What Is DynamoDB?

- A fully managed NoSQL datastore
- Removes the need to manage a database
- Handles any amount of throughput needed
- Data is encrypted at rest.
  - DynamoDB Encryption at Rest
- Used to store JSON documents
  - Not a relational database
- Provides High Availability and Durability

xerris

DynamoDB Introduction

DynamoDB is highly available

- DynamoDB data is spread across several servers to improve throughput
- ALL Dynamo data is stored on SSD drives

DynamoDB is Durable

- Data is replicated across availability zones
- Data can be global syncing between AWS Regions

xerris

DynamoDB

- **Table** - Similar to a relational database table
- **Partition Key** - Similar to a relational database primary key
- **Sort Key** - Similar to a composite primary key
- **Secondary Index** - Similar to an Alternate Key
- **Item** - Similar to a database row
- **Attribute** - Similar to a database column

xerris

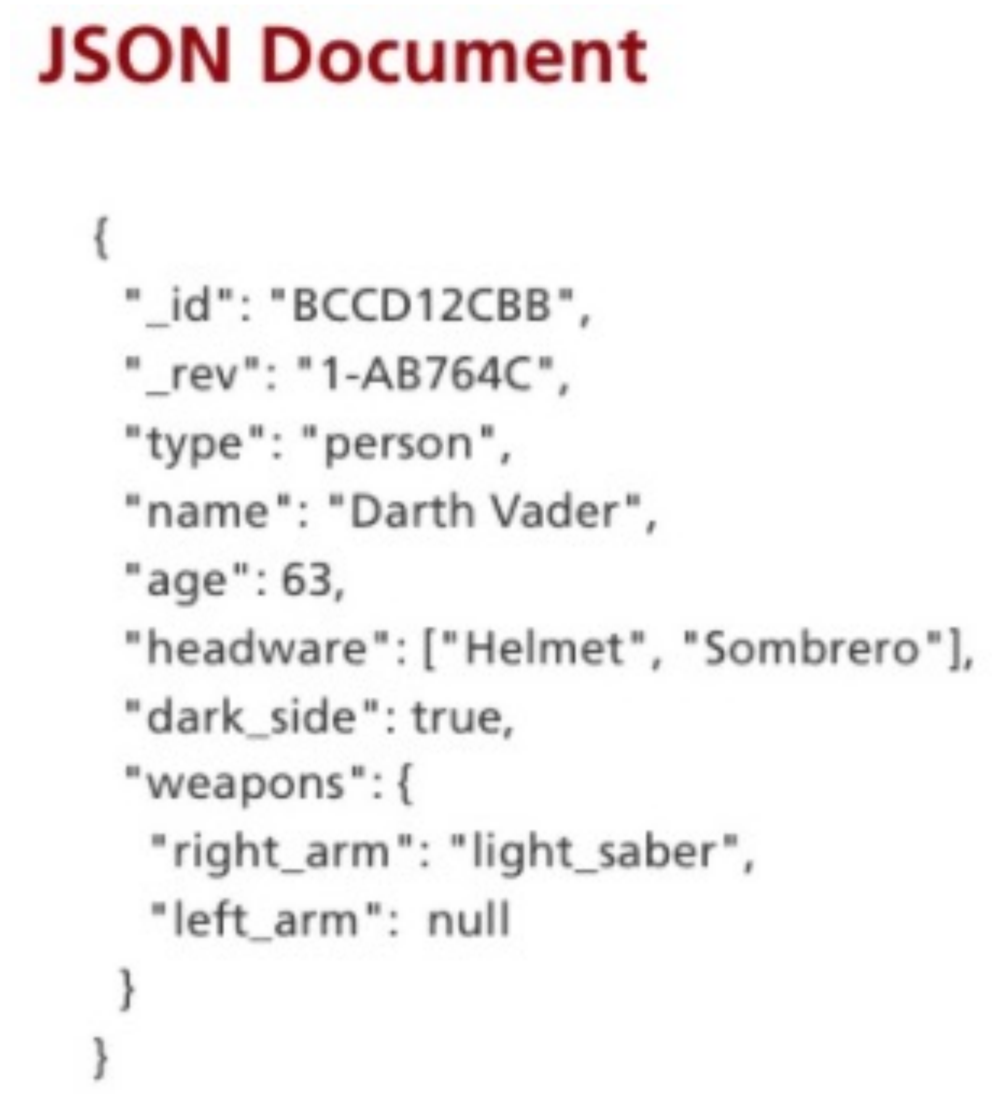
DynamoDB Anatomy

- **Partition Key** (Hash Attribute)
  - An attributed used to generate an internal hash
  - Used to physically store internally within Dynamo
  - Used to 'partition' data across different shards within Dynamo
- **Partition Key & Sort Key** (Composite Primary Key)
  - An Item is unique if the **Partition Key** & Sort **Key** are unique in that table
  - The partition key determines where physically the item is stored
  - *All items with the same partition key are stored together*
  - Example:
    - BlogPost.**Author** (Partition Key)
    - BlogPost.**PostedDate** (Sort Key)

DynamoDB Introduction

DynamoDB is a Document Database

- Used to store unstructured data



JSON Document

```
{
  "_id": "BCCD12CBB",
  "_rev": "1-AB764C",
  "type": "person",
  "name": "Darth Vader",
  "age": 63,
  "headware": ["Helmet", "Sombrero"],
  "dark_side": true,
  "weapons": {
    "right_arm": "light_saber",
    "left_arm":  null
  }
}
```

xerris

DynamoDB

DynamoDB is a NoSQL database

- Stores Key-value data
  - Use Dynamo to store JSON data based on a key.
- Graph databases
  - Stores graph nodes and allows for the graph schema to change over time
- Wide-column stores
  - Each row does not have to follow the same structure
  - Each column is stored separately

xerris

Working With Dynamo

Creating a Table (MacOS)

```
aws dynamodb create-table \
 --table-name Music \
 --attribute-definitions \
       AttributeName=Artist,AttributeType=S \
       AttributeName=SongTitle,AttributeType=S \
 --key-schema \
       AttributeName=Artist,KeyType=HASH \
       AttributeName=SongTitle,KeyType=RANGE \
 --provisioned-throughput \
       ReadCapacityUnits=10,WriteCapacityUnits=5
```

xerris

Working With Dynamo

Creating a Table (Windows PowerShell)

```
aws dynamodb create-table `
 --table-name Music `
 --attribute-definitions `
        AttributeName=Artist,AttributeType=S `
        AttributeName=SongTitle,AttributeType=S `
 --key-schema `
        AttributeName=Artist,KeyType=HASH `
        AttributeName=SongTitle,KeyType=RANGE `
 --provisioned-throughput `
        ReadCapacityUnits=10,WriteCapacityUnits=5
```

Working With Dynamo

Provisioned Read/Writes

- Specifies the Read/Write capacity units for your application
- Read Capacity - # of strongly consistent reads per second
  - For items 4kb or smaller
  - Items > 4kb require additional read units

- Write Capacity - # of writes per second for items <= 1kb
  - Items are rounded up to the nearest 1kb

Working With Dynamo

Adding an Item (Mac OS)

```
aws dynamodb put-item \
    --table-name Music \
    --item \
        '{"Artist": {"S": "No One You Know"}, "SongTitle": {"S": "Call Me Today"}, 
"AlbumTitle": {"S": "Somewhat Famous"}}' \
    --return-consumed-capacity TOTAL
```

Working With Dynamo

Adding an Item (Windows PowerShell)

```
aws dynamodb put-item `
    --table-name Music `
    --item `
        "{\`"Artist\`": {\`"S\`": \`"No One You Know\`"}, \`"SongTitle\`": {\`"S\`": \`"Call
Me Today\`"}, \`"AlbumTitle\`": {\`"S\`": \`"Somewhat Famous\`"}}" `
    --return-consumed-capacity TOTAL
```

Working With Dynamo

Reading an Item (Mac OS)

```
aws dynamodb get-item --consistent-read \
    --table-name Music \
    --key '{ "Artist": {"S": "No One You Know"}, "SongTitle": {"S": "Call Me 
Today"}}’
```

Querying Dynamo for an Item (Mac OS)

```
aws dynamodb query \
    --table-name Music \
    --key-condition-expression "Artist = :name" \
    --expression-attribute-values  '{":name":{"S":"No One You Know"}}'
```

Working With Dynamo

Reading an Item (Windows PowerShell)

```
aws dynamodb get-item --consistent-read `
    --table-name Music `
    --key "{ \`"Artist\`": {\`"S\`": \`"No One You Know\`"}, \`"SongTitle\`": {\`"S\`":
\`"Call Me Today\`"}}”
```

Querying Dynamo for an Item (Windows PowerShell)

```
aws dynamodb query `
 --table-name Music `
  --key-condition-expression "Artist = :name" `
 --expression-attribute-values  "{ \`":name\`":{\`"S\`":\`"No One You Know\`"}}"
```

Global Secondary Index

An index where the partition key and sort key differ from the base table.

Create Global Secondary Index (Mac OS)

```
aws dynamodb update-table \
    --table-name Music \
    --attribute-definitions AttributeName=AlbumTitle,AttributeType=S \
    --global-secondary-index-updates \
        "[{\"Create\":{\"IndexName\": \"AlbumTitle-
index\",\"KeySchema\":[{\"AttributeName\":\"AlbumTitle\",\"KeyType\":\"HASH\"}], \
        \"ProvisionedThroughput\": {\"ReadCapacityUnits\": 10, \"WriteCapacityUnits\": 5
},\"Projection\":{\"ProjectionType\":\"ALL\"}}}]"
```

Global Secondary Index

An index where the partition key and sort key differ from the base table.

Create Global Secondary Index (PowerShell)

```
aws dynamodb update-table `
    --table-name Music `
    --attribute-definitions AttributeName=AlbumTitle,AttributeType=S `
    --global-secondary-index-updates `
        "[{\`"Create\`":{\`"IndexName\`": \`"AlbumTitle-
index\`",\`"KeySchema\`":[{\`"AttributeName\`":\`"AlbumTitle\`",\`"KeyType\`":\`"HASH\`"}], `
        \`"ProvisionedThroughput\` ": {\`"ReadCapacityUnits\`": 10, \`"WriteCapacityUnits\`": 5
},\`"Projection\`":{\`"ProjectionType\`":\`"ALL\`"}}}]"
```

xerris

Working With Dynamo

Amazon AWS DotNet SDK

- Provides a client library for DynamoDB (nuget)

  Install-Package AWSSDK.DynamoDBv2

- AmazonDynamoDBClient Class
  - Provides connectivity to Dynamo
  - Used in conjunction with the DynamoDBContext

- DynamoDBContext
  - Used for create/read/update/delete (CRUD) operations

xerris