# Getting Started with Amazon DynamoDB

Padma Malligarjunan
Sr. Technical Account Manager
AWS Enterprise Support

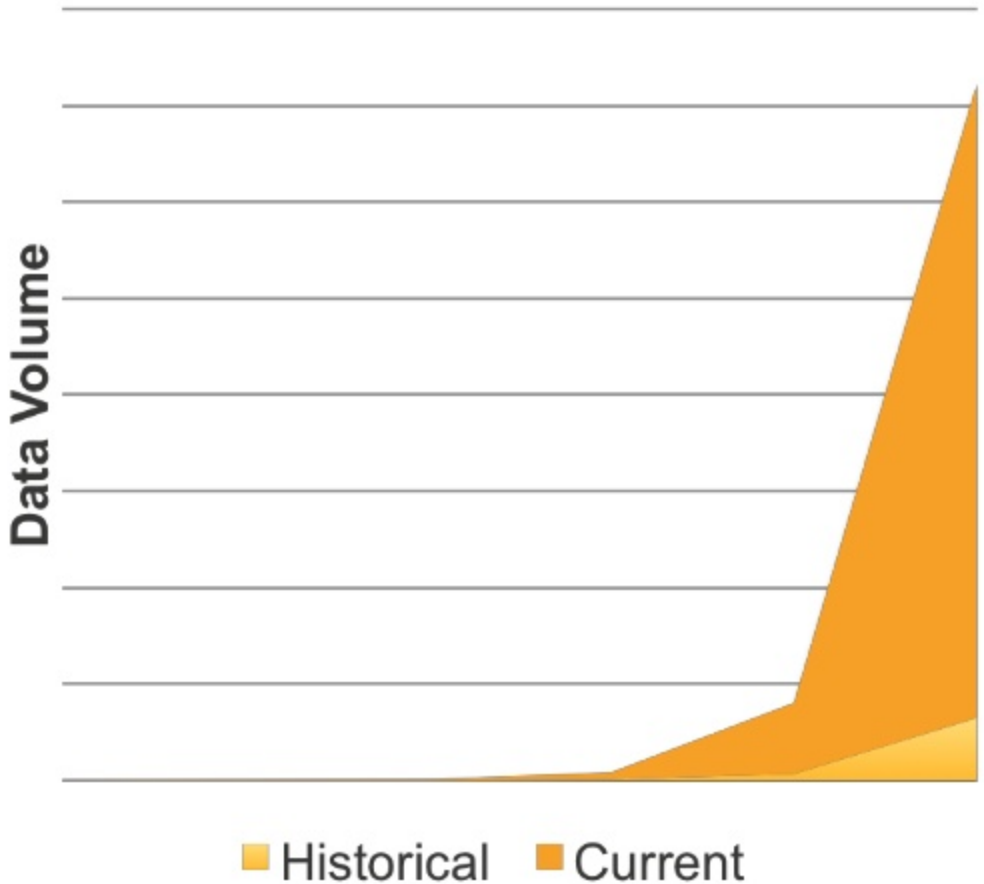

Pop-up Loft

November 02, 2016

# Agenda

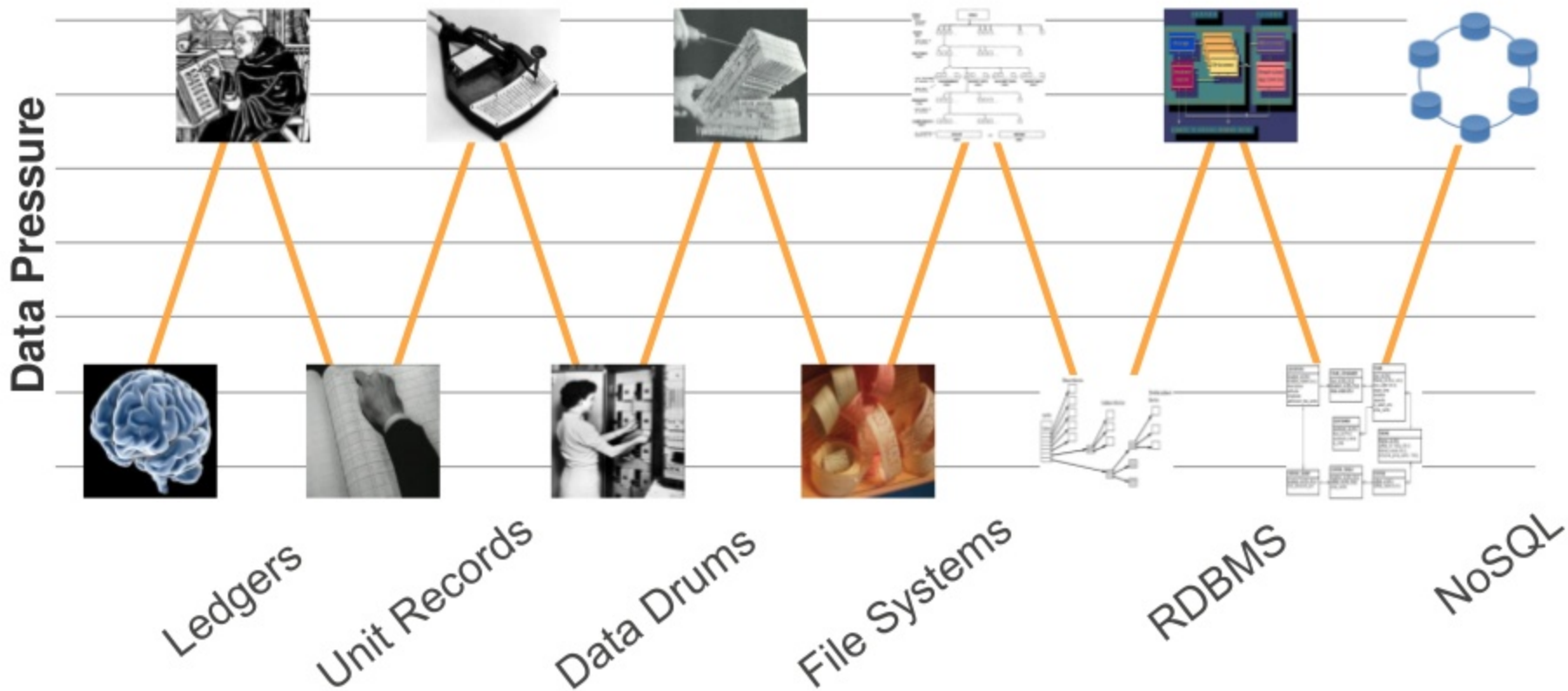
- Brief history of data processing
- Relational (SQL) vs. nonrelational (NoSQL)
- Fully managed features of DynamoDB
- Customer use cases
- Demo – serverless applications
- Pricing and Free Tier

# Data volume since 2010

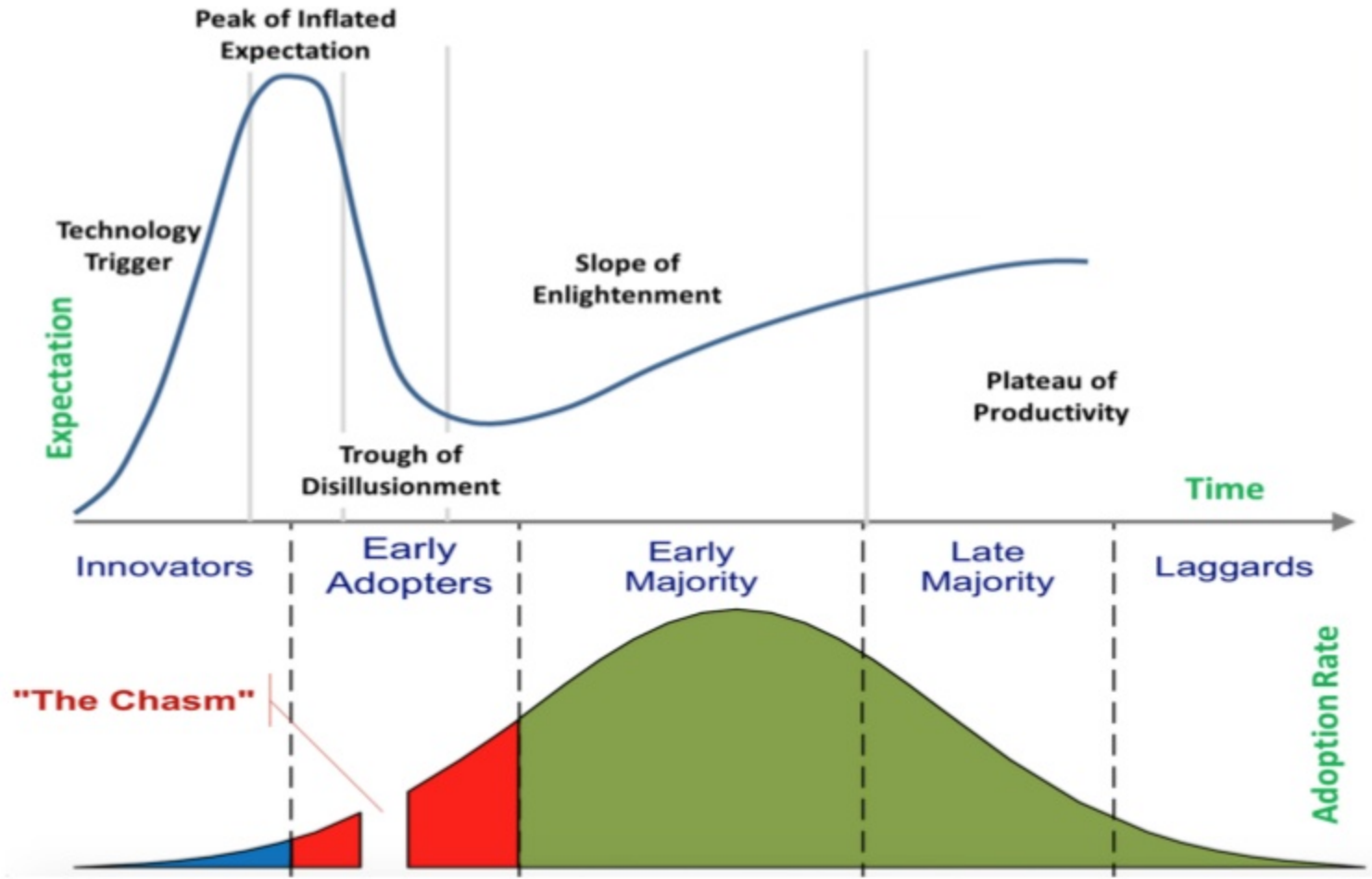



- 90% of stored data generated in last 2 years
- 1 terabyte of data in 2010 equals 6.5 petabytes today
- Linear correlation between data pressure and technical innovation
- No reason these trends will not continue over time

# Timeline of database technology

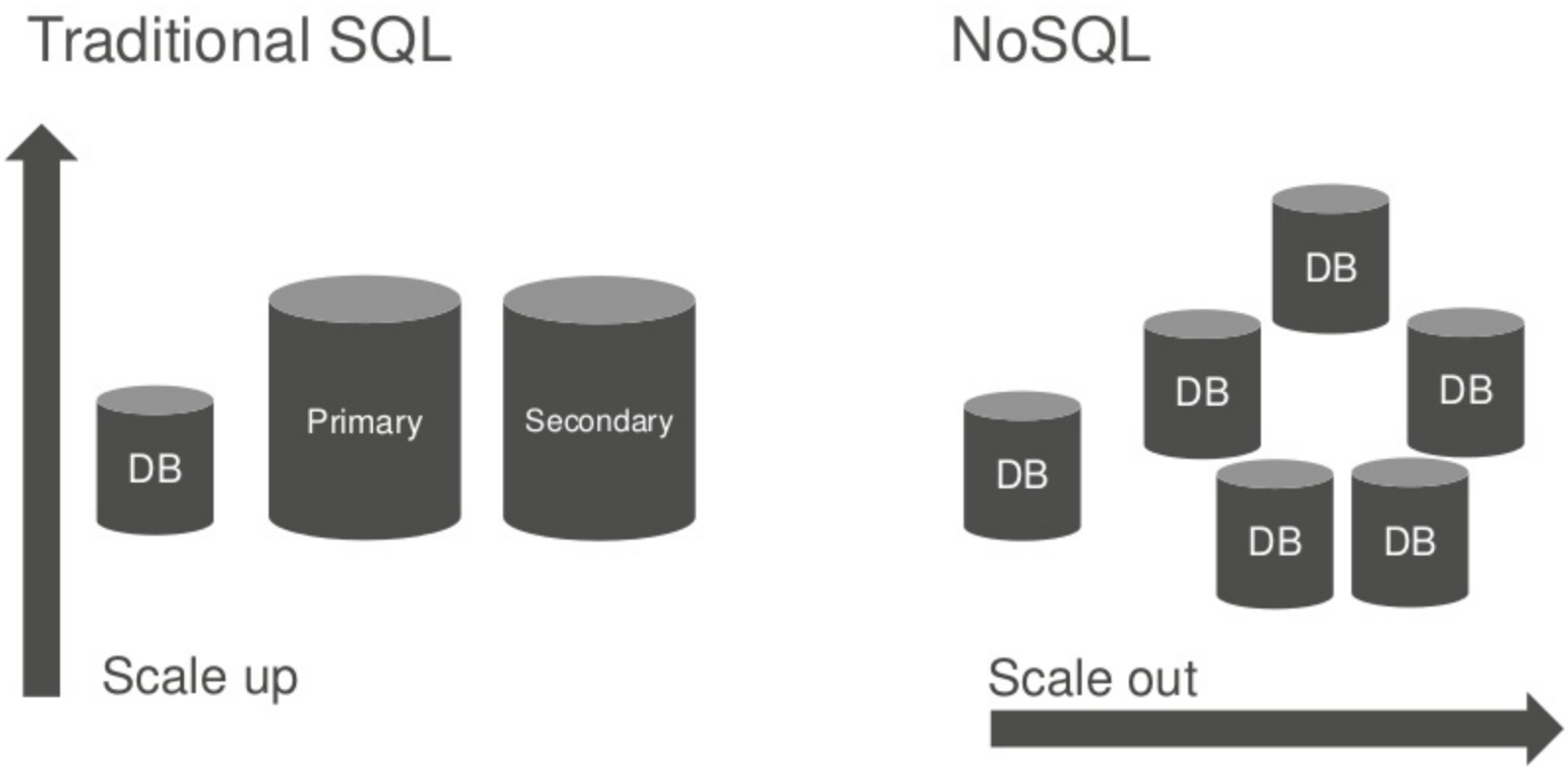


# Technology adoption and the hype curve

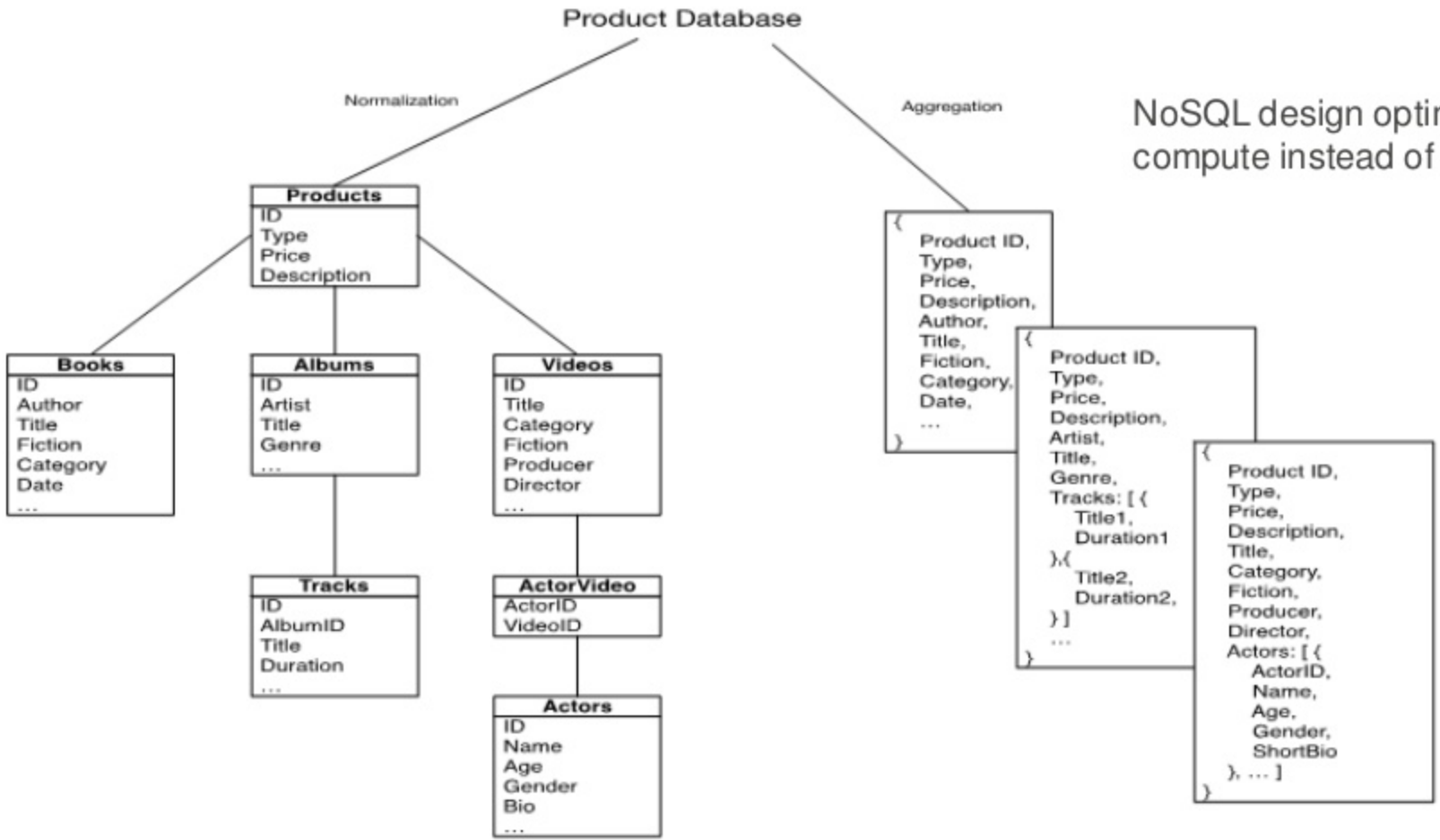


# Relational (SQL) vs. nonrelational (NoSQL)

# Relational vs. nonrelational databases



# SQL vs. NoSQL schema design




NoSQL design optimizes for compute instead of storage

# 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 |

# Amazon DynamoDB

Run your business, not your database

# DynamoDB Benefits

Fully managed

Fast, consistent performance

Highly scalable

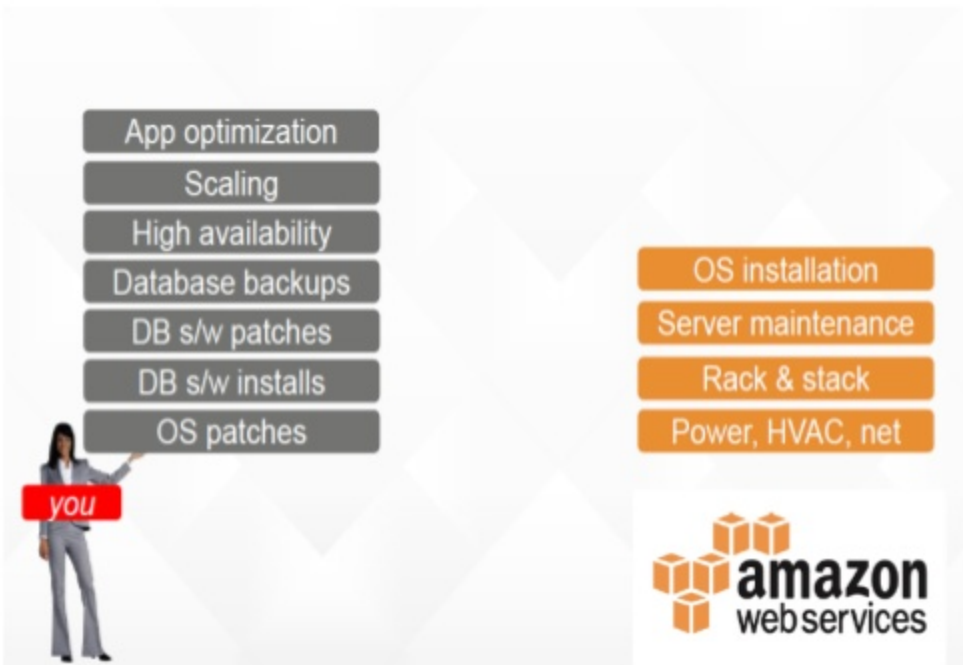
Flexible

Event-driven programming

Fine-grained access control

# Fully managed service = automated operations


DB hosted on-premises



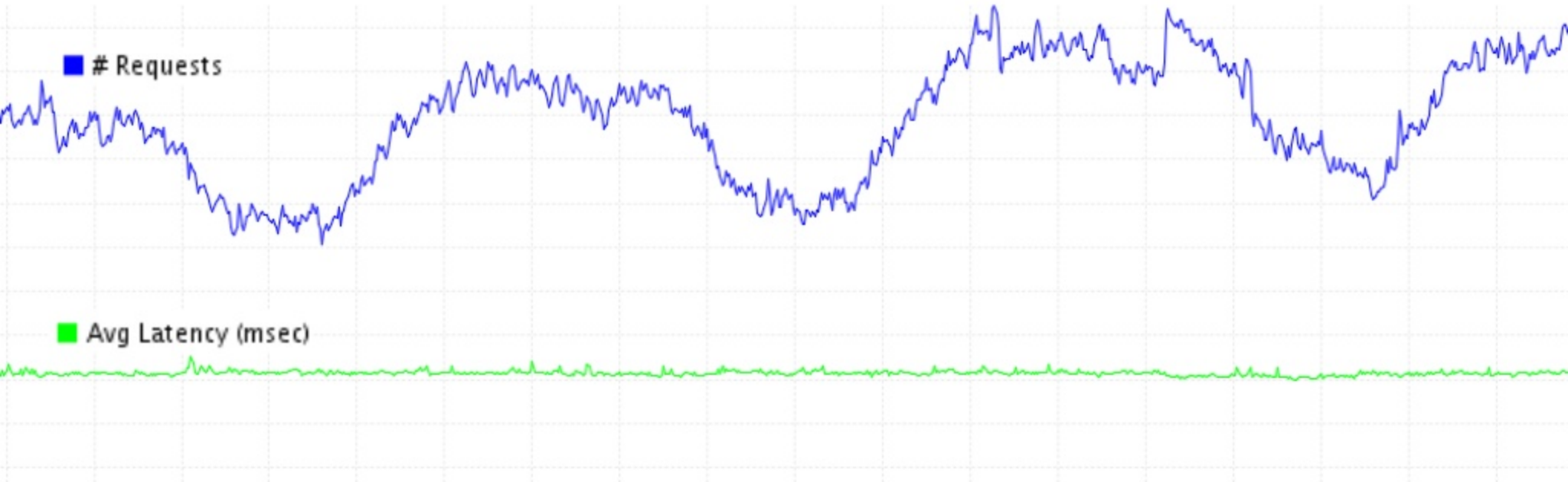

DB hosted on Amazon EC2

# Fully managed service = automated operations


DB hosted on premise

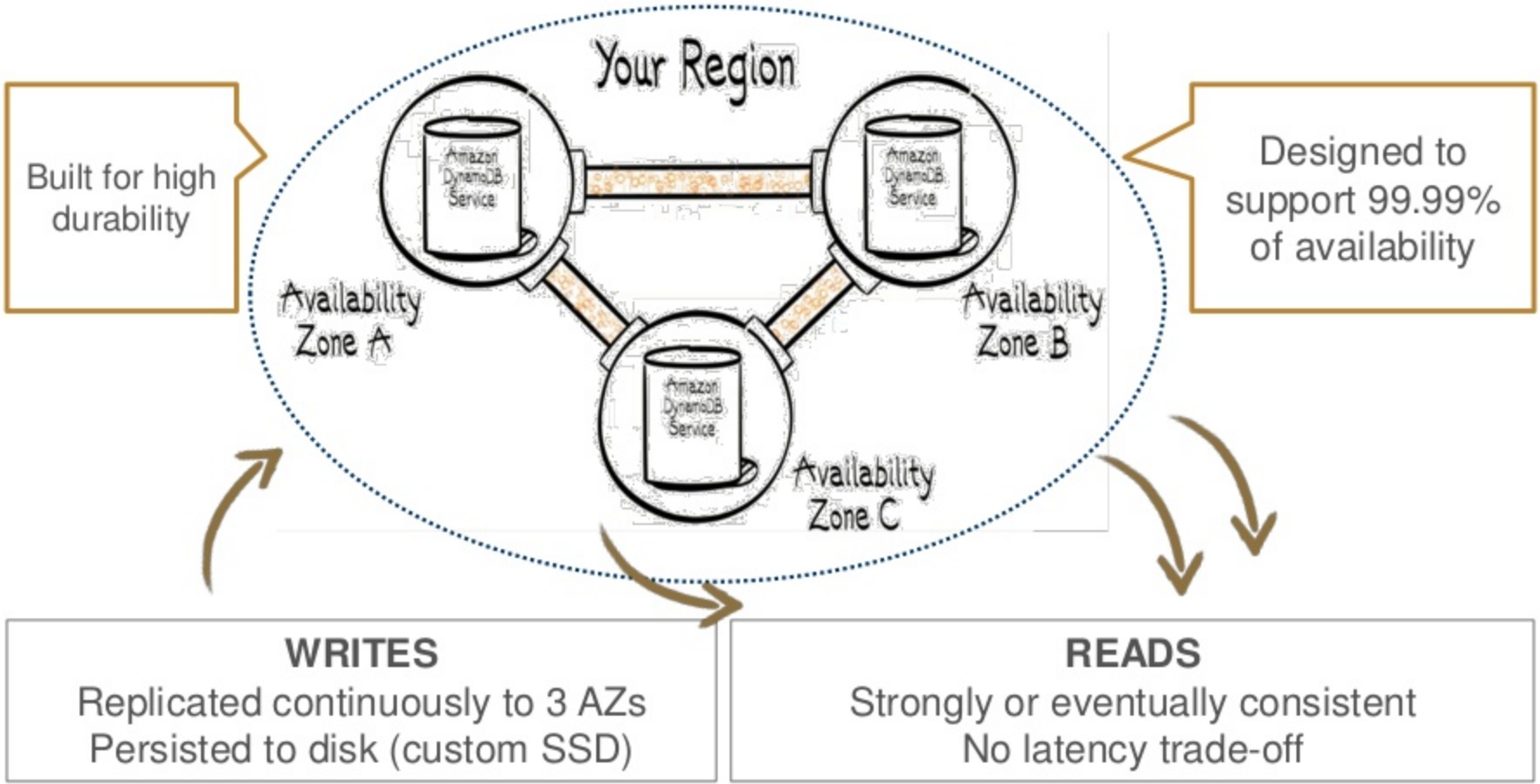

DynamoDB

# **Consistently** low latency at scale




PREDICTABLE
PERFORMANCE!

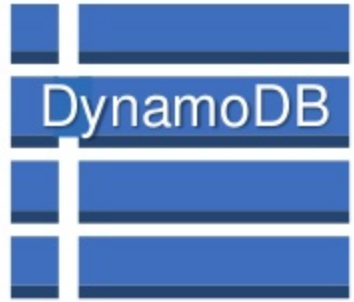
# High availability and durability



# Customer use cases

# Amazon's Path to DynamoDB




**Dynamo: Amazon's Highly Available Key-value Store**

Giuseppe DeCandia, Deniz Hastorun, Madan Jampani, Gunavardhan Kakulapati, Avinash Lakshman, Alex Pilchin, Swaminathan Sivasubramanian, Peter Vosshall and Werner Vogels

Amazon.com

**ABSTRACT**

Reliability at massive scale is one of the biggest challenges we face at Amazon.com, one of the largest e-commerce operations in the world; even the slightest outage has significant financial consequences and impacts customer trust. The Amazon.com platform, which provides services for many web sites worldwide, is implemented on top of an infrastructure of tens of thousands of servers and network components located in many datacenters around the world. At this scale, small and large components fail continuously and the way persistent state is managed in the face of these failures drives the reliability and scalability of the software systems.

This paper presents the design and implementation of Dynamo, a highly available key-value storage system that some of Amazon's core services use to provide an "always-on" experience. To achieve this level of availability, Dynamo sacrifices consistency under certain failure scenarios. It makes extensive use of object versioning and application-assisted conflict resolution in a manner that provides a novel interface for developers to use.

One of the lessons our organization has learned from operating Amazon's platform is that the reliability and scalability of a system is dependent on how its application state is managed. Amazon uses a highly decentralized, loosely coupled, service oriented architecture consisting of hundreds of services. In this environment there is a particular need for storage technologies that are always available. For example, customers should be able to view and add items to their shopping cart even if disks are failing, network routes are flapping, or data centers are being destroyed by tornados. Therefore, the service responsible for managing shopping carts requires that it can always write to and read from its data store, and that its data needs to be available across multiple data centers.

Dealing with failures in an infrastructure comprised of millions of components is our standard mode of operation; there are always a small but significant number of server and network components that are failing at any given time. As such Amazon's software systems need to be constructed in a manner that treats failure handling as the normal case without impacting availability or performance.

# Major League Baseball Fields Big Data, Excitement with Amazon DynamoDB

> For the first time, we can measure things we've never been able to measure before.
>
> **Joe Inzerillo**
> Executive Vice President and CTO, MLBAM

mlbam

MLBAM (MLB Advanced Media) is a full service solutions provider, operating a powerful content delivery platform.

- MLBAM can scale to support many games on a single day.
- Amazon DynamoDB powers queries and supports the fast data retrieval required.
- MLBAM distributes 25,000 live events annually and 10 million streams daily.

# Redfin Is Revolutionizing Home Buying and Selling with Amazon DynamoDB

> We have billions of records on DynamoDB being refreshed daily or hourly or even by seconds.
>
> **Yong Huang**
> Director, Big Data Analytics, Redfin

REDFIN

Redfin is a full-service real estate company with local agents and online tools to help people buy & sell homes.

- Redfin provides property and agent details and ratings through its websites and apps.
- With DynamoDB, latency for “similar” properties improved from 2 seconds to just 12 milliseconds.
- Redfin stores and processes five billion items in DynamoDB.

# Expedia's Real-time Analytics Application Uses Amazon DynamoDB

> With DynamoDB we were up and running in a less than day, and there is no need for a team to maintain.
>
> **Kuldeep Chowhan**
> Principal Engineer, Expedia


Expedia is a leader in the $1 trillion travel industry, with an extensive portfolio that includes some of the world's most trusted travel brands.

- Expedia's real-time analytics application collects data for its "test & learn" experiments on Expedia sites.
- The analytics application processes ~200 million messages daily.
- Ease of setup, monitoring, and scaling were key factors in choosing Amazon DynamoDB.

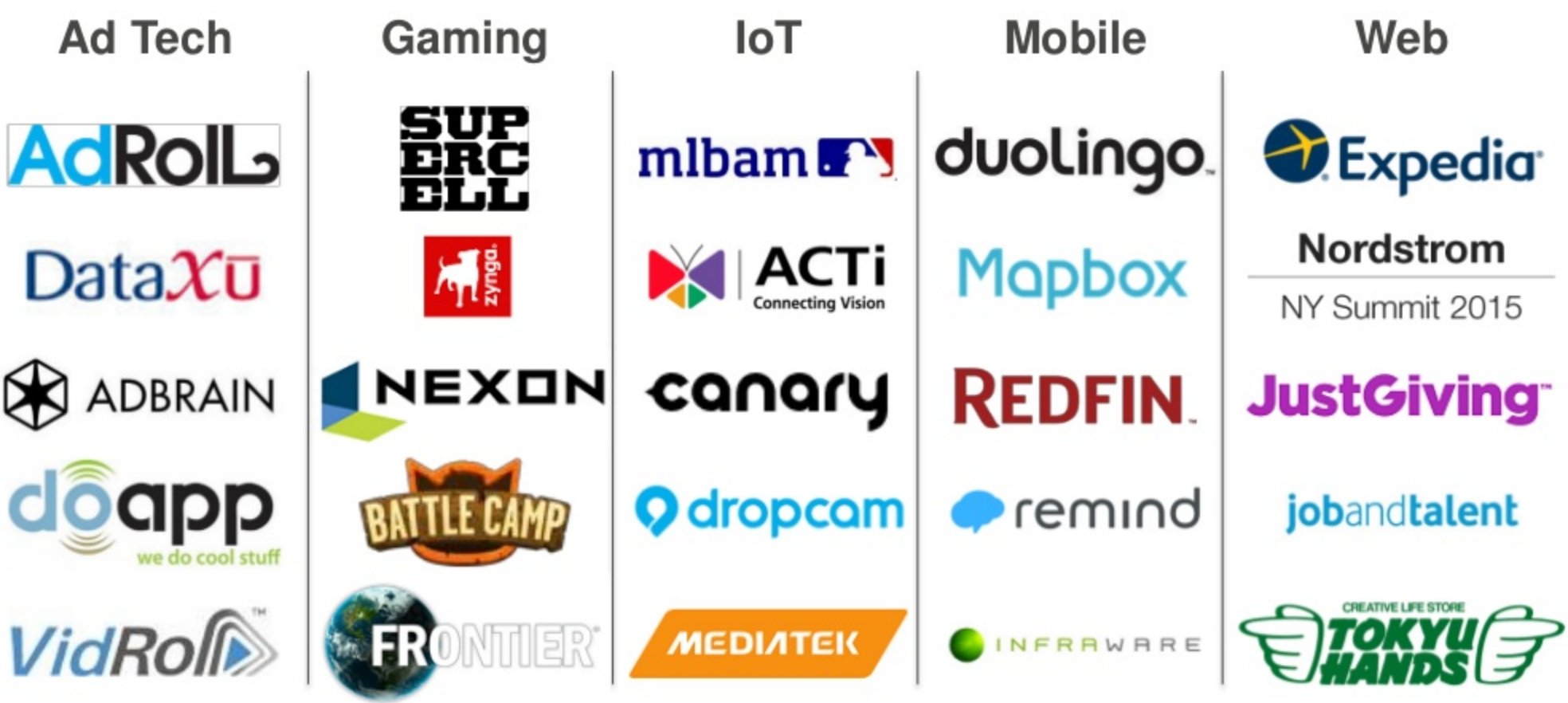
# Nexon Scales Mobile Gaming with Amazon DynamoDB

> By using AWS, we decreased our initial investment costs, and only pay for what we use.
>
> **Chunghoon Ryu**
> Department Manager, Nexon


Nexon is a leading South Korean video game developer and a pioneer in the world of interactive entertainment.

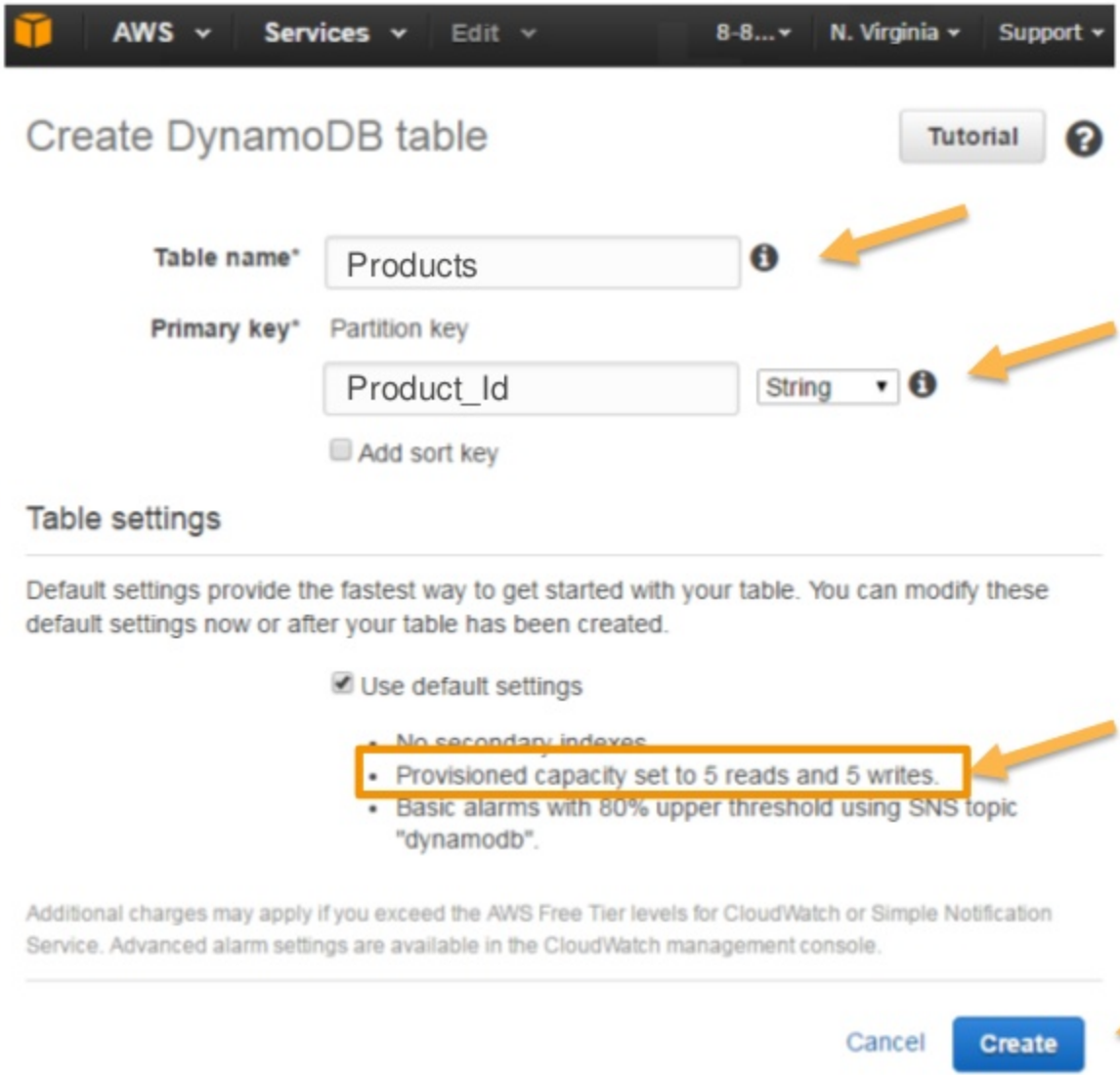
- Nexon used Amazon DynamoDB as its primary game database for a new blockbuster mobile game, HIT
- HIT became the #1 Mobile Game in Korea within the first day of launch and has > 2M registered users
- Nexon's HIT leverages DynamoDB to deliver steady latency of less than 10ms to deliver a fantastic mobile gaming experience for 170,000 concurrent players

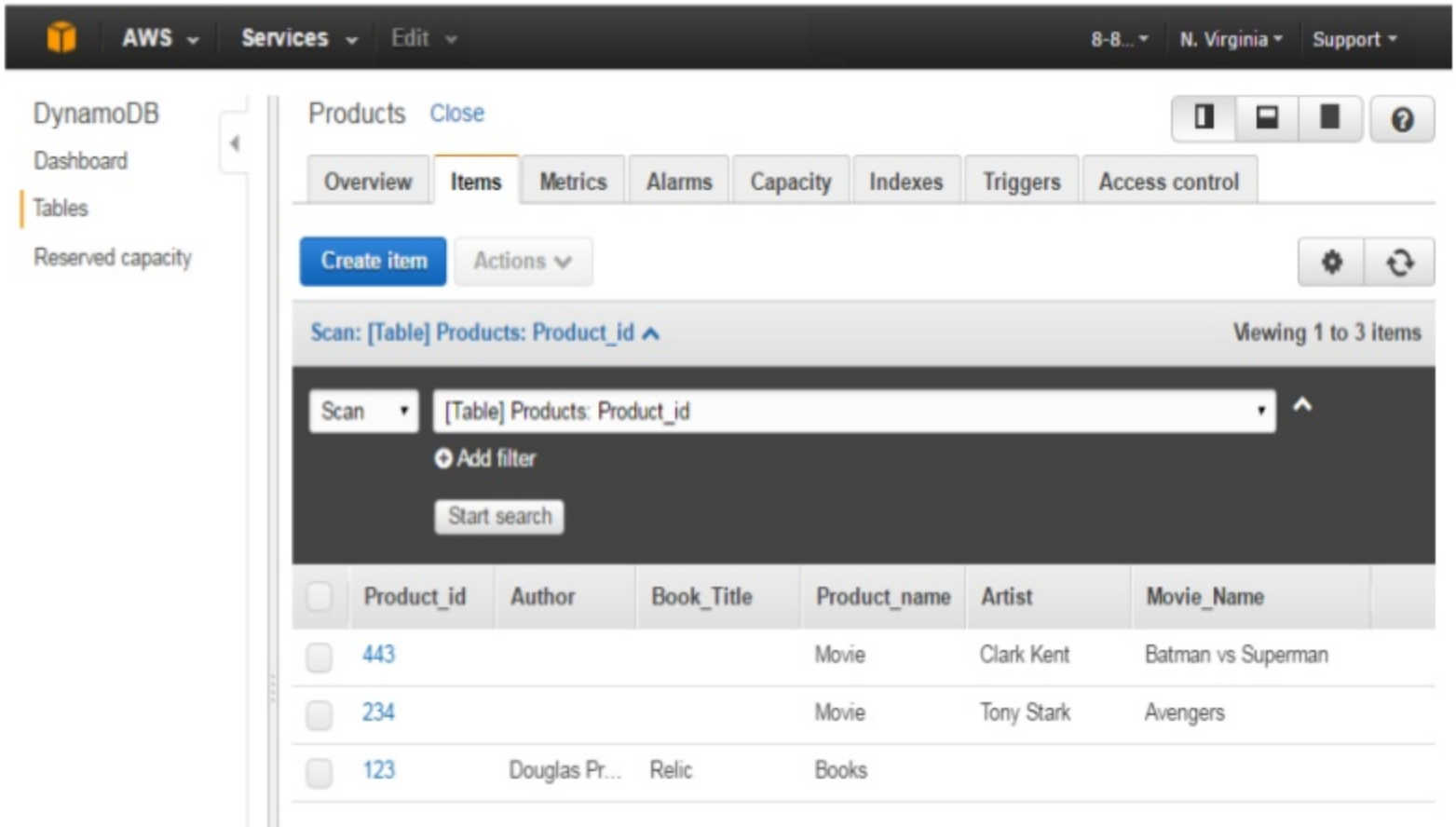
# Scaling high-velocity use cases with DynamoDB



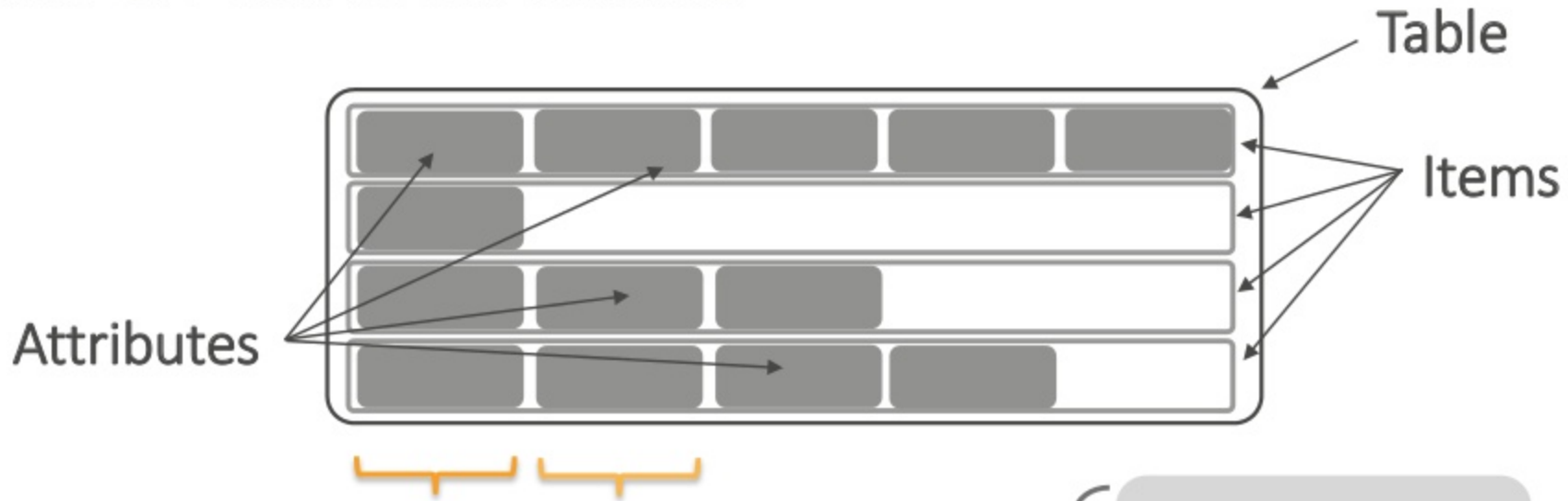
**That sounds really good. How do I get started?**

**Let’s create a table..**


AWS
Services
Edit
8-8...
N. Virginia
Support
Create DynamoDB table
Tutorial
Table name*
Products
Primary key*
Partition key
Product_Id
String
Add sort key
Table settings
Default settings provide the fastest way to get started with your table. You can modify these default settings now or after your table has been created.
Use default settings
Provisioned capacity set to 5 reads and 5 writes.
Basic alarms with 80% upper threshold using SNS topic "dynamodb".
Additional charges may apply if you exceed the AWS Free Tier levels for CloudWatch or Simple Notification Service. Advanced alarm settings are available in the CloudWatch management console.
Cancel
Create


AWS
Services
Edit
8-8...
N. Virginia
Support
DynamoDB
Dashboard
Tables
Reserved capacity
Products
Close
Overview
Items
Metrics
Alarms
Capacity
Indexes
Triggers
Access control
Create item
Actions
Scan: [Table] Products: Product_id
Viewing 1 to 3 items
Scan
[Table] Products: Product_id
Add filter
Start search
Product_id
Author
Book_Title
Product_name
Artist
Movie_Name
443
Movie
Clark Kent
Batman vs Superman
234
Movie
Tony Stark
Avengers
123
Douglas Pr...
Relic
Books

# DynamoDB table structure




Mandatory
Key-value access pattern
Determines data distribution

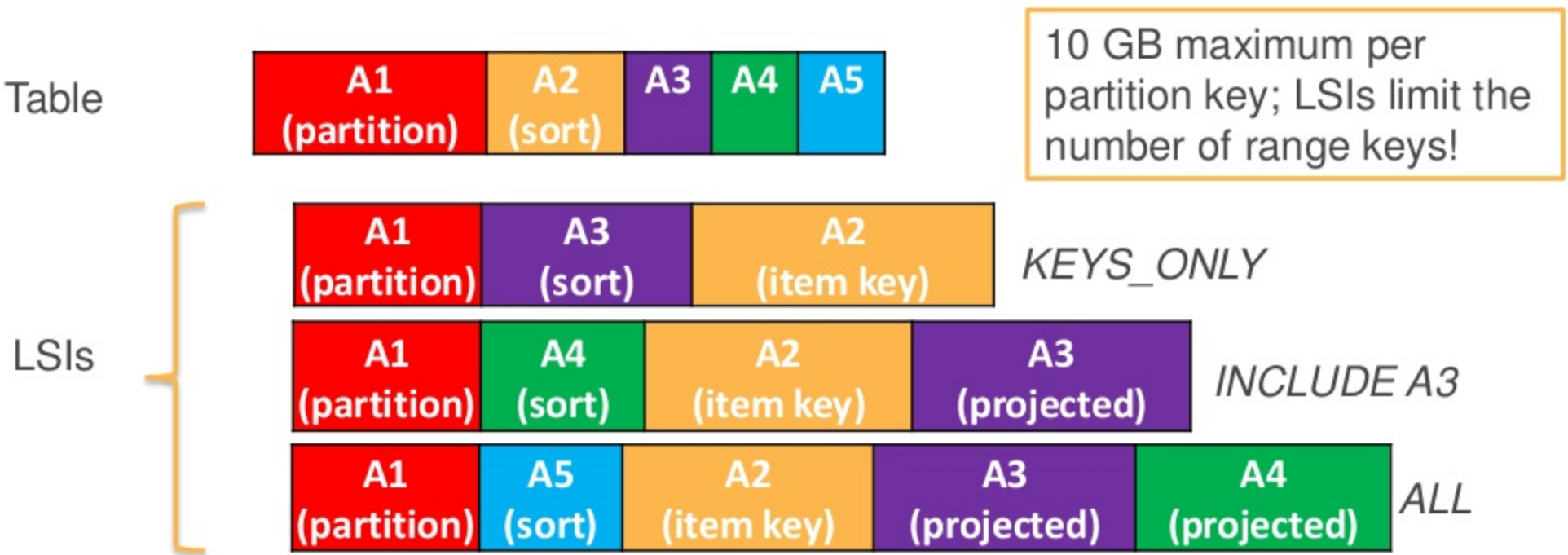
Optional
Model 1:N relationships
Enables rich query capabilities

All items for key
==, <, >, >=, <=
“begins with”
“between”
“contains”
“in”
sorted results
counts
top/bottom N values

# Local secondary index (LSI)

Alternate sort key attribute

Index is local to a partition key



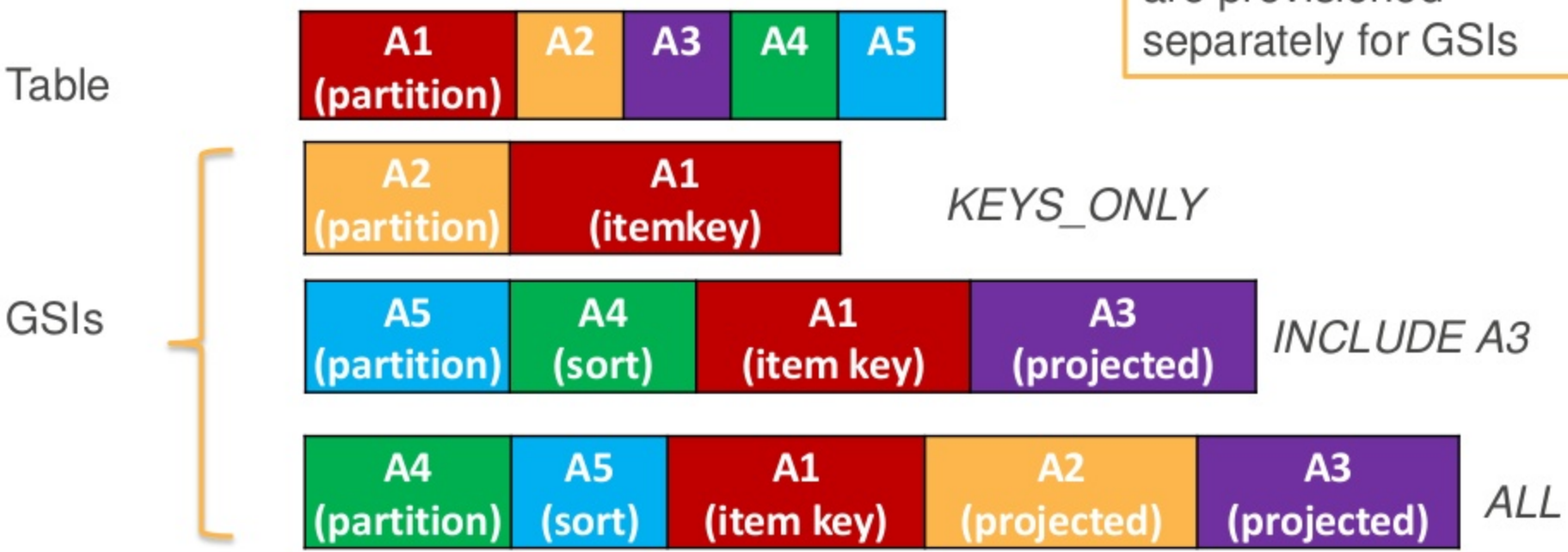
# Global secondary index (GSI)

Alternate partition and/or sort key

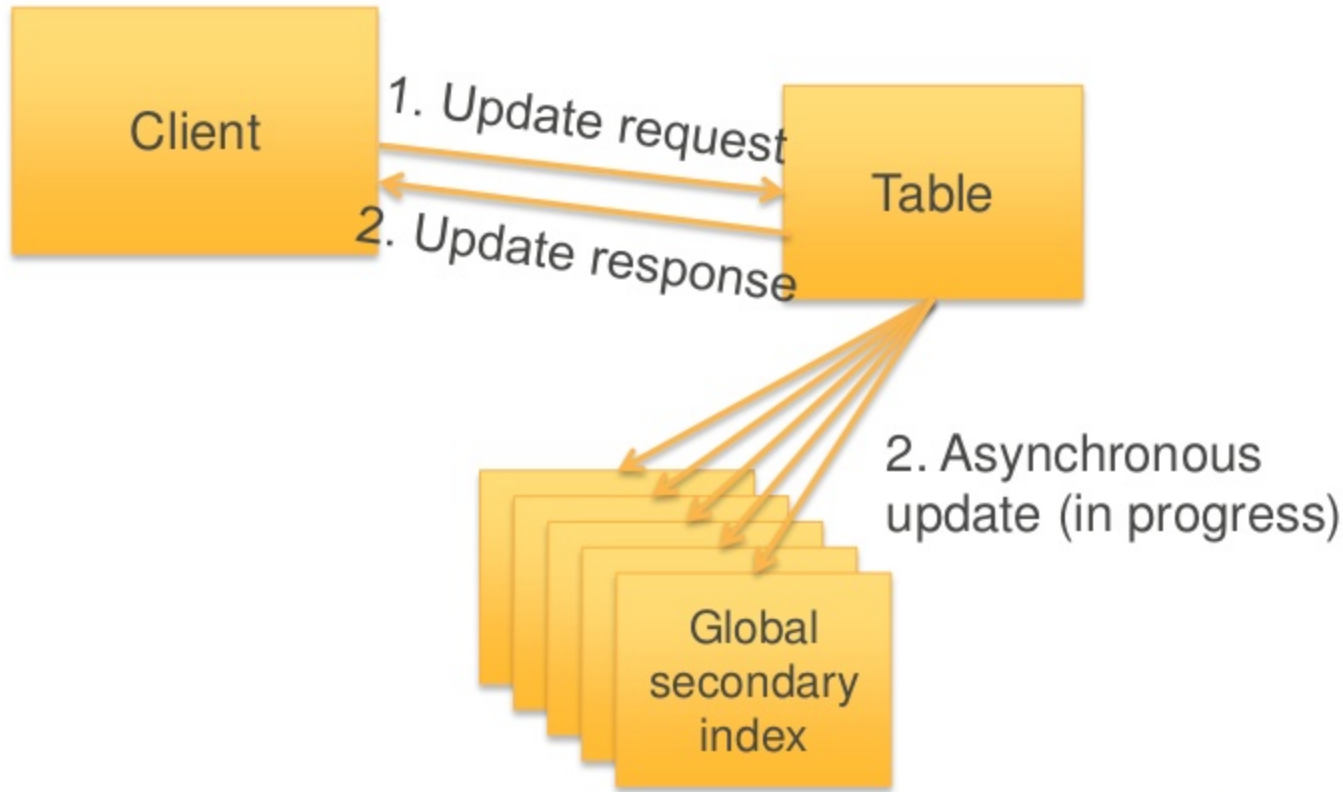
Index is across all partition keys

Online indexing

Read capacity units (RCUs) and write capacity units (WCUs) are provisioned separately for GSIs



# How do GSI updates work?




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

# LSI or GSI?

LSI can be modeled as a GSI

If data size in an item collection > 10 GB, use GSI

**If eventual consistency is okay for your scenario, use GSI!**

# Advanced topics in DynamoDB

- Data modeling
- Understanding Partitions
  - # of partitions depend on table throughput and size
- DynamoDB Scaling
- Design patterns and best practices

# To learn more, please attend:

**Deep Dive on Amazon DynamoDB**
Sean Shriver, AWS NoSQL Solutions Architect

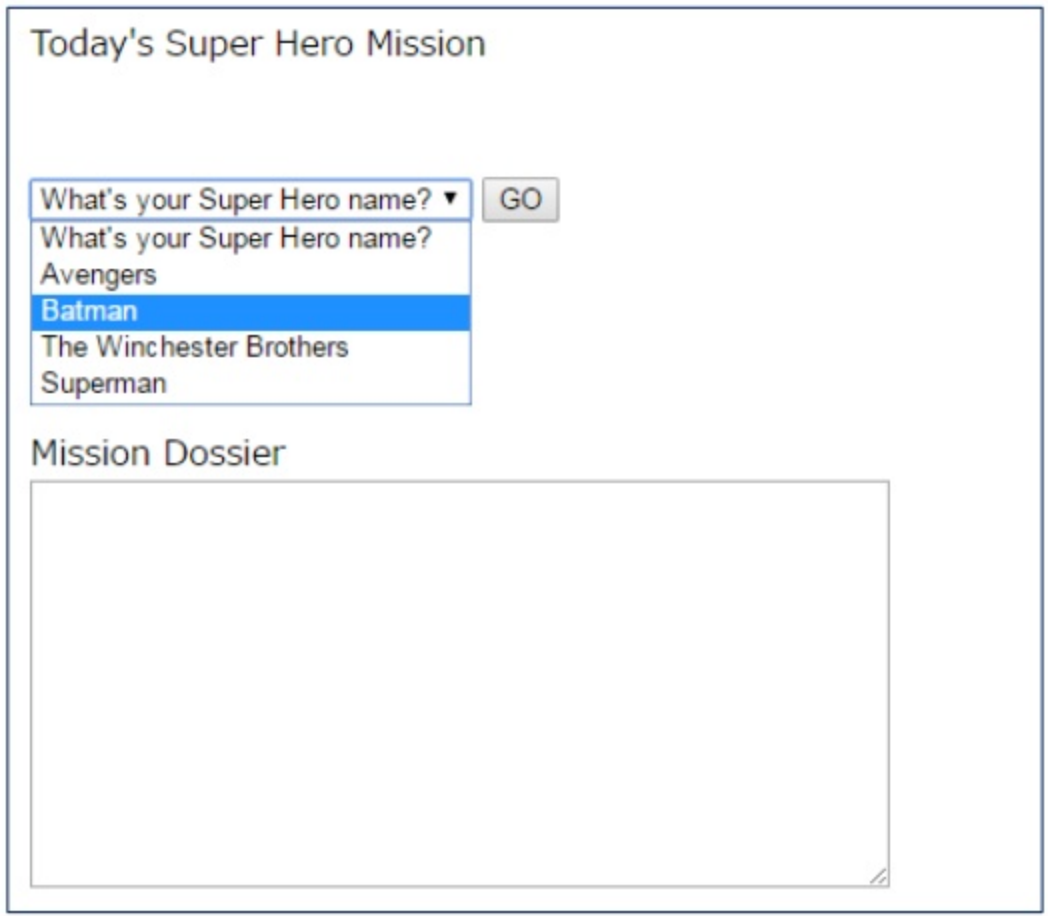
**Hands on 90 minute lab (Serverless Web apps)**
Sam Elmanak & Scott Kellish

**Amazon DynamoDB:** https://aws.amazon.com/dynamodb/

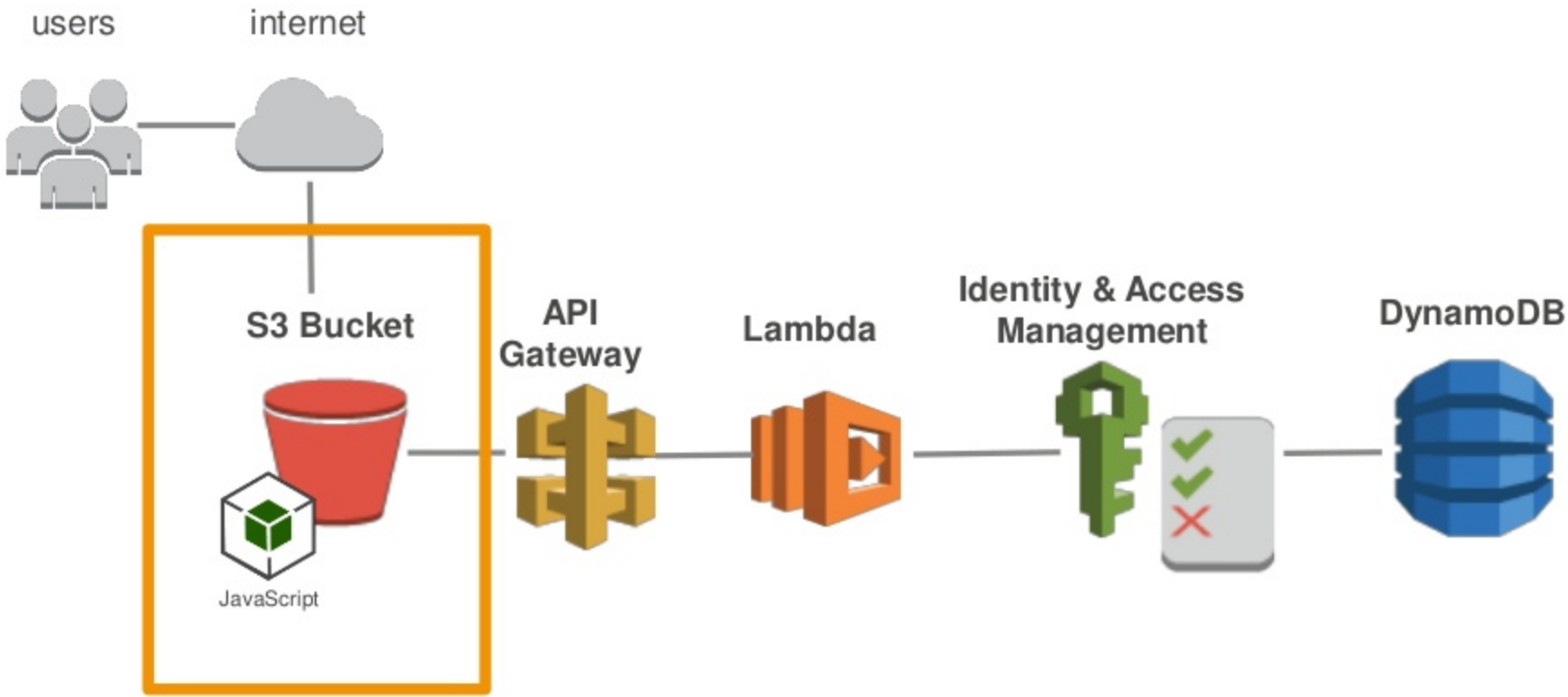
# Demo

# Serverless Web Apps with Amazon DynamoDB, API Gateway, and AWS Lambda

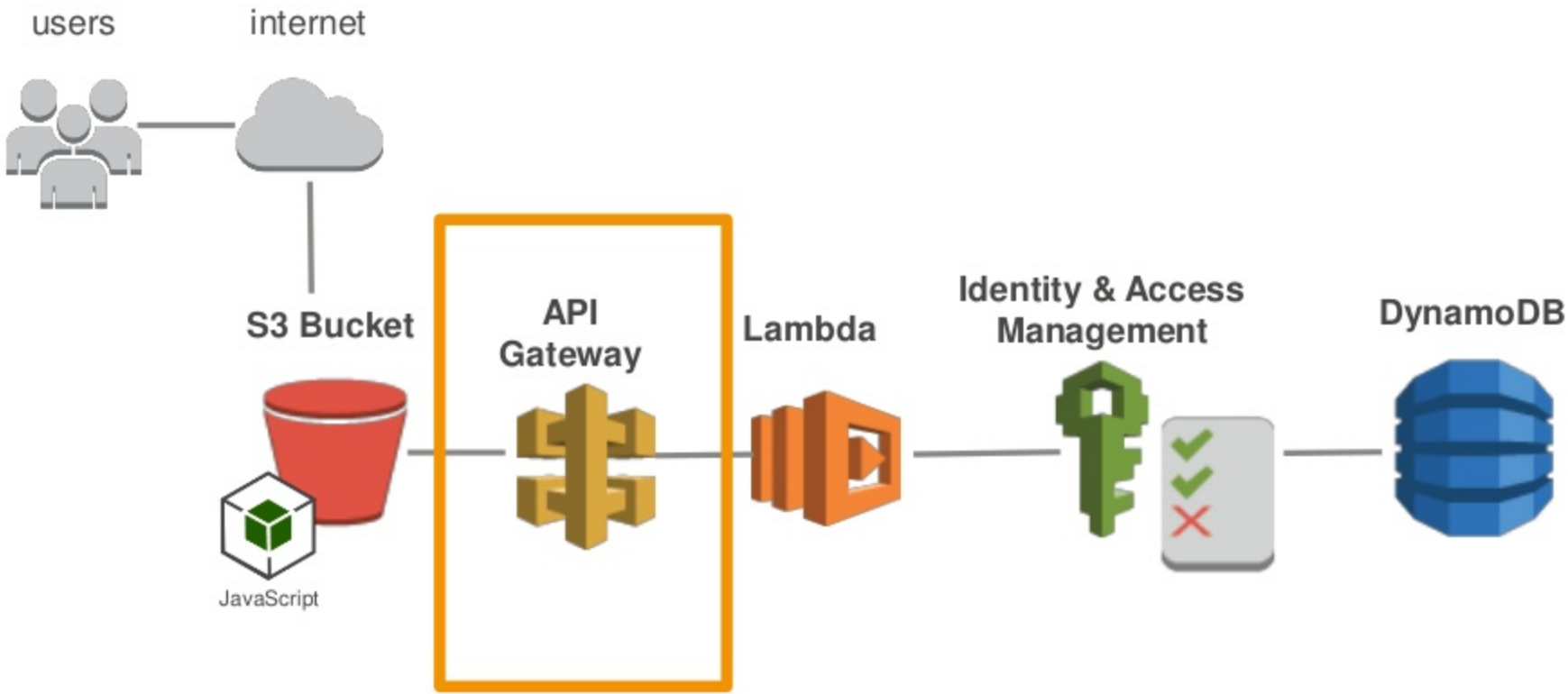
# Simple serverless web application – use case



# Architecture of a simple serverless web application

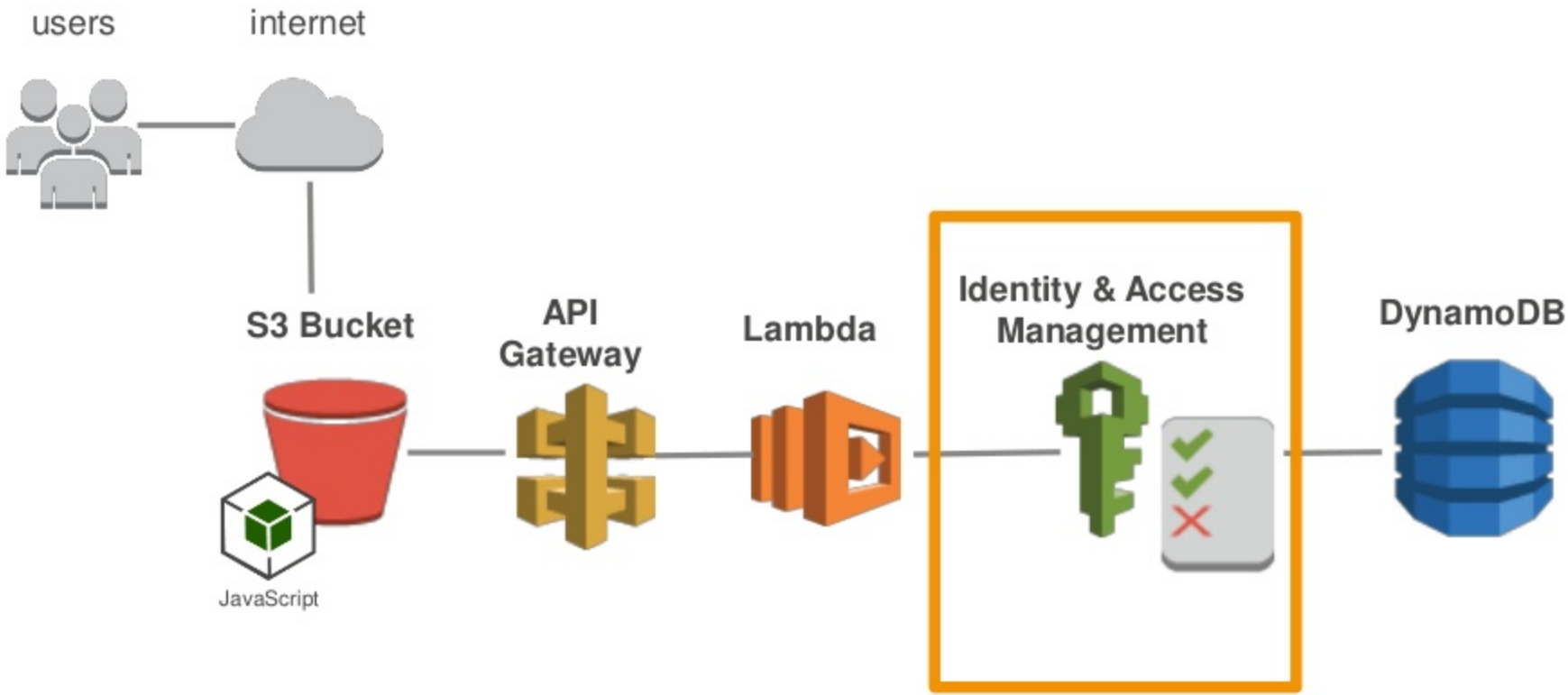


# Architecture of a simple serverless web application

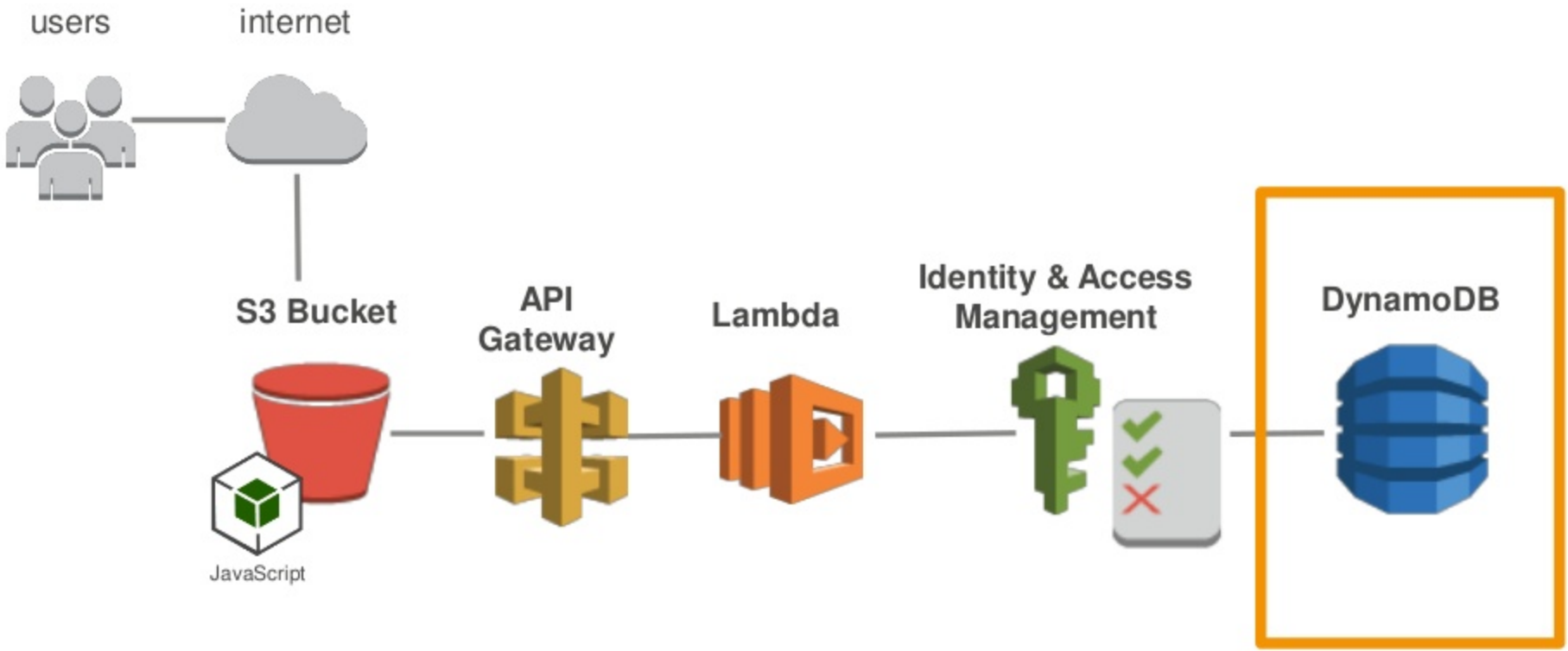


# Architecture of a simple serverless web application

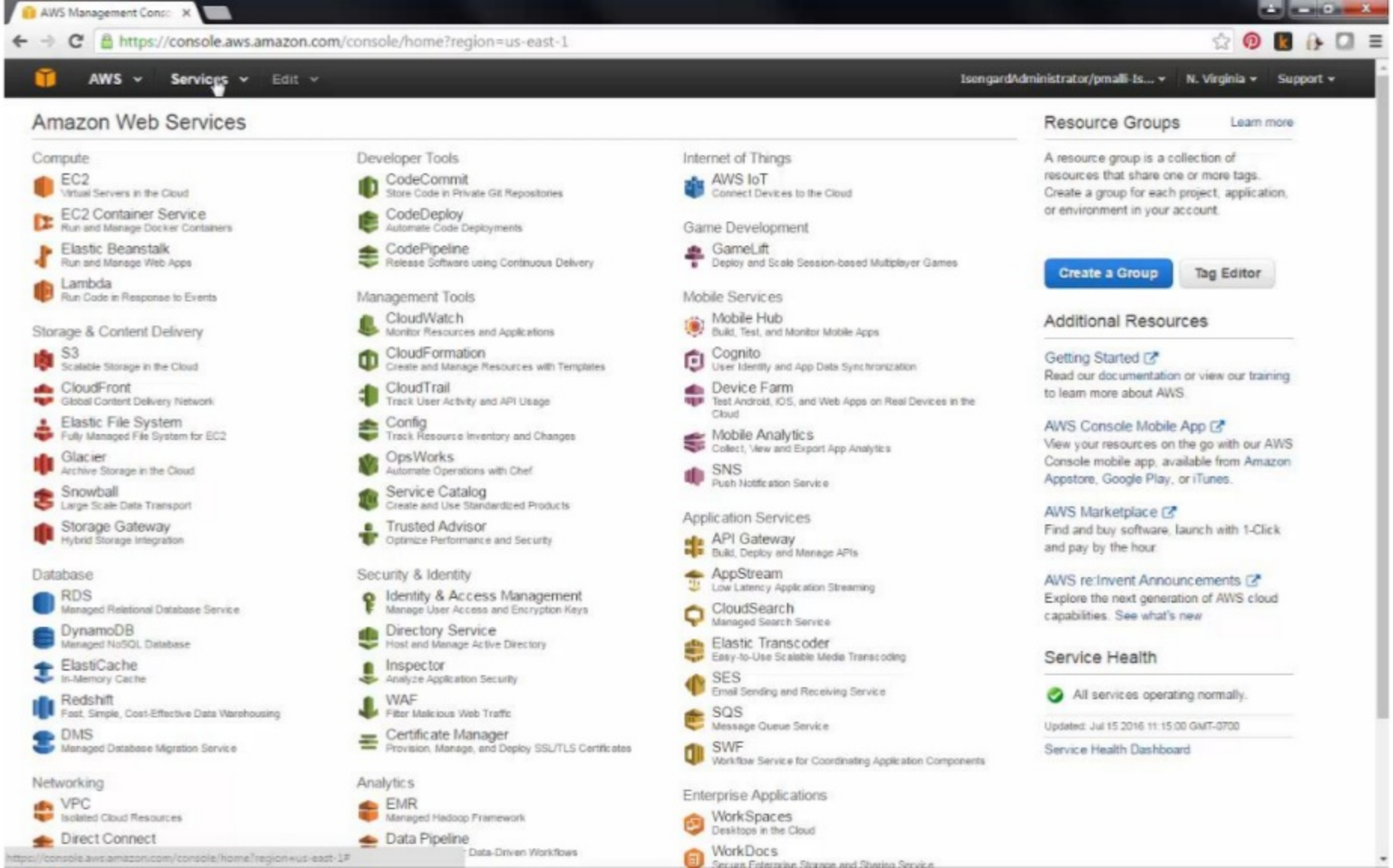
# Architecture of a simple serverless web application



# Architecture of a simple serverless web application



# Demo


AWS Management Conso
https://console.aws.amazon.com/console/home?region=us-east-1
AWS
Services
Edit
IsengardAdministrator/pmalli-Is...
N. Virginia
Support
Amazon Web Services
Compute
EC2
Virtual Servers in the Cloud
EC2 Container Service
Run and Manage Docker Containers
Elastic Beanstalk
Run and Manage Web Apps
Lambda
Run Code in Response to Events
Storage & Content Delivery
S3
Scalable Storage in the Cloud
CloudFront
Global Content Delivery Network
Elastic File System
Fully Managed File System for EC2
Glacier
Archive Storage in the Cloud
Snowball
Large Scale Data Transport
Storage Gateway
Hybrid Storage Integration
Database
RDS
Managed Relational Database Service
DynamoDB
Managed NoSQL Database
ElastiCache
In-Memory Cache
Redshift
Fast, Simple, Cost-Effective Data Warehousing
DMS
Managed Database Migration Service
Networking
VPC
Isolated Cloud Resources
Direct Connect
Developer Tools
CodeCommit
Store Code in Private Git Repositories
CodeDeploy
Automate Code Deployments
CodePipeline
Release Software using Continuous Delivery
Management Tools
CloudWatch
Monitor Resources and Applications
CloudFormation
Create and Manage Resources with Templates
CloudTrail
Track User Activity and API Usage
Config
Track Resource Inventory and Changes
OpsWorks
Automate Operations with Chef
Service Catalog
Create and Use Standardized Products
Trusted Advisor
Optimize Performance and Security
Security & Identity
Identity & Access Management
Manage User Access and Encryption Keys
Directory Service
Host and Manage Active Directory
Inspector
Analyze Application Security
WAF
Filter Malicious Web Traffic
Certificate Manager
Provision, Manage, and Deploy SSL/TLS Certificates
Analytics
EMR
Managed Hadoop Framework
Data Pipeline
Data-Driven Workflows
Internet of Things
AWS IoT
Connect Devices to the Cloud
Game Development
GameLift
Deploy and Scale Session-based Multiplayer Games
Mobile Services
Mobile Hub
Build, Test, and Monitor Mobile Apps
Cognito
User Identity and App Data Synchronization
Device Farm
Test Android, iOS, and Web Apps on Real Devices in the Cloud
Mobile Analytics
Collect, View and Export App Analytics
SNS
Push Notification Service
Application Services
API Gateway
Build, Deploy and Manage APIs
AppStream
Low Latency Application Streaming
CloudSearch
Managed Search Service
Elastic Transcoder
Easy-to-Use Scalable Media Transcoding
SES
Email Sending and Receiving Service
SQS
Message Queue Service
SWF
Workflow Service for Coordinating Application Components
Enterprise Applications
WorkSpaces
Desktops in the Cloud
WorkDocs
Resource Groups
Learn more
A resource group is a collection of resources that share one or more tags. Create a group for each project, application, or environment in your account.
Create a Group
Tag Editor
Additional Resources
Getting Started
Read our documentation or view our training to learn more about AWS.
AWS Console Mobile App
View your resources on the go with our AWS Console mobile app, available from Amazon Appstore, Google Play, or iTunes.
AWS Marketplace
Find and buy software, launch with 1-Click and pay by the hour
AWS re:Invent Announcements
Explore the next generation of AWS cloud capabilities. See what's new
Service Health
All services operating normally.
Updated: Jul 15 2016 11:15:00 GMT-0700
Service Health Dashboard
https://console.aws.amazon.com/console/home?region=us-east-1#

# DynamoDB Pricing & Free Tier

- Free Tier
  - 25GB of storage
  - 25 Reads per second
  - 25 Writes per second
- Pricing for additional usage in US East (N. Virginia)
  - $0.25 per GB per month
  - Write throughput: $0.0065 per hour for every 10 units of Write Capacity
  - Read throughput: $0.0065 per hour for every 50 units of Read Capacity

# THANK YOU!

**Don’t forget…**

- **SURVEY:** emailed to you within 12 hours. We value your feedback!
- **SLIDES:** links will display on screen when you complete the survey
- **MORE QUESTIONS?** Visit the Ask an Architect bar on the 1st Floor


Pop-up Loft

# Everything and Anything Startups Need to Get Started on AWS

aws.amazon.com/activate