AWS入門～第6回～

明けましておめでとうございます。今年もよろしくお願いします。

自己紹介

- 重本　尚志
- 略歴:
  - 徳島大学工学部卒業（2008-03）
  - 独立系IT企業に新卒として入社（2008-04）
  - C#やJavaを中心（クラサバ多め）に案件を転々とする。
  - 退職・独立（2017-01）
- 趣味:トレーディングカード収集
- 好きな食べ物:奈良漬、味噌ラーメン
- 最近購入した本:The Non Designer’s Design Book

今年の目標

- 法人化
- デザインを少しずつできるようになる
- PowerShell覚えたい（という願望）
- 日商簿記の2級を取得する（これは来年2月を受験目標にしている）

目次

- DynamoDBとは
- DynamoDBを使ってみる

DYNAMODBとは

- 完全マネージド型のNoSQLデータベースサービス
- Web版だけではなく、ダウンロードして利用できるローカル版が存在する
  - ローカル版を利用する場合、JRE6.X以降が必要
- インデックスを利用することも可能

- 毎月一定の無料枠が存在する
  - 初回ログインから●●ヵ月、初回利用から●●ヵ月といった制限はない
- 従量課金制で、データ転送、データ容量、読み込み及び書き込みのリクエスト数等によって料金が決定される

DYNAMODBとは

NoSQLとは？

- Not only SQLの略（SQL言語を利用せず、データを保存、参照することができる）
- RDBMSのように、複雑な構造・関連性を持たせることには向いていない
- スケーラビリティ（拡張性）が高いため、取得しておく必要があるが、頻繁には参照しないログ等のデータ保存に向いている
- 各データモデルの特徴については、下記URLを参照。
  - http://www.atmarkit.co.jp/ait/articles/1709/29/news001_2.html
  - http://www.atmarkit.co.jp/ait/articles/1102/24/news098_3.html

DYNAMODBとは

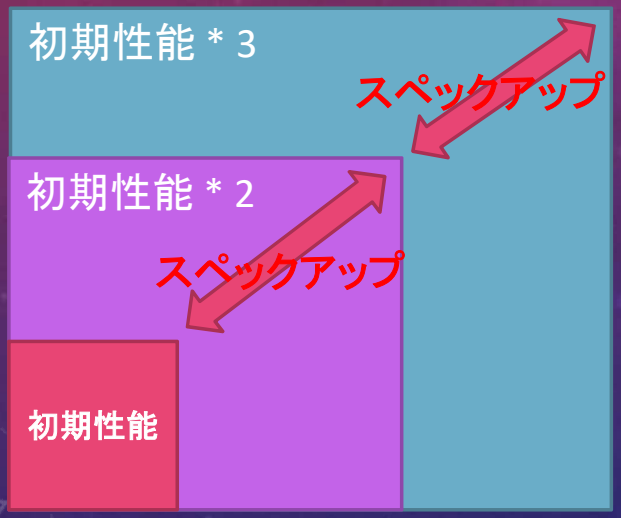
RDBMSとの比較（参考：https://aws.amazon.com/jp/nosql/）

	RDBMS	NoSQL
データモデル	データを行と列から成るテーブルと呼ばれる表形式の構造に正規化され、各データベース要素は、スキーマによって厳密に定義される。	スキーマは不要。ドキュメントの取得には、通常パーティションキーが使用される。
パフォーマンス	ディスクサブシステムに左右される。クエリ、インデックス、テーブル構造の最適化が必要。	基盤となるハードウェアクラスタのサイズ、ネットワークレイテンシー、呼び出すアプリケーションに依存する。
拡張性	高速なハードウェアを使用することで、簡単にスケール "アップ" できる。分散システムで使用するには追加の投資が必要。	低コストなハードウェアの分散クラスタを使用してスケール “アウト” することで、レイテンシーを増やすことなくスループットを改善できる。

DYNAMODBとは

スケールアップ・スケールアウト

スケールアップ：
サーバをスペックアップして性能を高める



スケールアウト：
サーバ台数を増やして性能を高める

DYNAMODBとは

DynamoDBのコアコンポーネント

テーブル

- 項目・・・全ての項目間で一意に識別可能な属性のグループ
- 項目
- 属性・・・・個別の項目
- 属性
- 属性

Personテーブル

- id
- company_code
- name
- address
- tel

DYNAMODBとは

毎月の無料枠

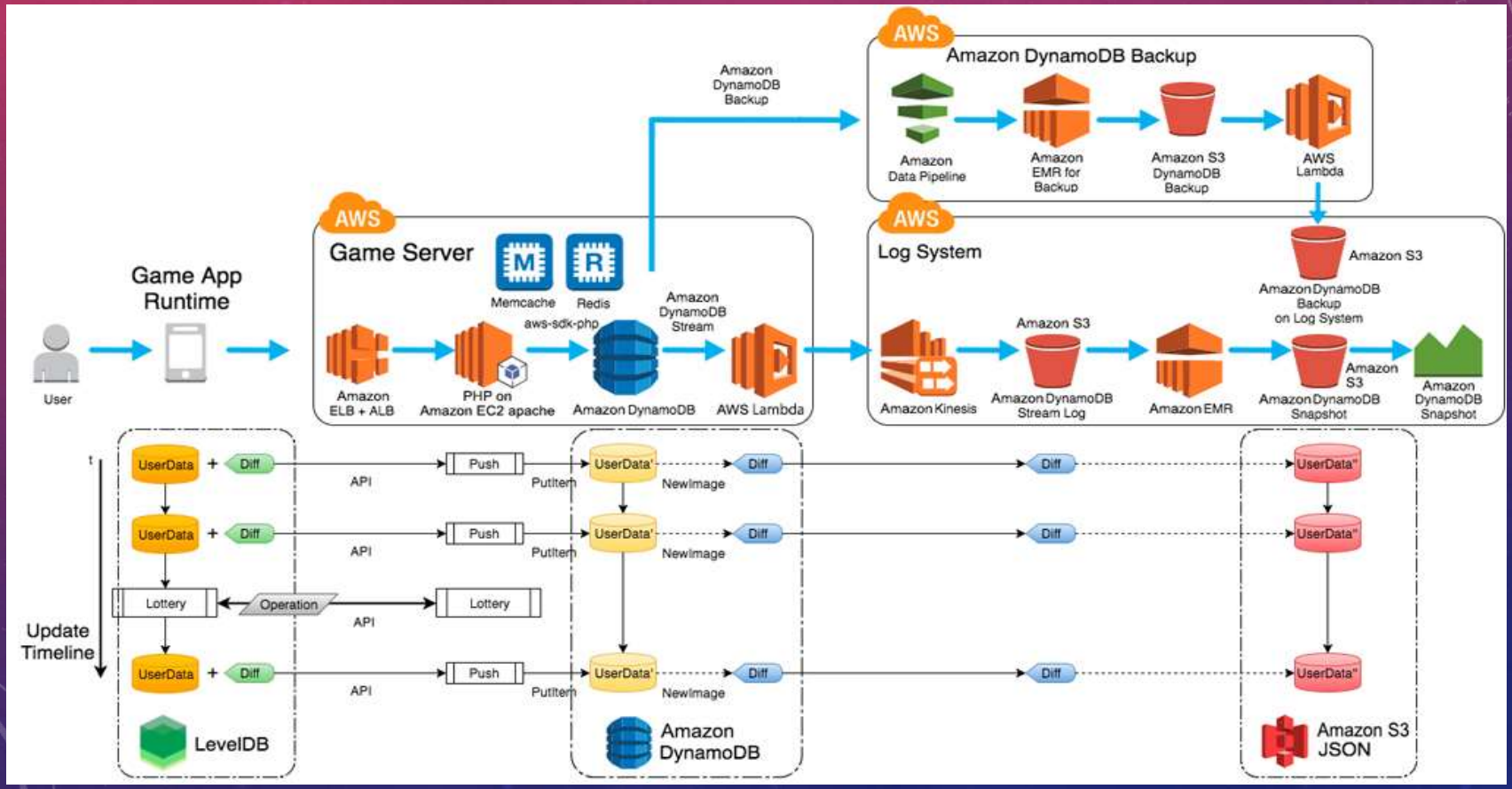
No.	無料枠の内容
①	1 か月あたり最大 2 億リクエスト (書き込みキャパシティー 25 ユニットおよび読み込みキャパシティー 25 ユニット) を処理するのに十分なスループット。 https://docs.aws.amazon.com/ja_jp/amazondynamodb/latest/developerguide/HowItWorks.ProvisionedThroughput.html
②	25 GB のインデックス化データストレージ。
③	DynamoDB ストリームからの 1 か月あたり 250 万件の読み込みリクエスト。

DYNAMODBとは

- 月々の詳細な料金については、下記URLを参照
  - https://aws.amazon.com/jp/dynamodb/pricing/

リソースタイプ	詳細	月額料金
プロビジョニングするスループット (書き込み)	1 つの書き込みキャパシティーユニット (WCU) は、1 秒あたり最大 1 回の書き込みを提供します。1 か月あたり最大 250 万回の書き込みが可能です。	1 WCU あたり最低 0.47 USD
プロビジョニングするスループット (読み込み)	1 つの読み込みキャパシティーユニット (RCU) は、1 秒あたり最大 2 回の読み込みを提供します。1 か月あたり最大 520 万回の書き込みが可能です。	1 RCU あたり最低 0.09 USD
インデックス化データストレージ	DynamoDB では、テーブルが消費するディスク容量の GB あたりの時間料金が課金されます	1 GB あたり最低 0.25 USD

DYNAMODBの利用例（GREE）



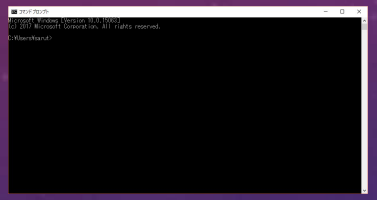
DYNAMODBを使ってみる

- DynamoDBの基本操作をいくつか行ってみる。
  - 投入データはJSONファイルを利用する。
  - ローカル版のDynamoDBを利用する。
    - https://docs.aws.amazon.com/ja_jp/amazondynamodb/latest/developerguide/DynamoDBLocal.html
- 今回は、以下のようなデータをテーブルに作成する。
  - テーブル名 : company

id	company_name	employee_num
1	testCompany1	50
2	testCompany2	100

DYNAMODBを使ってみる

今回の利用環境構成

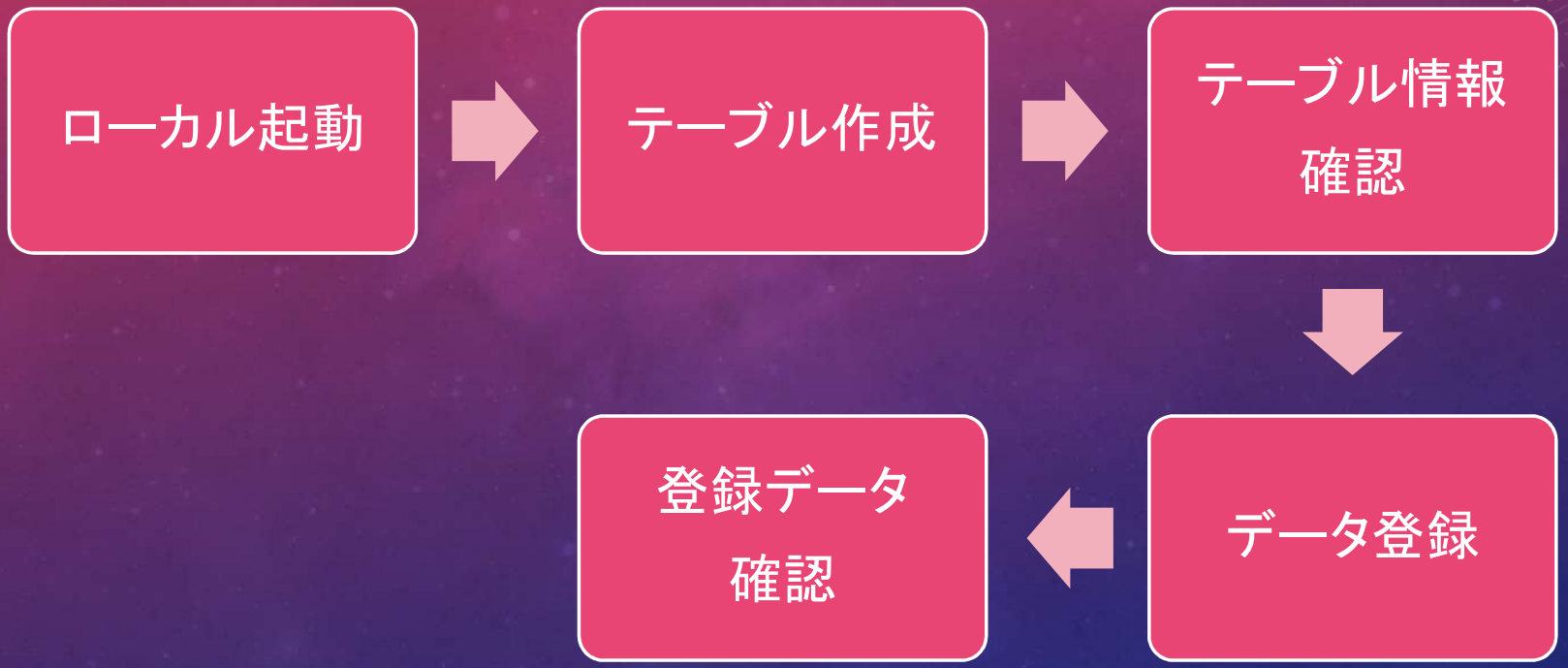


awscliで操作

Amazon DynamoDB

Java

DYNAMODBを使ってみる



DYNAMODBを使ってみる

ローカル起動

- 以下のコマンドを実行することにより、ローカル版のDynamoDBを起動することができる。
  - java -Djava.library.path=./DynamoDBLocal_lib -jar DynamoDBLocal.jar -sharedDb

- 注意点
  - コマンド実行は、ローカル版のDynamoDBのjarが存在するフォルダに移動して実行する必要がある。
  - ローカル版を利用する場合、AWS CLIのコマンド末尾に以下を付ける必要がある。
    - **--endpoint-url http://localhost:8000**

DYNAMODBを使ってみる

テーブル作成

- 以下のコマンド（create-table）を実行する。

```
aws dynamodb create-table --table-name company ^
--attribute-definitions AttributeName=id,AttributeType=S ^
--key-schema AttributeName=id,KeyType=HASH ^
--provisioned-throughput ReadCapacityUnits=1,WriteCapacityUnits=1 ^
--endpoint-url http://localhost:8000
```

- 上記コマンドを実行するとidという項目のみを保持するテーブルが作成される。
  - その他の項目はデータ投入時に属性を記述することで作成する。

DYNAMODBを使ってみる

テーブル情報確認

- 以下のコマンド（describe-table）を実行する。

```
aws dynamodb describe-table --table-name company ^
--endpoint-url http://localhost:8000
```

- テーブル情報がJSON形式で表示される。
  - スキーマ定義がないため、レコード毎に異なる項目を設定することもできる。

id	name	age	sex	
id	address			
id	tel	remarks	birthday	deleteflg

DYNAMODBを使ってみる

データ登録

- 以下のコマンド（batch-write-item）を実行する。

```
aws dynamodb batch-write-item ^
--request-items file://c:/PUreatio/putitem.json ^
--endpoint-url http://localhost:8000
```

- 今回は複数項目を書き込むためにbatch-write-itemを利用したが、個別登録の場合はput-itemコマンドを利用する。

```
aws dynamodb put-item ^
--table-name company ^
--item file://c:/PUreatio/putitem.json ^
--endpoint-url http://localhost:8000
```

DYNAMODBを使ってみる

登録データ確認

- 以下のコマンド（scan）を実行する。
  - 取得結果はJSON形式で表示される。

```
aws dynamodb scan ^
--table-name company ^
--endpoint-url http://localhost:8000
```

- キーを指定して取得する場合には、get-item、batch-get-itemコマンドを利用する。

```
aws dynamodb get-item ^
--table-name company ^
--key file://c:/PUreatio/getitem.json ^
--endpoint-url http://localhost:8000
```

DYNAMODBを使ってみる

登録時に利用するJSON

```
{"company": [{
   "PutRequest": { "Item": {
      "id": {"S": "1"},
      "company_name": {"S": "testCompany1"},
      "employee_num": {"N": "50"}
   }}
},
{
   "PutRequest": { "Item": {
      "id": {"S": "2"},
      "company_name": {"S": "testCompany2"},
      "employee_num": {"N": "100"}
   }}
}]}
```

取得時に利用するJSON

```
{
   "id": {"S": "1"}
}
```

まとめ

- データはキー、値のペアで保存され、AWSCLI上では主にJSONで処理する
- コンソールで有効なJSONを作るのは苦行の場合がある
- 毎月の無料枠が存在する

- Javaさえインストールしていれば動かせるローカル版が勉強に便利
  - 一通りコマンドを動かしてみる程度ならこれで十分

- スケールアウトで拡張可能なため、単純にハードディスクを追加すれば拡張可能。

ご清聴ありがとうございました。