の図 1 で説明すると、ディスク 1 自体が故障した場合、リカバリーに必要な資源の WAL も消失するので最新状態に復旧できません。Enterprise Postgres では WAL をバックアップデータ格納先のディスク（ディスク 2）に二重化しているため、ディスク 2 の WAL を使って最新状態に復旧できる仕組みです。日常的なデータ保全のバックアップツールとしては、増分バックアップの取得やバックアップデータの一覧表示ができる pg\_rman が便利です。ディスク障害などシステム全体の保全に対応するバックアップツールとしては、Enterprise Postgres をお勧めします。

pg\_rman と Enterprise Postgres のオンラインバックアップ・リカバリーの違いを以下にまとめます。なお、表中記号の意味は以下のとおりです。

レ：機能有り、N/A：機能無し

項目		pg_rman	Enterprise Postgres	
			コマンド機能	GUI ツール
全体バックアップ		レ	レ	レ
増分バックアップ		レ	N/A	N/A
バックアップデータの圧縮		レ	N/A	N/A
バックアップデータの世代管理		レ	N/A	N/A
バックアップ領域の自動生成		N/A	N/A	レ
ストレージ製品との連携による高速バックアップ		レ	レ	N/A
ディスク障害に備えた WAL の二重化		N/A	レ	レ
リカバリー の復旧 時点	最新時点	レ	レ（注 1）	レ（注 1）
	指定した任意の時点	レ	レ	N/A
	バックアップ時点	レ	レ	レ
異常箇所の自動特定		N/A	N/A	レ
動作環境	Linux	レ	レ	レ
	Windows	N/A	レ	レ

- 注 1 データだけではなく、WAL のディスク障害が発生した場合でも最新に復旧できます。

Enterprise Postgres では、バージョン 10 から、pg\_rman を同梱しています。

Enterprise Postgres のバックアップ・リカバリーの詳細については、Enterprise Postgres の製品マニュアル“FUJITSU Software Enterprise Postgres 運用ガイド”を参照してください。なお、同一のインスタンスに対するバックアップ処理として、pg\_rman と WAL の二重化は併用できません。

バックアップ・リカバリーの手段として、pg\_rman および Enterprise Postgres の機能・用途を理解して、ご利用のシステムに適した方法を選択してください。

2021 年 1 月 22 日