

Advanced Design Patterns for DynamoDB

Rick Houlihan
Principal Technologist, NoSQL
AWS



© 2018, Amazon Web Services, Inc. or its affiliates. All rights reserved.



Agenda

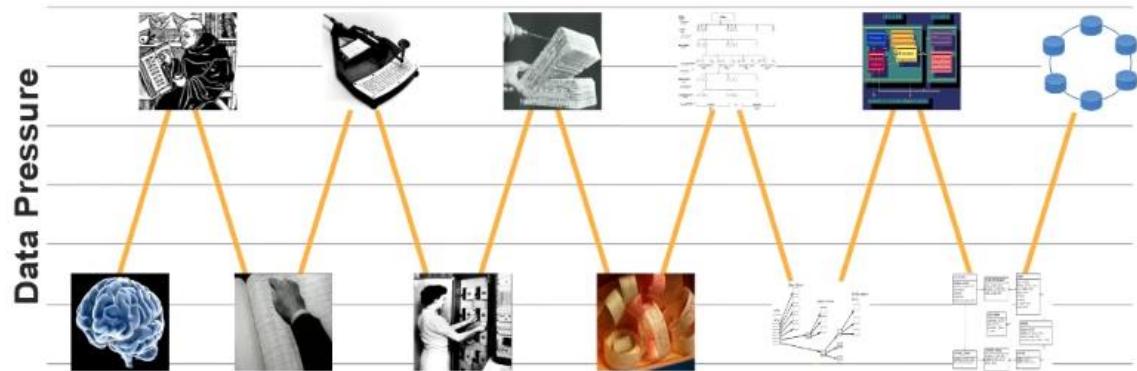
- Brief History of Data Processing (Why NoSQL?)
- Overview of DynamoDB
- NoSQL Data Modeling
 - Normalized versus De-normalized schema
- Common NoSQL Design Patterns
 - Composite Keys, Hierarchical Data, Relational Data
- Modeling Real Applications

History of Data Processing

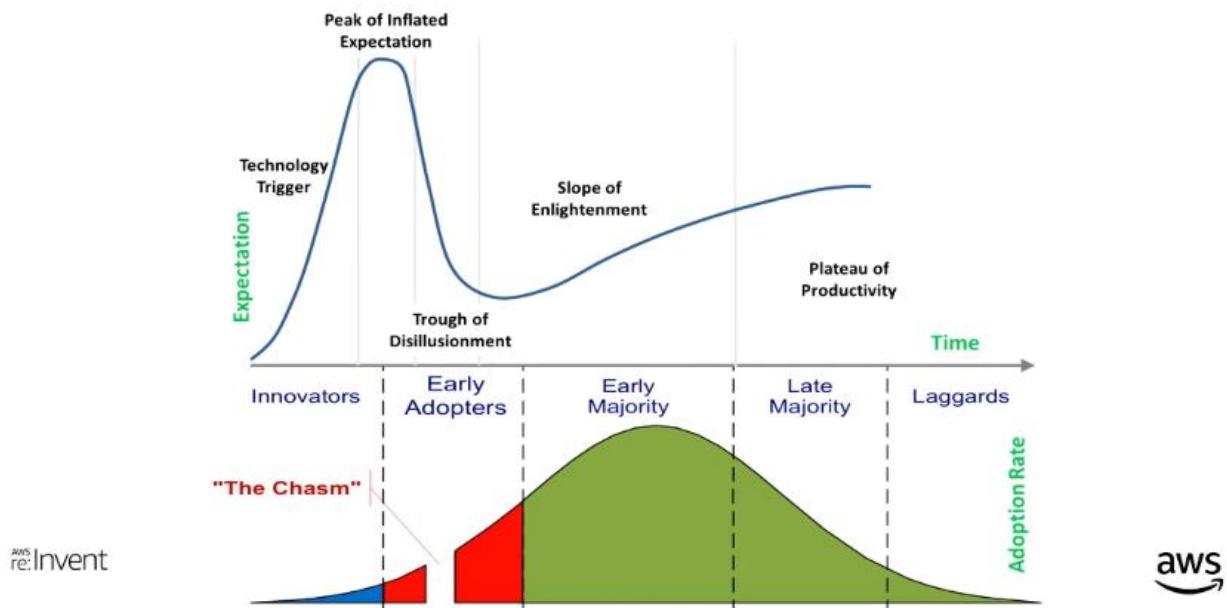
“History repeats itself because nobody was listening the first time”

- Anonymous

Timeline of Database Technology



Technology Adoption and the Hype Curve

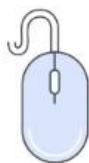


Why NoSQL?

SQL	NoSQL
Optimized for storage	Optimized for compute
Normalized/relational	Denormalized/hierarchical
Ad hoc queries	Instantiated views
Scale vertically	Scale horizontally
Good for OLAP	Built for OLTP at scale

When choosing a database for a system, we need to ask what type of access patterns are we dealing with in our application?

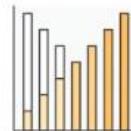
Amazon DynamoDB



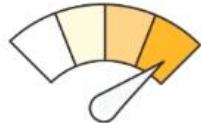
Fully Managed NoSQL



Document or Key-Value



Scales to Any Workload



Fast and Consistent

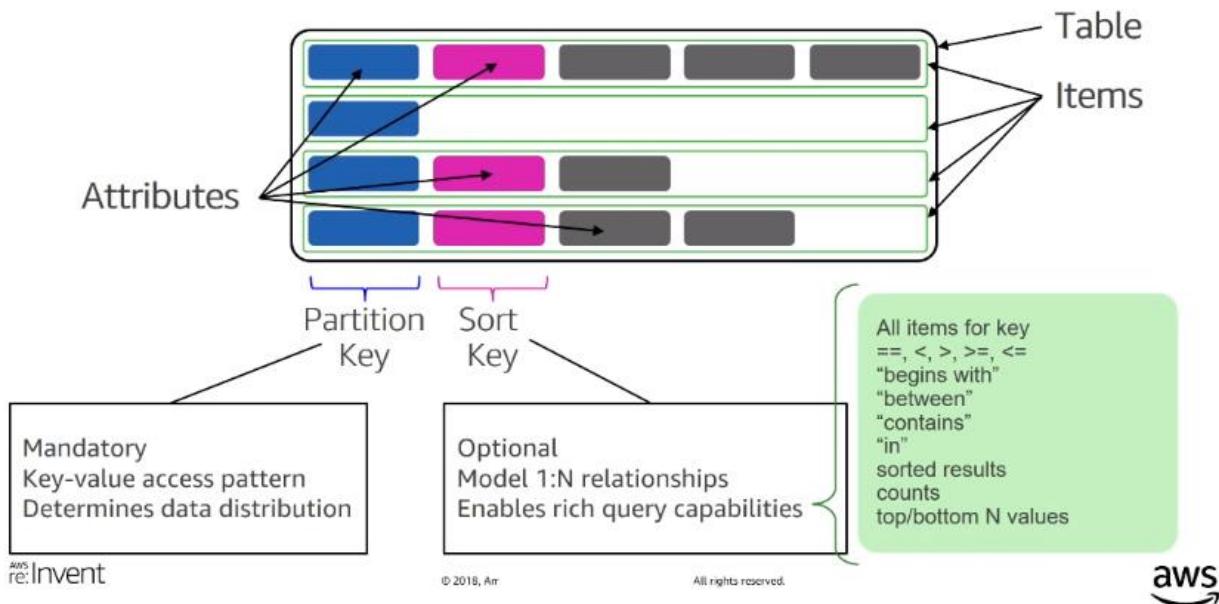


Access Control



Event Driven Programming

Table

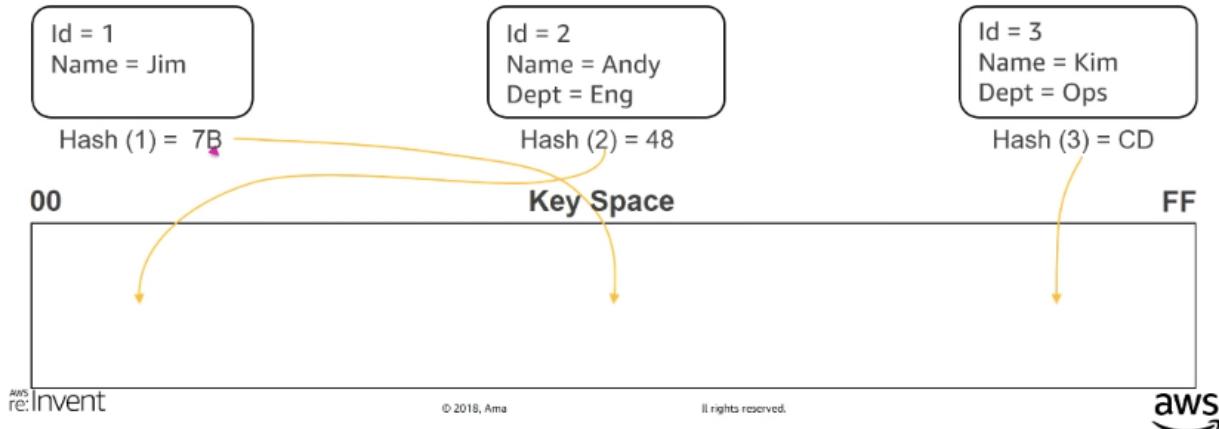


You also need to design the tables to support one of your primary data access patterns

Partition Keys

Partition Key uniquely identifies an item

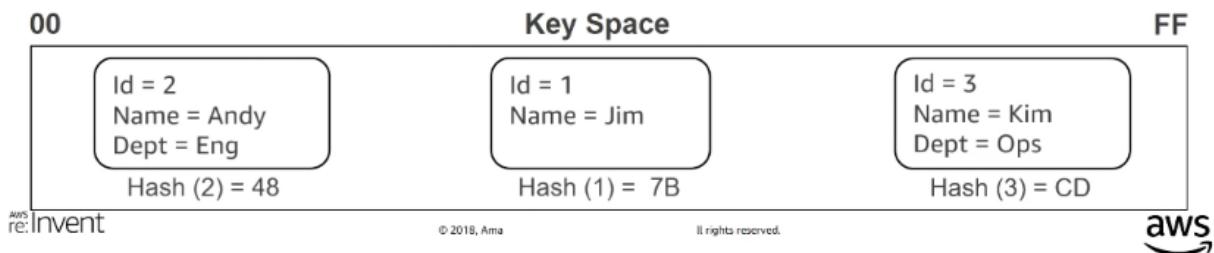
Partition Key is used for building an unordered hash index



Partition Keys

Partition Key uniquely identifies an item

Partition Key is used for building an unordered hash index



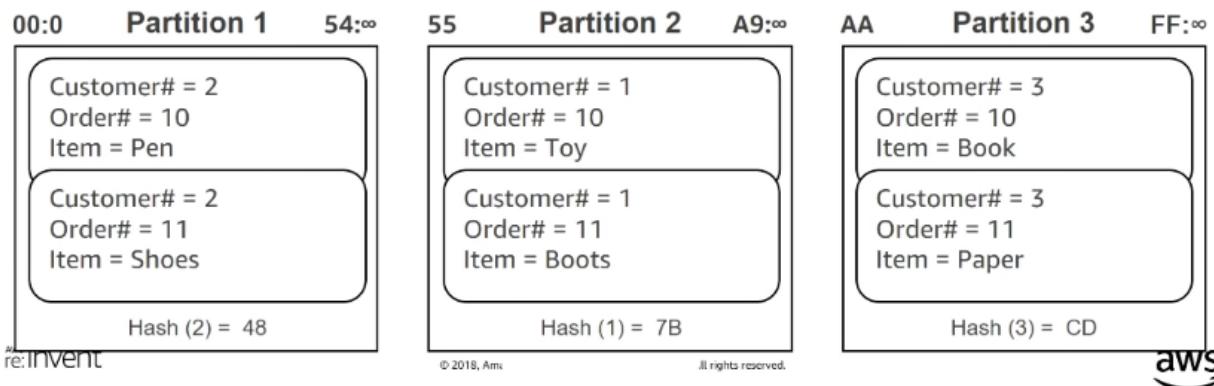
Partition:Sort Key

Partition:Sort Key uses two attributes together to uniquely identify an Item

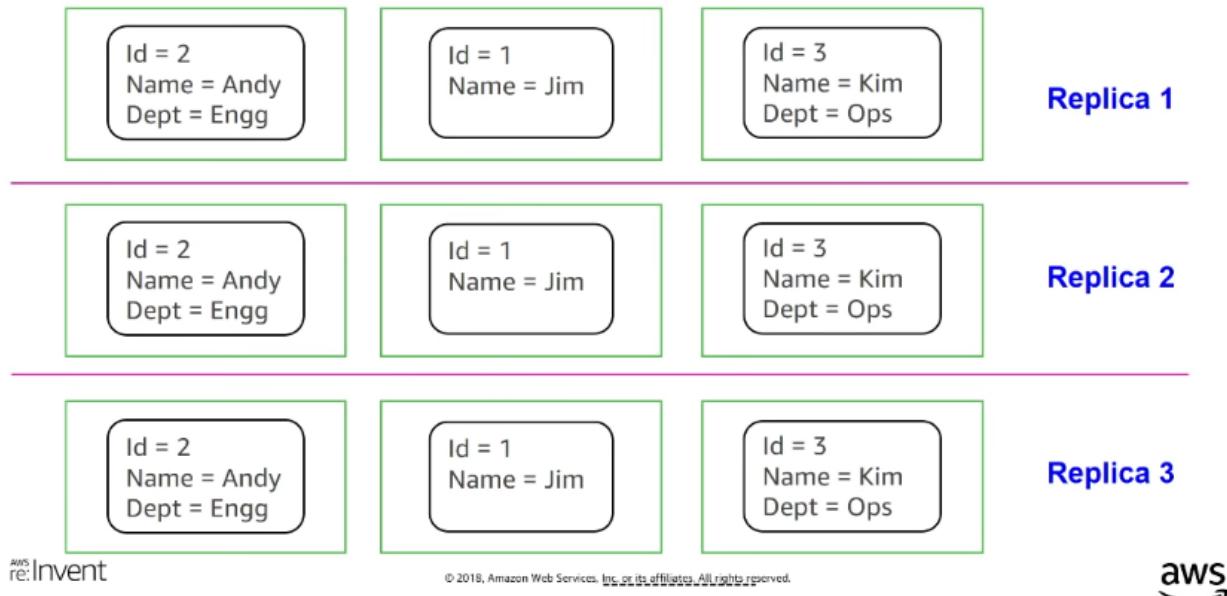
Within unordered hash index, data is arranged by the sort key

No limit on the number of items (∞) per partition key

- Except if you have local secondary indexes



Partitions are three-way replicated



When you read from DynamoDB, you have a choice between eventually consistent and strongly consistent reads. The difference between reading from the primary or the secondary node is a sub-1 millisecond delay, this allows eventually consistent reads to be half the cost of a strongly consistent read because we have more nodes to choose from and randomly read from one of those 3 replica partitions. Making a strongly consistent read means that the data is read from the primary node only since it always accepts the writes. A cheap way to double the capacity of your database/application is to use **ecreads**, the **ecreads** parameter is not on by default in your DynamoDB database.

Local Secondary Index (LSI)

Alternate sort key attribute

Index is local to a partition key

Table	A1 (partition)	A2 (sort)	A3	A4	A5	10 GB max per partition key, i.e. LSIs limit the # of range keys!
LSIs	A1 (partition)	A3 (sort)	A2 (item key)	KEYS_ONLY		
	A1 (partition)	A4 (sort)	A2 (item key)	A3 (projected)	INCLUDE A3	
	A1 (partition)	A5 (sort)	A2 (item key)	A3 (projected)	A4 (projected)	ALL

aws:Invent

aws

LSI allows you to re-sort the data in the partitions. If orderID is the primary key for orders data and the orderDate is the sort key, on the GSI or LSI we then create a partition key using the orderID to use to get data related to the orderID directly from the partition quickly. We can now use the LSI to get all the back ordered items for a customer using the orderID or get all the orders in the last 24 hours using the sort key orderDate less than 24 hours.

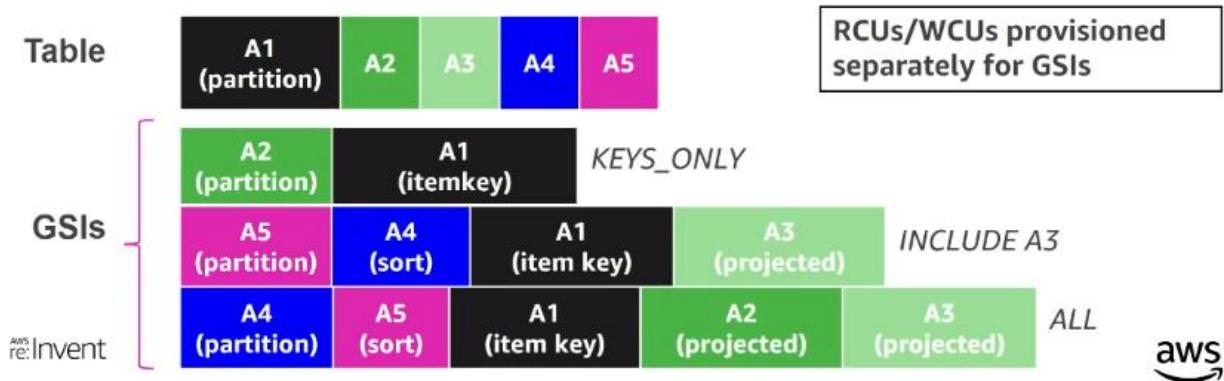
Global Secondary Index (GSI)

Online indexing

Alternate partition and/or sort key

Index is across all partition keys

Use composite sort keys for compound indexes



GSI allows us to create a completely new aggregation of the data, since the primary table is grouping the orders by customerID, then the GSI can group the orders by warehouseID. Then the partition key on the GSI will be the warehouseID and the sort key will be the orderDate. We can then get the latest orders in a warehouse using the warehouseID and the orderDate less than 1 hour. **As we begin to model the data, we will start to use the indexes LSI and GSI to re-group, re-sort, re-aggregate the data to support our secondary access patterns but using the same type of key structures.**

How Do GSI Updates Work?

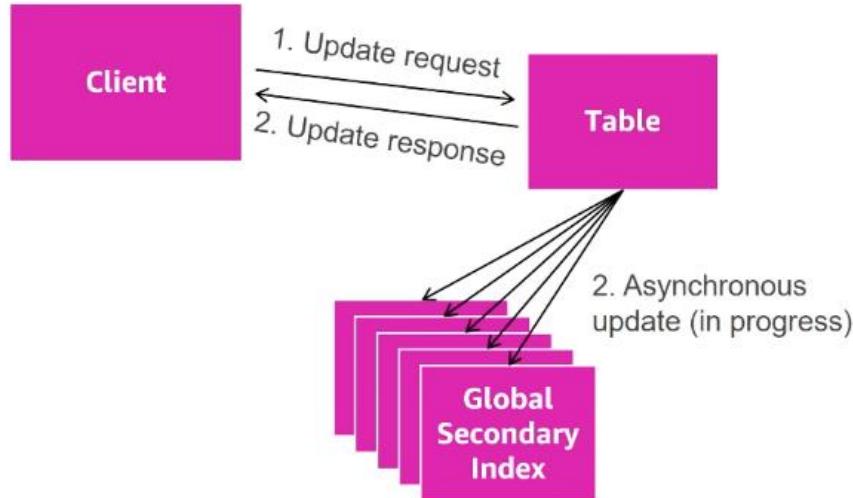


If GSIs don't have enough write capacity, table writes will be throttled!



GSI updates are eventually consistent, LSI updates are strongly consistent.

How Do GSI Updates Work?



If GSIs don't have enough write capacity, table writes will be throttled!

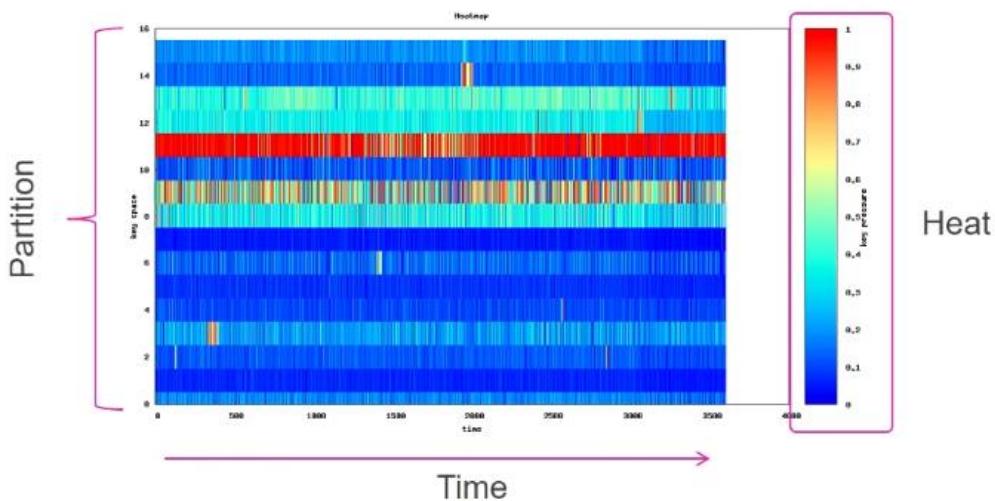


Scaling NoSQL

“We are stuck with technology when what we really want is just stuff that works.”

- Douglas Adams

What bad NoSQL looks like ...



All of our access here is hitting a single storage node within the sharded keyspace. This is an anti-pattern in NoSQL databases. We should be distributing the access patterns to spread out to the data storage nodes or partitions.

Getting the most out of Amazon DynamoDB throughput

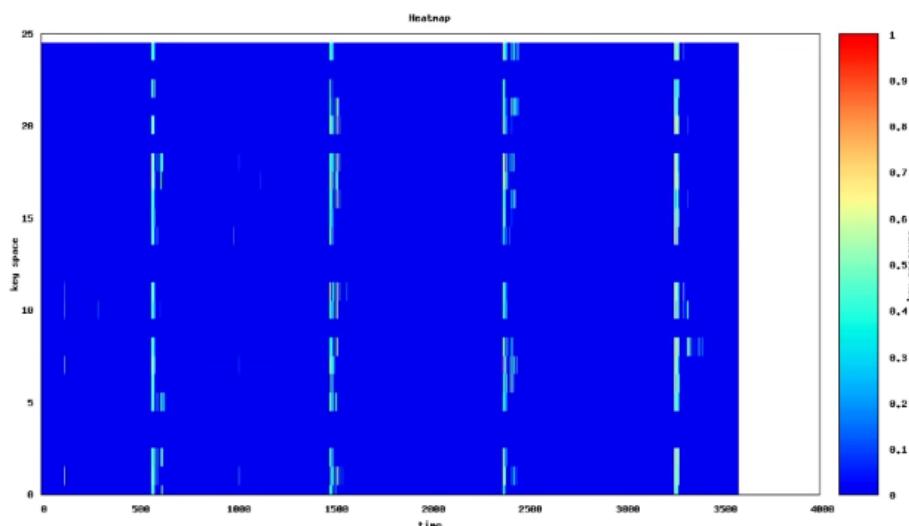
“To get the most out of DynamoDB throughput, create tables where the partition key element has a large number of distinct values, and values are requested fairly uniformly, as randomly as possible.”

—DynamoDB Developer Guide

Space: access is evenly spread over the key-space

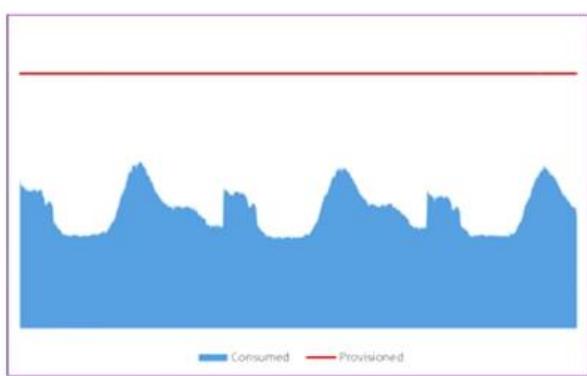
Time: requests arrive evenly spaced in time

Much better picture ...

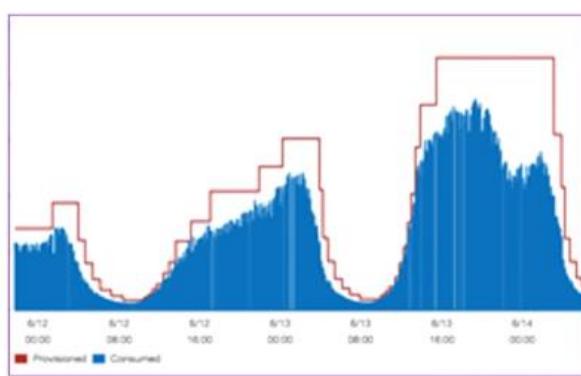


Auto Scaling

Throughput automatically adapts to your actual traffic



Without Auto Scaling



With Auto Scaling

NoSQL Data Modeling

“A ship in port is safe, but that’s not what ships were built for.”

- Grace Hopper

NoSQL data modelling design is to maximize the efficiency of your access patterns

It's all about relationships...



Social network



Document management



Process control



IT monitoring



Data trees

re:Invent

© 2018, Amazon Web Services, Inc. or its affiliates. All rights reserved.

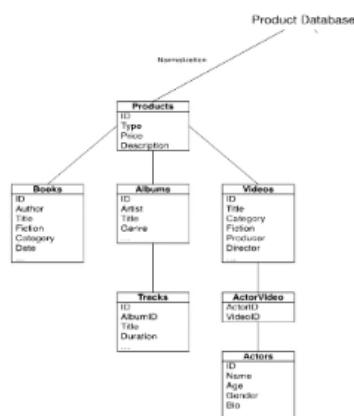


SQL vs. NoSQL design pattern

SELECT * FROM Products
INNER JOIN Books

SELECT * FROM Products
INNER JOIN Albums
INNER JOIN Tracks

SELECT * FROM Products
INNER JOIN Videos
INNER JOIN ActorVideo
INNER JOIN Actors



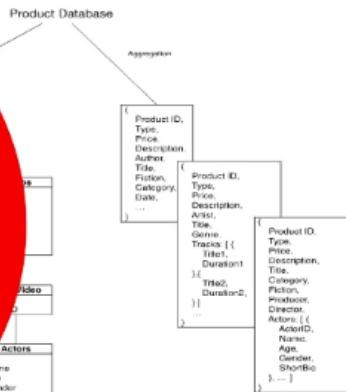
We used normalized modelling approach in the past with relational databases using complex queries and joins. The data ends up living in different tables and we need ACID transactions to make updates later

SQL vs. NoSQL design pattern

SELECT * FROM Products
INNER JOIN Books

SELECT * FROM Products
INNER JOIN Albums
INNER JOIN Tracks

SELECT * FROM Products
INNER JOIN Videos
INNER JOIN ActorVideo
INNER JOIN Actors



SELECT * FROM PRODUCTS

AWS re:Invent

© 2018, Amazon Web Services, Inc. or its affiliates. All rights reserved.



But we can use NoSQL databases with hierarchical data models and create simpler queries to get back the data.

AMAZON DYNAMODB - KEY CONCEPTS

Selecting a Partition Key

- Large number of distinct values
- Items are uniformly requested and randomly distributed

Examples:

- Bad: Status, Gender
- Good: CustomerId, DeviceId

Selecting a Sort Key

- Model 1:n and n:n relationships
- Efficient/selective patterns
 - Query multiple entities
- Leverage range queries

Examples:

- Orders and OrderItems
- Hierarchical relationships

TENETS OF NoSQL DATA MODELING

- Understand the use case
- Identify the access patterns
 - Read/Write workloads
 - Query dimensions and aggregations
- Data-modeling
 - Avoid relational design patterns, use one table
- Review -> Repeat -> Review

TENETS OF NoSQL DATA MODELING

- Understand the use case
- Identify the access patterns
 - Read/Write workloads
 - Query dimensions and aggregations
- Data-modeling
 - Avoid relational design patterns, use one table
- Review -> Repeat -> Review
- Nature of the application
 - OLTP / OLAP / DSS
- Define the Entity-Relationship Model
- Identify Data Life Cycle
 - TTL, Backup/Archival, etc.

TENETS OF NoSQL DATA MODELING

- Understand the use case
- Define the access patterns
 - Read/Write workloads
- Data-modeling
 - Avoid relational design patterns, use one table
- Review -> Repeat -> Review
- Identify data sources
- Define query aggregations
- Document all workflows

TENETS OF NoSQL DATA MODELING

- Understand the use case
- Identify the access patterns
 - Read/Write workloads
 - Query dimensions and aggregations
- Data-modeling
 - Avoid relational design patterns, use one table
- Review -> Repeat -> Review
- **1 application service = 1 table**
 - Reduce round trips
 - Simplify access patterns
- Identify Primary Keys
 - How will items be inserted and read?
 - Overload items into partitions
- Define indexes for secondary access patterns

TENETS OF NoSQL DATA MODELING

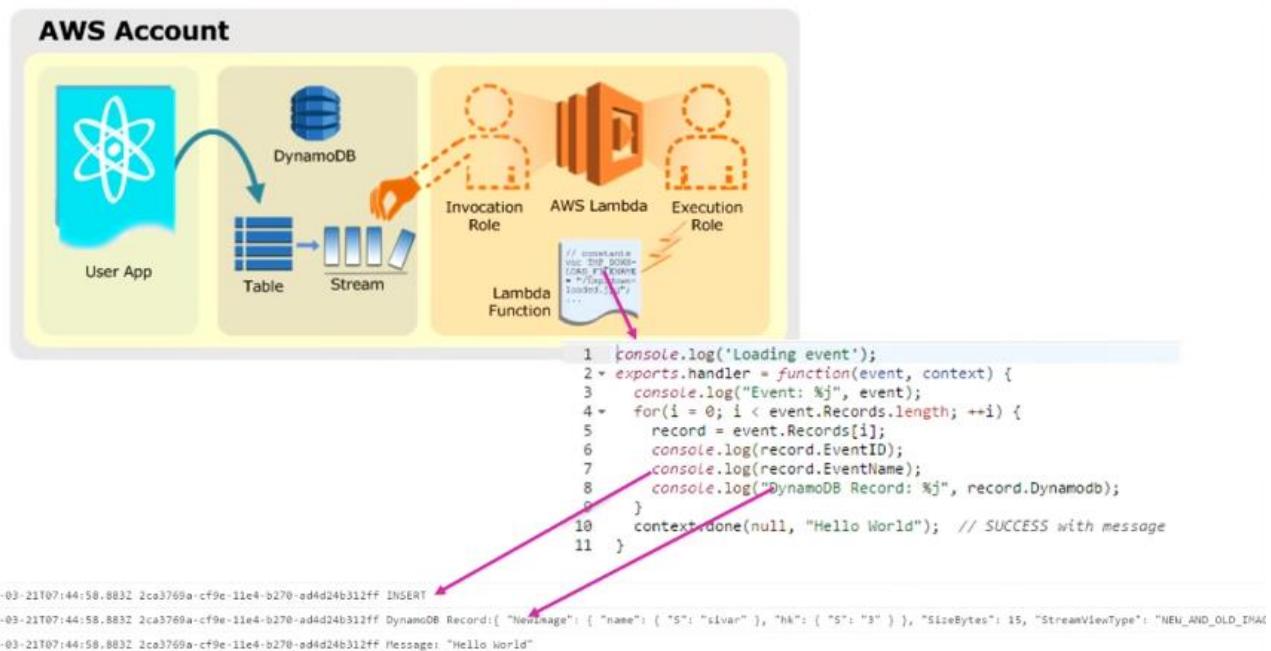
- Understand the use case
- Identify the access patterns
 - Read/Write workloads
 - Query dimensions and aggregations
- Data-modeling
 - Avoid relational design patterns, use one table
- Review -> Repeat-> Review

Complex Queries

“Computers are useless. They can only give you answers.”

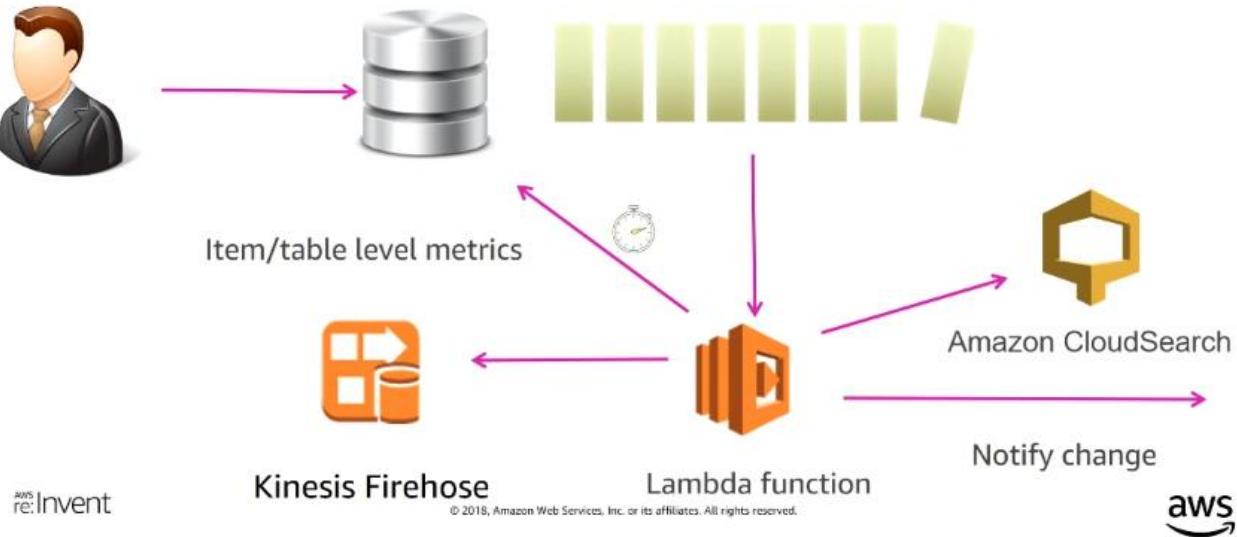
- Pablo Picasso

DynamoDB Streams and AWS Lambda



Using DynamoDB Streams (the change log of the DynamoDB tables) and we can kick off Lambdas on the events we want as they come through the stream) and Lambda is like the best stored procedure engines available because they decouple your implementation from the databases

Triggers



Using this pattern for stored procedure type problems, we can do computed running aggregations like counts and sums in DynamoDB Streams with Lambda, then write the metric like count back to the table as a metadata item.

Composite Keys

“Hierarchies are celestial. In hell all are equal.”

- Nicolás Gómez Dávila

Multi-value Sorts and Filters

The diagram shows a composite key structure. It consists of a primary key labeled 'Partition key' and a secondary key labeled 'Sort key'. The 'Sort key' is further divided into a 'Secondary index' and a 'Primary key'. Below this structure is a table with columns: Opponent, Date, GameId, Status, and Host. The table contains five rows of data. To the right of the table is a user icon labeled 'Bob'.

Partition key

Sort key

Secondary index

Opponent	Date	GameId	Status	Host
Alice	2014-10-02	d9bl3	DONE	David
Carol	2014-10-08	o2pnb	IN_PROGRESS	Bob
Bob	2014-09-30	72f49	PENDING	Alice
Bob	2014-10-03	b932s	PENDING	Carol
Bob	2014-10-03	ef9ca	IN_PROGRESS	David

aws re:Invent

aws

We can store hierarchical data using the primary and secondary indexes for our range queries

Approach 1: Query Filter



This is not good because you still get to pay for the full query that happens before the filtering

Approach 2: Composite Key

Status	Date	StatusDate
DONE	2014-10-02	DONE_2014-10-02
IN_PROGRESS	2014-10-08	IN_PROGRESS_2014-10-08
IN_PROGRESS	2014-10-03	IN_PROGRESS_2014-10-03
PENDING	2014-10-03	PENDING_2014-09-30
PENDING	2014-09-30	PENDING_2014-10-03

We can use this approach of creating and using a Composite key.

Approach 2: Composite Key

Partition key Sort key

Secondary Index

Opponent	StatusDate	GameId	Host
Alice	DONE_2014-10-02	d9bl3	David
Carol	IN_PROGRESS_2014-10-08	o2pnb	Bob
Bob	IN_PROGRESS_2014-10-03	ef9ca	David
Bob	PENDING_2014-09-30	72f49	Alice
Bob	PENDING_2014-10-03	b932s	Carol

re:Invent © 2018, Amazon Web Services, Inc. or its affiliates. All rights reserved. aws

Approach 2: Composite Key

```
SELECT * FROM Game  
WHERE Opponent='Bob'  
AND StatusDate BEGINS_WITH 'PENDING'
```



Secondary index

Opponent	StatusDate	GameId	Host
Alice	DONE_2014-10-02	d9bl3	David
Carol	IN_PROGRESS_2014-10-08	o2pnb	Bob
Bob	IN_PROGRESS_2014-10-03	ef9ca	David
Bob	PENDING_2014-09-30	72f49	Alice
Bob	PENDING_2014-10-03	b932s	Carol

re:Invent © 2018, Amazon Web Services, Inc. or its affiliates. All rights reserved. aws

This gives us a selective read querying that will cost less.

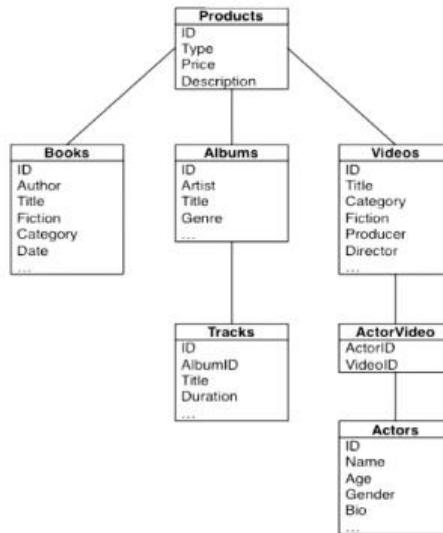
Advanced Data Modeling

“The great myth of our times is that technology is communication.”

- Libby Larsen

How OLTP Apps Use Data

- Mostly hierarchical structures
- Entity driven workflows
- Data spread across tables
- Requires complex queries
- Primary driver for ACID



Maintaining Version History

A diagram illustrating item versioning in Amazon DynamoDB. On the left is a table structure:

ItemID (PK)	Version (SK)	CurVer	Attrs
1	v0	2	...
	v1		...
	v2		...
	v3		...

(Many more item partitions)

On the right, a yellow box labeled 'Transaction' contains the following SQL-like code:

```
COPY Item.v0 -> Item1.v3 IF Item.v3 == NULL  
UPDATE Item1.v3 SET Attr1 += 1  
UPDATE Item1.v3 SET Attr2 = ...  
UPDATE Item1.v3 SET Attr3 = ...  
COPY Item1.v3 -> Item1.v0 SET CurVer = 3
```

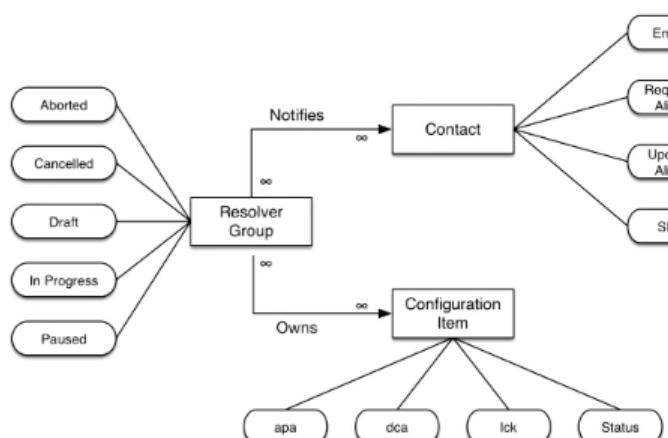
A pink bracket on the right side of the table is labeled 'Item versions'. A pink arrow points from the 'v3' row to the code above, labeled 'Overwrite v0 Item to Commit changes'.

aws re:Invent

© 2018, Amazon Web Services, Inc. or its affiliates. All rights reserved.

aws

Managing Relational Transactions



- Configuration Management Service
 - Resolver Groups
 - Contacts
 - Configuration Items
- Transactional Workflows
 - Add Config Items to Resolver Groups
 - Update Config Item status
 - Add Contacts to Resolver Groups

DynamoDB Transactions API

- TransactWriteItems
 - Synchronous update, put, delete, and check
 - Atomic
 - Automated Rollbacks
 - Up to 10 items within a transaction
 - Supports multiple tables
 - Complex conditional checks
- Good Use Cases
 - Commit changes across items
 - Conditional batch inserts/updates
- Bad Use Case
 - Maintaining normalized data



DynamoDB Table Schema

CM Data Reverse							
Partition	Sort	Attributes					
contact_1	resolver_1	team_email	requestor_alias	update_alias	onboard_sim		
	resolver_2	tauster0@imageshack.us	boernisrou0	nmoresht0	https://columba.acus.acus.png		
	resolver_1	team_email	requestor_alias	update_alias	onboard_sim		
	resolver_2	tauster0@imageshack.us	boernisrou0	nmoresht0	https://columba.acus.acus.png		
resolver_1	resolver_1	team_email	requestor_alias	update_alias	onboard_sim		
	resolver_2	dbourhoum6@medafra.com	cthy5	sjupmng5	https://fun.net/justo.jpg		
	resolver_1	team_email	requestor_alias	update_alias	onboard_sim		
	resolver_2	dbourhoum6@medafra.com	cthy5	sjupmng5	https://fun.net/justo.jpg		
resolver_1	1cc73e38-39ec-42cd-8392-ed8154834625	lastStatus	lck	dca	apa		
	2bc022c2-30bb-49b3-a3c1b78d48ed7f	completed	—		
	metadata	lastStatus	lck	dca	apa		
	90e368c4-6de2-41e5-822d-1c8be89ec50c	cancelled	—		
resolver_2	1cc73e38-39ec-42cd-8392-ed8154834625	aborted	cancelled	completed	draft	in_progress	paused
	90e368c4-6de2-41e5-822d-1c8be89ec50c	true	true	false	false	true	false
	metadata	lastStatus	lck	dca	apa		
	90e368c4-6de2-41e5-822d-1c8be89ec50c	in_progress	—		
Resolver Partitions	resolver_1	aborted	cancelled	completed	draft	in_progress	paused
	resolver_2	true	true	false	true	false	false

DynamoDB Table Schema

De-normalized Contacts

CM Data Reverse		Attributes					
Partition	Primary Key	Sort	team_email	requestor_alias	update_alias	onboard_sm	
Contact Partitions	resolver_1		tailey@imagestack.us	bcmresour0	nmresr0	https://columbia.edu/acus.png	
	resolver_2		team_email	requestor_alias	update_alias	onboard_sm	
	resolver_1		tailey@imagestack.us	bcmresour0	nmresr0	https://columbia.edu/acus.png	
	resolver_2		team_email	requestor_alias	update_alias	onboard_sm	
Resolver Partitions	resolver_1	1cc73e38-39ec-42cd-8392-ec8154834625	dbourloumco@medafro.com	cthy6	spurming6	https://fur.net/justo.jpg	
	resolver_2	2b0c22c2-30bb-49b3-a3c0-b78d48e6d7f	team_email	requestor_alias	update_alias	onboard_sm	
	resolver_1	1cc73e38-39ec-42cd-8392-ec8154834625	lastStatus	lock	dca	ape	
	resolver_2	90e30Bc4-6de2-41e5-822d-1c8be89ec50c	completed	—	
	resolver_1	1cc73e38-39ec-42cd-8392-ec8154834625	lastStatus	lock	dca	ape	
	resolver_2	90e30Bc4-6de2-41e5-822d-1c8be89ec50c	cancelled	—	
	resolver_1	1cc73e38-39ec-42cd-8392-ec8154834625	aborted	cancelled	completed	draft	in_progress paused
	resolver_2	90e30Bc4-6de2-41e5-822d-1c8be89ec50c	metadata	true	true	false	true false false

DynamoDB Table Schema

De-normalized Contacts

CM Data Reverse		Attributes					
Partition	Primary Key	Sort	team_email	requestor_alias	update_alias	onboard_sm	
Contact Partitions	resolver_1		tailey@imagestack.us	bcmresour0	nmresr0	https://columbia.edu/acus.png	
	resolver_2		team_email	requestor_alias	update_alias	onboard_sm	
	resolver_1		tailey@imagestack.us	bcmresour0	nmresr0	https://columbia.edu/acus.png	
	resolver_2		team_email	requestor_alias	update_alias	onboard_sm	
Resolver Partitions	resolver_1	1cc73e38-39ec-42cd-8392-ec8154834625	dbourloumco@medafro.com	cthy6	spurming6	https://fur.net/justo.jpg	
	resolver_2	2b0c22c2-30bb-49b3-a3c0-b78d48e6d7f	team_email	requestor_alias	update_alias	onboard_sm	
	resolver_1	1cc73e38-39ec-42cd-8392-ec8154834625	lastStatus	lock	dca	ape	
	resolver_2	90e30Bc4-6de2-41e5-822d-1c8be89ec50c	completed	—	
	resolver_1	1cc73e38-39ec-42cd-8392-ec8154834625	lastStatus	lock	dca	ape	
	resolver_2	90e30Bc4-6de2-41e5-822d-1c8be89ec50c	cancelled	—	
	resolver_1	1cc73e38-39ec-42cd-8392-ec8154834625	aborted	cancelled	completed	draft	in_progress paused
	resolver_2	90e30Bc4-6de2-41e5-822d-1c8be89ec50c	metadata	true	true	false	true false false

DynamoDB Table Schema

The diagram illustrates the de-normalized structure of contacts across six partitions:

- CM Data**: Contains primary key information.
- Reverse**: Contains attributes like team_email, requestor_alias, update_alias, onboard_slim, and URLs.
- Contact Partitions**: Contains contact_1 and contact_2.
- Configuration Items**: Contains configuration items.
- Resolver Partitions**: Contains resolver_1, resolver_2, and resolver_3.
- Resolver Metadata**: Contains metadata for each resolver partition.

Relationships are shown by lines connecting specific fields from the CM Data and Reverse partitions to their corresponding values in the other partitions. For example, the 'team_email' attribute in the Reverse partition connects to specific email addresses in the Contact Partitions and Configuration Items.

CM Data		Reverse	De-normalized Contacts					
Partition	Sort	Primary Key	Attributes					
			team_email	requestor_alias	update_alias	onboard_slim	URL	URL
contact_1	resolver_1	tauser@magistrack.us	bcenr10n0	nmcnra0c	https://columia.edu/eius.png			
	resolver_2	tauser@magistrack.us	bcenr10n0	nmcnra0c	https://columia.edu/eius.png			
contact_2	resolver_1	team_email	requestor_alias	update_alias	onboard_slim			
	resolver_2	team_email	requestor_alias	update_alias	onboard_slim			
	resolver_1	dbourlouimco@modifro.com	cthy5	spouhng5	https://ur.net/etc.jpg			
	resolver_2	dbourlouimco@modifro.com	cthy5	spouhng5	https://ur.net/etc.jpg			
	resolver_1	lastStatus	lock	dca	apa			
	resolver_2	lastStatus	lock	dca	apa			
	resolver_1	completed	—	—	—			
	resolver_2	completed	—	—	—			
	resolver_1	lastStatus	lock	dca	apa			
	resolver_2	lastStatus	lock	dca	apa			
	resolver_1	cancelled	—	—	—			
	resolver_2	cancelled	—	—	—			
	resolver_1	aborted	cancelled	completed	draft	in_progress	paused	
	resolver_2	aborted	cancelled	completed	draft	in_progress	paused	
	resolver_1	true	true	false	false	true	false	
	resolver_2	true	true	false	false	true	false	
	resolver_1	lastStatus	lock	dca	apa			
	resolver_2	lastStatus	lock	dca	apa			
	resolver_1	completed	—	—	—			
	resolver_2	completed	—	—	—			
	resolver_1	lastStatus	lock	dca	apa			
	resolver_2	lastStatus	lock	dca	apa			
	resolver_1	in_progress	—	—	—			
	resolver_2	in_progress	—	—	—			
	resolver_1	aborted	cancelled	completed	draft	in_progress	paused	
	resolver_2	aborted	cancelled	completed	draft	in_progress	paused	
	resolver_1	true	true	false	true	false	false	
	resolver_2	true	true	false	true	false	false	
	resolver_1	metadata	—	—	—			
	resolver_2	metadata	—	—	—			

We can also add configuration items into those resolver groups when needed.

DynamoDB Table Schema

For transaction updates, we can execute inserts to both of those partitions and guarantee if they will succeed or not.

DynamoDB Table Schema

Contact Partitions

Configuration Items

Resolver Partitions

Resolver Metadata

CM Data Reverse		De-normalized Contacts			
Partition	Sort	Attributes			
resolver_1	team_email	requestor_alias	update_alias	onboard_slm	
contact_1	tauser0@imageshack.us	boenriscou0	nmcreath0	https://columbia.edu/eacus.png	
resolver_2	team_email	requestor_alias	update_alias	onboard_slm	
resolver_1	tauser0@imageshack.us	boenriscou0	nmcreath0	https://columbia.edu/eacus.png	
contact_2	team_email	requestor_alias	update_alias	onboard_slm	
resolver_2	dbourthouim6@mediasite.com	cthy6	spurnig6	https://fun.net/justo.jpg	
resolver_1	dbourthouim6@mediasite.com	cthy6	spurnig6	https://fun.net/justo.jpg	
resolver_1	lastStatus	lock	dca	ape	
resolver_2	lastStatus	lock	dca	ape	
resolver_1	completed	
resolver_2	cancelled	
resolver_1	aborted	cancelled	completed	draft	in_progress paused
resolver_2	true	true	false	false	true false
resolver_1	lastStatus	lock	dca	ape	
resolver_2	completed	
resolver_1	lastStatus	lock	dca	ape	
resolver_2	in_progress	
resolver_1	aborted	cancelled	completed	draft	in_progress paused
resolver_2	true	true	false	true	false false
resolver_1	lastStatus	lock	dca	ape	
resolver_2	completed	
resolver_1	lastStatus	lock	dca	ape	
resolver_2	in_progress	
resolver_1	aborted	cancelled	completed	draft	in_progress paused
resolver_2	true	true	false	true	false false
resolver_1	lastStatus	lock	dca	ape	
resolver_2	completed	
resolver_1	lastStatus	lock	dca	ape	
resolver_2	in_progress	
resolver_1	aborted	cancelled	completed	draft	in_progress paused
resolver_2	true	true	false	true	false false

We can also update the transaction status for those transaction updates.

DynamoDB Table Schema

Contact Partitions

Configuration Items

Resolver Partitions

Resolver Metadata

CM Data Reverse		De-normalized Contacts			
Partition	Sort	Attributes			
resolver_1	team_email	requestor_alias	update_alias	onboard_slm	
contact_1	tauser0@imageshack.us	boenriscou0	nmcreath0	https://columbia.edu/eacus.png	
resolver_2	team_email	requestor_alias	update_alias	onboard_slm	
resolver_1	tauser0@imageshack.us	boenriscou0	nmcreath0	https://columbia.edu/eacus.png	
contact_2	team_email	requestor_alias	update_alias	onboard_slm	
resolver_2	dbourthouim6@mediasite.com	cthy6	spurnig6	https://fun.net/justo.jpg	
resolver_1	dbourthouim6@mediasite.com	cthy6	spurnig6	https://fun.net/justo.jpg	
resolver_1	lastStatus	lock	dca	ape	
resolver_2	completed	
resolver_1	cancelled	
resolver_1	aborted	cancelled	completed	draft	in_progress paused
resolver_2	true	true	false	false	true false
resolver_1	lastStatus	lock	dca	ape	
resolver_2	completed	
resolver_1	lastStatus	lock	dca	ape	
resolver_2	in_progress	
resolver_1	aborted	cancelled	completed	draft	in_progress paused
resolver_2	true	true	false	true	false false
resolver_1	lastStatus	lock	dca	ape	
resolver_2	completed	
resolver_1	lastStatus	lock	dca	ape	
resolver_2	in_progress	
resolver_1	aborted	cancelled	completed	draft	in_progress paused
resolver_2	true	true	false	true	false false
resolver_1	lastStatus	lock	dca	ape	
resolver_2	completed	
resolver_1	lastStatus	lock	dca	ape	
resolver_2	in_progress	
resolver_1	aborted	cancelled	completed	draft	in_progress paused
resolver_2	true	true	false	true	false false

We can also update contact emails across a transaction group.

Reverse Lookup GSI

CM Data		Reverse							
Primary Key		Sort	Partition	Attributes					
resolver_1	contact_1			team_email	requestor_alias	update_alias	onboard_slim		
				tauster0@imageshack.us	bconnistoun0	rmoreath0	https://columbia.edu/facus.org		
resolver_2	contact_1			team_email	requestor_alias	update_alias	onboard_slim		
				dbourthouloum5@mediasfire.com	cthy5	sjouhing5	https://url.net/justo.pg		
metadata	contact_1			team_email	requestor_alias	update_alias	onboard_slim		
				tauster0@imageshack.us	bconnistoun0	rmoreath0	https://columbia.edu/facus.org		
resolver_1	aborted			team_email	requestor_alias	update_alias	onboard_slim		
	true			tauster0@imageshack.us	bconnistoun0	rmoreath0	https://columbia.edu/facus.org	in_progress	paused
resolver_2	aborted			team_email	requestor_alias	update_alias	onboard_slim		
	true			dbourthouloum5@mediasfire.com	cthy5	sjouhing5	https://url.net/justo.pg	in_progress	paused
resolver_1	lastStatus			lock	dca	dpa			
	completed					
resolver_2	lastStatus			lock	dca	dpa			
	completed					
resolver_1	lastStatus			lock	dca	dpa			
	cancelled					
resolver_2	lastStatus			lock	dca	dpa			
	in_progress					

re:Invent

© 2018, Amazon Web Services, Inc. or its affiliates. All rights reserved.



We can add one GSI to do a reverse lookup by reading the partition

Reverse Lookup GSI

CM Data		Reverse							
Primary Key		Sort	Partition	Attributes					
resolver_1	contact_1			team_email	requestor_alias	update_alias	onboard_slim		
				tauster0@imageshack.us	bconnistoun0	rmoreath0	https://columbia.edu/facus.org		
resolver_2	contact_1			team_email	requestor_alias	update_alias	onboard_slim		
				tauster0@imageshack.us	bconnistoun0	rmoreath0	https://columbia.edu/facus.org		
metadata	contact_1			team_email	requestor_alias	update_alias	onboard_slim		
				dbourthouloum5@mediasfire.com	cthy5	sjouhing5	https://url.net/justo.pg		
resolver_1	aborted			team_email	requestor_alias	update_alias	onboard_slim		
	true			tauster0@imageshack.us	bconnistoun0	rmoreath0	https://columbia.edu/facus.org	in_progress	paused
resolver_2	aborted			team_email	requestor_alias	update_alias	onboard_slim		
	true			dbourthouloum5@mediasfire.com	cthy5	sjouhing5	https://url.net/justo.pg	in_progress	paused
resolver_1	lastStatus			lock	dca	dpa			
	completed					
resolver_2	lastStatus			lock	dca	dpa			
	completed					
resolver_1	lastStatus			lock	dca	dpa			
	cancelled					
resolver_2	lastStatus			lock	dca	dpa			
	in_progress					

re:Invent

© 2018, Amazon Web Services, Inc. or its affiliates. All rights reserved.

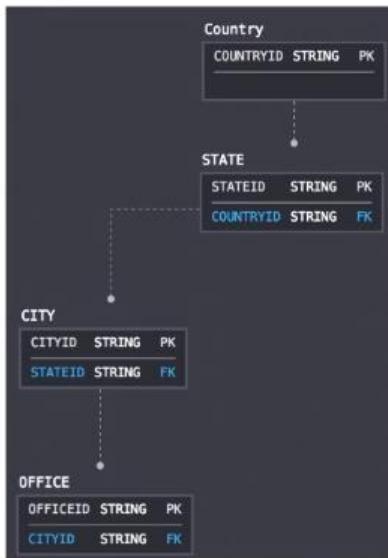


Or we can also do a lookup using the sort key

Hierarchical Data

Joins? We don't need no stinkin' joins!

Hierarchical Data Structures as Items



- Use composite sort key to define a hierarchy
- Highly selective queries with sort conditions
- Reduce query complexity

Primary Key		Attributes		
COUNTRY	LOCATION	Address	EmployeeCount	BuildingManager
USA	NY#NYC#JFK11			
	NY#NYC#JFK14	Address	EmployeeCount	BuildingManager
	WA#SEA#BLACKFOOT	Address	EmployeeCount	BuildingManager
	WA#SEA#KUMO	Address	EmployeeCount	BuildingManager
	WA#SEA#MAYDAY	Address	EmployeeCount	BuildingManager

© 2017, Amazon Web Services, Inc. or its Affiliates. All rights reserved.



This model allows us to search for an office location. This allows us to use a single table to support multiple access patterns using sort keys and composite keys.

Modeling Relational Data

Dude, where's my Lookup Table?

Modeling a Delivery Service – GetMeThat!

The Business Model:

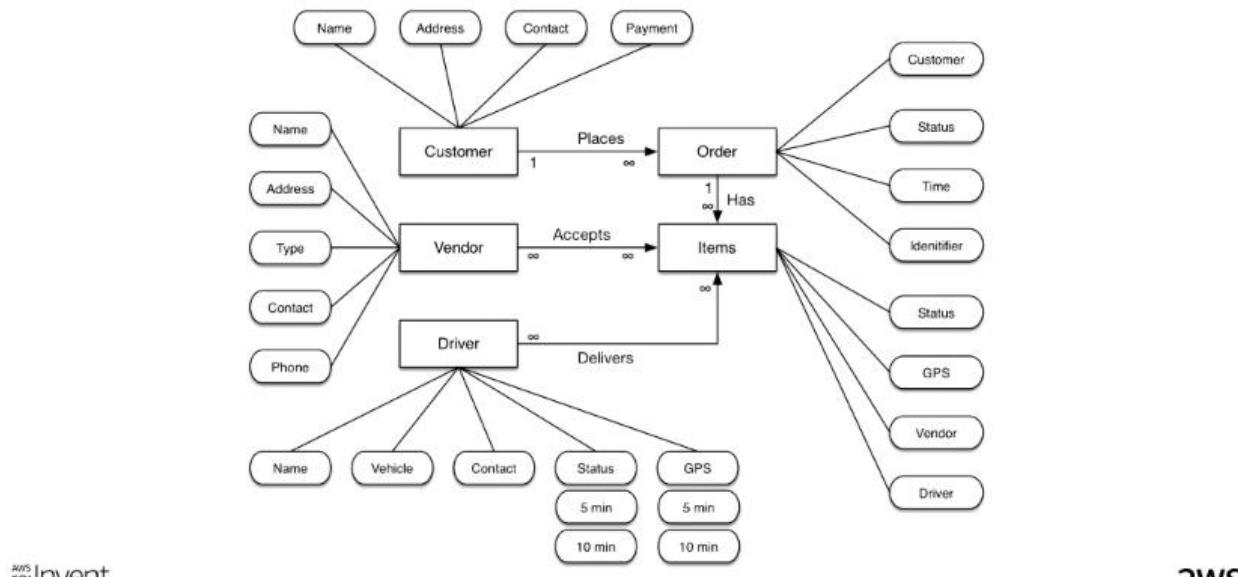
Customers want things, we get them things.

How it works:

People are busy, they need stuff. GetMeThat! lets users create lists of stuff they need and have it dropped where and when they need it.

- Download GetMeThat!
- Browse Stuff
- Get stuff
- Tell us where to put your stuff

The Entity Model



The Access Patterns

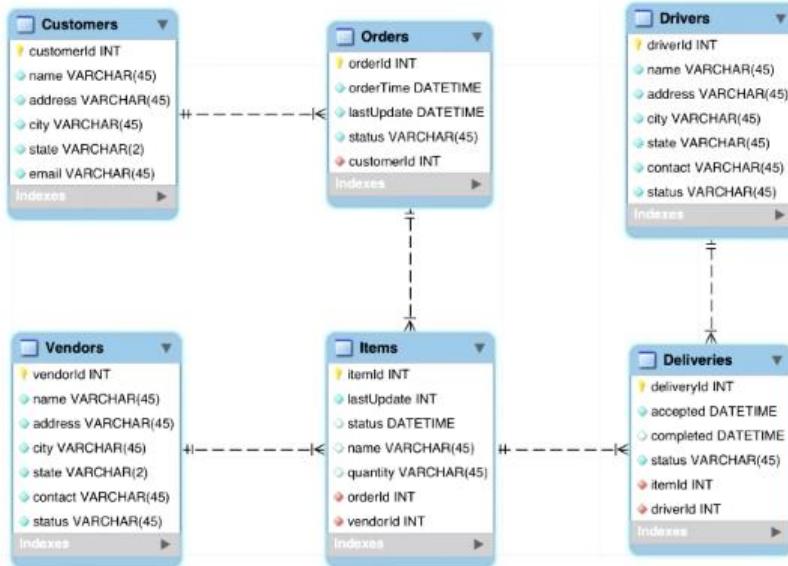
- Get Order(s)
 - by date range
 - by Customer
 - by Vendor
- Get Order Details
 - Status
 - Items
- Get Deliveries
 - by date range
 - by Driver
- Get available drivers
 - by Geo and Time
- Get Customer Data
- Get Vendor Data



There are about 10 – 12 different access patterns that needs to be supported for this application

The Relational Approach

re:Invent



The NoSQL Approach

re:Invent



Table	Primary Key		Attributes						
	PK	SK	name	status	update	GSI1PK	GSI1SK	GSI2PK	GSI2SK
kpakes@Samsung.com	2018-11-01T12:10:25Z	customer	Delivered	2018-11-01T12:45:10Z	order_1		kpakes@Samsung.com	brchys-eu1	2018-11-01T12:10:25Z
		full_name	address	city	state		email		
lmanchester@qpm.jp	2018-11-01T12:03:37Z	customer	Kelly Fisher	657 La Follette Center	Austin	Ready	kpakes@Samsung.com		
		full_name	address	city	state	update	GSI1PK	GSI1SK	GSI2PK
brchys-aus1	2018-11-01T12:03:37Z	vendor	Umojo Manchester	90 Aranicas Crossing	Austin	Texas	lmanchester@qpm.jp	satlick-aus1	2018-11-01T12:03:37Z
		vendorName	address	city	state		username	satlick-aus1	
satlick-aus1	2018-11-01T12:03:37Z	vendor	Tochya Tacos	7621 Usain Drive	Austin	Texas	brchys-aus1	satlick-aus1	2018-11-01T12:03:37Z
		vendorName	address	city	state		username	satlick-aus1	
csommerton@mgur.com	2018-11-01T12:10:25Z	driver	Seti Lok	7750 Dennis Avenue	Austin	Texas	satlick-aus1	seti Lok	2018-11-01T12:10:25Z
		name	address	city	state		contact	status	driver
lgross1@google.com.br	2018-11-01T12:10:25Z	driver	Comai Somerton	3603 Chinese Avenue	Austin	Active	csommerton@mgur.com	lgross1@google.com.br	2018-11-01T12:10:25Z
		name	status	update	GSI1PK	GSI1SK			
lgross1@google.com.br	2018-11-01T12:10:25Z	gpa	Available	Assigned	9:45	Assigned			
		gpa	status	update	GSI1PK	GSI1SK			
lgross1@google.com.br	2018-11-01T12:10:25Z	gpa_05	Available	Assigned	9:45	Assigned	gpa_05		
		gpa	status	update	GSI1PK	GSI1SK			
lgross1@google.com.br	2018-11-01T12:10:25Z	gpa_10	Available	Available	9:45	Available 10min			
		gpa	status	update	GSI1PK	GSI1SK			
lgross1@google.com.br	2018-11-01T12:10:25Z	driver	Felipe Gross	7630 Lulu View Hill	Austin	Active	lgross1@google.com.br	lgross1@google.com.br	2018-11-01T12:10:25Z
		name	status	update	GSI1PK	GSI1SK			
lgross1@google.com.br	2018-11-01T12:10:25Z	gpa	Available	Assigned	9:45	Assigned			
		gpa	status	update	GSI1PK	GSI1SK			
lgross1@google.com.br	2018-11-01T12:10:25Z	gpa_05	Available	Assigned	9:45	Available 05min			
		gpa	status	update	GSI1PK	GSI1SK			
lgross1@google.com.br	2018-11-01T12:10:25Z	gpa_10	Available	Available	9:45	Available 10min			
		gpa	status	update	GSI1PK	GSI1SK			
lgross1@google.com.br	2018-11-01T12:10:25Z	item_1	committer@mgur.com	description	GSI1PK	GSI1SK	GSI2PK	GSI2SK	
		committer@mgur.com	Mark Special	order_1	item_1	committer@mgur.com	2018-11-01T12:10:25Z		
lgross1@google.com.br	2018-11-01T12:10:25Z	item_2	committer@mgur.com	description	GSI1PK	GSI1SK	GSI2PK	GSI2SK	
		committer@mgur.com	Onionoids	order_1	item_1	committer@mgur.com	2018-11-01T12:10:25Z		
lgross1@google.com.br	2018-11-01T12:10:25Z	item_3	lgross1@google.com.br	description	GSI1PK	GSI1SK	GSI2PK	GSI2SK	
		lgross1@google.com.br	Basket	order_2	item_2	lgross1@google.com.br	2018-11-01T12:03:37Z		
lgross1@google.com.br	2018-11-01T12:10:25Z	item_4	lgross1@google.com.br	description	GSI1PK	GSI1SK	GSI2PK	GSI2SK	
		lgross1@google.com.br	Pork Ribs	order_2	item_2	lgross1@google.com.br	2018-11-01T12:03:37Z		

We will design and use a single DynamoDB table that will satisfy all those different access patterns using a combination of a primary key, sort keys and composite keys.

The NoSQL Approach

aws re:Invent

Table		GSI1	GSI2				
PK	SK	Attributes					
sparker@kommung.com	2018-11-01T12:10:28Z	Items	status	update	GSI1PK	GSI1SK	GSI2PK
Customer	Keto Patis	fullName	address	city	state	email	tachys-aus1
lmanchester@jugem.jp	2018-11-01T12:03:37Z	Items	status	update	GSI1PK	GSI1SK	GSI2PK
Customer	Limao Manchester	En Route	address	city	state	email	lmanchester@jugem.jp
tachys-aus1	vendor	Tachy's Tacos	7620 Lake Drive	Austin	Texas	agathyc0	tachys-aus1
satrick-aus1	vendor	Set Loko	7750 Demar Avenue	Austin	Texas	esleyt1	satrick-aus1
oscommerton@mguru.com	driver	Comal Somerton	3600 Chinook Avenue	Austin	Texas	contact	status
gpa	gpa	gpa	status	GSI1PK	GSI1SK	GSI2PK	GSI2SK
gpa-05	DriverAssigned	gpa	status	GSI1PK	GSI1SK	GSI2PK	GSI2SK
gpa-10	DriverAssigned	gpa	status	GSI1PK	GSI1SK	GSI2PK	GSI2SK
gpa-15	DriverAvailable	gpa	status	GSI1PK	GSI1SK	GSI2PK	GSI2SK
gpa-20	DriverAvailable	gpa	status	GSI1PK	GSI1SK	GSI2PK	GSI2SK
gpa-25	DriverAvailable	gpa	status	GSI1PK	GSI1SK	GSI2PK	GSI2SK
gpa-30	DriverAvailable	gpa	status	GSI1PK	GSI1SK	GSI2PK	GSI2SK
gpa-35	DriverAvailable	gpa	status	GSI1PK	GSI1SK	GSI2PK	GSI2SK
gpa-40	DriverAvailable	gpa	status	GSI1PK	GSI1SK	GSI2PK	GSI2SK
item_1	oscommerton@mguru.com	Milk Special	order_1	item_1	oscommerton@mguru.com	2018-11-01T12:10:28Z	GSI2PK
item_2	oscommerton@mguru.com	Crossroads	order_1	item_1	oscommerton@mguru.com	2018-11-01T12:10:28Z	GSI2PK
item_3	lgross1@google.com.br	Bread	order_2	item_2	lgross1@google.com.br	2018-11-01T12:03:37Z	GSI2PK
item_4	lgross1@google.com.br	Pork Ribs	order_2	item_2	lgross1@google.com.br	2018-11-01T12:03:37Z	GSI2PK



We can get a **customer information** by simply querying the table by the customer's email address as the partition key and the sort key will be the **customer** value for that email.

The NoSQL Approach

aws re:Invent

Table		GSI1	GSI2				
PK	SK	Attributes					
sparker@kommung.com	2018-11-01T12:10:28Z	Items	status	update	GSI1PK	GSI1SK	GSI2PK
Customer	Keto Patis	fullName	address	city	state	email	tachys-aus1
lmanchester@jugem.jp	2018-11-01T12:03:37Z	Items	status	update	GSI1PK	GSI1SK	GSI2PK
Customer	Limao Manchester	En Route	address	city	state	email	lmanchester@jugem.jp
tachys-aus1	vendor	Tachy's Tacos	7620 Lake Drive	Austin	Texas	agathyc0	tachys-aus1
satrick-aus1	vendor	Set Loko	7750 Demar Avenue	Austin	Texas	esleyt1	satrick-aus1
oscommerton@mguru.com	driver	Comal Somerton	3600 Chinook Avenue	Austin	Texas	contact	status
gpa	gpa	gpa	status	GSI1PK	GSI1SK	GSI2PK	GSI2SK
gpa-05	DriverAssigned	gpa	status	GSI1PK	GSI1SK	GSI2PK	GSI2SK
gpa-10	DriverAssigned	gpa	status	GSI1PK	GSI1SK	GSI2PK	GSI2SK
gpa-15	DriverAvailable	gpa	status	GSI1PK	GSI1SK	GSI2PK	GSI2SK
gpa-20	DriverAvailable	gpa	status	GSI1PK	GSI1SK	GSI2PK	GSI2SK
gpa-25	DriverAvailable	gpa	status	GSI1PK	GSI1SK	GSI2PK	GSI2SK
gpa-30	DriverAvailable	gpa	status	GSI1PK	GSI1SK	GSI2PK	GSI2SK
gpa-35	DriverAvailable	gpa	status	GSI1PK	GSI1SK	GSI2PK	GSI2SK
gpa-40	DriverAvailable	gpa	status	GSI1PK	GSI1SK	GSI2PK	GSI2SK
item_1	oscommerton@mguru.com	Milk Special	order_1	item_1	oscommerton@mguru.com	2018-11-01T12:10:28Z	GSI2PK
item_2	oscommerton@mguru.com	Crossroads	order_1	item_1	oscommerton@mguru.com	2018-11-01T12:10:28Z	GSI2PK
item_3	lgross1@google.com.br	Bread	order_2	item_2	lgross1@google.com.br	2018-11-01T12:03:37Z	GSI2PK
item_4	lgross1@google.com.br	Pork Ribs	order_2	item_2	lgross1@google.com.br	2018-11-01T12:03:37Z	GSI2PK



We can also use the **orderdate** value as a sort key to **get orders for that email address** in a particular data or date range.

The NoSQL Approach

Table GSI1		Table GSI2						
Primary Key	SK	Attributes						
PK	SK	item	status	update	GSI1PK	GSI1SK	GSI2PK	GSI2SK
sparker@empresasing.com	customer	fullName	address	city	state	email	sparkers@gmail.com	brazil-sa01
	customer	Kota Palor	957 La Follette Drive	Austin	Texas	sparkers@gmail.com	sparkers@gmail.com	2018-11-01T12:10:26Z
umanchester1@upem.p	customer	fullName	status	update	GSI1PK	GSI1SK	GSI2PK	GSI2SK
	customer	—	In-Route	2018-11-01T12:46:16Z	order_2	umanchester1@upem.p	umanchester1@upem.p	satellite-aus1
tachys@aus1	vendor	Umoja Manchester	90 Anaroma Crossing	Austin	Texas	email	umanchester1@upem.p	2018-11-01T12:03:07Z
	vendor	Tachys Tacos	762 Utah Cross	Austin	Texas	username	tachys@aus1	tachys@aus1
satnick@aus1	vendor	vehicleName	address	city	state	username	satnick@aus1	satnick@aus1
	vendor	Sat Lick	7750 Denver Avenue	Austin	Texas	vendor	satnick@aus1	satnick@aus1
cameron@imgur.com	driver	name	address	city	contact	status	driver	driver
	driver	Camel Compton	5503 Chinook Avenue	Austin	cameron@imgur.com	Active	cameron@imgur.com	cameron@imgur.com
cameron@imgur.com	gpa	gpa	status	GSI1PK	GSI1SK			
	gpa-05	Unlocked	assigned	94%	assigned			
	gpa-06	Unlocked	assigned	94%	assigned	05min		
igress1@google.com.br	gpa-10	Unlocked	available	94%	available	10min		
	driver	name	address	city	contact	status	driver	driver
	driver	Field Officer	7650 Lake View Hill	Austin	fores1@google.com.br	Active	fores1@google.com.br	fores1@google.com.br
igress1@google.com.br	gpa	gpa	status	GSI1PK	GSI1SK			
	gpa-05	Unlocked	assigned	94%	assigned			
	gpa-06	Unlocked	available	94%	available	05min		
igress1@google.com.br	gpa-10	Unlocked	available	94%	available	10min		
item_1	caimmerito@imgur.com	description	GSI1PK	GSI1SK	GSI2PK	GSI2SK		
	caimmerito@imgur.com	Monk Special	order_1	item_1	caimmerito@imgur.com	2018-11-01T12:10:26Z		
	caimmerito@imgur.com	description	GSI1PK	GSI1SK	GSI2PK	GSI2SK		
	caimmerito@imgur.com	Croissant	order_1	item_1	caimmerito@imgur.com	2018-11-01T12:10:26Z		
item_2	caimmerito@imgur.com	description	GSI1PK	GSI1SK	GSI2PK	GSI2SK		
	caimmerito@imgur.com	Crème Brûlée	order_2	item_2	caimmerito@imgur.com	2018-11-01T12:03:07Z		
item_3	igress1@google.com.br	Unlocked	order_2	item_2	igress1@google.com.br	2018-11-01T12:03:07Z	GSI2PK	GSI2SK
	igress1@google.com.br	Pork Rib	order_2	item_2	igress1@google.com.br	2018-11-01T12:03:07Z	GSI2PK	GSI2SK

We can also get vendor data using the vendorID as the primary key as above

The NoSQL Approach

Table		GSI1	GSI2					
Primary Key	SK	Attributes						
PK	SK	item	status	update	GSI1PK	GSI1SK	GSI2PK	GSI2SK
sparker@amazon.com	customer	item1	Debtored	2016-11-01T12:45:10Z	order_1	sparker@amazon.com	sparky@sparky1	2016-11-01T12:10:20Z
		fullName	address	city	state	email		
	Keto Baker	657 La Follette Center	Austin	Texas		sparker@amazon.com		
umanchester@jumpr.p	customer	item2	status	update	GSI1PK	GSI1SK	GSI2PK	GSI2SK
		item3	En Route	2016-11-01T12:46:10Z	order_2	umanchester@jumpr.p	satnick.aust1	2016-11-01T12:03:37Z
		fullName	address	city	state	email		
	Lemon Manchester	90 Ananasus Crossing	Austin	Texas		umanchester@jumpr.p		
birchys-aust1	vendor	item4	address	city	state	username	vendor	
		Torches Tacos	2627 Linn Chie	Austin	Texas	birchys0	birchys-aust1	
satnick.aust1	vendor	item5	address	city	state	username	vendor	
		Salt Lick	7750 Denita Avenue	Austin	Texas	satnick1	satnick-aust1	
		item6	address	city	contact	status	driver	
		Comal Somerton	3603 Chinook Avenue	Austin	clsmerton@lmjmr.com	Active	clsmerton@lmjmr.com	
clsmerton@lmjmr.com	driver	item7	status	GSI1PK	GSI1SK			
		driv001	assigned	94%	assigned			
	driv001	item8	status	GSI1PK	GSI1SK			
		driv002	assigned	94%	assigned	driv001@lmjmr.com		
	driv002	item9	status	GSI1PK	GSI1SK			
		driv003	available	94%	available	10min		
gretta1@google.com.br	driver	item10	name	address	city	contact	status	driver
		Field Order	7630 Lake View Ht	Austin	fgretta1@google.com.br	Active	fgretta1@google.com.br	
		item11	status	GSI1PK	GSI1SK			
		driv004	assigned	94%	assigned			
	driv004	item12	status	GSI1PK	GSI1SK			
		driv005	available	94%	available	05min		
	driv005	item13	status	GSI1PK	GSI1SK			
		driv006	available	94%	available	10min		
	driv006	item14	status	GSI1PK	GSI1SK			
		driv007	available	94%	available	10min		
item_1	clsmerton@lmjmr.com	description	GSI1PK	GSI1SK	GSI2PK	GSI2SK		
		Mark Special	order_1	item_1	clsmerton@lmjmr.com	2016-11-01T12:10:20Z		
item_2	clsmerton@lmjmr.com	description	GSI1PK	GSI1SK	GSI2PK	GSI2SK		
		Overcooked	order_4	item_1	clsmerton@lmjmr.com	2016-11-01T12:10:20Z		
item_3	fgretta1@google.com.br	description	GSI1PK	GSI1SK	GSI2PK	GSI2SK		
		Baked	order_2	item_2	fgretta1@google.com.br	2016-11-01T12:03:37Z		
item_4	fgretta1@google.com.br	description	GSI1PK	GSI1SK	GSI2PK	GSI2SK		
		Pork Ribs	order_2	item_2	fgretta1@google.com.br	2016-11-01T12:03:37Z		

We can get driver data by selecting using the **driver's email** as the primary key and the **driver** value as the sort key.

The NoSQL Approach

AWS re:Invent



Table		GSI1	GSI2	Attributes						
PK	SK			Items	status	update	GSI1PK	GSI1SK	GSI2PK	GSI2SK
spaker0@kemung.com	2016-11-01T12:10:25Z	customer		Delivered		2016-11-01T12:05:10Z	order_1	spaker0@kemung.com	tachy-aus1	2016-11-01T12:10:21M
umanchester@uguru.com	2016-11-01T12:03:37Z	customer		fullName	address	city	state	email		
umanchester@uguru.com	2016-11-01T12:03:37Z	customer	Kelta Fisher	957 La Felizte Center	Austin	Texas	spaker0@kemung.com			
umanchester@uguru.com	2016-11-01T12:03:37Z	customer	En-Route	2016-11-01T12:06:16Z	order_2	umanchester@uguru.com	satrick-aus1			2016-11-01T12:03:37Z
umanchester@uguru.com	2016-11-01T12:03:37Z	customer	Umesh Marichander	90 Anantas Crossing	Austin	Texas	umanchester@uguru.com			
tachy-aus1	vendor	Tachy's Tacos	2620 Utah One	Austin	Texas	spaker0@kemung.com	tachy-aus1			
satrick-aus1	vendor	Sat.Loki	7750 Denrite Avenue	Austin	Texas	esetly1	satrick-aus1			
osammerito@uguru.com	2016-11-01T12:03:37Z	driver	Corral Somerton	3603 Chinook Avenue	Austin	osammerito@uguru.com	Active	osammerito@uguru.com		
osammerito@uguru.com	2016-11-01T12:03:37Z	gpa	gpa	status	GSI1PK	GSI1SK				
osammerito@uguru.com	2016-11-01T12:03:37Z	gpa	Defuzzify	assigned	94%	assigned				
osammerito@uguru.com	2016-11-01T12:03:37Z	gpa	gpa	status	GSI1PK	GSI1SK				
osammerito@uguru.com	2016-11-01T12:03:37Z	gpa	Defuzzifying	assigned	94%	assigned	GSI1PK	GSI1SK		
osammerito@uguru.com	2016-11-01T12:03:37Z	gpa	gpa	status	GSI1PK	GSI1SK				
osammerito@uguru.com	2016-11-01T12:03:37Z	gpa	94%99%100%	available	94%	available	10min			
osammerito@uguru.com	2016-11-01T12:03:37Z	driver	Fred Griss	2630 Lake View Hill	Austin	igress1@google.com.br	Active	igress1@google.com.br		
osammerito@uguru.com	2016-11-01T12:03:37Z	gpa	gpa	status	GSI1PK	GSI1SK				
osammerito@uguru.com	2016-11-01T12:03:37Z	gpa	Defuzzified	assigned	94%	assigned				
osammerito@uguru.com	2016-11-01T12:03:37Z	gpa	gpa	status	GSI1PK	GSI1SK				
osammerito@uguru.com	2016-11-01T12:03:37Z	gpa	Defuzzifying	assigned	94%	assigned	GSI1PK	GSI1SK		
osammerito@uguru.com	2016-11-01T12:03:37Z	gpa	gpa	status	GSI1PK	GSI1SK				
osammerito@uguru.com	2016-11-01T12:03:37Z	gpa	94%99%100%	available	94%	available	10min			
osammerito@uguru.com	2016-11-01T12:03:37Z	item_1	osammerito@uguru.com	description	GSI1PK	GSI1SK	GSI2PK	GSI2SK		
osammerito@uguru.com	2016-11-01T12:03:37Z	item_1	Mink Special	order_1	item_1	osammerito@uguru.com	2016-11-01T12:10:20Z			
osammerito@uguru.com	2016-11-01T12:03:37Z	item_1	description	GSI1PK	GSI1SK	GSI2PK	GSI2SK			
osammerito@uguru.com	2016-11-01T12:03:37Z	item_1	Crossroads	order_1	item_1	osammerito@uguru.com	2016-11-01T12:10:20Z			
osammerito@uguru.com	2016-11-01T12:03:37Z	item_1	description	GSI1PK	GSI1SK	GSI2PK	GSI2SK			
osammerito@uguru.com	2016-11-01T12:03:37Z	item_1	Baked	order_2	item_2	igress1@google.com.br	2016-11-01T12:03:37Z			
osammerito@uguru.com	2016-11-01T12:03:37Z	item_1	description	GSI1PK	GSI1SK	GSI2PK	GSI2SK			
osammerito@uguru.com	2016-11-01T12:03:37Z	item_1	Pork Ribs	order_2	item_2	igress1@google.com.br	2016-11-01T12:03:37Z			

The NoSQL Approach

AWS re:Invent



Table		GSI1	GSI2	Attributes						
PK	SK			Items	status	update	GSI1PK	GSI1SK	GSI2PK	GSI2SK
spaker0@kemung.com	2016-11-01T12:10:25Z	customer		Delivered		2016-11-01T12:05:10Z	order_1	spaker0@kemung.com	tachy-aus1	2016-11-01T12:10:21M
umanchester@uguru.com	2016-11-01T12:03:37Z	customer		fullName	address	city	state	email		
umanchester@uguru.com	2016-11-01T12:03:37Z	customer	Kelta Fisher	957 La Felizte Center	Austin	Texas	spaker0@kemung.com			
umanchester@uguru.com	2016-11-01T12:03:37Z	customer	En-Route	2016-11-01T12:06:16Z	order_2	umanchester@uguru.com	satrick-aus1			2016-11-01T12:03:37Z
umanchester@uguru.com	2016-11-01T12:03:37Z	customer	Umesh Marichander	90 Anantas Crossing	Austin	Texas	umanchester@uguru.com			
tachy-aus1	vendor	Tachy's Tacos	2620 Utah One	Austin	Texas	spaker0@kemung.com	tachy-aus1			
satrick-aus1	vendor	Sat.Loki	7750 Denrite Avenue	Austin	Texas	esetly1	satrick-aus1			
osammerito@uguru.com	2016-11-01T12:03:37Z	driver	Corral Somerton	3603 Chinook Avenue	Austin	osammerito@uguru.com	Active	osammerito@uguru.com		
osammerito@uguru.com	2016-11-01T12:03:37Z	gpa	gpa	status	GSI1PK	GSI1SK				
osammerito@uguru.com	2016-11-01T12:03:37Z	gpa	Defuzzify	assigned	94%	assigned				
osammerito@uguru.com	2016-11-01T12:03:37Z	gpa	gpa	status	GSI1PK	GSI1SK				
osammerito@uguru.com	2016-11-01T12:03:37Z	gpa	Defuzzifying	assigned	94%	assigned	GSI1PK	GSI1SK		
osammerito@uguru.com	2016-11-01T12:03:37Z	gpa	gpa	status	GSI1PK	GSI1SK				
osammerito@uguru.com	2016-11-01T12:03:37Z	gpa	94%99%100%	available	94%	available	10min			
osammerito@uguru.com	2016-11-01T12:03:37Z	driver	Fred Griss	2630 Lake View Hill	Austin	igress1@google.com.br	Active	igress1@google.com.br		
osammerito@uguru.com	2016-11-01T12:03:37Z	gpa	gpa	status	GSI1PK	GSI1SK				
osammerito@uguru.com	2016-11-01T12:03:37Z	gpa	Defuzzified	assigned	94%	assigned				
osammerito@uguru.com	2016-11-01T12:03:37Z	gpa	gpa	status	GSI1PK	GSI1SK				
osammerito@uguru.com	2016-11-01T12:03:37Z	gpa	Defuzzifying	assigned	94%	assigned	GSI1PK	GSI1SK		
osammerito@uguru.com	2016-11-01T12:03:37Z	gpa	gpa	status	GSI1PK	GSI1SK				
osammerito@uguru.com	2016-11-01T12:03:37Z	gpa	94%99%100%	available	94%	available	10min			
osammerito@uguru.com	2016-11-01T12:03:37Z	item_1	osammerito@uguru.com	description	GSI1PK	GSI1SK	GSI2PK	GSI2SK		
osammerito@uguru.com	2016-11-01T12:03:37Z	item_1	Mink Special	order_1	item_1	osammerito@uguru.com	2016-11-01T12:10:20Z			
osammerito@uguru.com	2016-11-01T12:03:37Z	item_1	description	GSI1PK	GSI1SK	GSI2PK	GSI2SK			
osammerito@uguru.com	2016-11-01T12:03:37Z	item_1	Crossroads	order_1	item_1	osammerito@uguru.com	2016-11-01T12:10:20Z			
osammerito@uguru.com	2016-11-01T12:03:37Z	item_1	description	GSI1PK	GSI1SK	GSI2PK	GSI2SK			
osammerito@uguru.com	2016-11-01T12:03:37Z	item_1	Baked	order_2	item_2	igress1@google.com.br	2016-11-01T12:03:37Z			
osammerito@uguru.com	2016-11-01T12:03:37Z	item_1	description	GSI1PK	GSI1SK	GSI2PK	GSI2SK			
osammerito@uguru.com	2016-11-01T12:03:37Z	item_1	Pork Ribs	order_2	item_2	igress1@google.com.br	2016-11-01T12:03:37Z			

Indexing is about overloading the key attributes like order_1

The NoSQL Approach

Table		GS1	GS12									
PK	SK	Primary Key							Attributes			
		item	status	update	GS1PK	GS1SK	GS2PK	GS2SK				
sparker@salesforce.com	2016-11-01T12:10:28Z	item	Defined	2016-11-01T12:10:10Z	order_1	sparker@salesforce.com	tomchya-aus1	2016-11-01T12:10:28Z				
	customer	full_name	address	city	state	email	sparker@salesforce.com	tomchya-aus1				
lmanchester@quorum.p	2016-11-01T12:03:37Z	item	status	update	GS1PK	GS1SK	GS2PK	GS2SK				
	customer	full_name	address	city	state	email	lmanchester@quorum.p	tomchya-aus1	2016-11-01T12:03:37Z			
tomchya-aus1	vendor	name	address	city	state	username	tomchya-aus1	tomchya-aus1				
selfwick-aus1	vendor	name	address	city	state	username	selfwick-aus1	selfwick-aus1				
caommerton@imgur.com	driver	name	address	city	contact	status	driver	caommerton@imgur.com				
	gpa	gpa	status	GS1PK	GS1SK							
caommerton@imgur.com	gpa_05	gpa	status	gpa	assigned							
	gpa_06	gpa	status	GS1PK	GS1SK							
	gpa_10	gpa	status	GS1PK	GS1SK							
llynn1@google.com.br	driver	name	address	city	contact	status	driver	llynn1@google.com.br				
	gpa	gpa	status	GS1PK	GS1SK							
llynn1@google.com.br	gpa_05	gpa	status	gpa	assigned							
	gpa_06	gpa	status	GS1PK	GS1SK							
	gpa_10	gpa	status	GS1PK	GS1SK							
llynn1@google.com.br	gpa_10	gpa	status	gpa	available	10min						
	item_1	description	GS1PK	GS1SK	GS2PK	GS2SK						
item_2	caommerton@imgur.com	Minx Special	order_1	item_1	caommerton@imgur.com	2016-11-01T12:10:28Z						
	caommerton@imgur.com	description	GS1PK	GS1SK	GS2PK	GS2SK						
item_3	llynn1@google.com.br	Oranges	order_1	item_1	caommerton@imgur.com	2016-11-01T12:10:28Z						
item_4	llynn1@google.com.br	Broccoli	order_2	item_2	llynn1@google.com.br	2016-11-01T12:03:37Z						
	llynn1@google.com.br	Pork Ribs	order_2	item_2	llynn1@google.com.br	2016-11-01T12:03:37Z						

We can also index the drivers GPS coordinates and store the data for easy access later

The NoSQL Approach

Table		GSI1	GSI2					
Primary Key		Attributes						
PK	SK	Item	status	update	GSI1PK	GSI1SK	GSI2PK	GSI2SK
spiderman@spider.com	customer	fullName	Delivered	2018-11-01T12:45:10Z	order_1	spiderman@spider.com	spiderman1	2018-11-01T12:45:10Z
	customer	address	city	Texas	state	email		
keto@keto.com	Keto	name	status	updated	GSI1PK	GSI1SK	GSI2PK	GSI2SK
	Keto	address	city	Texas	state	email		
umanchester@jugem.jp	customer	fullName	En Route	2018-11-01T12:46:16Z	order_2	umanchester1@jugem.jp	satlich_aus1	2018-11-01T12:46:16Z
	customer	address	city	Texas	state	email		
tortys-aus1	vendor	Umojo Manchester	99 Arkansas Crossing	Austin	Texas	username	vendor	
	vendor	vendorName	address	city	state	username	vendor	
satlich-aus1	vendor	Tucky's Tacos	262 Lash Drive	Austin	Texas	username	satlich_aus1	
	vendor	vendorName	address	city	state	username	satlich_aus1	
caommerton@mgur.com	driver	Set.Loc	7750 Demme Avenue	Austin	Texas	username	driver	
	driver	name	address	city	contact	status	driver	
caommerton@mgur.com	driver	Comet Sommers	3603 Chinook Avenue	Austin	caommerton@mgur.com	Active	caommerton@mgur.com	
	driver	git	status	GSI1PK	GSI1SK			
gpa-06	git	Debtocracy	assigned	9Vu	assigned			
	git	status	GSI1PK	GSI1SK				
gpa-10	git	Debtocracy	assigned	9Vu	assigned			
	git	status	GSI1PK	GSI1SK				
gpa-06	git	Debtocracy	available	9Vu	available:10min			
	git	status	GSI1PK	GSI1SK				
gpa-10	git	Debtocracy	available	9Vu	available:10min			
	git	status	GSI1PK	GSI1SK				
gpa-06	driver	Field Gross	7630 Lake View Hn	Austin	gross1@google.com.br	Active	gross1@google.com.br	
	driver	git	status	GSI1PK	GSI1SK			
gross1@google.com.br	driver	Debtocracy	assigned	9Vu	assigned			
	driver	git	status	GSI1PK	GSI1SK			
gross1@google.com.br	driver	Debtocracy	available	9Vu	available:6min			
	driver	git	status	GSI1PK	GSI1SK			
gross1@google.com.br	driver	Debtocracy	available	9Vu	available:10min			
	driver	git	status	GSI1PK	GSI1SK			
item_1	caommerton@mgur.com	description	GSI1PK	GSI1SK	GSI2PK	GSI2SK		
	caommerton@mgur.com	Mark Special	order_1	item_1	caommerton@mgur.com	2018-11-01T12:10:20Z		
item_2	caommerton@mgur.com	description	GSI1PK	GSI1SK	GSI2PK	GSI2SK		
	caommerton@mgur.com	Onionside	order_1	item_1	caommerton@mgur.com	2018-11-01T12:10:20Z		
item_3	gross1@google.com.br	description	GSI1PK	GSI1SK	GSI2PK	GSI2SK		
	gross1@google.com.br	Basket	order_2	item_2	gross1@google.com.br	2018-11-01T12:03:37Z		
item_4	gross1@google.com.br	description	GSI1PK	GSI1SK	GSI2PK	GSI2SK		
	gross1@google.com.br	Pork Ribs	order_2	item_2	gross1@google.com.br	2018-11-01T12:03:37Z		

We can index the ordered items across those GSI's too using timestamp by customers and vendors

The NoSQL Approach (Orders and Drivers GSI)

Table		GSI1	GSI2				
Primary Key	GS1PK	Attributes					
order_1	item_1	PK	SK	description	GS1PK	GS1SK	
	item_1	csommeron0@imgur.com	Mark Special	csommeron0@imgur.com	2018-11-01T12:10:28Z		
	item_2	PK	SK	description	GS1PK	GS1SK	
order_2	kpaiser0@samsung.com	PK	SK	Items	status	update	GS1PK GS1SK
	kpaiser0@samsung.com	2018-11-01T12:10:28Z	...	Delivered	2018-11-01T12:45:13Z	torchy-aus1	2018-11-01T12:10:28Z
	item_2	PK	SK	description	GS1PK	GS1SK	
order_2	item_2	item_3	lgreen1@google.com.br	Broker	lgreen1@google.com.br	2018-11-01T12:03:37Z	
	item_2	PK	SK	description	GS1PK	GS1SK	
	item_4	lgreen1@google.com.br	Park. Rts:	lgreen1@google.com.br	2018-11-01T12:03:37Z		
9vfu	umanchester1@judem.p	PK	SK	Items	status	update	GS1PK GS1SK
	umanchester1@judem.p	2018-11-01T12:03:37Z	...	En Route	2018-11-01T12:46:18Z	satlick-aus1	2018-11-01T12:03:37Z
	assigned	PK	SK	gps			
9vfu	csommeron0@imgur.com	gps	9vfuwckevy				
	assigned	PK	SK	gps	9vfuwckcf	status	
	assigned 05min	PK	SK	gps	9vfuycsgpeq	status	
9vfu	available-05mn	csommeron0@imgur.com	gps-05	9vfuycsgpeq	assigned		
	available-05mn	PK	SK	gps	9vfuycsfya	status	
	available-10mn	lgreen1@google.com.br	gps-05	9vfuycsfya	available		
9vfu	available-10mn	lgreen1@google.com.br	gps-10	9vfuycsfya	available		
	available-10mn	PK	SK	gps	9vfuib917w	status	
	available-10mn	csommeron0@imgur.com	gps-10	9vfuib917w	available		

re:Invent

© 2018, Amazon Web Services, Inc. or its affiliates. All rights reserved.



We can now use the GSI and the orderId like **order_1** to get all order items from a specific customer

The NoSQL Approach (Orders and Drivers GSI)

Table		GSI1	GSI2				
Primary Key	GS1PK	Attributes					
order_1	item_1	PK	SK	description	GS1PK	GS1SK	
	item_1	csommeron0@imgur.com	Mark Special	csommeron0@imgur.com	2018-11-01T12:10:28Z		
	item_2	PK	SK	description	GS1PK	GS1SK	
order_2	kpaiser0@samsung.com	PK	SK	Items	status	update	GS1PK GS1SK
	kpaiser0@samsung.com	2018-11-01T12:10:28Z	...	Delivered	2018-11-01T12:45:13Z	torchy-aus1	2018-11-01T12:10:28Z
	item_2	PK	SK	description	GS1PK	GS1SK	
order_2	item_2	item_3	lgreen1@google.com.br	Broker	lgreen1@google.com.br	2018-11-01T12:03:37Z	
	item_2	PK	SK	description	GS1PK	GS1SK	
	item_4	lgreen1@google.com.br	Park. Rts:	lgreen1@google.com.br	2018-11-01T12:03:37Z		
9vfu	umanchester1@judem.p	PK	SK	Items	status	update	GS1PK GS1SK
	umanchester1@judem.p	2018-11-01T12:03:37Z	...	En Route	2018-11-01T12:46:18Z	satlick-aus1	2018-11-01T12:03:37Z
	assigned	PK	SK	gps			
9vfu	csommeron0@imgur.com	gps	9vfuwckevy				
	assigned	PK	SK	gps	9vfuwckcf	status	
	assigned 05min	csommeron0@imgur.com	gps-05	9vfuycsgpeq	assigned		
9vfu	available-05mn	lgreen1@google.com.br	gps-05	9vfuycsgpeq	status		
	available-05mn	PK	SK	gps	9vfuycsfya	status	
	available-10mn	lgreen1@google.com.br	gps-10	9vfuycsfya	available		
9vfu	available-10mn	PK	SK	gps	9vfuib917w	status	
	available-10mn	csommeron0@imgur.com	gps-10	9vfuib917w	available		

re:Invent

© 2018, Amazon Web Services, Inc. or its affiliates. All rights reserved.



We can also query by sector to get driver details also

The NoSQL Approach (Vendor and Deliveries GSI)

Table	GSI1	GSI2					
Primary Key GSI2PK	Attributes						
	GS1SK						
torchys-aus1	2018-11-01T12:10:28Z	PK	SK	Items	status	update	GS1PK
		kpaiser0@samsung.com	2018-11-01T12:10:28Z	...	Delivered	2018-11-01T12:45:13Z	order_1
sattick-aus1	2018-11-01T12:03:37Z	PK	SK	Items	status	update	GS1PK
		umanchester1@jugem.jp	2018-11-01T12:03:37Z	...	En Route	2018-11-01T12:48:18Z	order_2
csommerton0@imgur.com	2018-11-01T12:10:28Z	PK	SK	description	GS1PK	GS1SK	
		item_1	csommerton0@imgur.com	Mork Special	order_1	item_1	
	2018-11-01T12:10:28Z	PK	SK	description	GS1PK	GS1SK	
		item_2	csommerton0@imgur.com	Crossroads	order_1	item_1	
tgross1@google.com.br	2018-11-01T12:03:37Z	PK	SK	description	GS1PK	GS1SK	
		item_3	tgross1@google.com.br	Briket	order_2	item_2	
	2018-11-01T12:03:37Z	PK	SK	description	GS1PK	GS1SK	
		item_4	tgross1@google.com.br	Pork Ribs	order_2	item_2	

We can also query GSI2 by vendor and date

The NoSQL Approach (Vendor and Deliveries GSI)

Table	GSI1	GSI2					
Primary Key GSI2PK	Attributes						
	GS1SK						
torchys-aus1	2018-11-01T12:10:28Z	PK	SK	Items	status	update	GS1PK
		kpaiser0@samsung.com	2018-11-01T12:10:28Z	...	Delivered	2018-11-01T12:45:13Z	order_1
sattick-aus1	2018-11-01T12:03:37Z	PK	SK	Items	status	update	GS1PK
		umanchester1@jugem.jp	2018-11-01T12:03:37Z	...	En Route	2018-11-01T12:48:18Z	order_2
csommerton0@imgur.com	2018-11-01T12:10:28Z	PK	SK	description	GS1PK	GS1SK	
		item_1	csommerton0@imgur.com	Mork Special	order_1	item_1	
	2018-11-01T12:10:28Z	PK	SK	description	GS1PK	GS1SK	
		item_2	csommerton0@imgur.com	Crossroads	order_1	item_1	
tgross1@google.com.br	2018-11-01T12:03:37Z	PK	SK	description	GS1PK	GS1SK	
		item_3	tgross1@google.com.br	Briket	order_2	item_2	
	2018-11-01T12:03:37Z	PK	SK	description	GS1PK	GS1SK	
		item_4	tgross1@google.com.br	Pork Ribs	order_2	item_2	

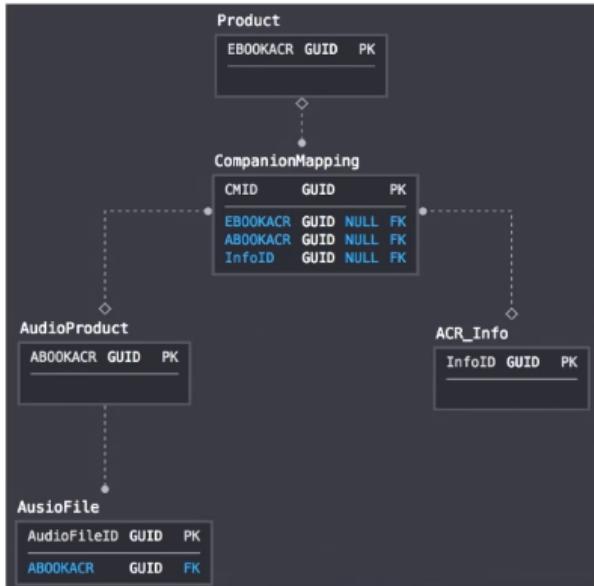
We can also query by email and date

A Real World Example

“Reality is that which, when you stop believing in it, does not go away.”

- Philip K. Dick

Audible eBook Sync Service



© 2017, Amazon Web Services, Inc. or its Affiliates. All rights reserved.

- Allows users to save session state for Audible eBooks
- Maintains mappings per user for eBooks and audio products
- Spikey load patterns require significant overprovisioning
- Large number of access patterns

Access Patterns

#	USE CASE
1	CompanionMapping
2	getCompanionMappingsByAsin
2	CompanionMapping
2	getCompanionMappingsByEbookAndAudiobookContentId
3	CompanionMapping
3	getCompanionMappingsFromCache
4	CompanionMapping
4	getCompanionMappings
5	CompanionMapping
5	getCompanionMappingsAvailable
6	Acrinfo
6	getACRInfo
7	Acrinfo
7	getACRs
8	Acrinfo
8	getACRInfos
9	Acrinfo
9	getACRInfosbySKU
10	AudioProduct
10	getAudioProductsForACRs
11	AudioProduct
11	getAudioProducts
12	AudioProduct
12	deleteAudioProductsMatchingSkuVersions
13	AudioProduct
13	getChildAudioProductsForSKU
14	Product
14	getProductInfoByAsins
15	Product
15	getParentChildDataByParentAsins
16	AudioFile
16	getAudioFilesForACR
17	AudioFile
17	getAudioFilesForChildACR
18	AudioFile
18	getAudioFilesByParentAsinVersionFormat
19	AudioFile
19	getAudioFiles
20	AudioFile
20	getAudioFilesForChildAsin

© 2018, Amc

All rights reserved.

aws Invent



We have 20 access patterns and we have 5 GSIs to use in our single DynamoDB table implementation.

Primary Table

Primary Key		Attributes	
PK	SK (GSI 3)		
T A B L E	v0#ABOOKACR1	GSI-1	GSI-2
	ABOOK-ASIN1	ABOOK-ASIN1	ABOOK-SKU1
	EBOOKACR1	GSI-1	GSI-2
	SyncFileAcr	SyncFileAcr	ABOOK-ASIN1
	ABOOKACR1#TRACK#1	GSI-1	GSI-2
	ABOOK-ASIN1	ABOOK-ASIN1	ABOOK-SKU1
EBOOKACR1	ABOOKACR1#TRACK#2	GSI-1	GSI-2
	ABOOK-ASIN1	ABOOK-ASIN1	ABOOK-SKU1
EBOOKACR1		GSI-1	EBookAsin
EBOOK-SKU1		EBOOK-SKU1	ASIN

This is the proposed table structure we implemented. The current version of a book is always the v0 version. We have 2 partitions on the table, the ABOOK and the EBOOK partitions. We then insert an item into the ABOOK called ABOOKACR partition that associates the ABOOK to the EBOOK. We end up with 3 GSIs/indexes on this table that don't always contain the same data type because we have so many access data patterns.

Indexes

Partition Key		Projected Attributes	
G S I 1	ABOOK-ASIN1	ABOOKACR1	ABOOKACR1-v1
			ABOOKACR1#TRACK#1
			ABOOKACR1#TRACK#2
	SyncFileAcr	ABOOKACR1	MAP-EBOOKACR1
	EBOOK-SKU1	ABOOKACR1	EBOOKACR1
5 5 2	ABOOK-ASIN1	ABOOKACR1	MAP-EBOOKACR1
	ABOOK-SKU1	ABOOKACR1	ABOOKACR1-v1
			ABOOKACR1#TRACK#1
			ABOOKACR1#TRACK#2
GSI Partition Key		Projected Attributes	
G S I 3	V0#ABOOKACR1	ABOOKACR1	ABOOKACR1-v1
	EBOOKACR1	ABOOKACR1	MAP-EBOOKACR1
	ABOOKACR1#TRACK#1	ABOOKACR1	ABOOKACR1#TRACK#1
	ABOOKACR1#TRACK#2	ABOOKACR1	ABOOKACR1#TRACK#2
	EBOOKACR1	ABOOKACR1	EBOOKACR1

re:Invent



This is what ends up happening on the table, the GSI entries look meaningless because we have items sorted by arbitrary values.

Query Conditions

#	USE CASE	Lookup parameters	INDEX	Key Conditions	Filter Conditions
1	CompanionMapping	getCompanionMappingsByAsin	audiobookAsin/ebookSku	GSI2	GSI-2=ABOOK-ASIN1 None
2	CompanionMapping	getCompanionMappingsIfBookAndAudiobookContentId	ebookAcr/sku,version,format or audiobookAcr/asin,version,format	GSI-3 on TargetACR attribute OR PrimaryKey on Table	GSI-3=MAP-FBOOKACR1 version=v and format=f
3	CompanionMapping	getCompanionMappingsFromCache	ebookAcr/sku,version,format or audiobookAcr/asin,version,format	GSI-3 on TargetACR attribute OR PrimaryKey on Table	GSI-3=MAP-EBOOKACR1 version=v and format=f
4	CompanionMapping	getCompanionMappings	syncfileAcr, ebookAcr?, audiobookAcr?	GSI1	GSI-1=SyncFileAcr None
5	CompanionMapping	getCompanionMappingsAvailable	ebookAcr, audiobookAcr	Primary Key on Table	Acr=ABOOKACR1 and TargetACR beginswith "MAP_"
6	AcrInfo	getACRInfo	acr	Primary Key on Table	Acr=ABOOKACR1 and TargetACR beginswith "ABOOKACR1-v"
7	AcrInfo	getACRs	acr / asin,version,format	Primary Key on Table	Acr=ABOOKACR1 version=v and format=f
8	AcrInfo	getACRInfos	acr	Primary Key on table	Acr=ABOOKACR1 and TargetACR beginswith "ABOOKACR1"
9	AcrInfo	getACRInfosbySKU	sku	GSI2	GSI-2=ABOOK-SKU1
10	AudioProduct	getAudioProductsForACRs	acr	Primary Key on Table	Acr=ABOOKACR1 and TargetACR beginswith "ABOOKACR1"
11	AudioProduct	getAudioProducts	sku, version, format	GSI2	GSI-2=ABOOK-SKU1 version=v and format=f
12	AudioProduct	deleteAudioProductsMatchingSkuVersion	sku, version	GSI2	GSI-2=ABOOK-SKU1 version=v
13	AudioProduct	getChildAudioProductsForSKU	sku	GSI2	GSI-2=ABOOK-SKU1
14	Product	getProductInfoByAsins	asin	GSI1	GSI-1=ABOOK-ASIN1
15	Product	getParentChildDataByParentAsins	asin	GSI1	GSI-1=ABOOK-ASIN1
16	AudioFile	getAudioFilesForACR	acr	Primary Key on table	Acr=ABOOKACR1 and TargetACR beginswith "ABOOKACR1#"
17	AudioFile	getAudioFilesForChildACR	acr, parent_asin	Primary Key on table	Acr=ABOOKACR1 version=v and format=f
18	AudioFile	getAudioFilesByParentAsinVersionFormat	parent_asin, version, format	GSI1	GSI-1=ABOOK-ASIN1 version=v and format=f
19	AudioFile	getAudioFiles	sku, version, format	GSI2	GSI-2=ABOOK-SKU1 version=v and format=f
20	AudioFile	getAudioFilesForChildAsin	asin, parent_asin, version, format	GSI1	GSI-1=ABOOK-ASIN1 version=v and format=f

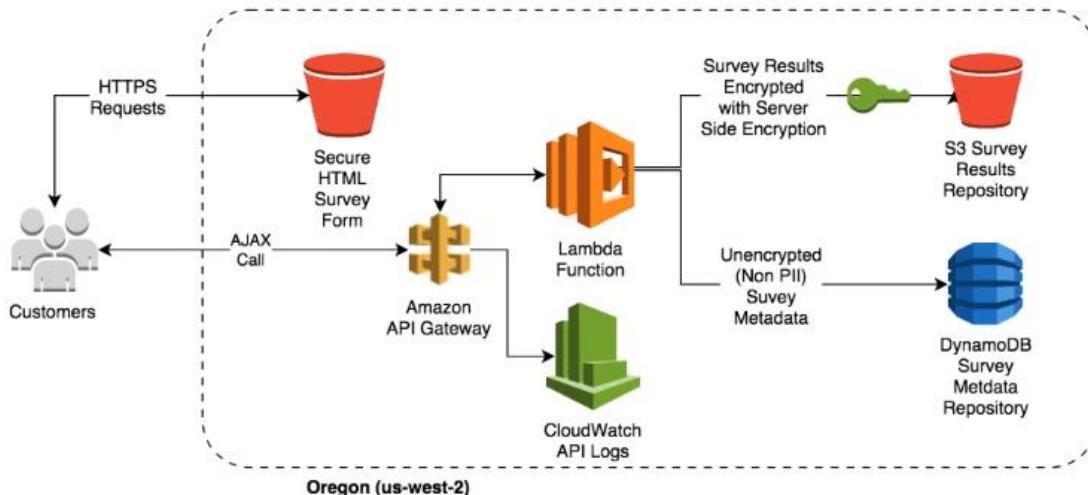
We ended up expanding the tables to show the needed sort key and filter conditions for the 20 different access patterns are satisfied using the proposed single table.

The Serverless Paradigm

“Fairly cheap home computing was what changed my life.”

- Linus Torvalds

Elastic Serverless Applications



This is the rate my response interaction service for Amazon services feedback feature.

Conclusions

- NoSQL does not mean non-relational
- The ERD still matters
- RDBMS is not deprecated by NoSQL
- Use NoSQL for OLTP or DSS at scale.
- Use RDBMS for OLAP or OLTP when scale is not important.

