AWS re:Invent

NOV. 29 – DEC. 3, 2021 | LAS VEGAS, NV

DAT328-R2

Building modern applications with AWS purpose-built databases

Arabinda Pani
Principal Partner Database SA

Ryan Shevchik
Sr. Data and Analytics Specialist SA

© 2021, Amazon Web Services, Inc. or its affiliates. All rights reserved.

Agenda

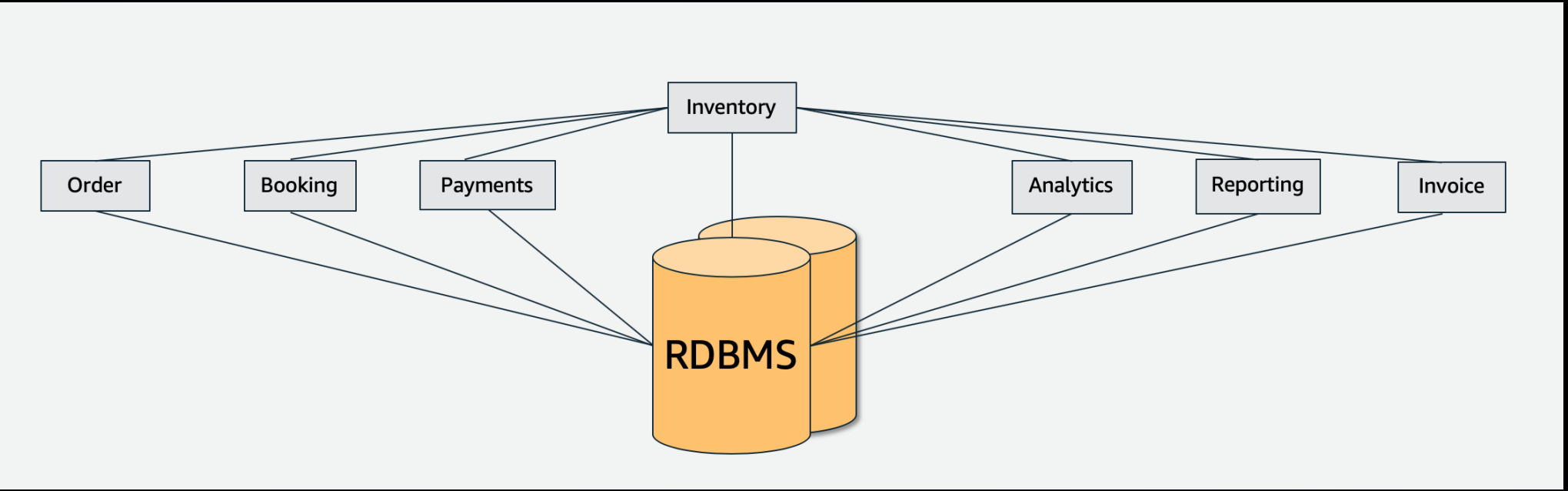
- What’s a modern application?
- Purpose-built databases use cases and AWS services
- Workshop reference architecture
- Lab walkthrough

© 2021, Amazon Web Services, Inc. or its affiliates. All rights reserved.

What’s a modern application?

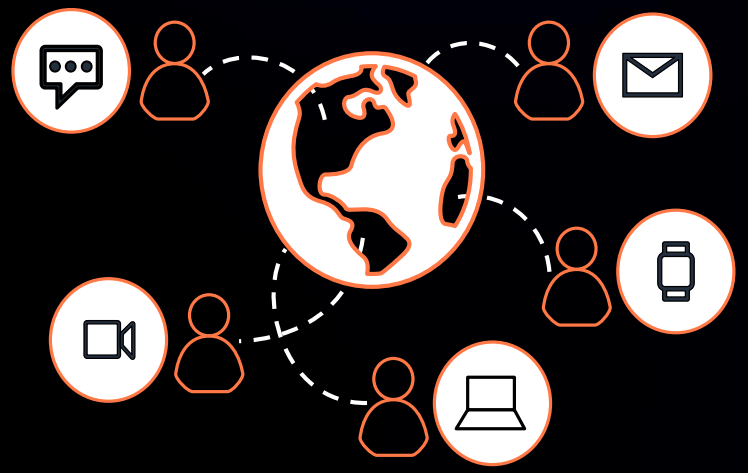
© 2021, Amazon Web Services, Inc. or its affiliates. All rights reserved.

Traditional application



© 2021, Amazon Web Services, Inc. or its affiliates. All rights reserved.

Modern real-time applications require performance, scale, and availability

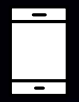



Ecommerce


Media streaming


Social media


Online gaming


Shared economy

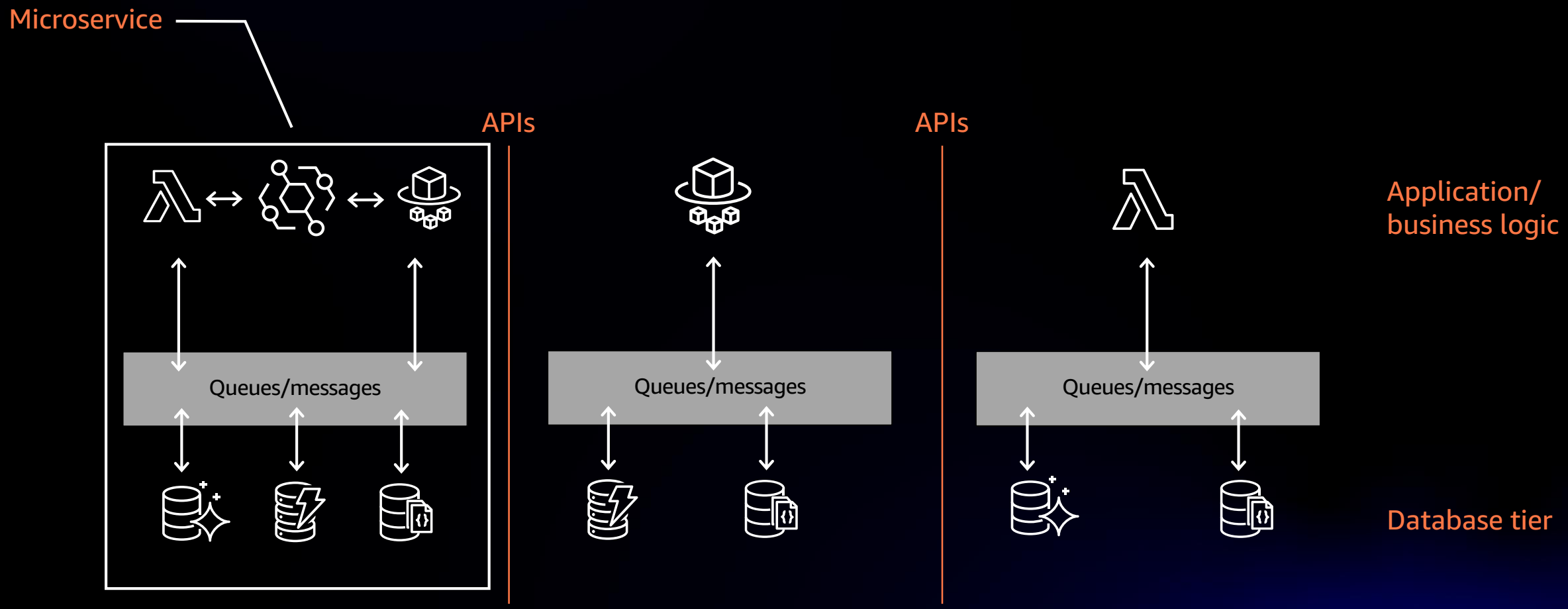
Users	1 million+
Data volume	Terabytes – petabytes
Locality	Global
Performance	Microsecond latency
Request rate	Millions per second
Access	Mobile, IoT, devices
Scale	Virtually unlimited
Economics	Pay as you go
Developer access	Instance API access
Development	Apps and storage are decoupled

aws

© 2021, Amazon Web Services, Inc. or its affiliates. All rights reserved.

Keeping up with modern app requirements

CUSTOMERS ARE SHIFTING TO MICROSERVICES ARCHITECTURES WITH PURPOSE-BUILT DATABASES



© 2021, Amazon Web Services, Inc. or its affiliates. All rights reserved.

Purpose-built databases use cases and AWS services

© 2021, Amazon Web Services, Inc. or its affiliates. All rights reserved.

Common data models and use cases

	Relational	Key-value	Document	In-memory	Graph	Time-series	Ledger	Wide column
Data models	Referential integrity, ACID transactions, schema-on-write	High throughput, low-latency reads and writes, endless scale	Store documents and quickly access querying on any attribute	Query by key with microsecond latency	Quickly and easily create and navigate relationships between data	Collect, store, and process data sequenced by time	Complete, immutable, and verifiable history of all changes to application data	Scalable, highly available, and managed Apache Cassandra–compatible service
Common use cases	Lift and shift, ERP, CRM, finance	Real-time bidding, shopping cart, social, product catalog, customer preferences	Content management, personalization, mobile	Leaderboards, real-time analytics, caching	Fraud detection, social networking, recommendation engine	IoT applications, event tracking	Systems of record, supply chain, healthcare, registrations, financial	Build low-latency applications, leverage open source, migrate Cassandra to the cloud
	Amazon Aurora, Amazon RDS	Amazon DynamoDB	Amazon DocumentDB	Amazon ElastiCache, Amazon MemoryDB for Redis	Amazon Neptune	Amazon Timestream	Amazon QLDB	Amazon Keyspaces (for Apache Cassandra)

© 2021, Amazon Web Services, Inc. or its affiliates. All rights reserved.



Airbnb uses different databases based on the purpose

User search history: **Amazon DynamoDB**
- Massive data volume
- Need quick lookups for personalized search

Session state: **Amazon ElastiCache**
- In-memory store for submillisecond site rendering

Relational data: **Amazon RDS**
- Referential integrity
- Primary transactional database

aws

© 2021, Amazon Web Services, Inc. or its affiliates. All rights reserved.

Workshop reference architecture

© 2021, Amazon Web Services, Inc. or its affiliates. All rights reserved.

AnyCompany RideShare scenario

AnyCompany RideShare* is a rideshare application that quickly became popular in the United States due to its innovation in customer experience. Currently, AnyCompany RideShare leverages an on-premises data center, and its tech stack consists of Java-based mobile/web applications and an Oracle database for its RDBMS.

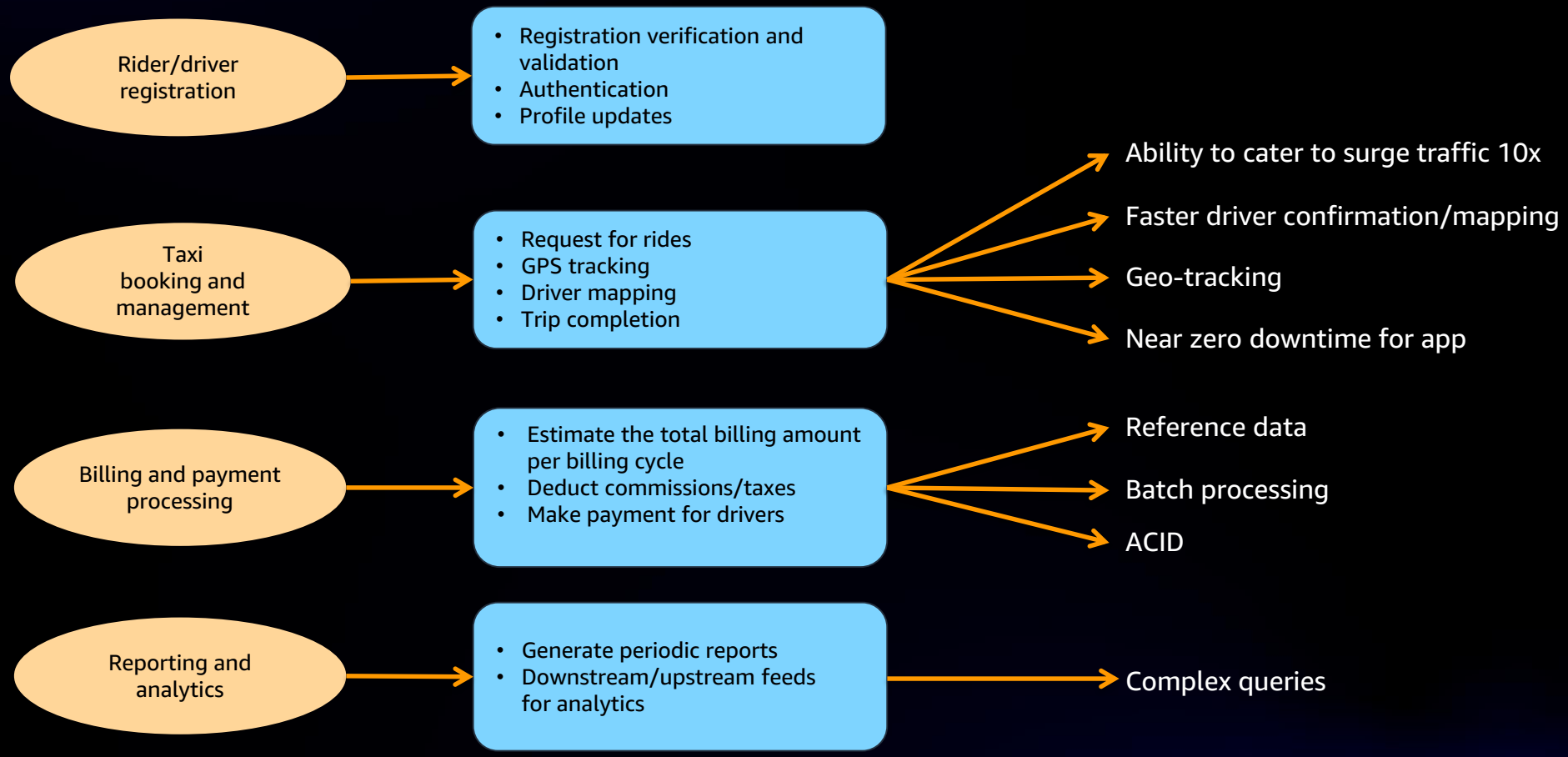
As AnyCompany's customer base grows, it has become clear that its existing technology stack will not scale well. AnyCompany is experiencing high licensing costs and high operational complexity. AnyCompany has not been able to scale quickly to meet demand. Additionally, the company has been facing increased security and compliance requirements, especially as it plans to expand to global markets.

*AnyCompany RideShare is a fictional company

aws

© 2021, Amazon Web Services, Inc. or its affiliates. All rights reserved.

AnyCompany RideShare key use cases



© 2021, Amazon Web Services, Inc. or its affiliates. All rights reserved.

Why NoSQL for taxi booking use case

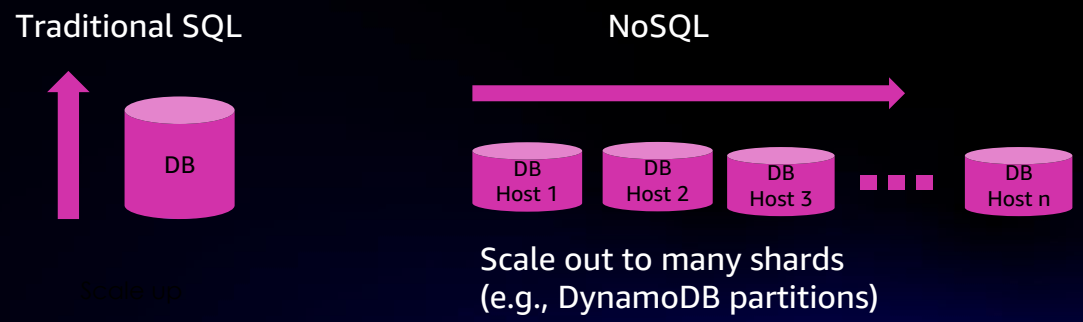
Taxi booking and management

- Request a trip by rider
- Map the driver
- Update the trip information
- Query the trips by rider
- Query the trips by driver



Key advantages

- OLTP at scale – scale horizontally
- Key-value lookup
- Instantiated views
- Flexible schema
- Denormalized

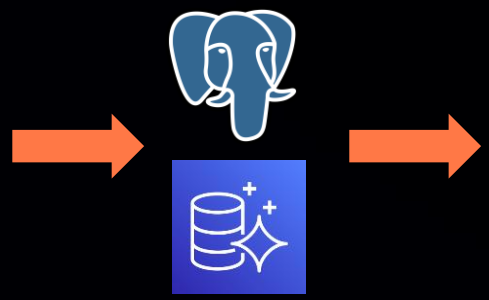


© 2021, Amazon Web Services, Inc. or its affiliates. All rights reserved.

Why SQL for billing and payment use case

Billing and payment workflow

- Find list of rides for current billing cycle
- Estimate total billing adjusting commissions
- Make the payment
- Complete in one transaction (all or nothing)
- Provide real-time reporting capabilities (top N, group by analysis) for business users



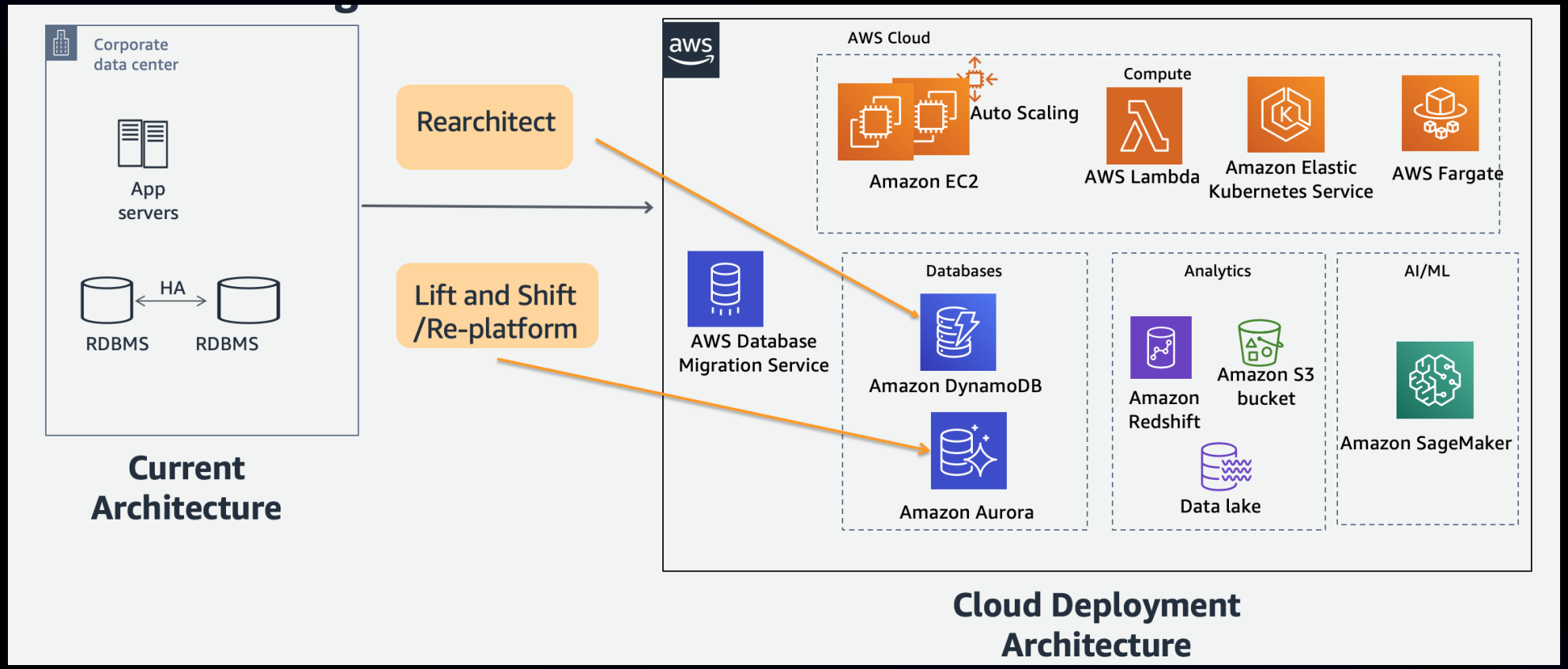
Key advantages

- Lift and shift data
- No licensing cost
- Data is normalized and requires joins
- Transactions involving multiple tables
- Relationships established via keys enforced by the system
- Run it as a batch on a schedule/ on demand
- Scaling via read replicas

aws

© 2021, Amazon Web Services, Inc. or its affiliates. All rights reserved.

AnyCompany proposed target architecture



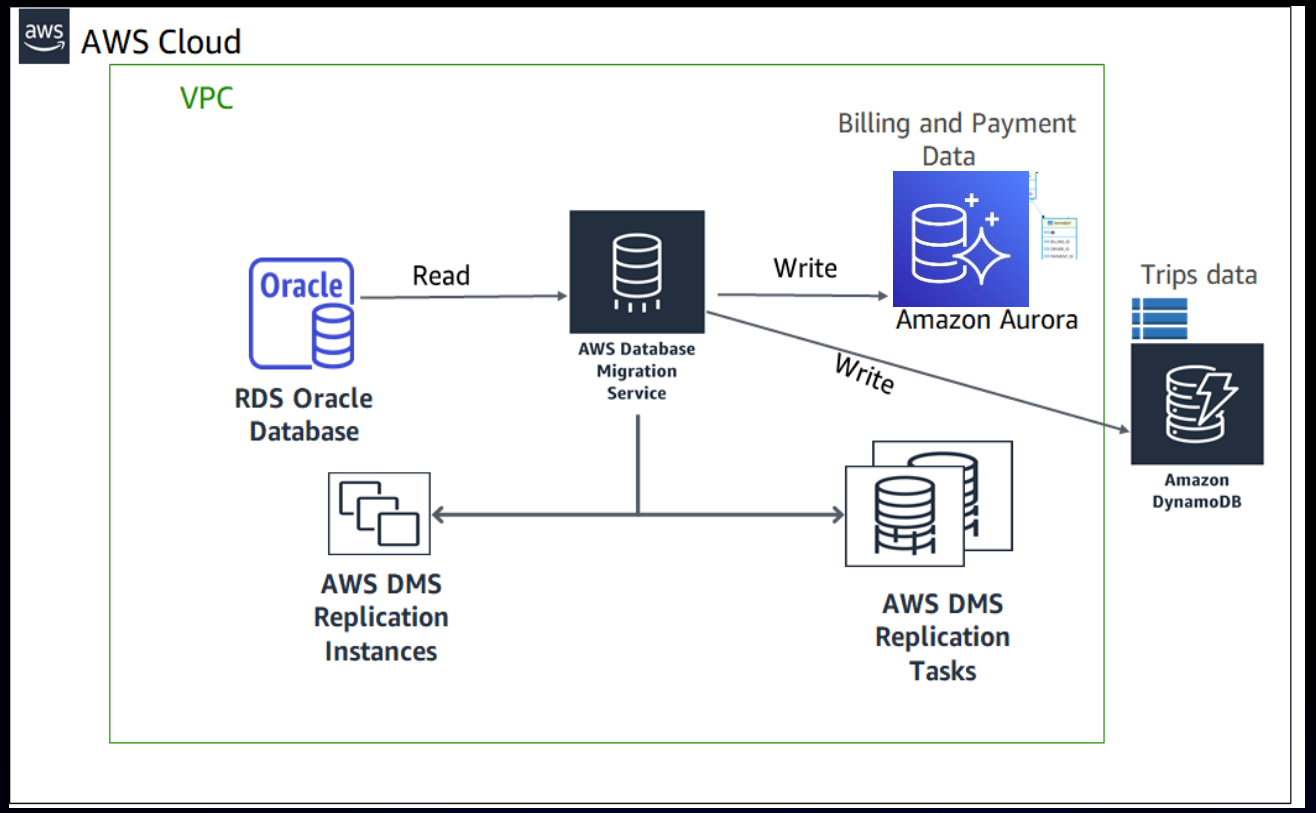
© 2021, Amazon Web Services, Inc. or its affiliates. All rights reserved.

Lab walkthrough

© 2021, Amazon Web Services, Inc. or its affiliates. All rights reserved.

Lab 1: Migrating data from Oracle to Amazon Aurora and DynamoDB

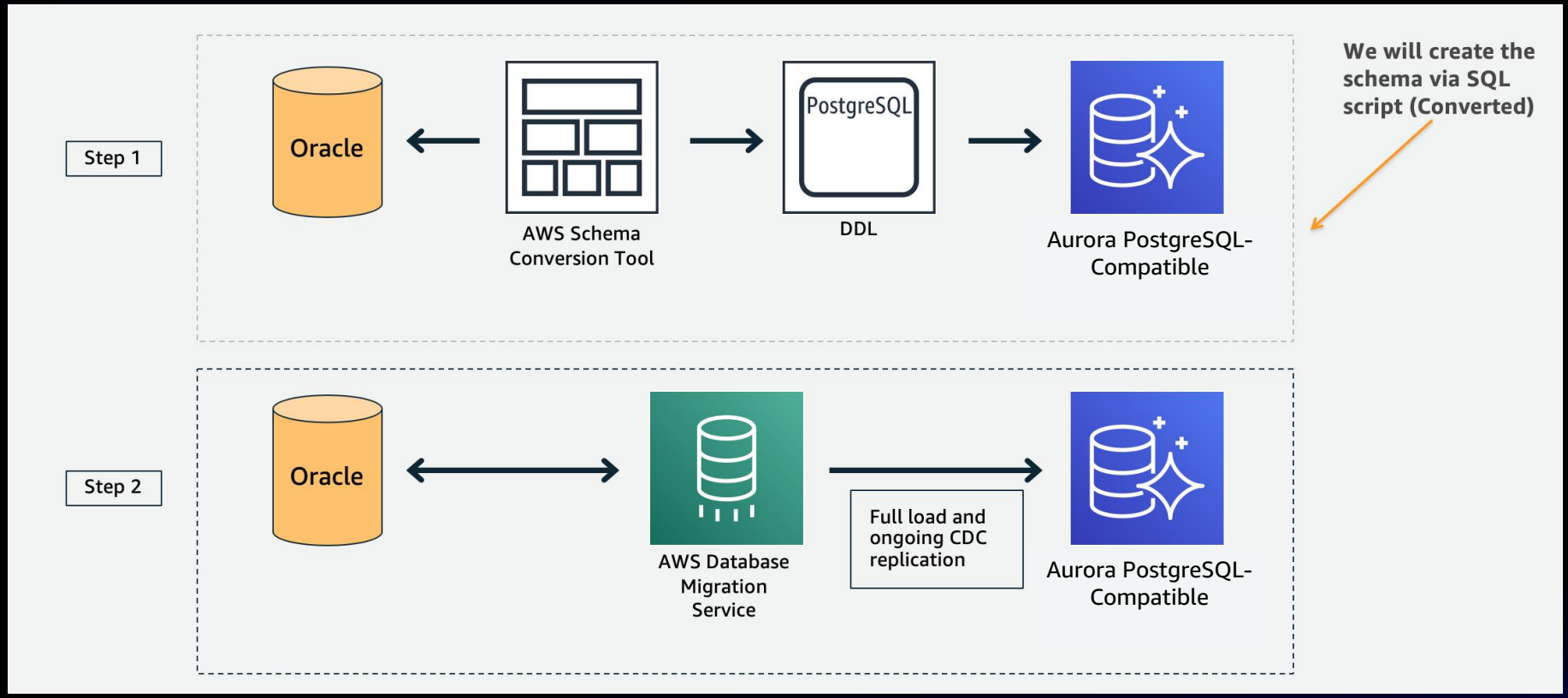
- AWS resources for the labs already pre-provisioned using AWS CloudFormation
- Amazon RDS for Oracle (source) with sample data pre-loaded
- Use AWS DMS to
  - Migrate the trips table from Amazon RDS for Oracle to DynamoDB using AWS DMS
  - Migrate the billing and payment tables to Aurora PostgreSQL-Compatible database



© 2021, Amazon Web Services, Inc. or its affiliates. All rights reserved.

Lab 1: Migrating data from Oracle to Aurora and DynamoDB

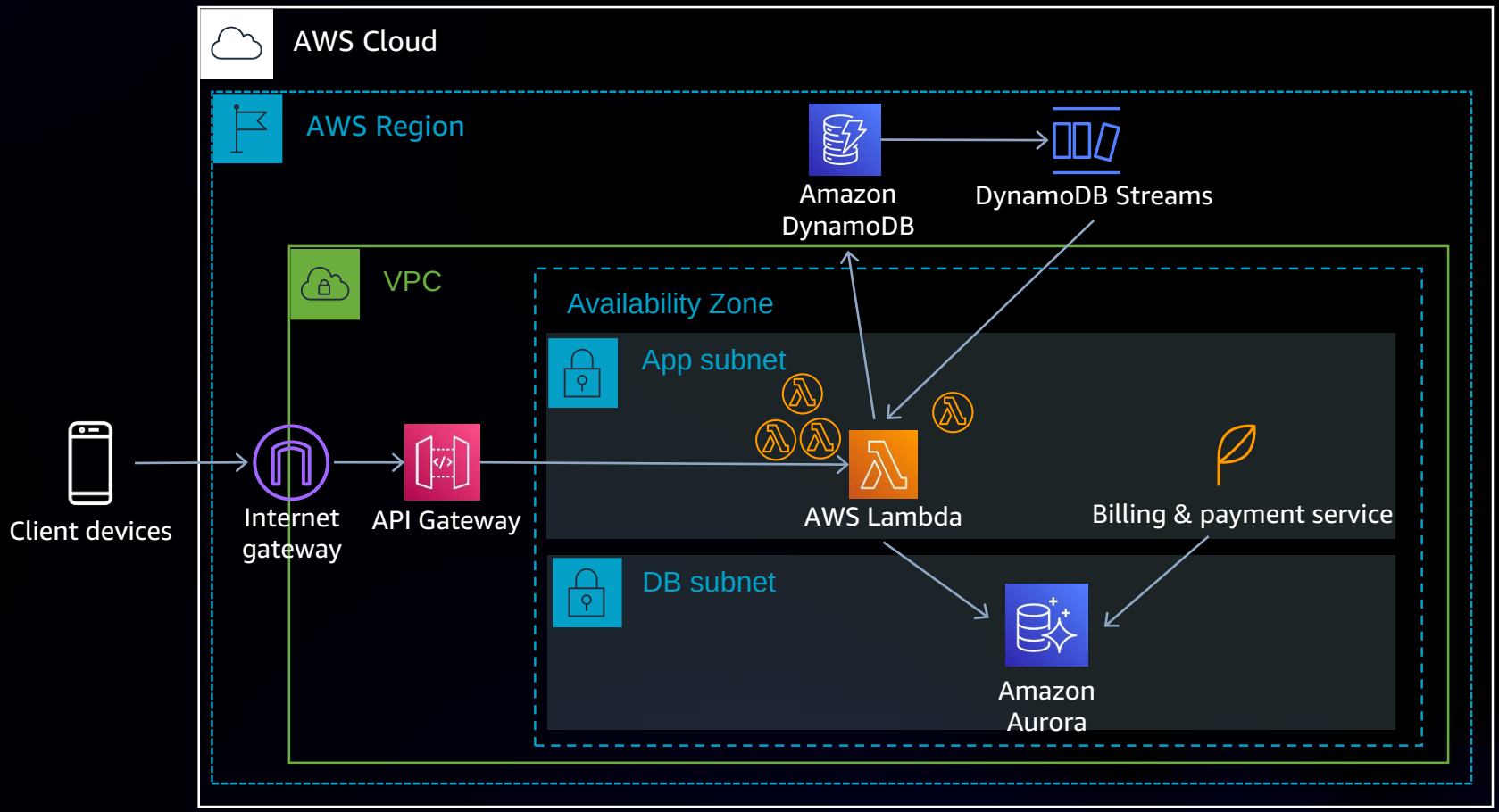
Oracle to Aurora data migration



© 2021, Amazon Web Services, Inc. or its affiliates. All rights reserved.

Lab 2: Data processing using DynamoDB and Aurora

- Enable DynamoDB Streams
- Deploy AWS Lambda functions
- Deploy Amazon API Gateway
- Simulate taxi ride workflow
- Process billing and payment

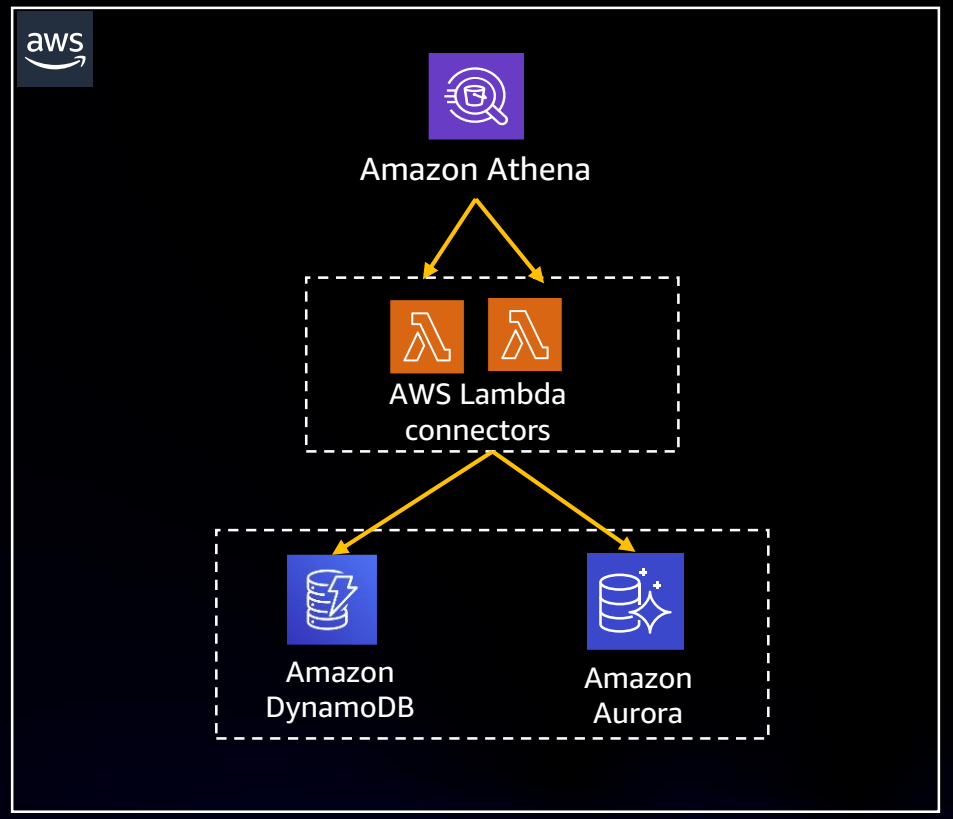


© 2021, Amazon Web Services, Inc. or its affiliates. All rights reserved.

Lab 3: Query multiple data sources using Athena Federated Query

Use Athena Federated Query to validate data accuracy and consistency of trip records across DynamoDB and Aurora PostgreSQL

- Deploy Lambda connectors
- Set up data sources
- Query the data

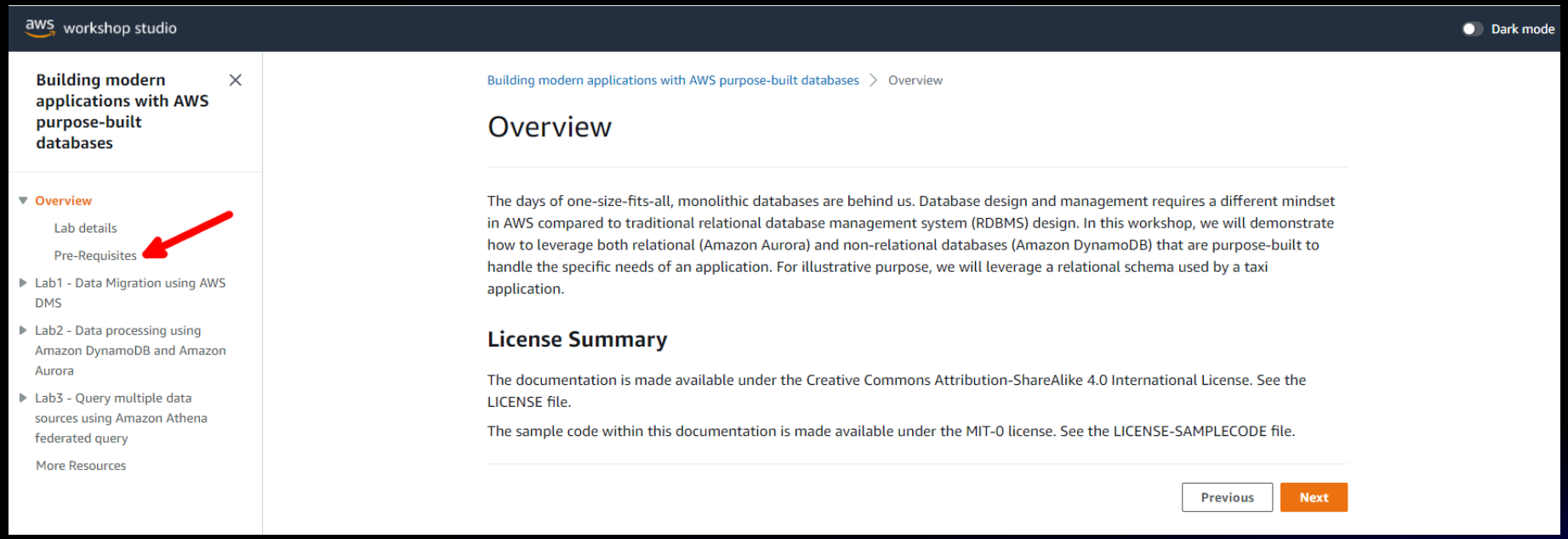


© 2021, Amazon Web Services, Inc. or its affiliates. All rights reserved.

Workshop details

Lab guide: https://bit.ly/pdb-ws

Event hash code: 901e-14b3ef9934-78



© 2021, Amazon Web Services, Inc. or its affiliates. All rights reserved.

Thank you!

© 2021, Amazon Web Services, Inc. or its affiliates. All rights reserved.