DynamoDB Design Patterns

Ivan Mushketyk

@mushketyk   brewing.codes

Overview

DynamoDB best practices

Data Modeling with DynamoDB

Hot Keys

Reducing cost with DynamoDB

Data Modeling in DynamoDB

Data modeling for

- 1:1 relationships
- 1:N relationships
- M:N relationships

Composite keys

1:1 Relationship

OrderId	ItemId	Date	TotalPrice
1	1	2017_05_11	117.8
2	2	2017_08_11	167
3	2	2017_11_22	167

OrderId	Status	Courier
1	DELIVERED	FedEx
2	IN_TRANSIT	DHL
3	DELIVERED	DHL

Why Use 1:1 Relationship

- DynamoDB item limitation
- Update an attribute less costly
- Can create more indexes
- Can help to save money

1:M Relationship

Partitionkey

OrderId	ItemId	Name	Price
1	4	DynamoDB sticker	1
1	5	DynamoDB book	25
2	9	Quadrocopter	300
3	4	Phone	10
3	5	Bitcoin miner	454

Sort key

M:N Relationship

Partition key | Sort key

OrderId	ItemId	Name	Price
1	4	DynamoDB sticker	1
1	5	DynamoDB book	25
2	9	Quadrocopter	300
3	4	Phone	10
3	5	Bitcoin miner	454

GSI sort key | GSI partition key

Composite Keys

Can query using only one index

Find all orders delivered in May

Need to search by status AND date

Can use composite keys

Composite Keys Example

OrderId	ItemId	Status	Date
1	1	DELIVERED	2017_01_02
2	1	DELIVERED	2017_05_02
3	2	SHIPPED	2017_05_08
4	2	CANCELED	2017_06_11
5	1	DELIVERED	2017_05_14

Composite Keys Example

OrderId	ItemId	Status	Date	Status_Date
1	1	DELIVERED	2017_01_02	DELIVERED_2017_01_02
2	1	DELIVERED	2017_05_02	DELIVERED_2017_05_02
3	2	SHIPPED	2017_05_08	SHIPPED_2017_05_08
4	2	CANCELED	2017_06_11	CANCELED_2017_06_11
5	1	DELIVERED	2017_05_14	DELIVERED_2017_05_14

Composite Keys Example

OrderId	ItemId	Status	Date	Status_Date
1	1	DELIVERED	2017_01_02	DELIVERED_2017_01_02
2	1	DELIVERED	2017_05_02	DELIVERED_2017_05_02
3	2	SHIPPED	2017_05_08	SHIPPED_2017_05_08
4	2	CANCELED	2017_06_11	CANCELED_2017_06_11
5	1	DELIVERED	2017_05_14	DELIVERED_2017_05_14

GSI Partition key

Sort key

Query:
Status_Date **BEGINS_WITH** “DELIVERED_2017_05”

Hot Keys

What are hot keys

Symptoms of hot keys

How to deal with them

Selecting good partition keys

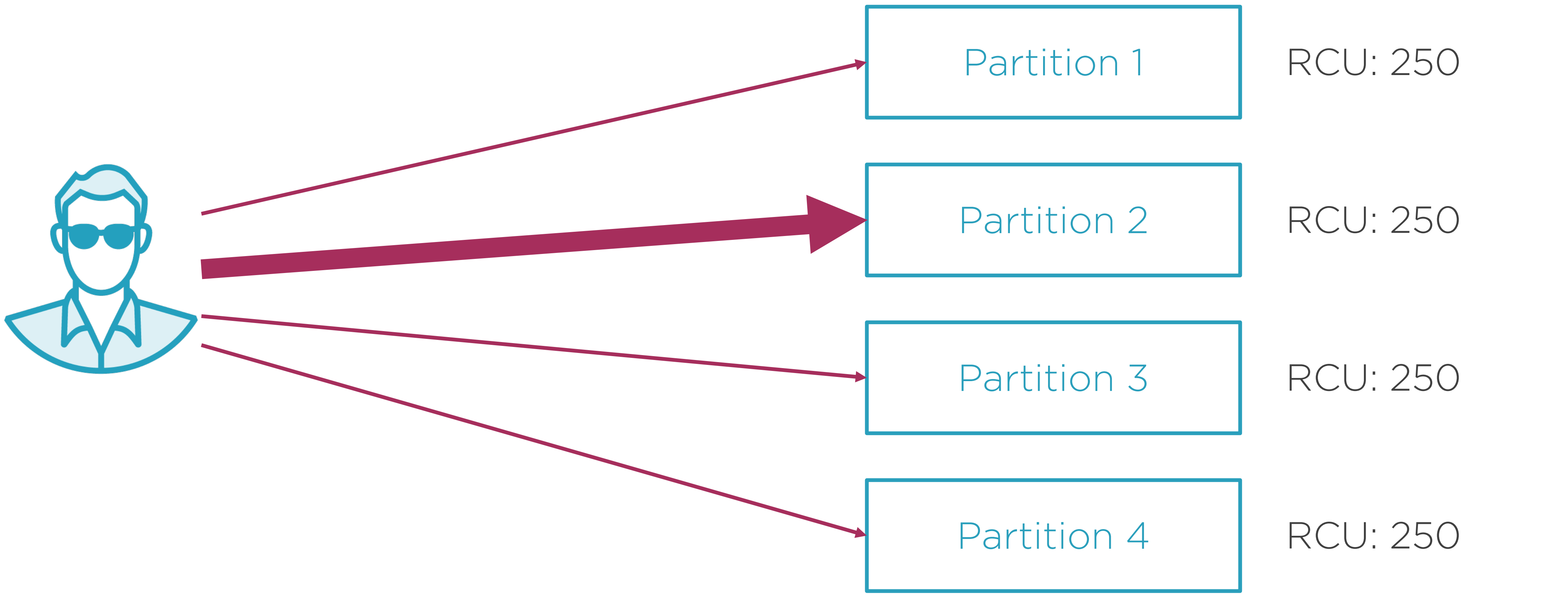
Will This Work?

Your table has 1000 RCUs

It receives 500 RCUs of requests

Any reasons to worry?

Hot Keys


Total RCU: 1000
Partition 1
RCU: 250
Partition 2
RCU: 250
Partition 3
RCU: 250
Partition 4
RCU: 250

Avoid Hot Keys

Ensure uniform load on all partitions

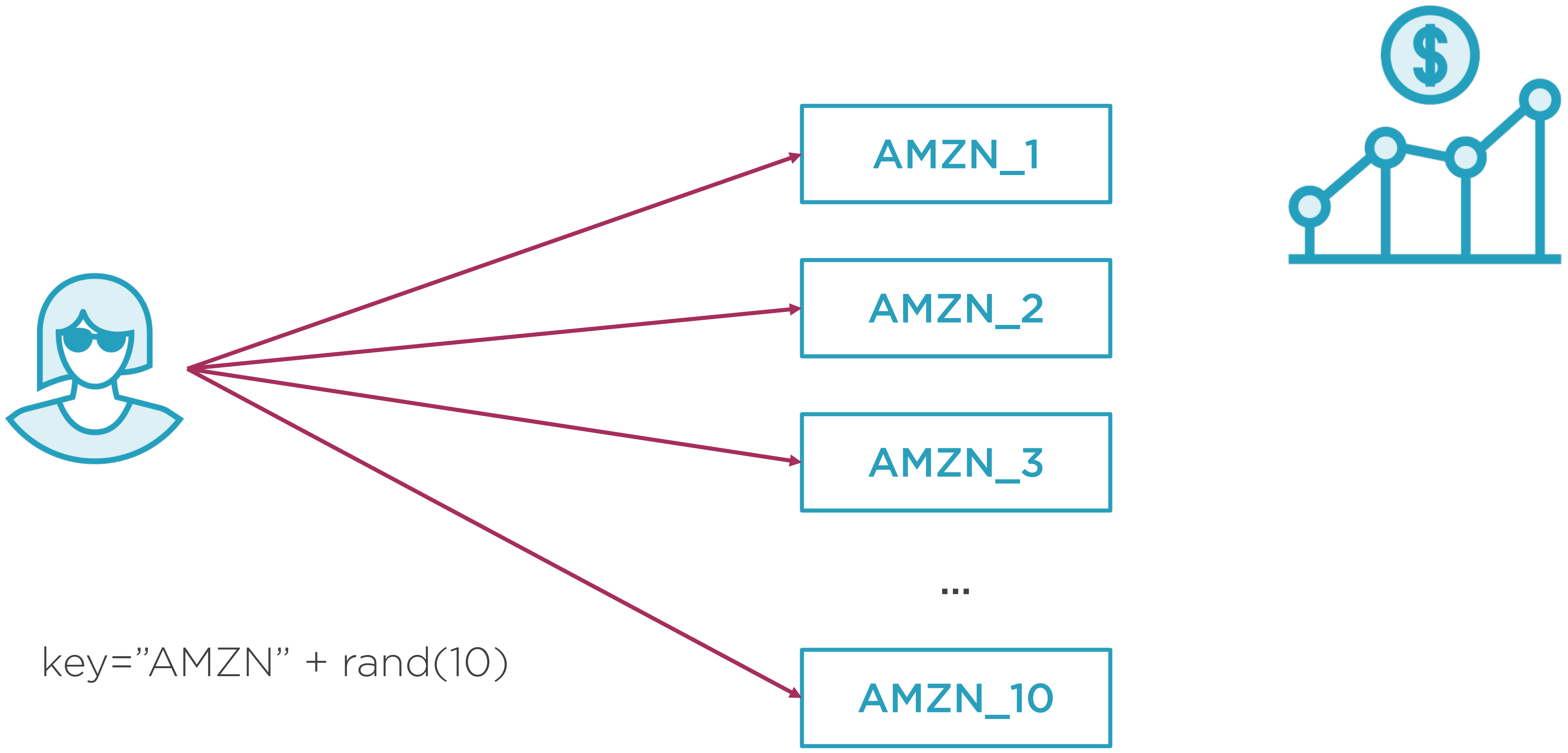
Use caching

Select good partition key

How to Select Partition Key

- ⊖ **Boolean value**
- ⊖ **Limited range of values**
- ⊕ **UUIDs**
- ⊕ **Numbers (range is unlimited)**

Randomized Values



key=”AMZN” + rand(10)

Reducing Costs of DynamoDB

High RCU and WCU lead to high cost

How to reduce RCUs/WCUs

Use Less Data

Store big items in S3

Use data compression

Attributes projection

Split into big items into multiple tables

Minimize RCUs

PostId	Time	Text
1	1498916052	Let me tell the story of my life...
2	1498185048	Once upon a time...

PostId	Time	Excerpt
1	1498916052	Let me tell
2	1498185048	Once upon

PostId	Text
1	Let me tell the story of my life...
2	Once upon a time...

Exploit Temporal Access Patterns

Tables	
2018_April	RCU: 2000 WCU:2000
2018_March	RCU: 1000 WCU:1
2018_February	RCU: 10 WCU:1

Time-to-live

DynamoDB removes item when they expire

No extra cost

Allows to save money

No need to manually remove items

Time-to-live Use Cases

Legal requirements

Session data

Temporary data

Enable Time-to-Live

```
TimeToLiveSpecification ttls = new TimeToLiveSpecification()
ttls.setAttributeName("TTL"); ttls.setEnabled(true);

UpdateTimeToLiveRequest request = new UpdateTimeToLiveRequest();
request.setTableName("Orders");
request.setTimeToLiveSpecification(ttls);

client.updateTimeToLive(ttls);
```

Sparse Indexes

Table

OrderId	ItemId	OrderType
1	3	
2	4	
3	5	Collection
4	6	

GSI Sort key

GSI Partition key

Index

OrderType	OrderId	ItemId
Collection	3	5

Other Tactics

Avoid scans (expensive)

Use queries

Distribute read/write operations

Reserved capacity

Summary

NoSQL database should be used differently

How to use indexes to represent relationships

How to avoid hot keys

How to save money using DynamoDB