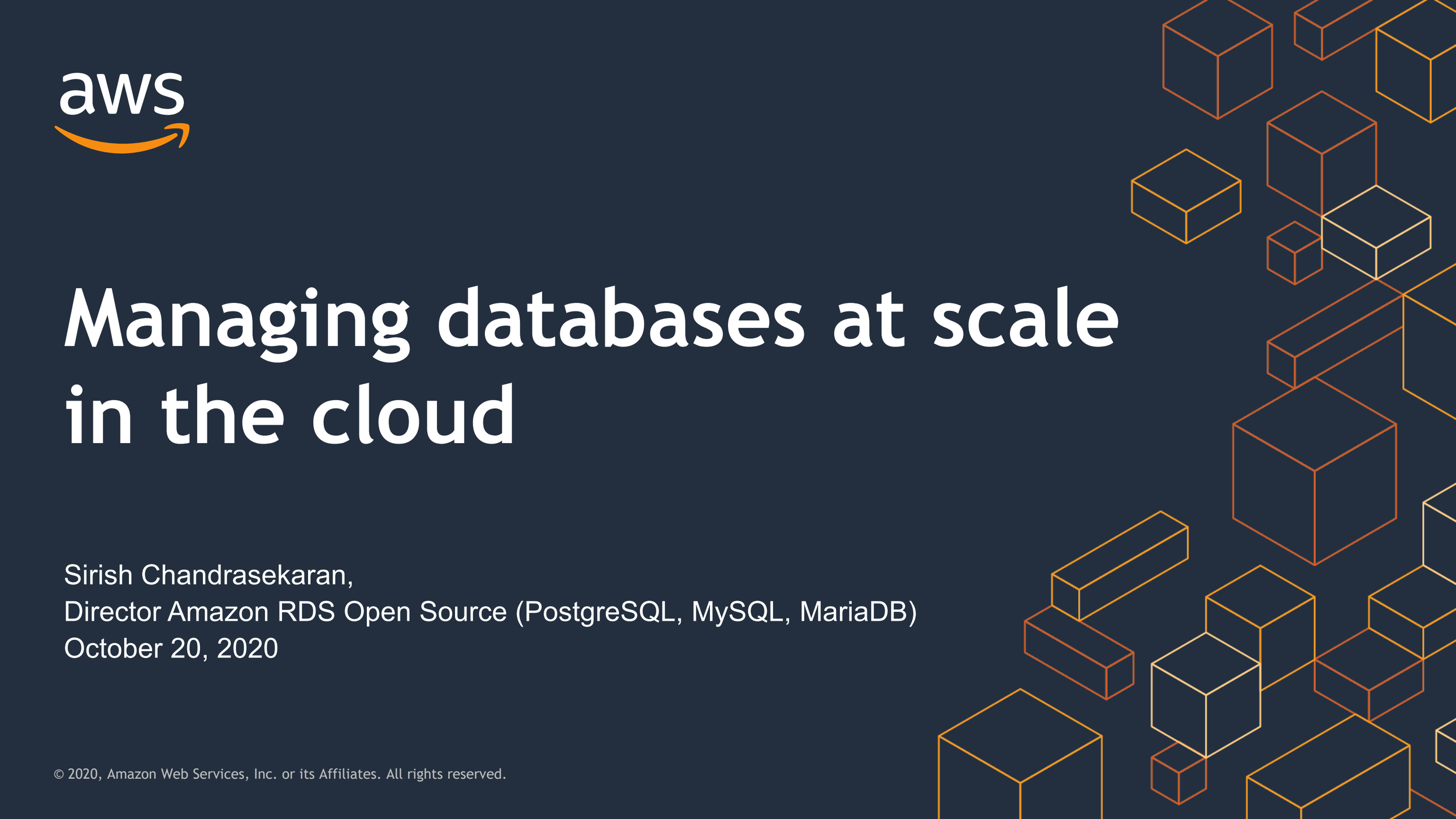


Managing databases at scale in the cloud

Sirish Chandrasekaran,
Director Amazon RDS Open Source (PostgreSQL, MySQL, MariaDB)
October 20, 2020

© 2020, Amazon Web Services, Inc. or its Affiliates. All rights reserved.

Our best practices come from a set of amazing customers

Hundreds of thousands of customers use Amazon Relational Database Service

- 7 database engines
    - RDS for: PostgreSQL, MySQL, MariaDB, SQL Server, Oracle
    - Aurora with: MySQL compatibility, PostgreSQL compatibility
- 25 AWS Regions, 77 Availability Zones, RDS on Outposts, RDS on VMware

© 2020, Amazon Web Services, Inc. or its Affiliates. All rights reserved.

aws

…who span the spectrum of scale

1 dev/test Free Tier instance

1000s of instances

Internet-scale workloads

Example: Amazon.com’s Inventory Management System runs on Aurora

© 2020, Amazon Web Services, Inc. or its Affiliates. All rights reserved.

aws

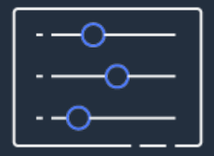
Which problems become especially hard at scale?

© 2020, Amazon Web Services, Inc. or its Affiliates. All rights reserved.

Database fleet management at scale
In our experience, it falls into 3 categories

Provisioning: Accounts, users, database configurations

Operating: Replication, monitoring, backups, disaster recovery, cost management

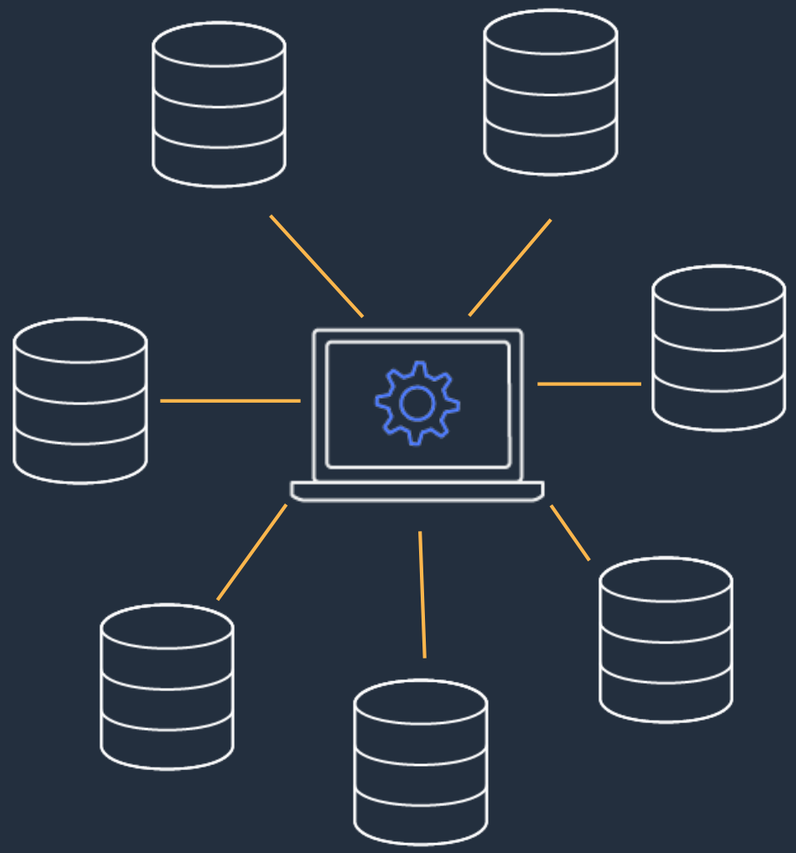


Patching: Database upgrades, instance upgrades, OS patching

Most importantly: self-managing is hard – and the undifferentiated heavy lifting grows with scale

aws

So you adopt a managed database service…



Now what can you do to provision, operate and patch your database fleet at scale?

Let us share a few of our learnings.

(Note: we'll show generic points in white and AWS-specific points in orange)

© 2020, Amazon Web Services, Inc. or its Affiliates. All rights reserved.

aws

Tips for database management at scale

1. Use fine-grained resources with flexible hierarchy
2. Centralize management of those resources
3. Standardize and script your deployments
4. Auto-scale where possible
5. Offload the undifferentiated heavy lifting
6. Watch the application-database interface carefully

© 2020, Amazon Web Services, Inc. or its Affiliates. All rights reserved.

aws

#1: Use fine-grained resources with flexible hierarchy

- Use separate accounts for different BUs, dev vs. prod etc.
- Set permissions at a fine grain
- Tag, tag, and tag again
- Use cross-account features where available

© 2020, Amazon Web Services, Inc. or its Affiliates. All rights reserved.

aws

#1: Use fine-grained resources with flexible hierarchy

- Use separate accounts for different BUs, dev vs. prod etc.
- Set permissions at a fine grain (AWS Identity and Access Management – IAM)
- Tag, tag, and tag again (AWS Resource Groups)
- Use cross-account features where available (RDS cross-account snapshot restore, …)

© 2020, Amazon Web Services, Inc. or its Affiliates. All rights reserved.

aws

#2: Centralize management of those resources

- Use centralized governance and management tools
- Centralize user/policy management across all cloud services
- Store passwords centrally, and rotate them frequently
- Maintain a global view of your spend

© 2020, Amazon Web Services, Inc. or its Affiliates. All rights reserved.

aws

#2: Centralize management of those resources

- Use centralized governance and management tools
  (AWS Organizations)

- Centralize user/policy management across all cloud services
  (AWS Identity and Access Management)

- Store passwords centrally, and rotate them frequently (AWS Secrets Manager)

- Maintain a global view of your spend (AWS Budgets; AWS Cost Explorer)

© 2020, Amazon Web Services, Inc. or its Affiliates. All rights reserved.

aws

#3: Standardize and script your deployments

- Set standard DB configurations for dev and prod
- Treat infrastructure as code

© 2020, Amazon Web Services, Inc. or its Affiliates. All rights reserved.

#3: Standardize and script your deployments

- Set standard DB configurations for dev and prod
  (RDS DB parameter groups)

- Treat infrastructure as code (CloudFormation)

© 2020, Amazon Web Services, Inc. or its Affiliates. All rights reserved.

aws

#4: Auto-scale where possible

Use cloud-native automation for scaling:

- Writes
- Reads
- Storage
- I/O

© 2020, Amazon Web Services, Inc. or its Affiliates. All rights reserved.

#4: Auto-scale where possible

Use cloud-native automation for scaling:

- Writes (Aurora Serverless)
- Reads (Auto-scaled read replicas)
- Storage (Auto-scaled storage)
- I/O (Aurora - *by default*)

© 2020, Amazon Web Services, Inc. or its Affiliates. All rights reserved.

aws

#5: Offload the undifferentiated heavy lifting

Use cloud-native automation for:

- HA and read scaling
- Logging
- Monitoring
- Backup and DR
- Patching

© 2020, Amazon Web Services, Inc. or its Affiliates. All rights reserved.

aws

#5: Offload the undifferentiated heavy lifting

Use cloud-native automation for:

- HA and read scaling (RDS Multi-AZ, managed in-region replicas)
- Logging (CloudWatch Logs)
- Monitoring (CloudWatch Metrics, Enhanced Monitoring)
- Backup and DR (RDS automated backups, RDS cross-region replicas, Aurora Global Database)
- Patching (RDS Auto Minor Version Upgrades)

© 2020, Amazon Web Services, Inc. or its Affiliates. All rights reserved.

aws

#6: Watch the application-database interface carefully

- Tune connection behavior across the stack
- Monitor query-level performance
- Use cross-stack tools where possible

© 2020, Amazon Web Services, Inc. or its Affiliates. All rights reserved.

aws

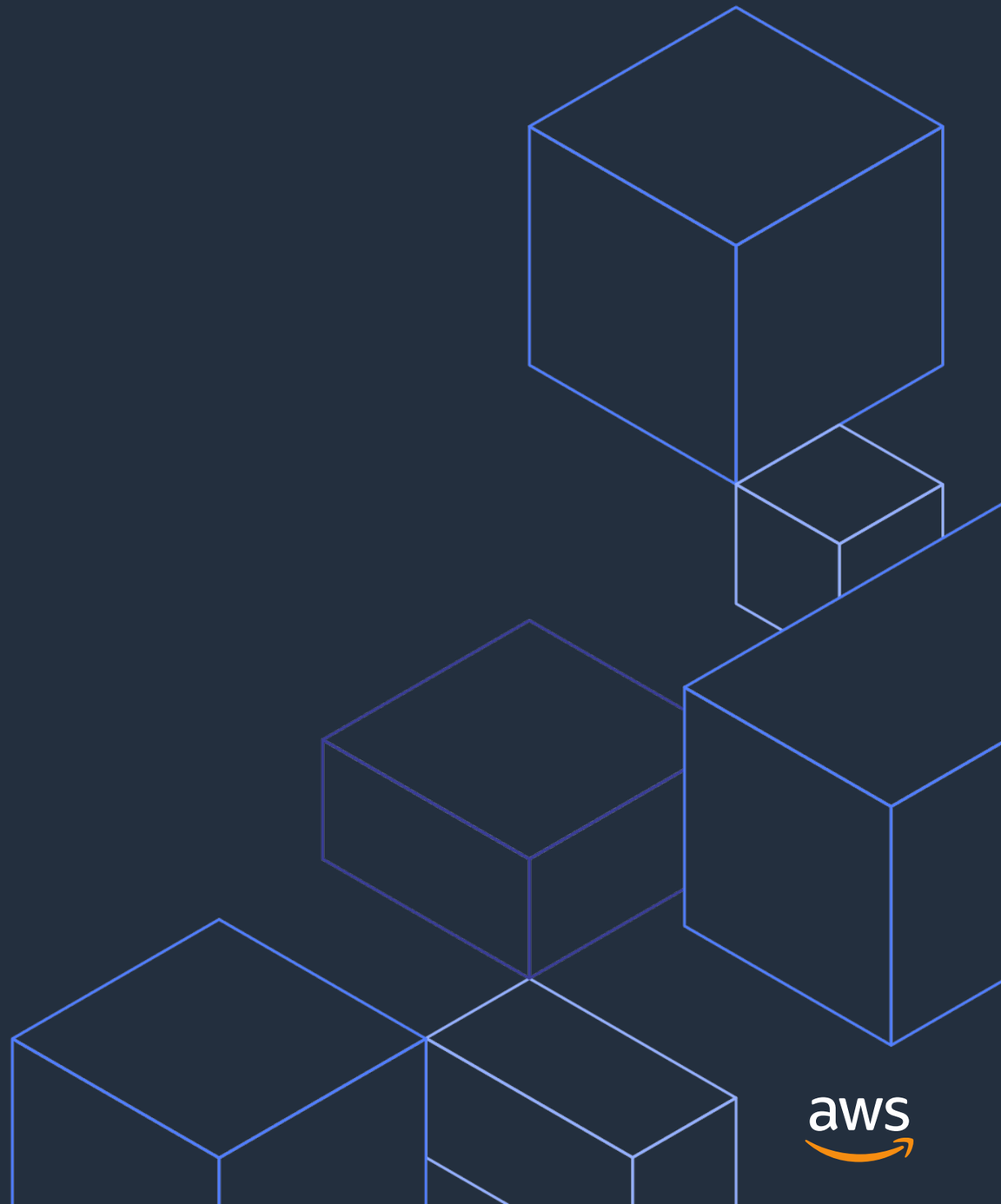
#6: Watch the application-database interface carefully

- Tune connection behavior across the stack (RDS Proxy)
- Monitor query-level performance (RDS Performance Insights)
- Use cross-stack tools where possible (AWS Backups)

© 2020, Amazon Web Services, Inc. or its Affiliates. All rights reserved.

aws

Closing thoughts



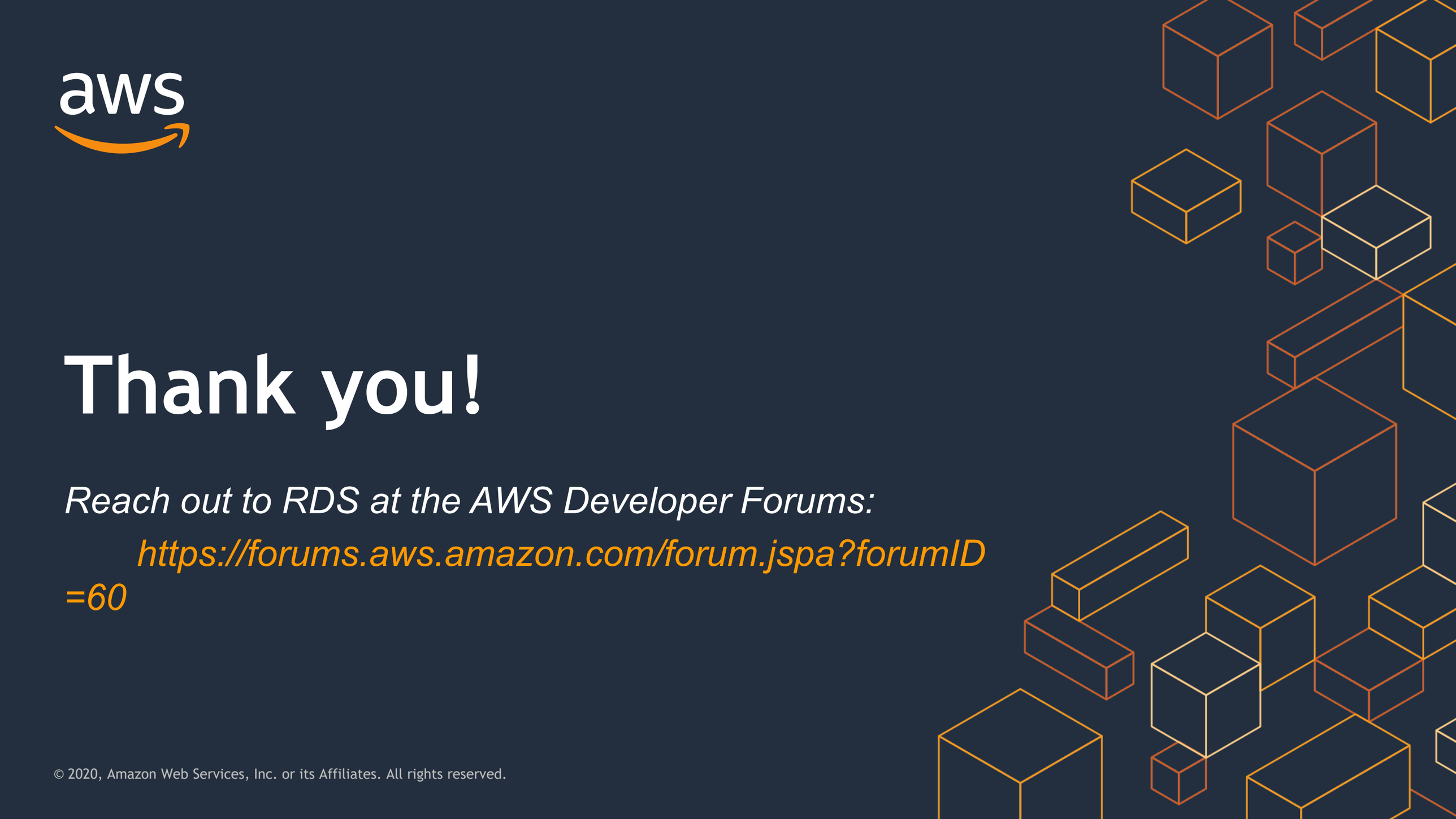
© 2020, Amazon Web Services, Inc. or its Affiliates. All rights reserved.

Managing databases in the cloud is a shared responsibility

- *Offload the undifferentiated heavy lifting to the cloud vendor…*
- *…which frees you to focus on your applications so you can give them the fast performance, high availability, security and compatibility they need*

© 2020, Amazon Web Services, Inc. or its Affiliates. All rights reserved.

aws



Thank you!

Reach out to RDS at the AWS Developer Forums:

https://forums.aws.amazon.com/forum.jspa?forumID=60

© 2020, Amazon Web Services, Inc. or its Affiliates. All rights reserved.