Availability and Performance Tradeoffs in Global Databases

Kevin Jernigan, Principal Database SA, Aurora PostgreSQL

ONLINE, AROUND-THE-WORLD OPEN SOURCE DATABASE CONFERENCE

PERCONA LIVEONLINE

20-21 OCTOBER 2020

Agenda

- Why make your database global?
- How can you make your database global?
  - Logical replication
  - Physical replication
  - Tradeoffs: latency, consistency, complexity, manageability
- Amazon Aurora examples

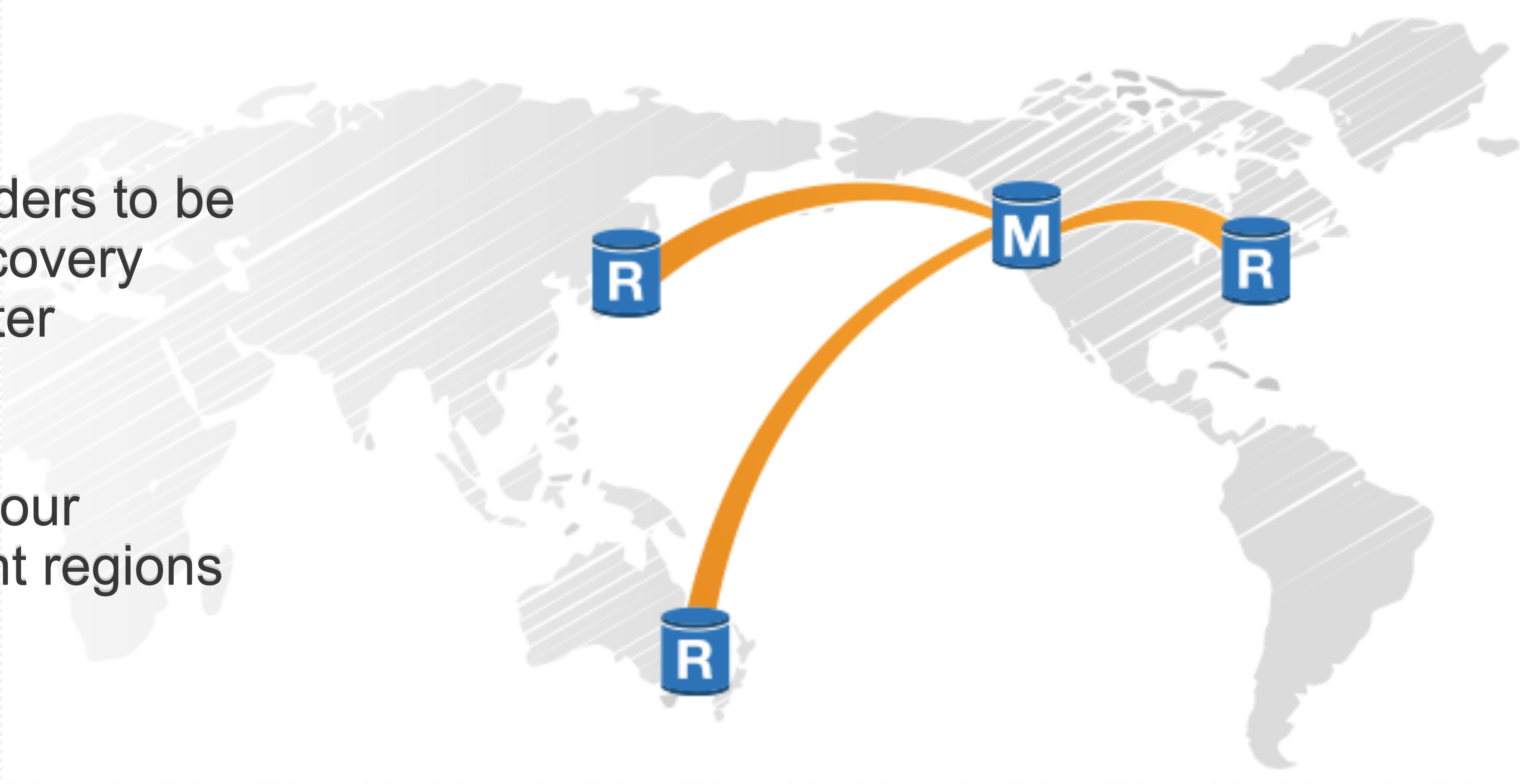
Why make your database global?

Why make your database global?

Faster disaster recovery and enhanced data locality

- Promote remote readers to be primary for faster recovery in the event of disaster

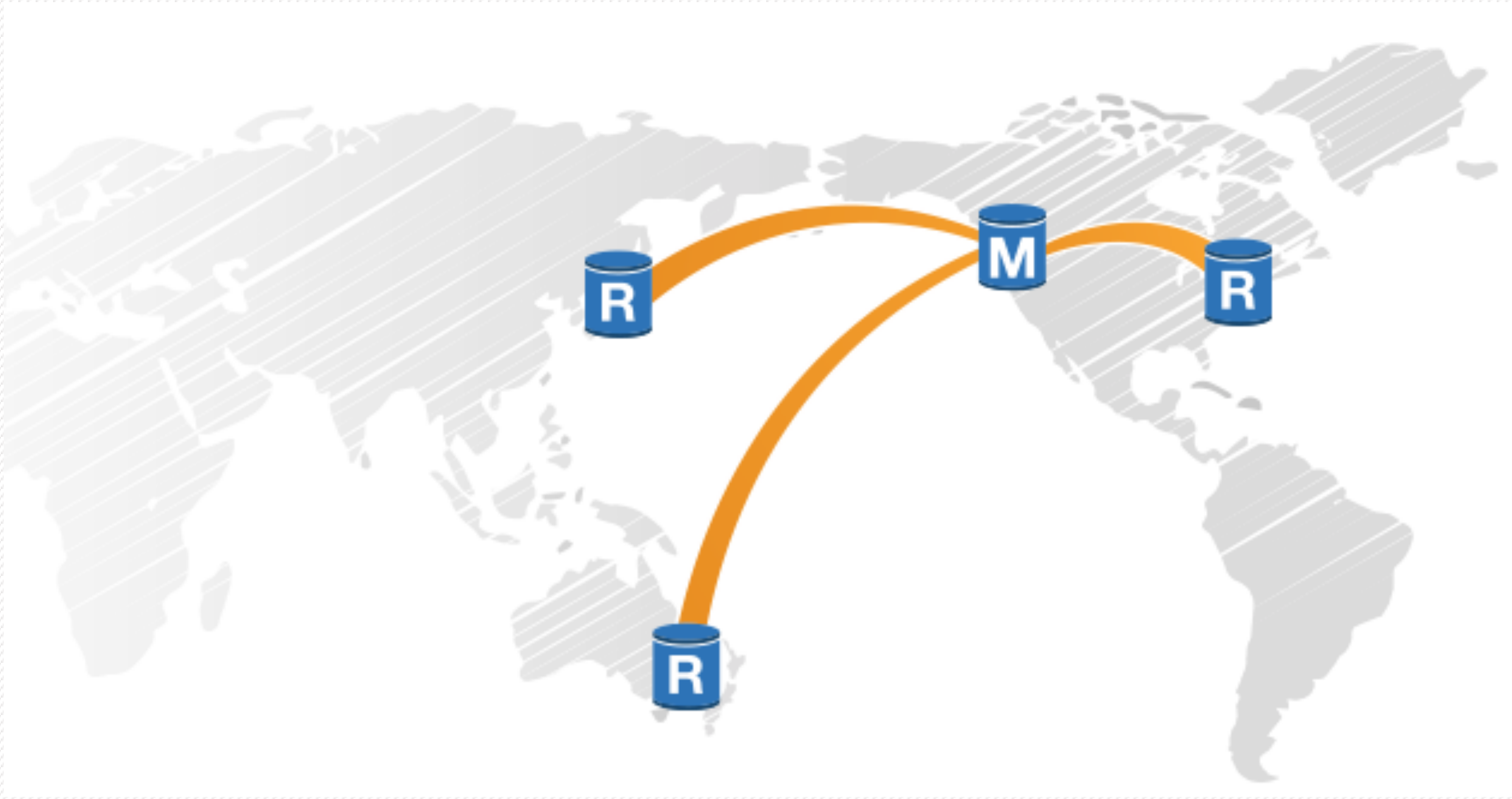
- Bring data close to your customers in different regions



How can you make your database global?

Methods for global database

- Logical replication
- Physical replication



Global database: logical replication

Customer use cases for logical replication

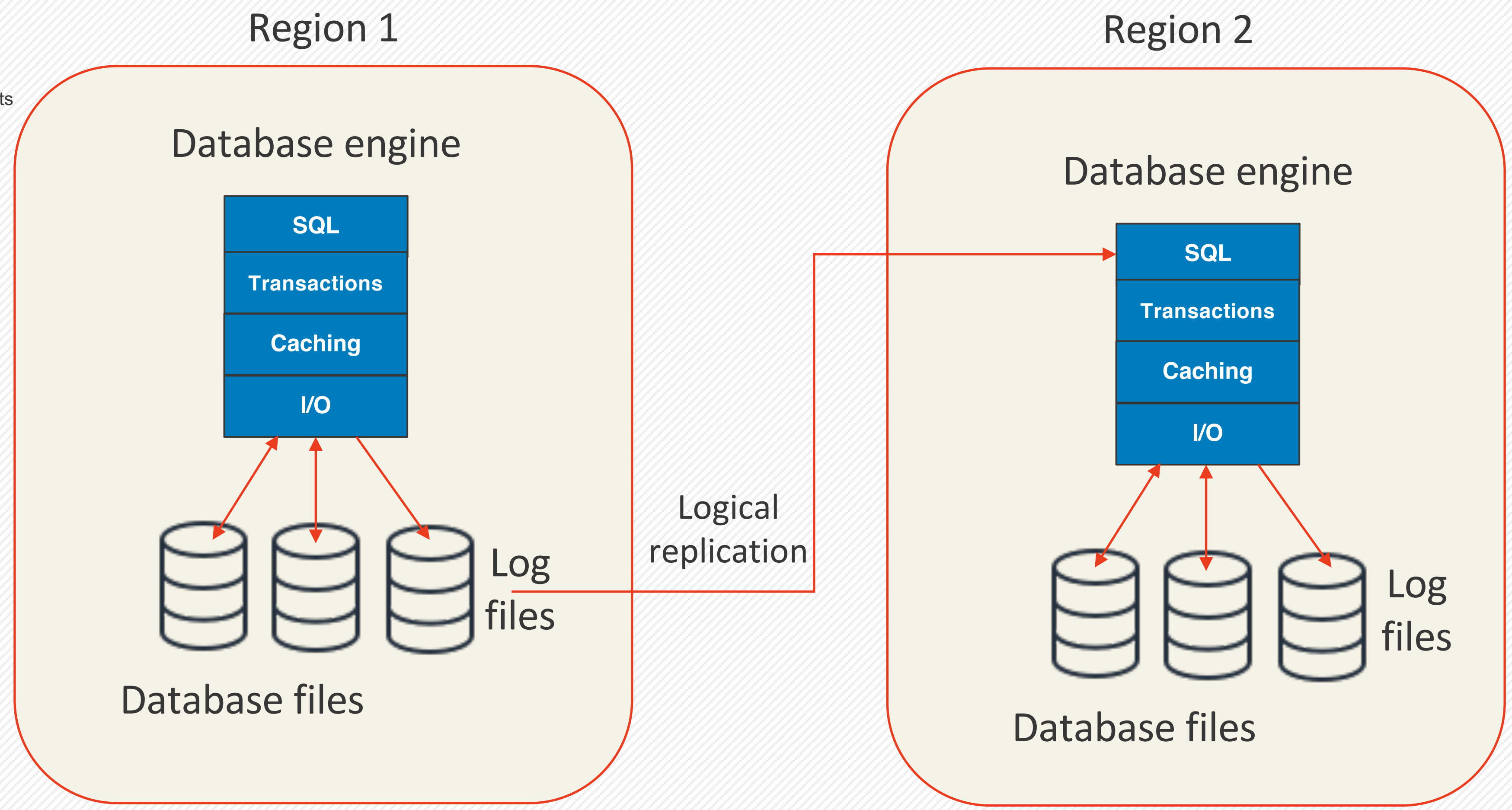
Telco, e-commerce, financial services

- Migrate from one region or version to another
- Maintain copies of data in multiple Regions
- Replicate changes for some tables but not others
- Selectively disable replication for specific statements
- Manage conflicts automatically
- Manage DDL / schema changes

Logical replication

Telco, e-commerce, financial services

- Migrate from one region or version to another
- Maintain copies of data in multiple Regions
- Replicate changes for some tables but not others
- Selectively disable replication for specific statements
- Manage conflicts automatically
- Manage DDL / schema changes



Logical replication: challenges

- Complex to set up and to manage
- Primary keys are usually required on all tables
- Replication lag will be measured in seconds or minutes
- Multi-directional (BDR) is tempting…but challenging
    - How to configure conflict management?
    - What happens when tables get out of sync?
    - What about DDL?

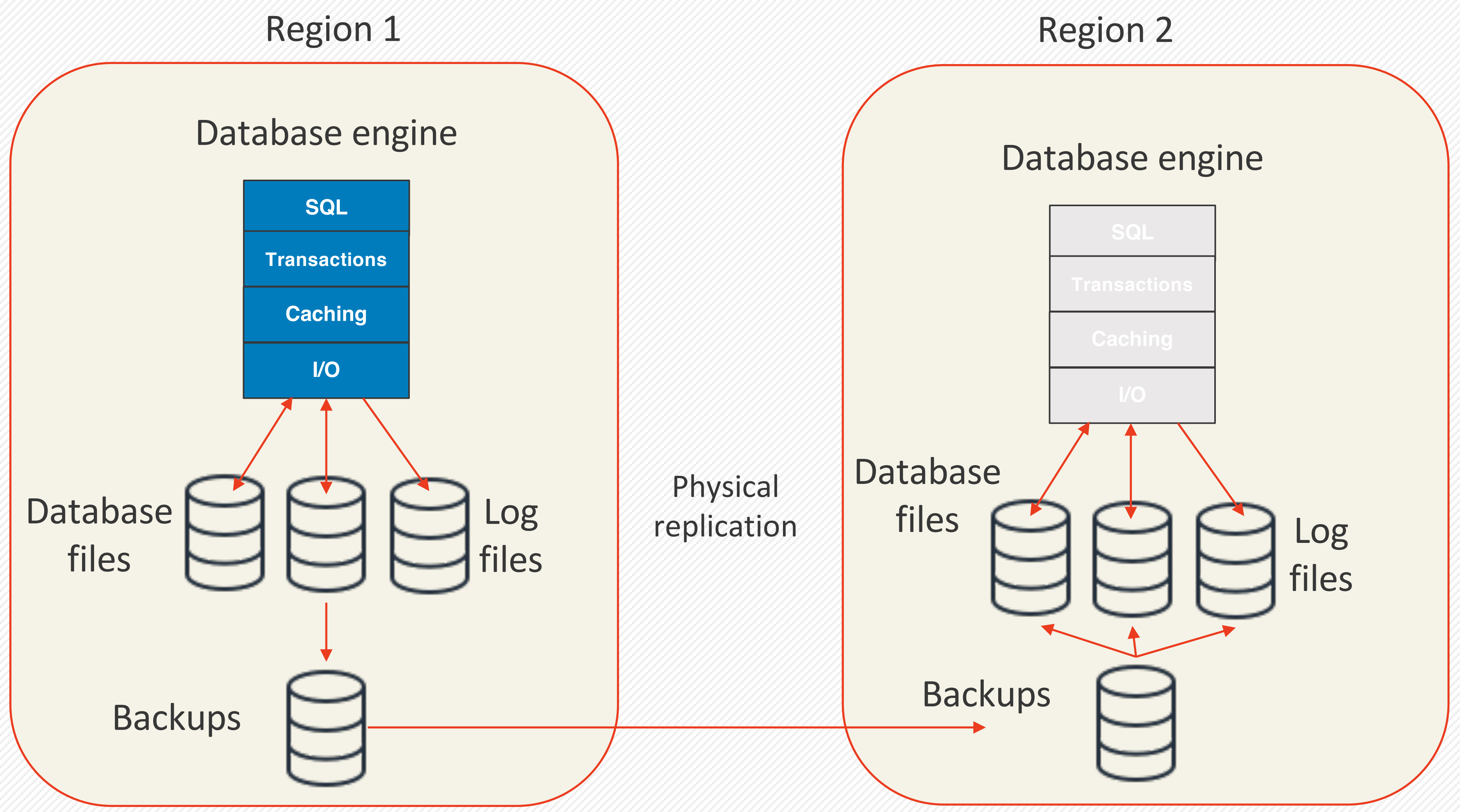
Global database: physical replication

Customer use cases for physical replication

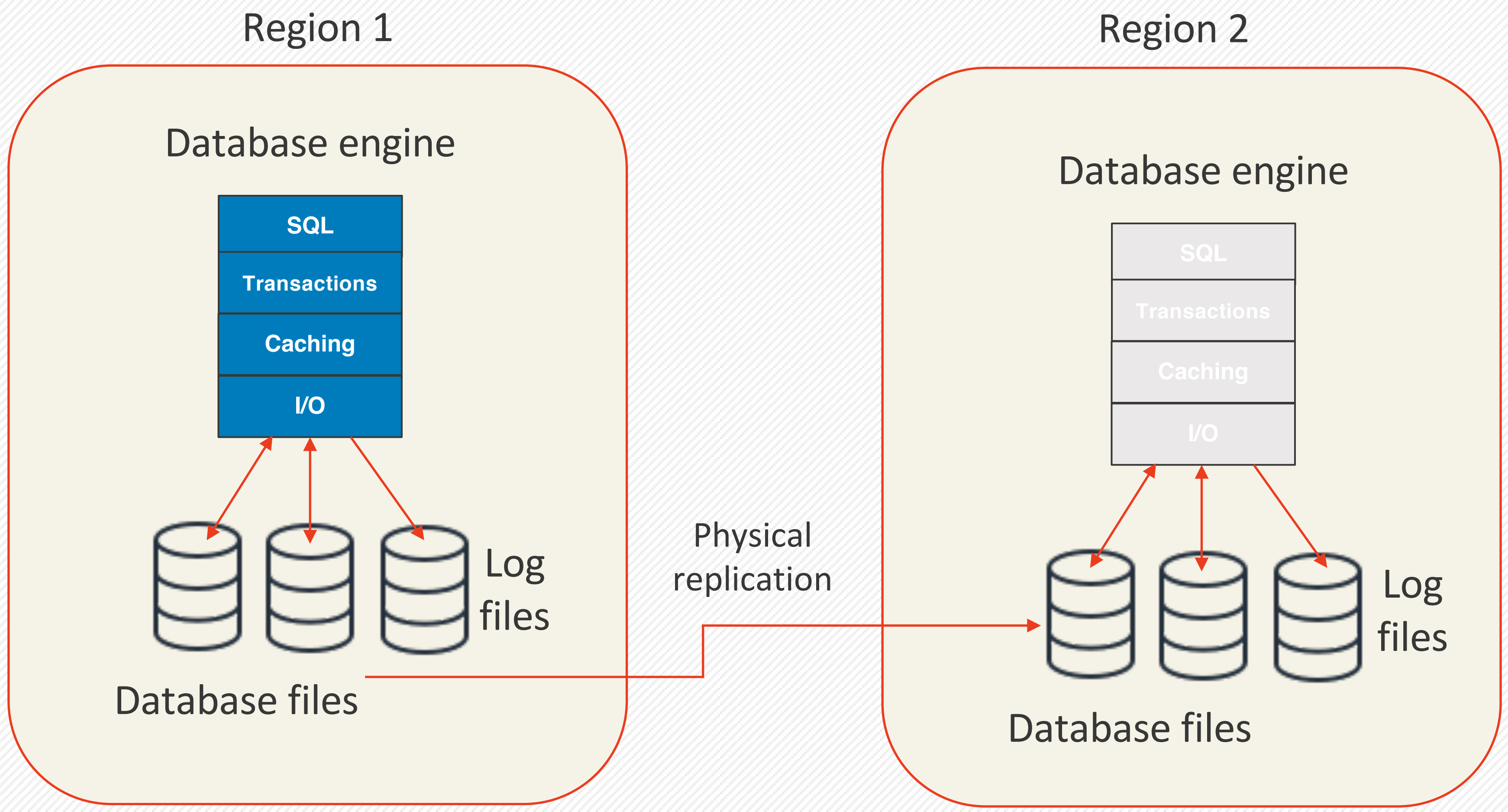
Financial services, telco, gaming, etc

- Disaster Recovery: protect against regional outages
  - Minimize data loss (RPO)
  - Minimize failover time (RTO)
- Geo-distributed reads: low latency read access

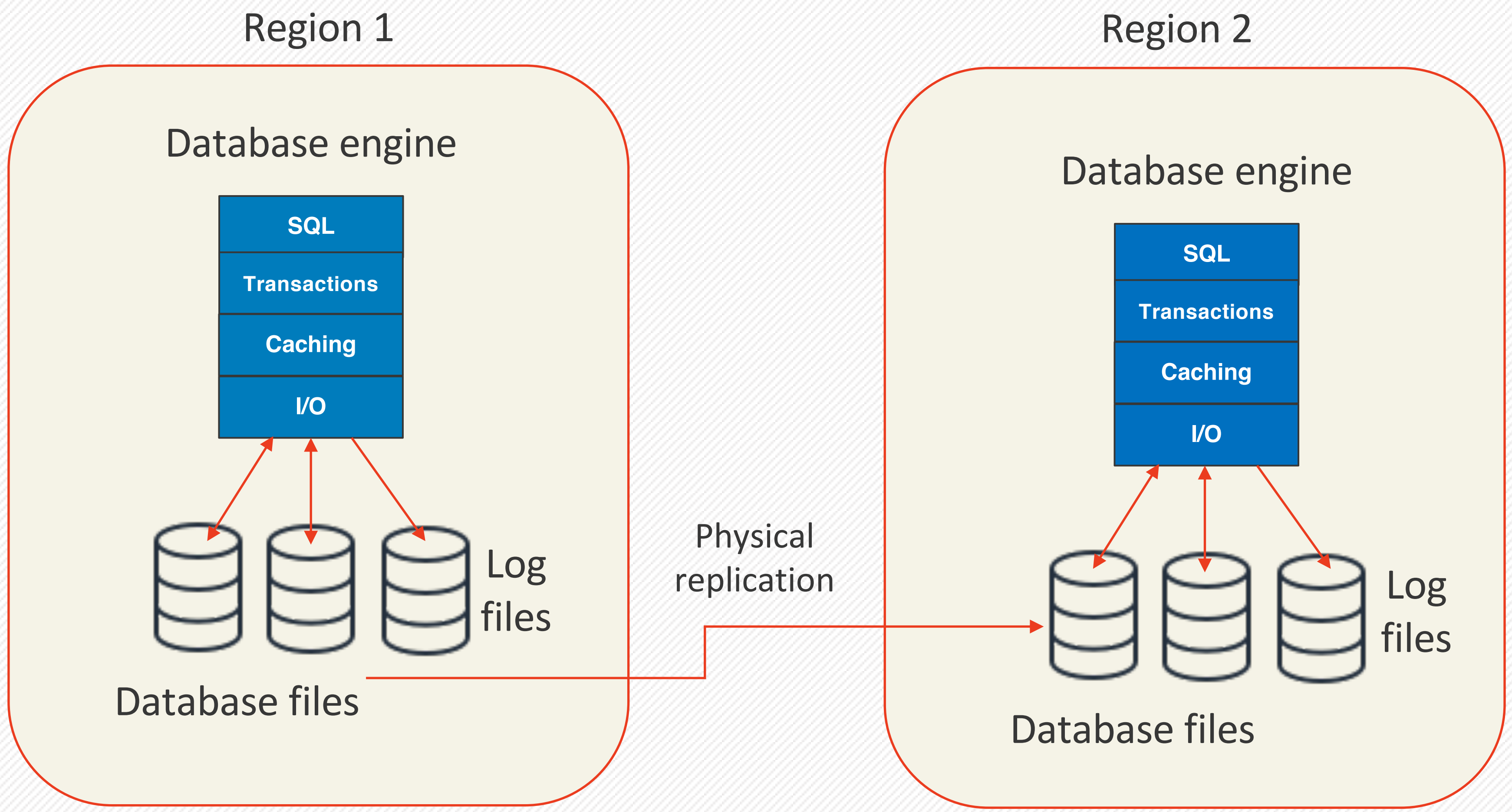
Physical replication



Physical replication



Physical replication



Physical replication: tradeoffs

- All or nothing
- Performance depends on implementation
  - Block-level versus log-based
- Secondary database(s) might not be available for reads

Amazon Aurora

Amazon Aurora

Enterprise database at open source price,
delivered as a managed service

Amazon Aurora

Speed and **availability** of high-end commercial databases

Simplicity and **cost-effectiveness** of open source databases

Drop-in **compatibility** with MySQL and PostgreSQL

Simple **pay as you go** pricing

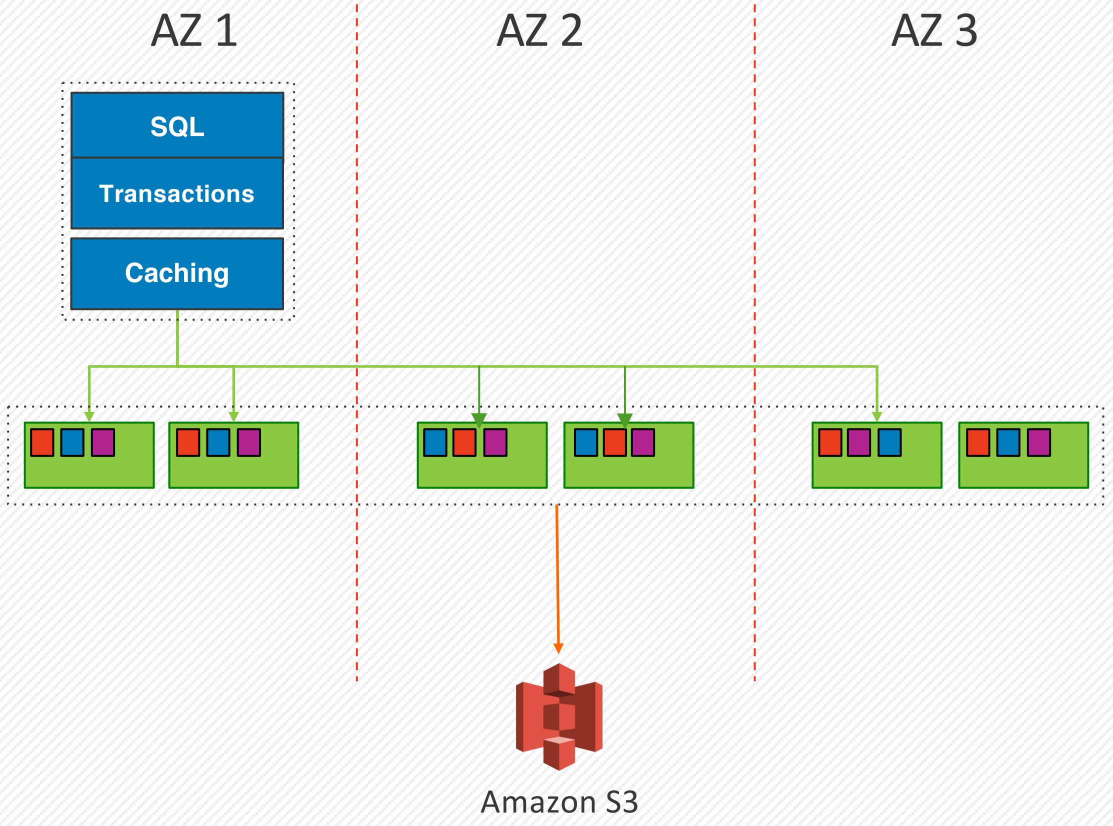
Aurora: Offload Redo to Storage

Database Tier

- Writes redo log records on network
- Use 4/6 quorum protocol
- No full data block writes for
  - Checkpointing, cache eviction, bg writes
- Push log apply to storage

Storage Tier

- Highly parallel scale out redo processing
- Generate database blocks on demand (redo)
- Materialize database blocks in background (redo)



The Log is the Database

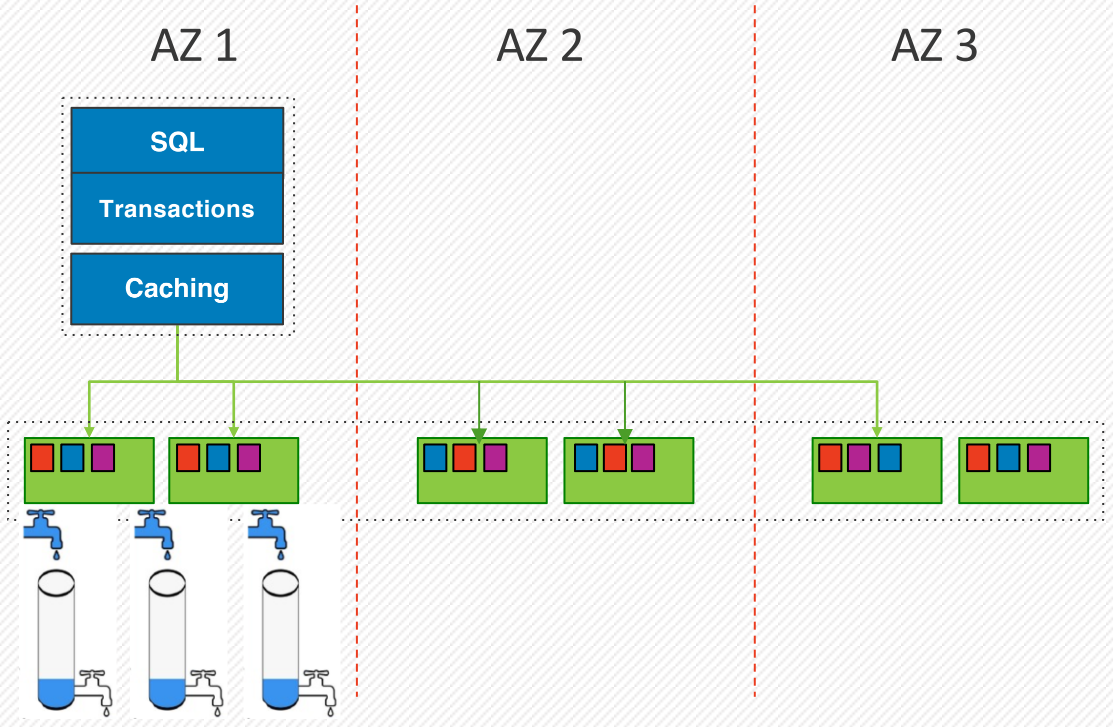
Aurora logical replication

Database Tier

- Writes redo log / WAL records to storage

Aurora PostgreSQL: replication slots and pglogical

Aurora MySQL: binlog



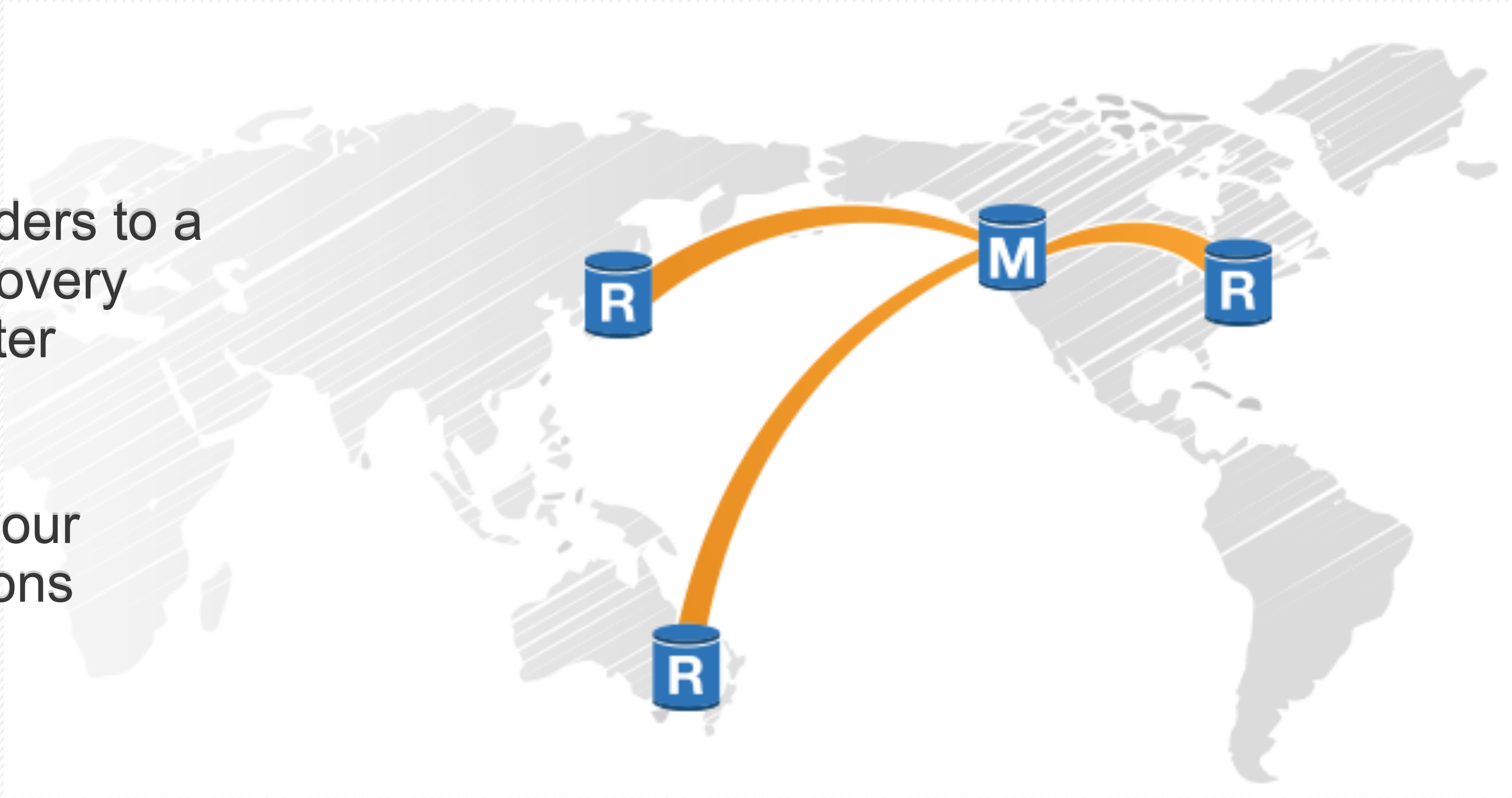
pglogical / binlog

Aurora Global Database

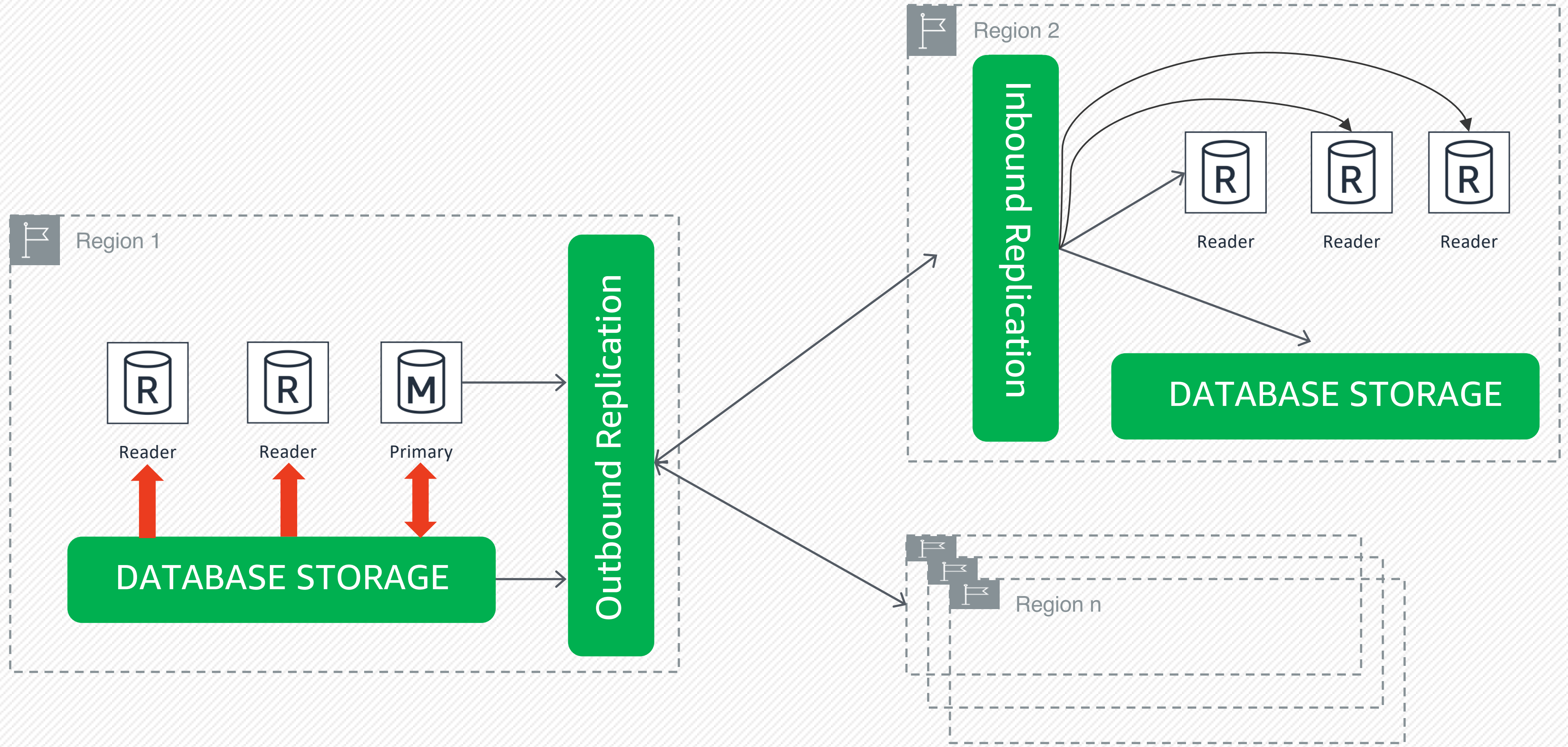
Faster disaster recovery and enhanced data locality

- Promote remote readers to a master for faster recovery in the event of disaster

- Bring data close to your customer's applications in different regions

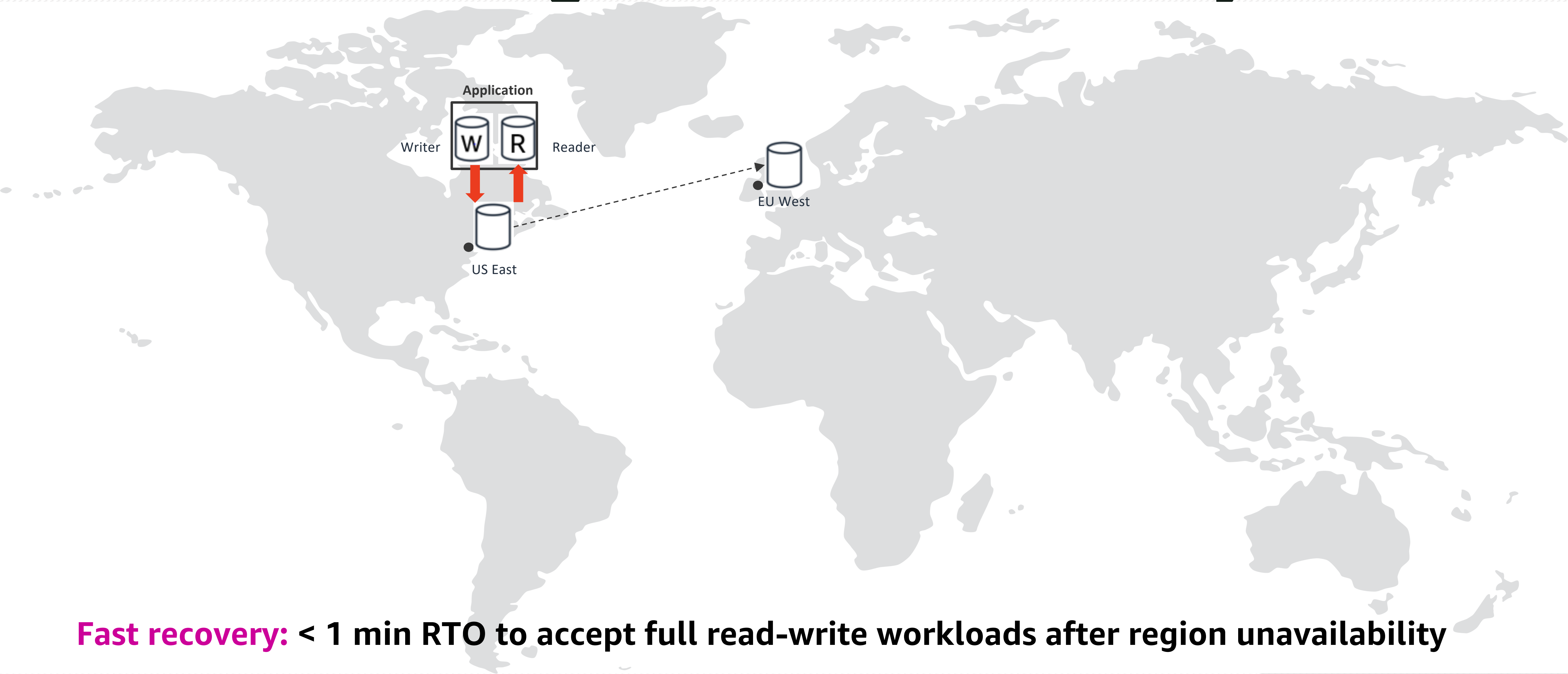


How does it work?



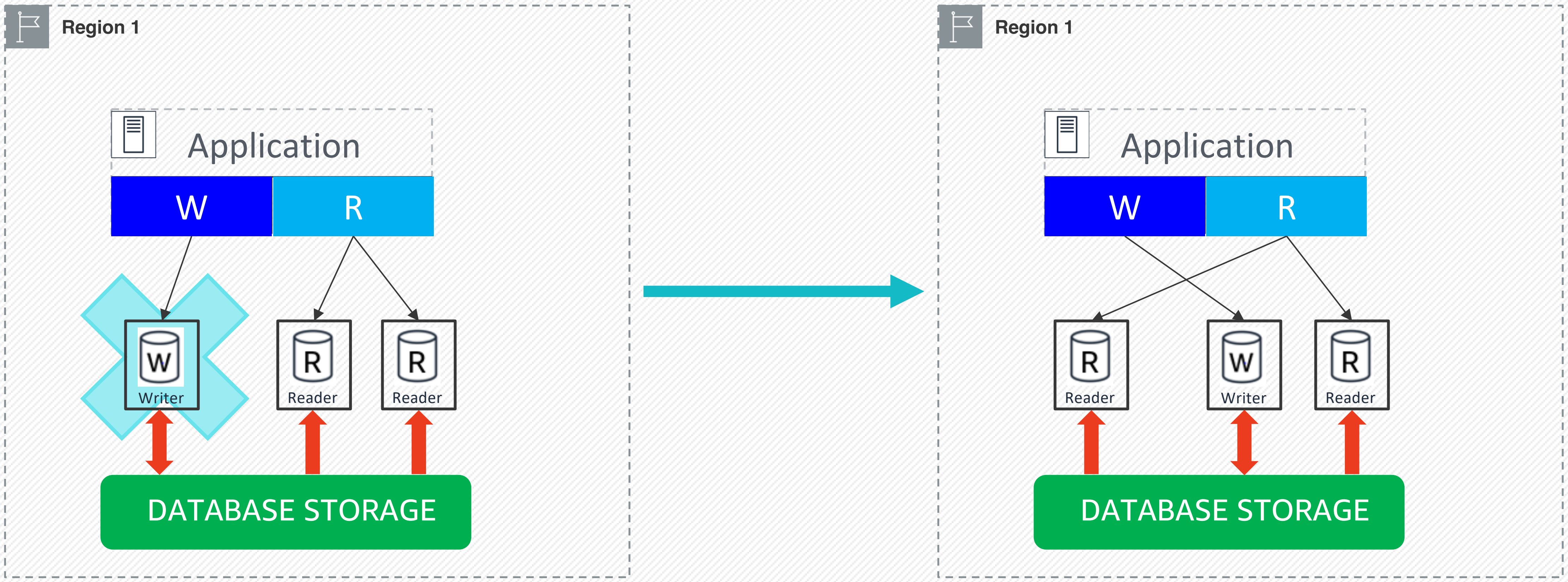
High throughput: Up to 200K writes/sec – negligible performance impact

Fast cross-region disaster recovery

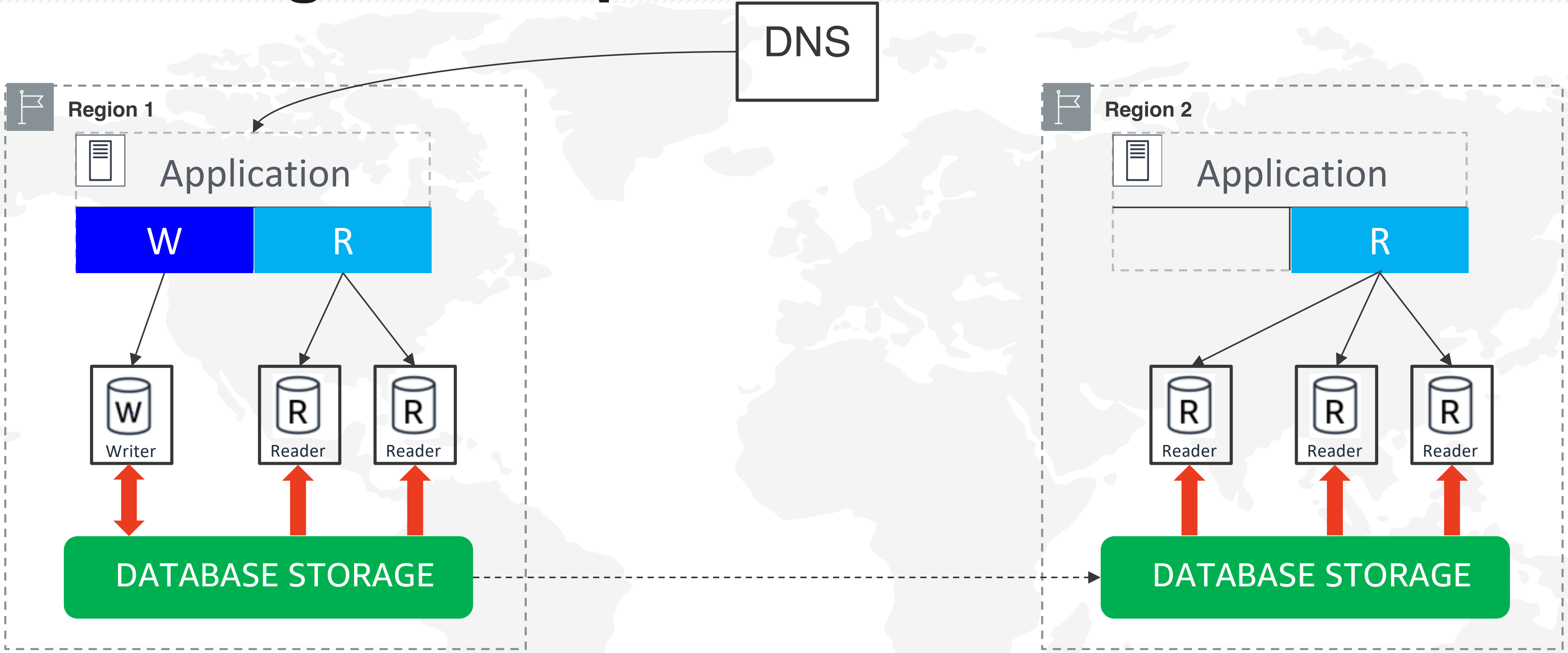


Fast recovery: < 1 min RTO to accept full read-write workloads after region unavailability

Failover within a single region

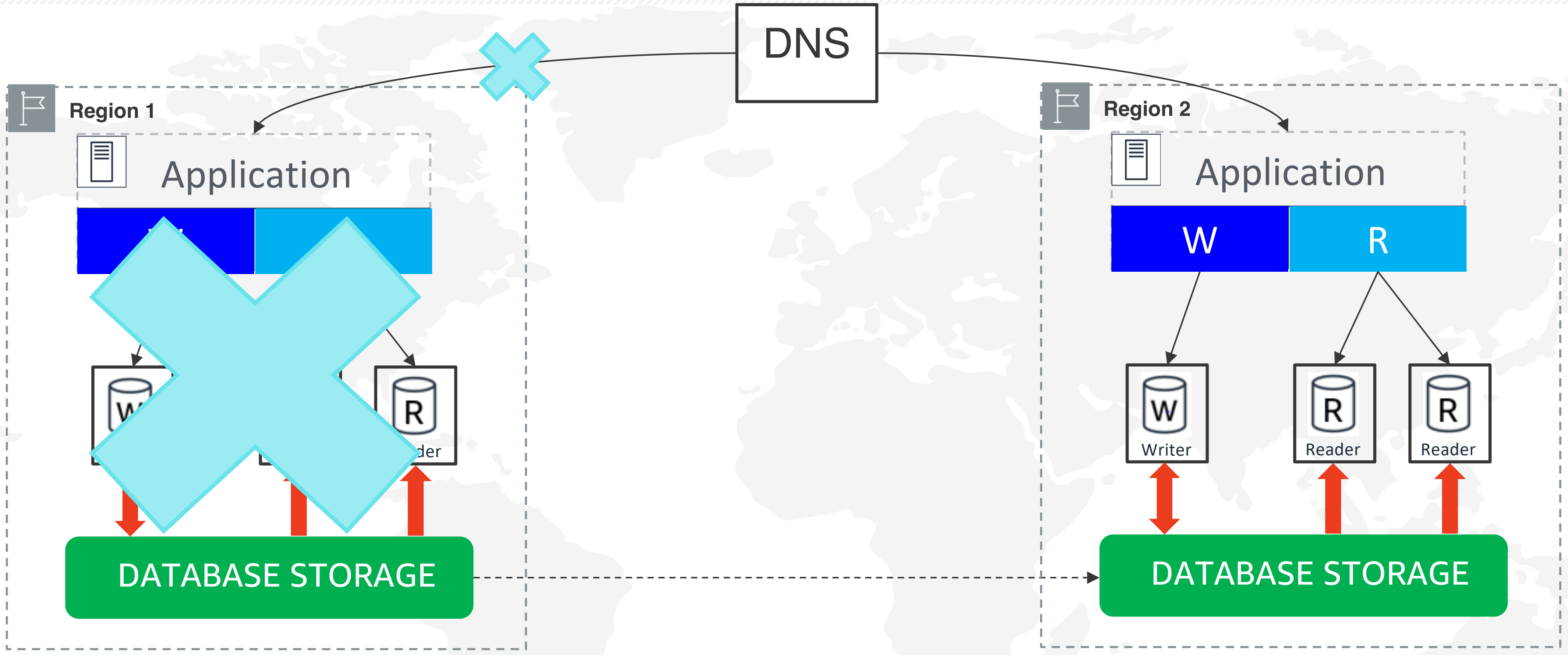


Cross-region setup with Aurora Global Database



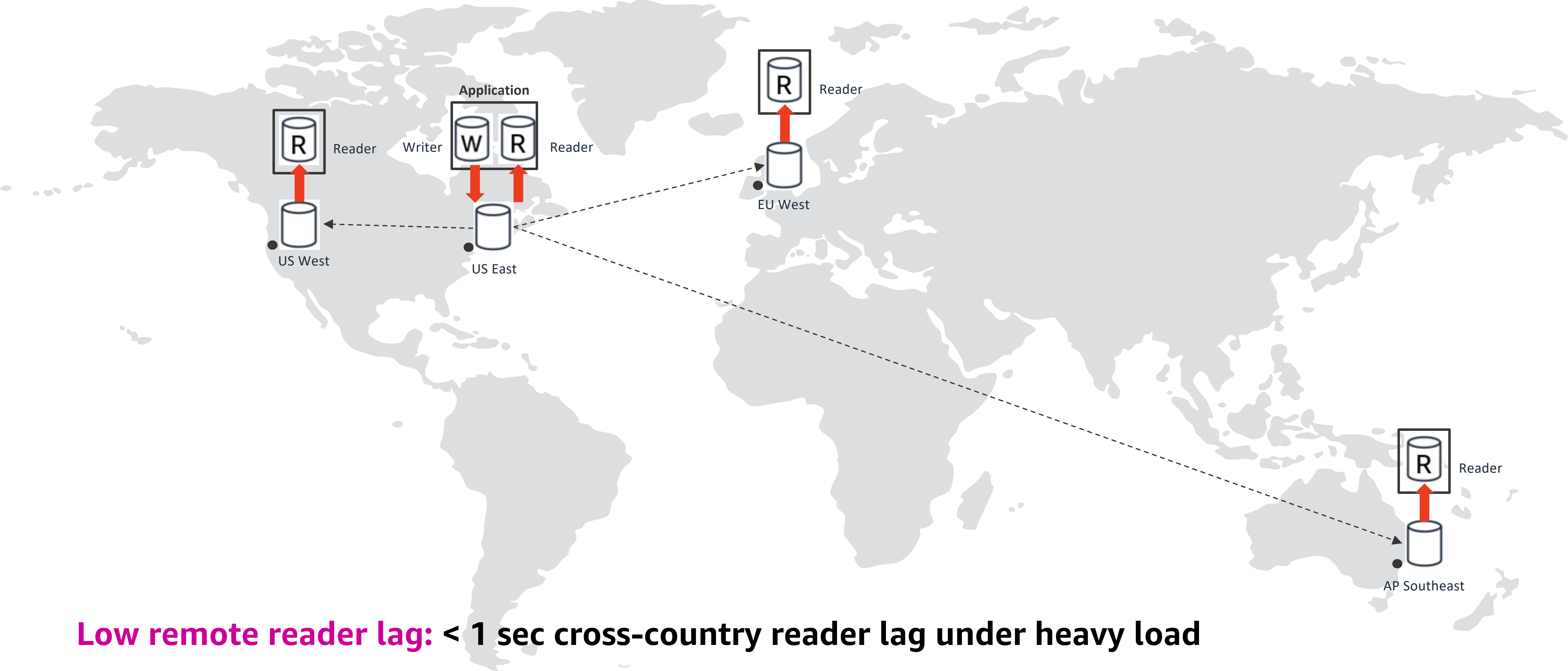
Version of the application stack is set up in another region to provide disaster recovery and serve fast local reads

Failover with Aurora Global Database



Application stack can be spun up in the secondary region and traffic can begin to be routed there (via DNS) with the promoted writer serving write requests within 1 minute

Global reads with low replication latency



Low remote reader lag: **< 1 sec cross-country reader lag under heavy load**

Q&A and Thank You!

kmj@amazon.com