

Amazon RDS+EC2+PGroonga+ ロジカルレプリケーション を使った低コスト 高速全文検索

堀本泰弘

クリアコード

PostgreSQL Conference Japan 2019
2019-11-15





自己紹介



堀本 泰弘

全文検索エンジン

gr^onga pgr^onga

mr^onga rr^onga

の開発・サポート



自己紹介

- 他のOSSへもコントリビュート
 - 昨日、PostgreSQLのドキュメントの改善を提案
 - 問題は「**開発元**」で修正が弊社のポリシー



今日のテーマ

なるべく**楽**に
高機能で高速な
全文検索
がしたい



実現のための要素





特に大事な要素





楽



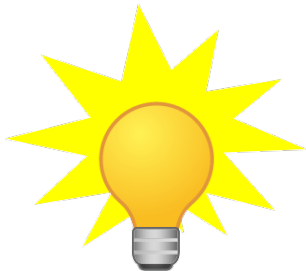


楽

なぜ楽なのか？



設定が楽



Optimization

設定は最適化済み



構築が楽



**A few
minutes...**

**設定は最適化されているので
構築は数分!**



スケールアップが楽

MAX

CPU:32

Memory:244GiB

Storage:32TB



スケールアップが楽



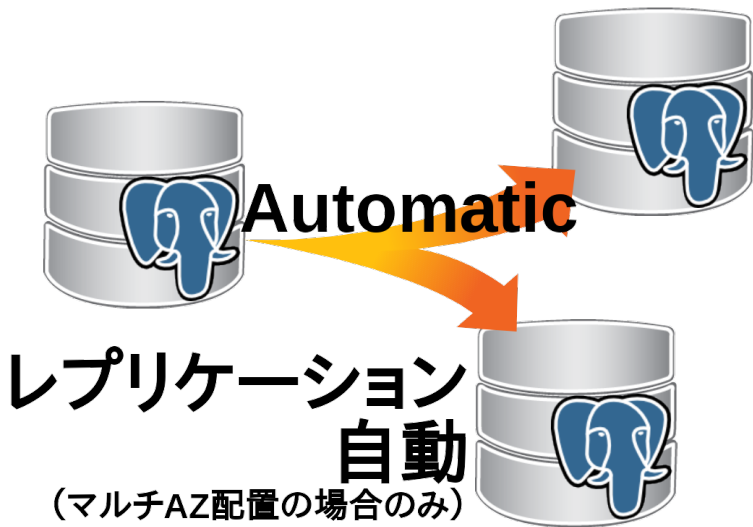
スケールリングは
数分

Storageは
ダウンタイムなし





障害対策が楽





障害対策が楽



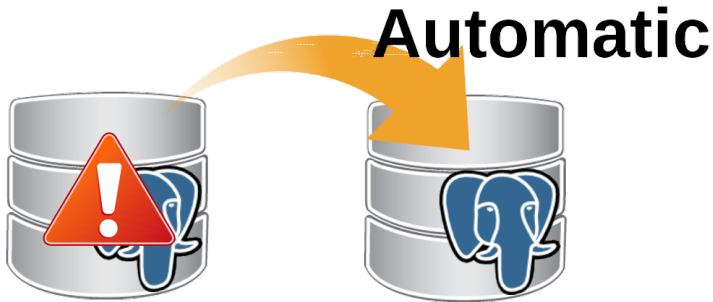
Automatic

バックアップ
自動





障害対策が楽



故障ハードの交換
自動



今日のテーマ

なるべく**楽**に
高機能で高速な
全文検索
がしたい



全文検索





全文検索

なぜPGroonga
なのか？



対応言語

全言語対応



速い

高速



速い

ベンチマーク

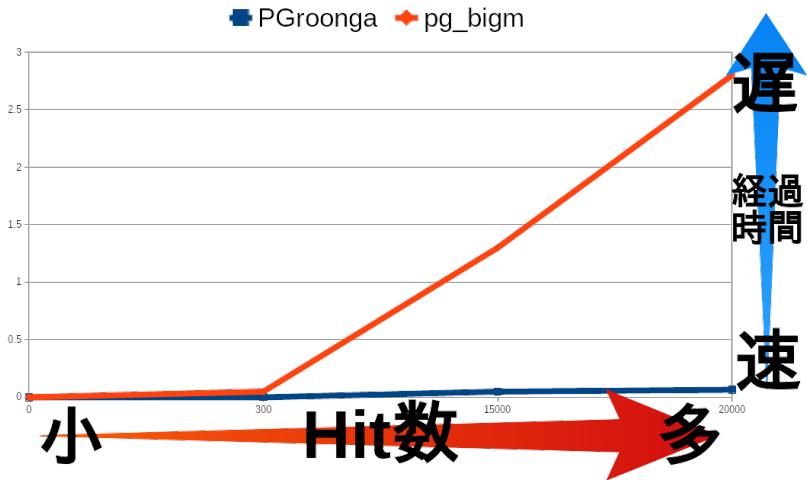


速い

- 日本語Wikipedia
 - レコード数：約90万レコード
 - 平均テキストサイズ：6.7KiB



速い





速い

- ベンチマークの詳細な条件は以下を参照

<https://pgroonga.github.io/ja/reference/pgroonga-versus-pg-bigm.html>



SQLが使える



専用のクエリー
を覚えなくてOK!





実例

具体的に
どう書くのか？



実行例：テーブル定義

```
CREATE TABLE entries (  
  title text,  
  content text  
);
```



実行例： インデックス定義

```
-- 全文検索用インデックス  
CREATE INDEX entries_full_text_search  
ON entries  
-- 「USING pgroonga」 = 「PGroongaを使う」  
USING pgroonga (title, content);
```



実行例：データ挿入

```
-- 普通に挿入するだけでよい  
INSERT INTO entries  
VALUES ('PGroongaで高速全文検索！',  
       '高速に全文検索したいですね！');
```



実行例：全文検索

```
SELECT title FROM entries
WHERE
-- &@~で全文検索
-- 「検索」と「高速」をAND検索
title &@~ '検索 高速' OR
content &@~ '検索 高速';
```



実行例 : LIKE

```
SELECT title FROM entries  
WHERE
```

- *LIKE*でもインデックスが効く
- = アプリを書き換えずに高速化可能
- ただし&@~より性能が落ちる

```
title LIKE '%検索%' OR  
content LIKE '%検索%';
```



機能

- 全文検索に必要なような機能は一通り揃っている
 - 同義語検索
 - 類似文書検索
 - 読みがな検索
 - 入力補完 etc..



読みがな検索

「やきにく」
ってどう書きますか？



読みがな検索

- やきにく
- 焼き肉
- 焼肉
- やき肉
- ヤキニク



読みがな検索

当然ですがどれ
も「やきにく」
と読みます



読みがな検索

- 読みが同じなので、以下は全部同じものとして扱えます
 - やきにく
 - 焼き肉
 - 焼肉
 - やき肉
 - ヤキニク



読みがな検索

例えば
「やきにく」で
検索すると



読みがな検索

- 「やきにく」 **Hit!**
- 「焼き肉」 **Hit!**
- 「焼肉」 **Hit!**
- 「やき肉」 **Hit!**
- 「ヤキニク」 **Hit!**



読みがな検索

異体字



読みがな検索

「広」と「廣」



読みがな検索

例えば人名の
検索



読みがな検索

検索キーワード「広瀬」で

- 「広瀬」 **Hit**
- 「廣瀬」 **Hit**

となつてほしい



読みがな検索

通常 of 検索

検索キーワード「広瀬」で

- 「広瀬」のみ**Hit**



読みがな検索

読みがな検索なら
検索キーワード「広瀬」で

- 「広瀬」 **Hit**
- 「廣瀬」 **Hit**



読みがな検索

両方ヒット！



読みがな検索

「広瀬」も
「廣瀬」も
読みが同じ



他にも

- それっぽい順でソート
- キーワードハイライト
- キーワードの周辺テキスト表示



他にも

- 電話番号検索
 - 090-1234-5678 と 090 1234 5678、
(090)1234-5678 等
- fuzzy検索
 - typo対策
 - テクノロジーとテノクロジー



他にも

継続的に
メンテナンス
されている



他にも

PostgreSQL12
に対応！（昨日）



高機能で高速

PGroongaなら
高機能で高速に
全文検索できる



Amazon RDS + PGroonga

しかし。 。 。



Amazon RDS + PGroonga



インストール不可





Amazon RDS + PGroonga

他の2つの要素
の出番！



Amazon RDS + PGroonga





構成



Logical
Replication



更新



検索





ロジカル レプリケーションの特徴

複製元と複製先
の構造が同一で
なくてもよい



ロジカル レプリケーションの特徴

Publisher



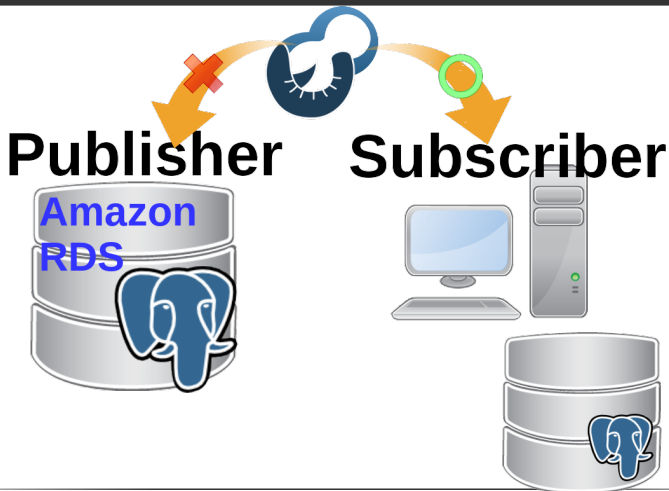
Subscriber

≠





ロジカル レプリケーションの特徴





検索

Publisher



Logical Replication





更新

Publisher



Write Only!

Logical Replication

Subscriber





問題点

Subscriberが
1台しかない

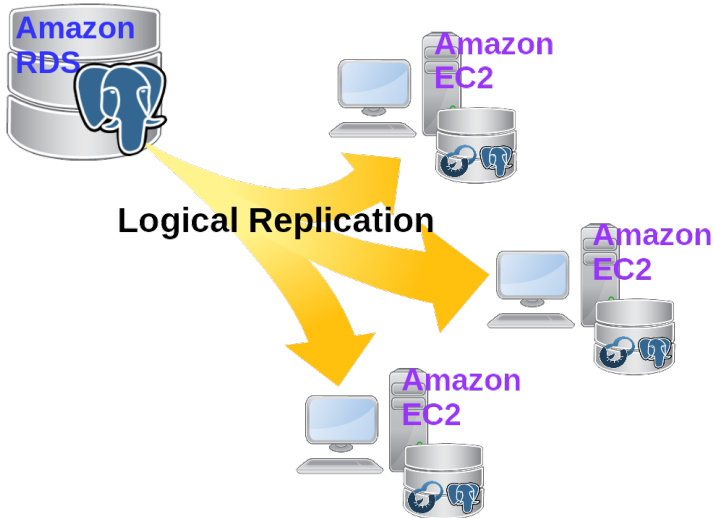


問題点

- リクエストが増加しつづけた場合、この構成では耐えられない



負荷分散





復旧

検索できなく
なったら

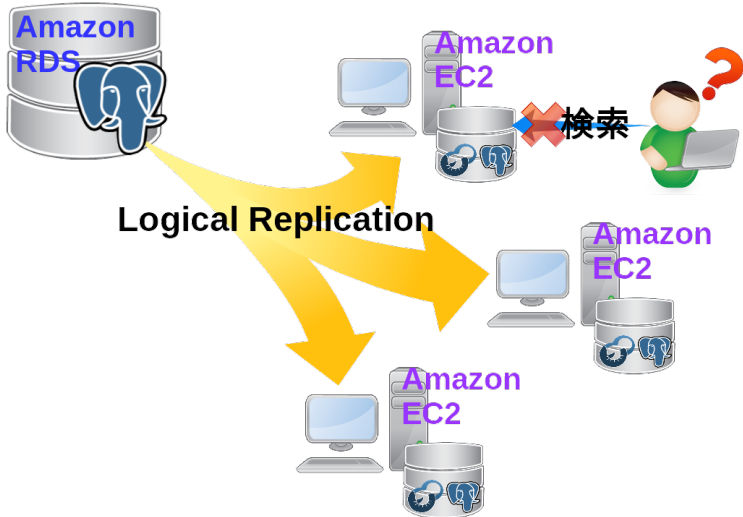


復旧

使い捨てる
復旧は
がんばらない

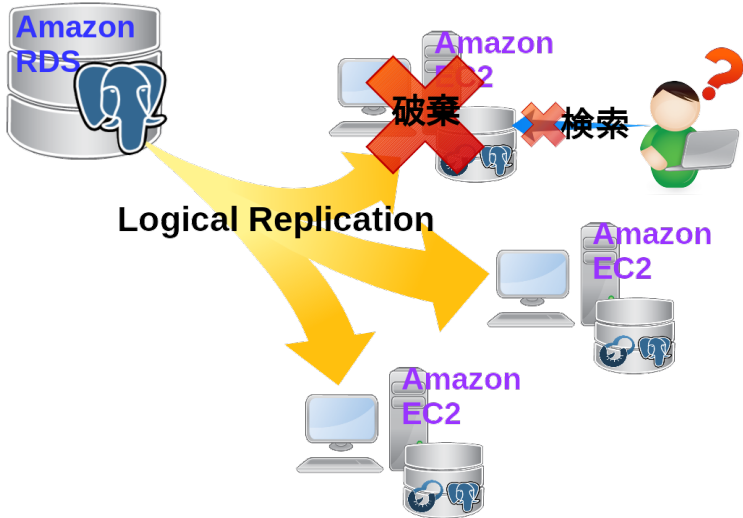


復旧



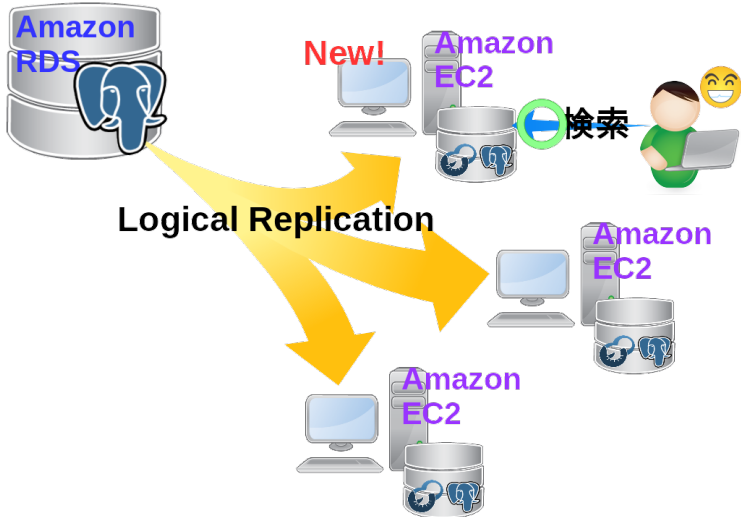


復旧





復旧





サービス開始時間

もう一つの問題

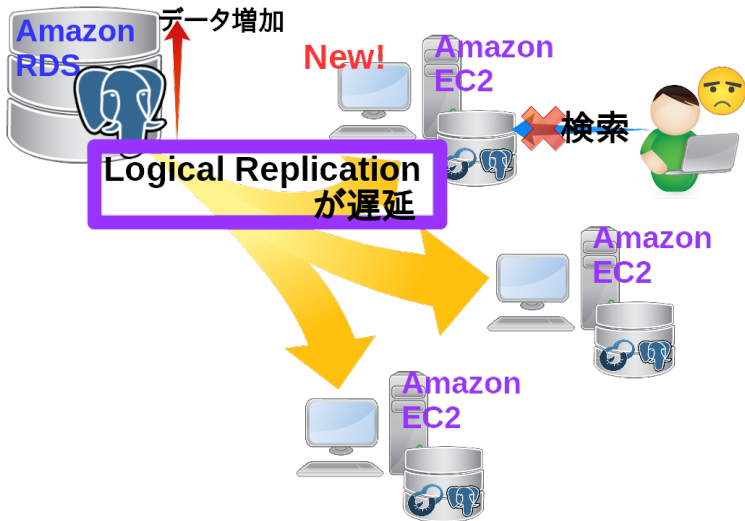


サービス開始時間

データが増える
と復旧が遅延



サービス開始時間





停止時間の見積もり

許容できる停止
時間は？



停止時間の見積もり

停止時間を
見積もる



停止時間の見積もり

- 例えば...
 - サービス復旧時間：1日
 - 新しくAmazon EC2を作成して、サービス開始できるまでの時間
 - 故障：1ヶ月に1回の頻度で故障



停止時間の見積もり

- リクエスト：
 - 各EC2には均等にリクエストが振り分けられる
- サービス継続：
 - Amazon EC2が3台あれば、サービスを提供可能なくらいの負荷

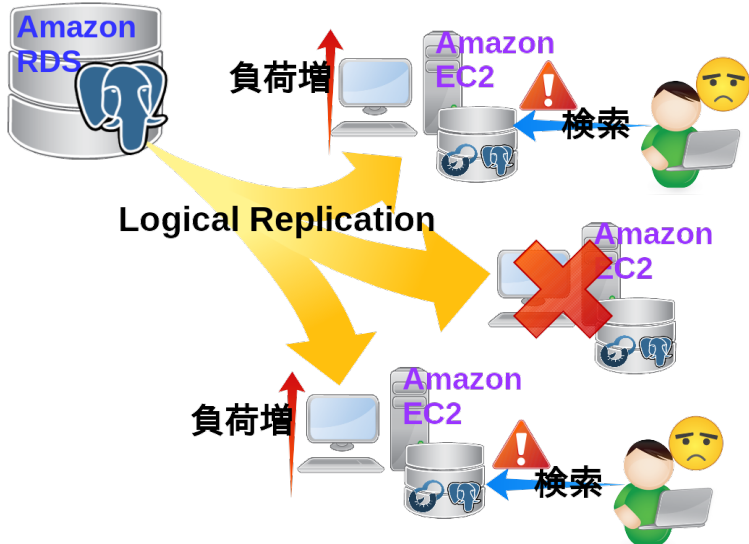


停止時間の見積もり

- Amazon EC2を3台で運用する場合
 - 1台でも故障するとサービス継続不可



停止時間の見積もり





停止時間の見積もり

- この場合の稼働率
 - 平均故障間隔 =
 $365 * 24 / 12 = 730$ 時間
 - 平均復旧時間 = 24時間
 - 稼働率 = $730 / 754 =$
 $0.9681697612732095 \doteq 96.8\%$



停止時間の見積もり

つまり



停止時間の見積もり

- この構成では、1ヶ月に1日程度システムが停止する



停止時間の見積もり

- では、4台運用の場合ではどうなるのか？



停止時間の見積もり

- 1台あたりの稼働率 : 96.8%
- 1台あたりの故障率 :
 $100\% - 96.8\% = 3.2\%$
- 2台同時に故障する確率 :
 $0.032 * 0.032 \div 0.001 = 0.1\%$

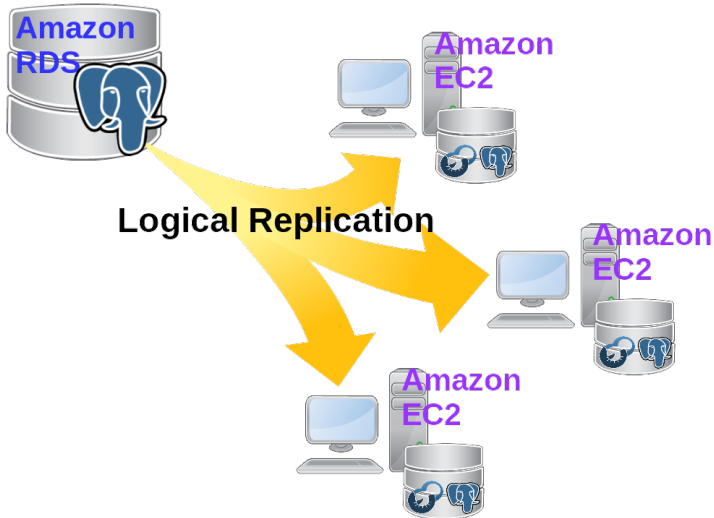


停止時間の見積もり

- したがって、1ヶ月に約45分程度システムが停止する



まとめ





まとめ

なるべく**楽**に
高機能で**高速**な
全文検索が
できました！



最後に

- PGroongaについての疑問等は、GitHub、Gitterにて
- ドキュメントも充実
 - <https://pgroonga.github.io/ja/>



最後に

より突っ込んだお話がしたい場合

は↓↓

問い合わせ先：

<https://www.clear-code.com/contact/?type=groonga>



最後に

ご静聴ありがとうございました