Replication for Multiple Partition Keys

Leonard Lobel
CTO, SLEEK TECHNOLOGIES

lennilobel.wordpress.com

Curing Analysis Paralysis

Choose a partition key

Based on your most common querying patterns

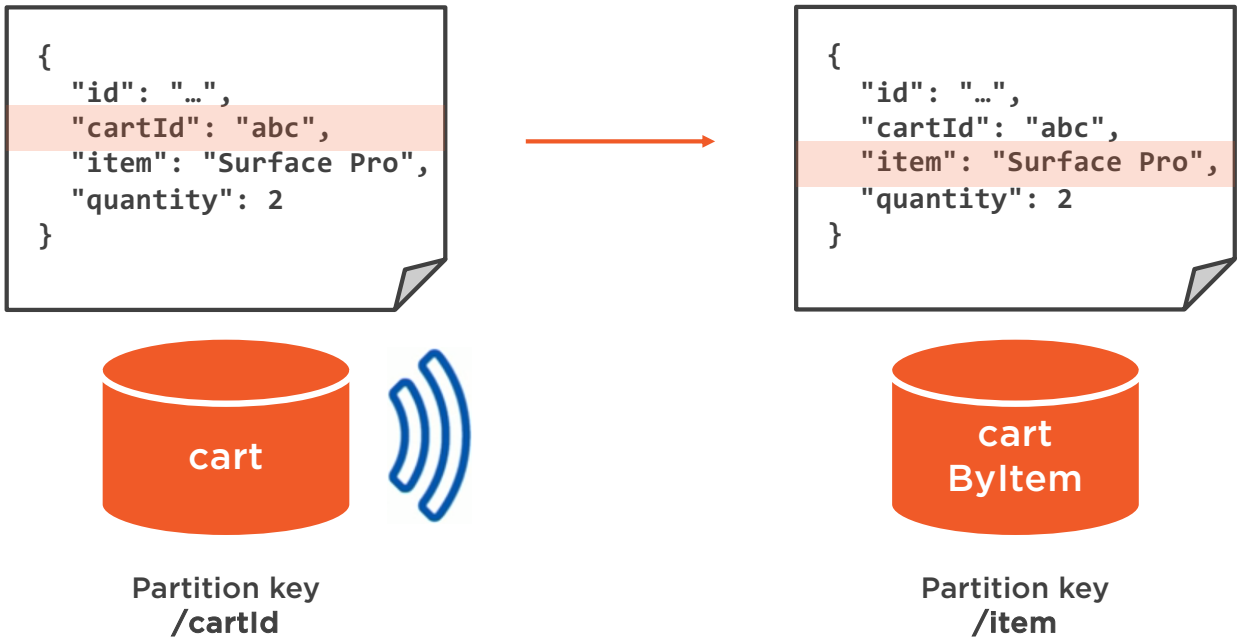
Only ONE partition key

What if you have more than just one common querying pattern?

Replication

Synchronize multiple containers with the same data, partitioned differently

Using Change Feed for Replication




```
SELECT * FROM c WHERE c.cartId = 'abc'
SELECT * FROM c WHERE c.item = 'Surface Pro'
```

Partition key

Processing Updates and Deletes

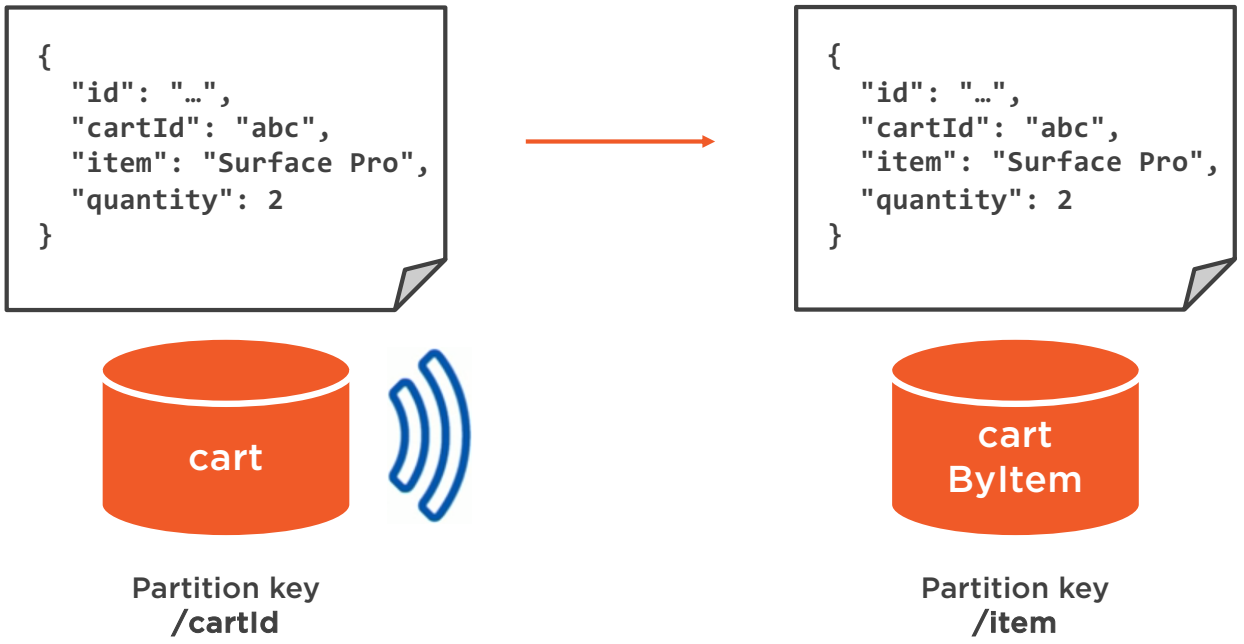
Updates

Interim updates are not retained by the change feed

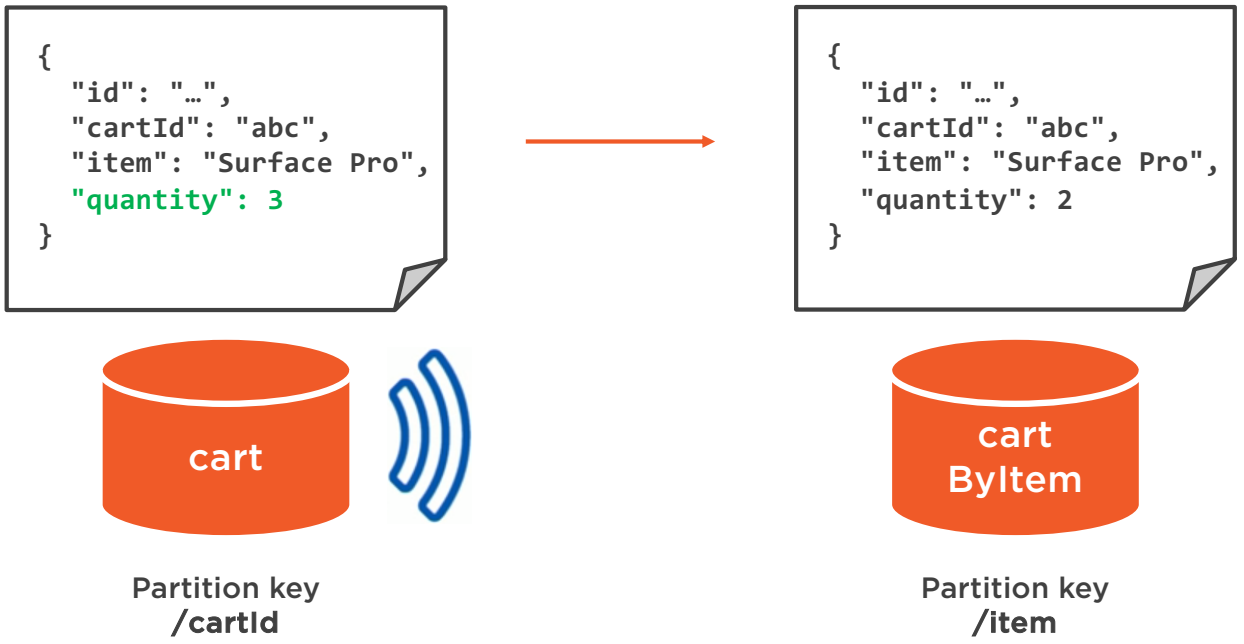
Deletes

Deletes are not written to the change feed

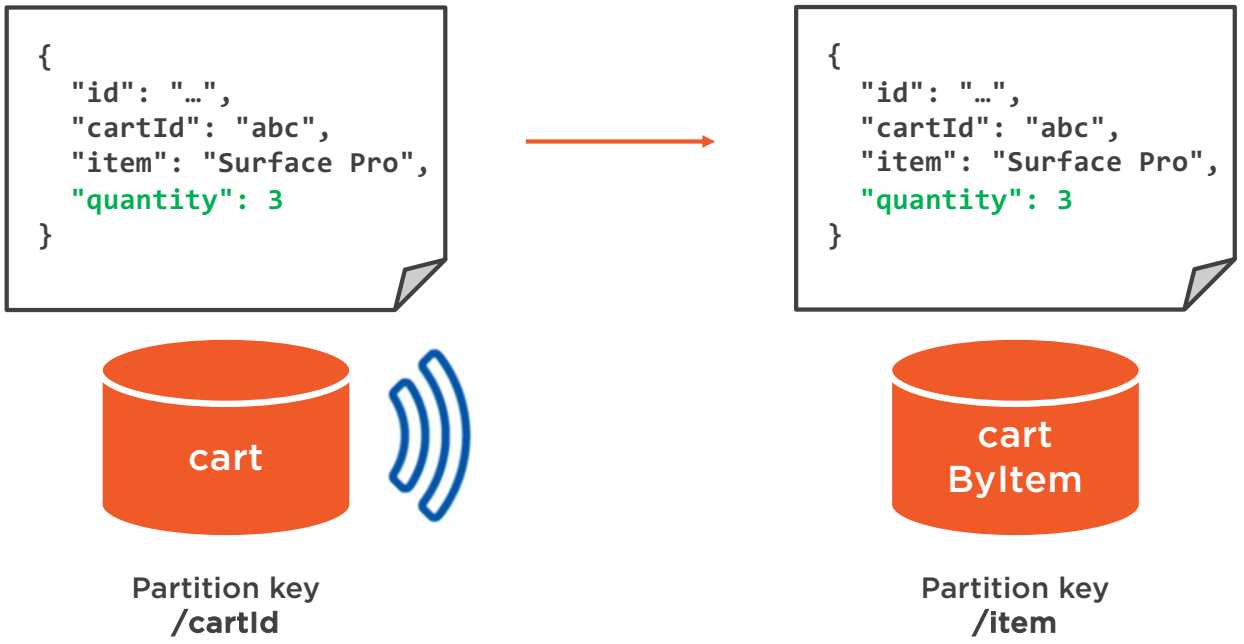
Processing Updates and Deletes



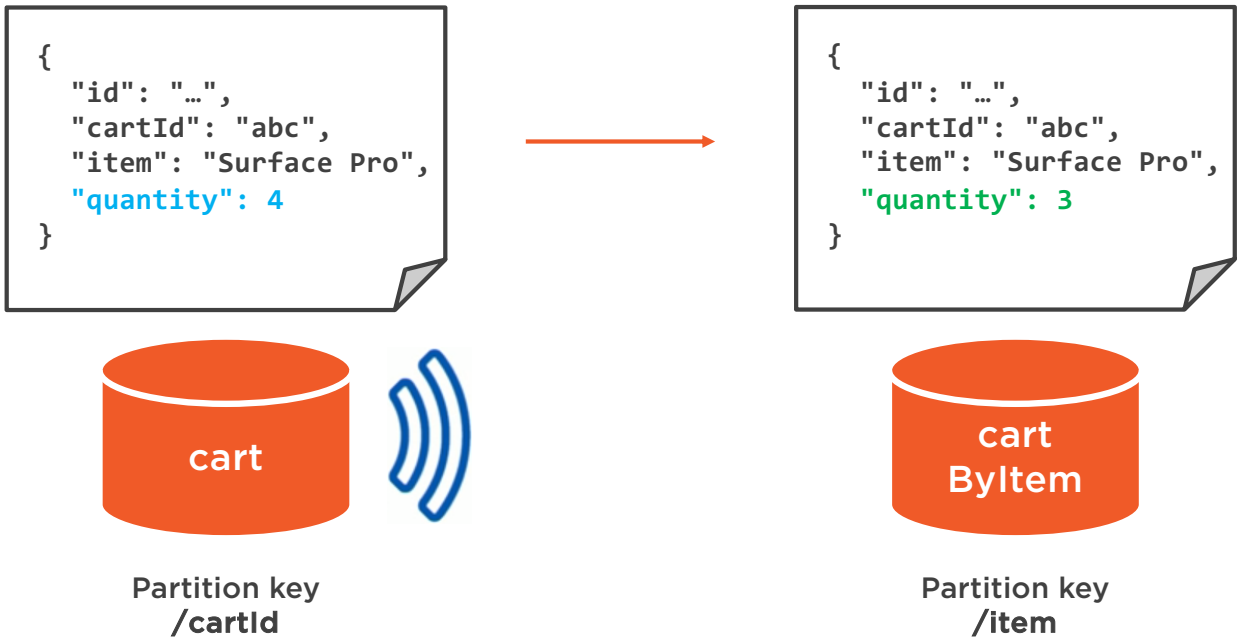
Processing Updates and Deletes



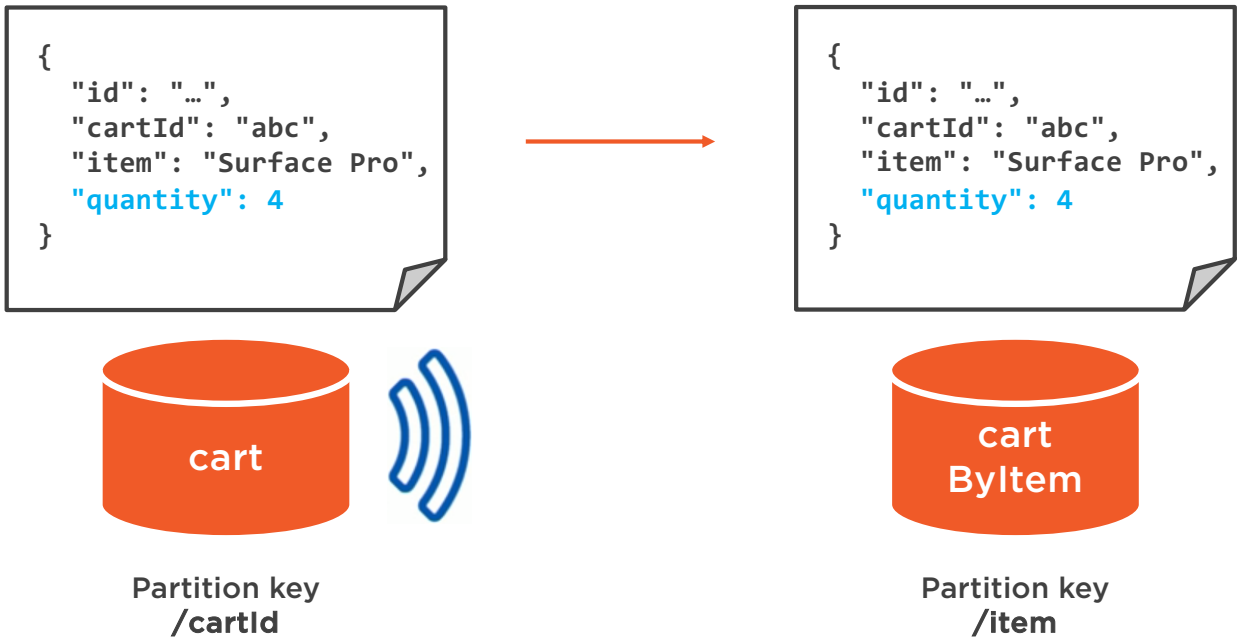
Processing Updates and Deletes



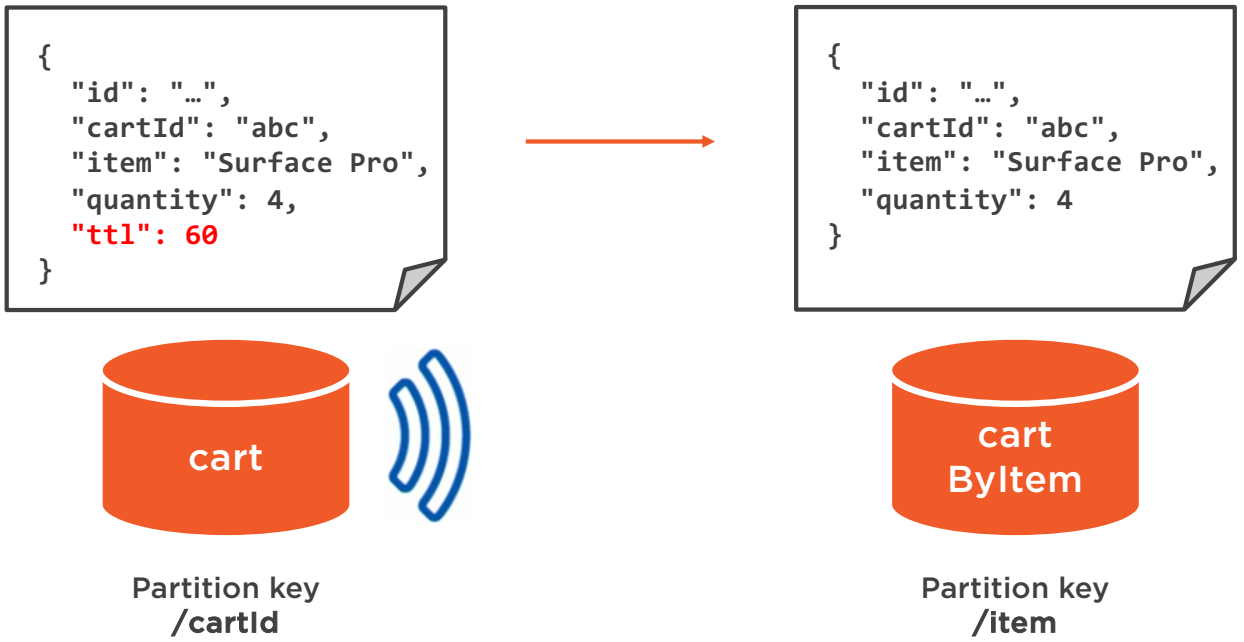
Processing Updates and Deletes



Processing Updates and Deletes



Processing Updates and Deletes



Using Azure Functions

CFP Library

Host is deployed to an Azure service

E.g., web job, worker role

Dedicated capacity

Azure Functions

Serverless, reactive functions

Shared capacity

Dedicated capacity with AF Premium

Demo

Preparing for Replication

Demo

Creating the Replication Microservice

Demo

Testing the Replication Microservice

Demo

Deploying the Replication Microservice

Summary

Replication microservice

- Multiple containers with identical data
- Each partitioned to satisfy different query patterns
- Recommended only for read-heavy workloads

Updates and deletes

- Interim updates are dropped
- Deletes are dropped (use TTL)

Azure Functions

- CFP Library wrapper
- Serverless, reactive functions
- Create, test, and deploy a microservice