Amazon Aurora DSQL
The Future of Scalable, Distributed Relational Databases

AVAILABLE IN PUBLIC PREVIEW

Samuel Baruffi

Principal Solutions Architect @ aws

Traditional relational databases

Challenges

Scale

Capacity limits of traditional databases results in significant effort to right-size workloads

Availability

Lower resiliency leads to unplanned downtimes impacting database availability

Functional challenges

Management Overhead

Infrastructure management, patching, upgrades requires high engineering effort for maintenance

Complexity

Infrastructure setup, database installation, configuration, tuning and management is complex and high effort

Operational challenges

Aurora DSQL

AVAILABLE IN PREVIEW TODAY

CLOUD-NATIVE, SERVERLESS DISTRIBUTED SQL DATABASE WITH VIRTUALLY UNLIMITED SCALABILITY AND HIGHEST AVAILABILITY

Virtually unlimited scaling

Scales compute, read, write and storage resources independently, both up and down

Business Continuity

99.999% availability with active-active access in a multi-region cluster

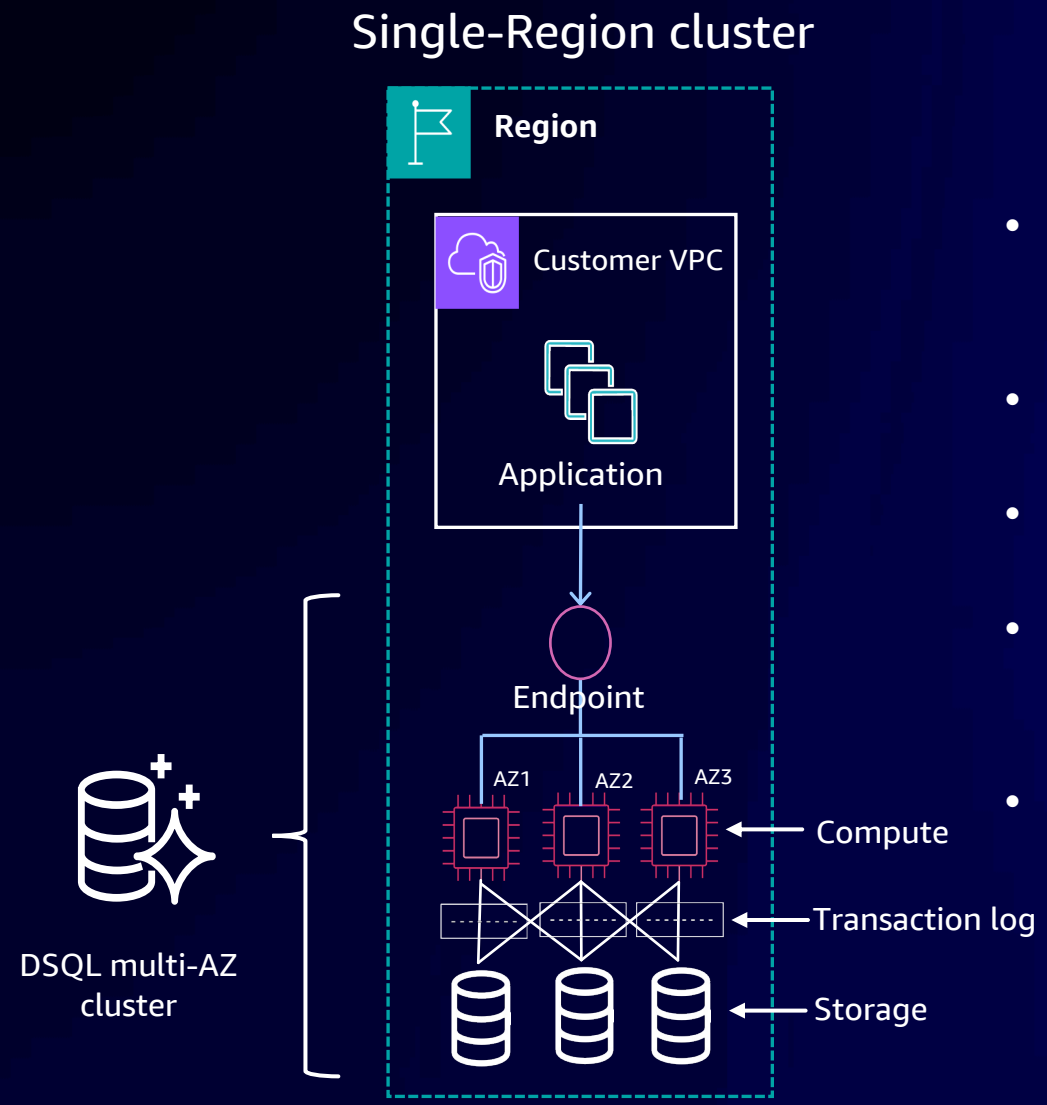
Serverless

No servers to provision, patch, or manage, and no software to install, maintain, or operate.

Fast and Easy

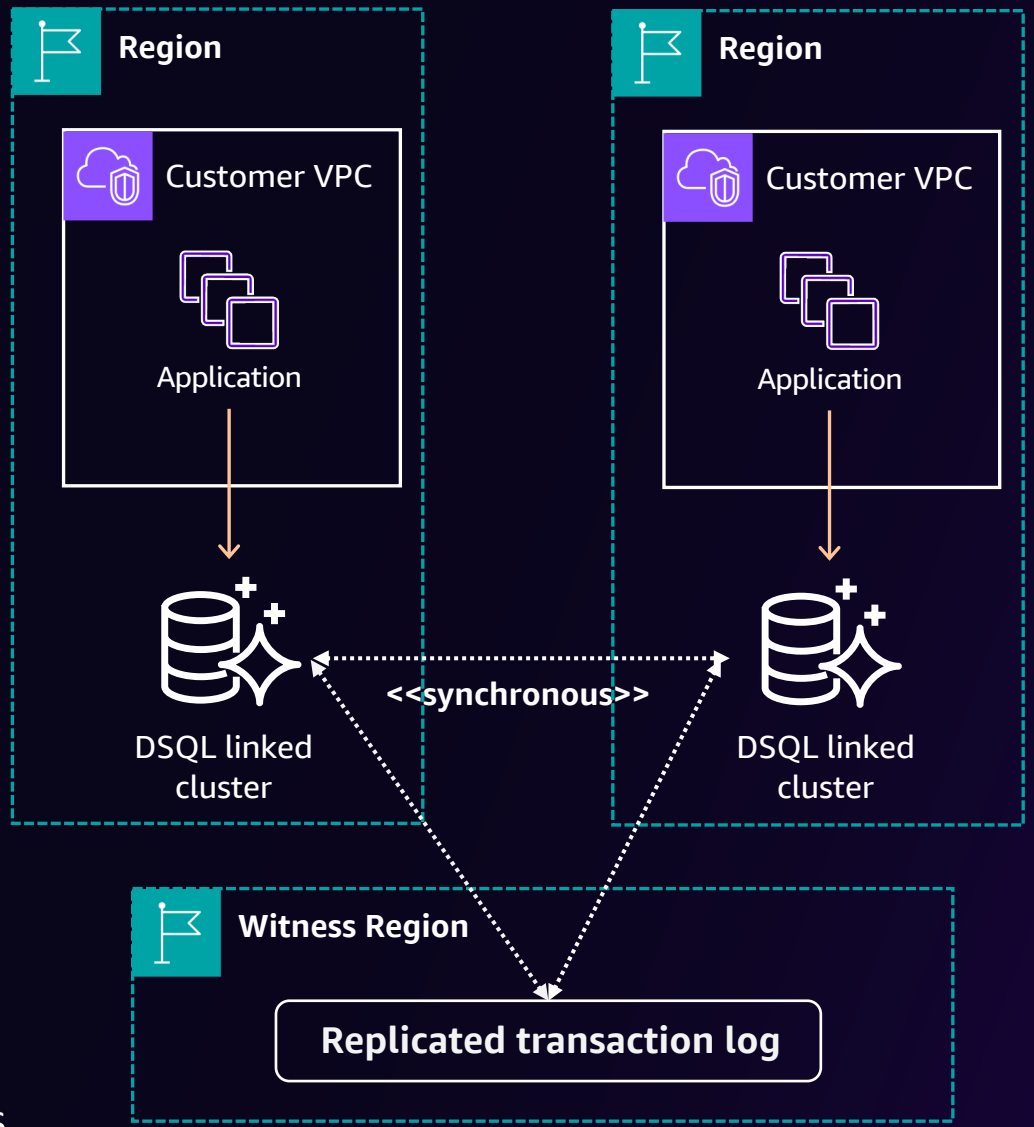
Create and query a database in less than a minute with familiarity of PostgreSQL

Aurora DSQL: Single Region Cluster



- Single-region clusters operate actively across 3 AZs always
- Transactions maintain ACID properties
- Transactions, are fast and local
- Transaction commit goes across AZs ensuring that your transactions are durable, isolated, and atomic.
- Designed for 99.99% availability

Aurora DSQL: Multi-Region Cluster

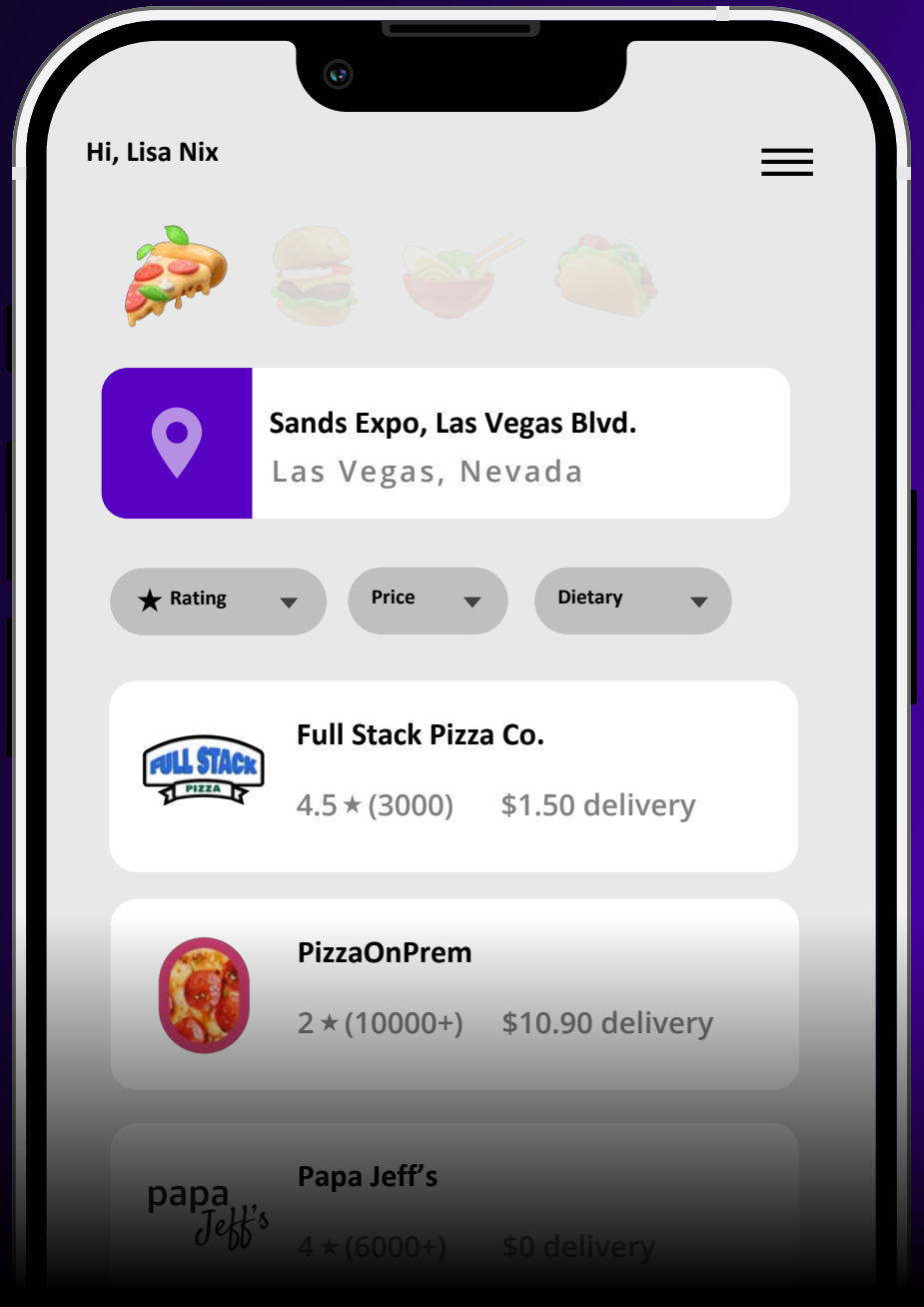


- Aurora DSQL delivers 99.999% availability across multiple Regions
- Multi-Region clusters provide two Regional endpoints,
- A witness Region stores encrypted transaction logs without hosting a cluster, supporting multi-Region durability and availability
- Read operations are local
- The regions are equal peers
- Synchronous replication between regions, always RPO 0

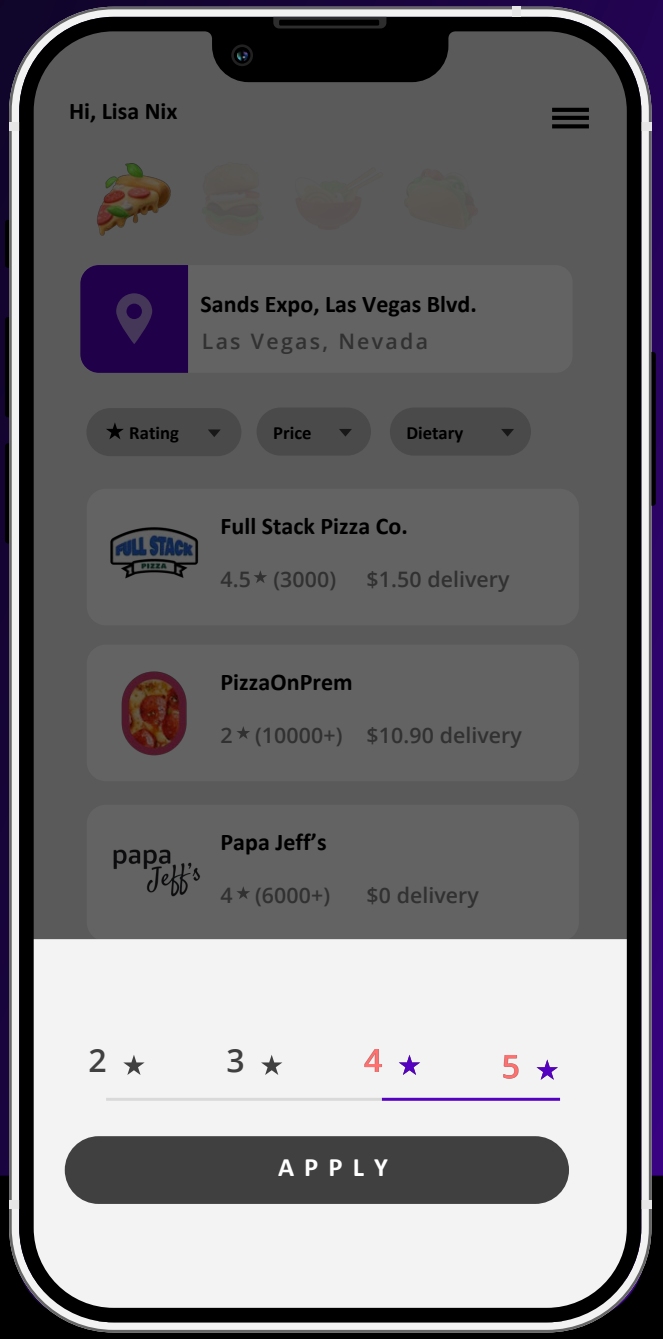
Aurora DSQL: Components

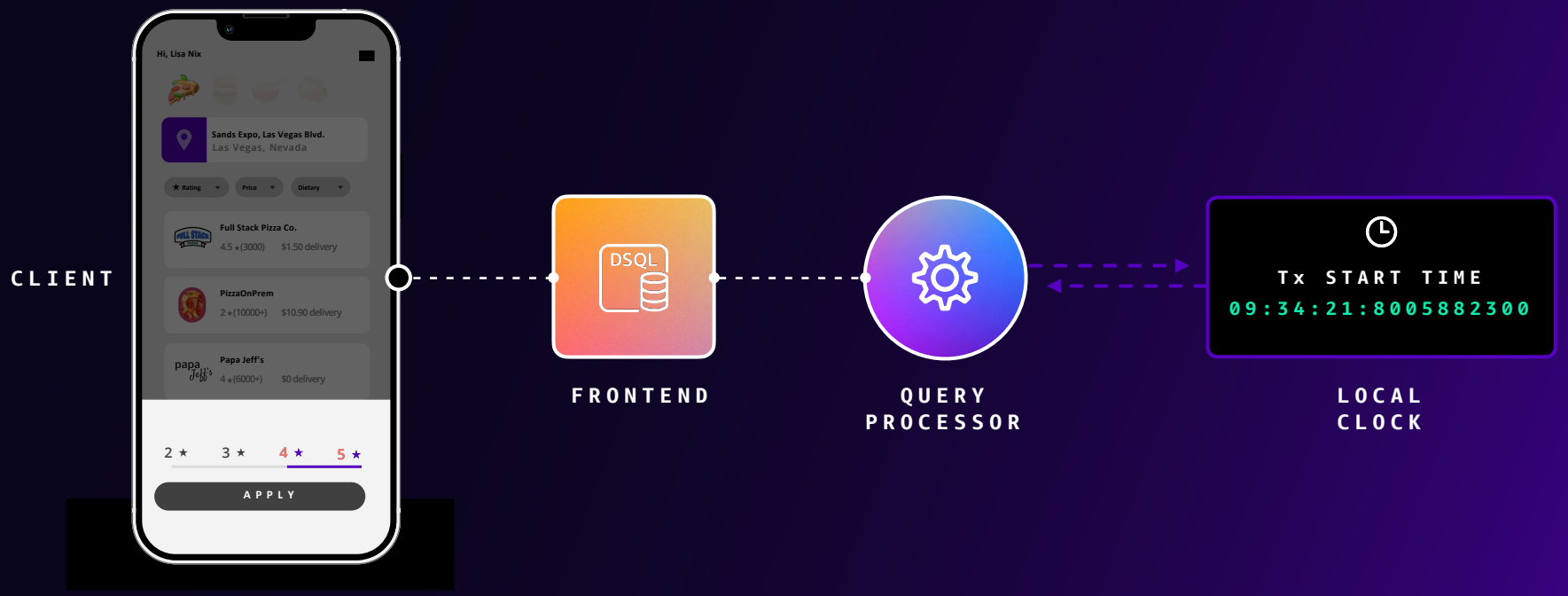
Amazon Aurora DSQL

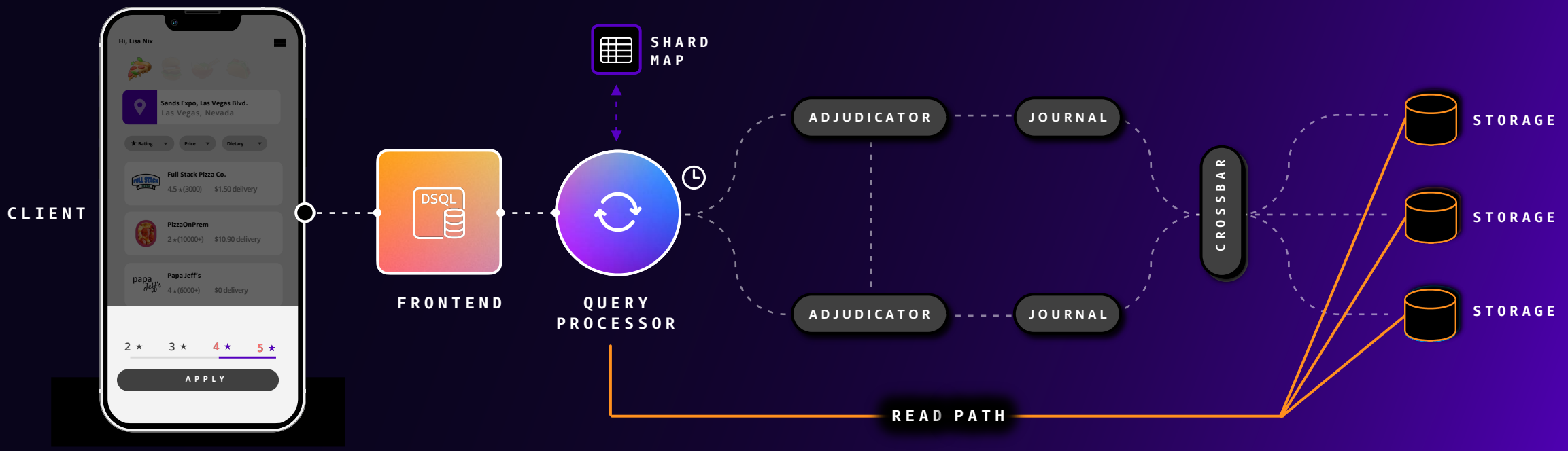
READ transaction

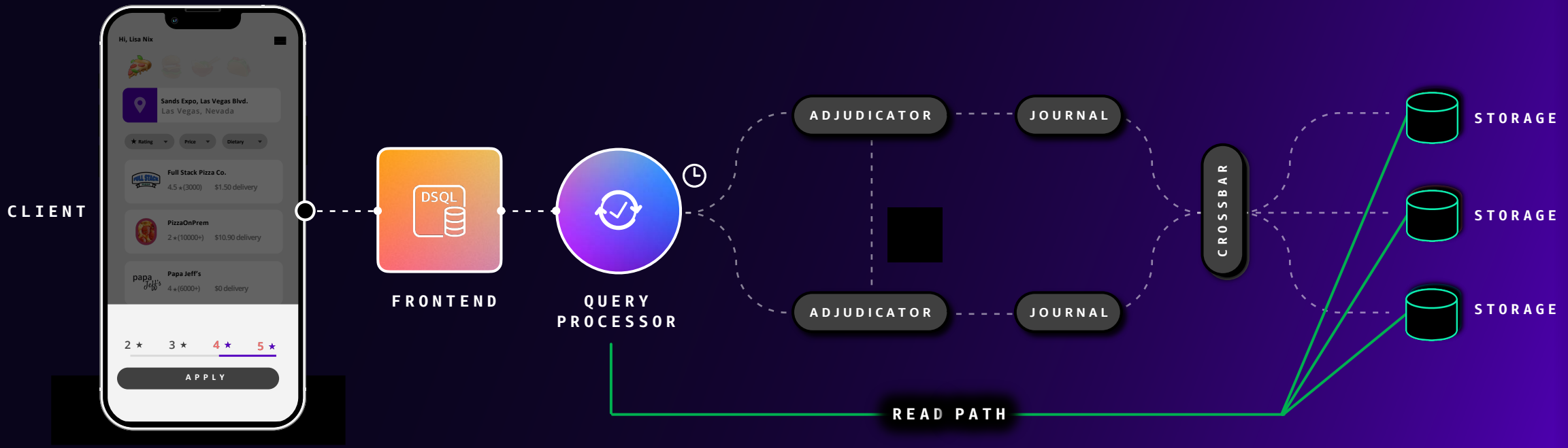


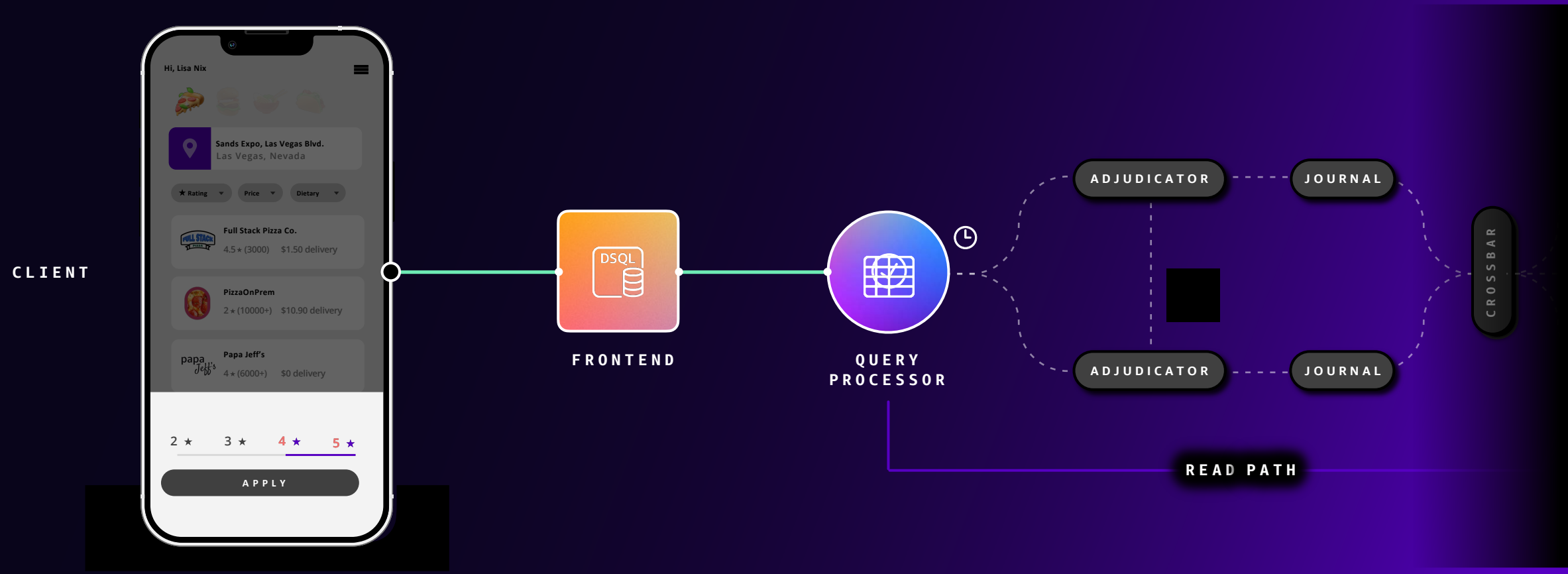
```
SELECT * FROM Restaurants
WHERE rating >= 4.0;
```







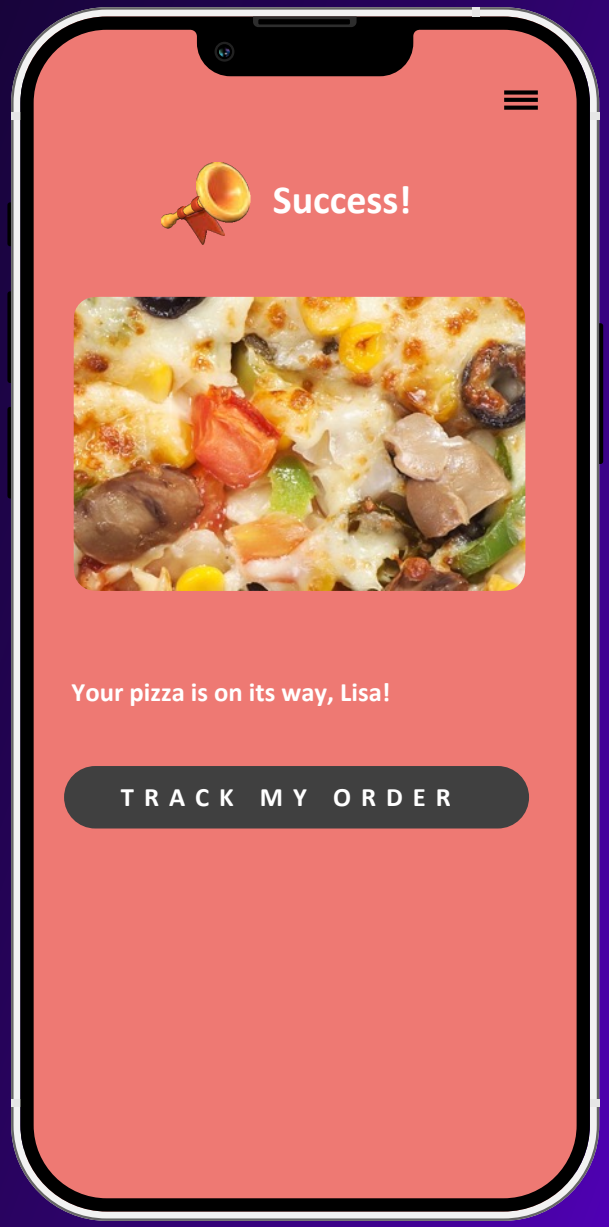


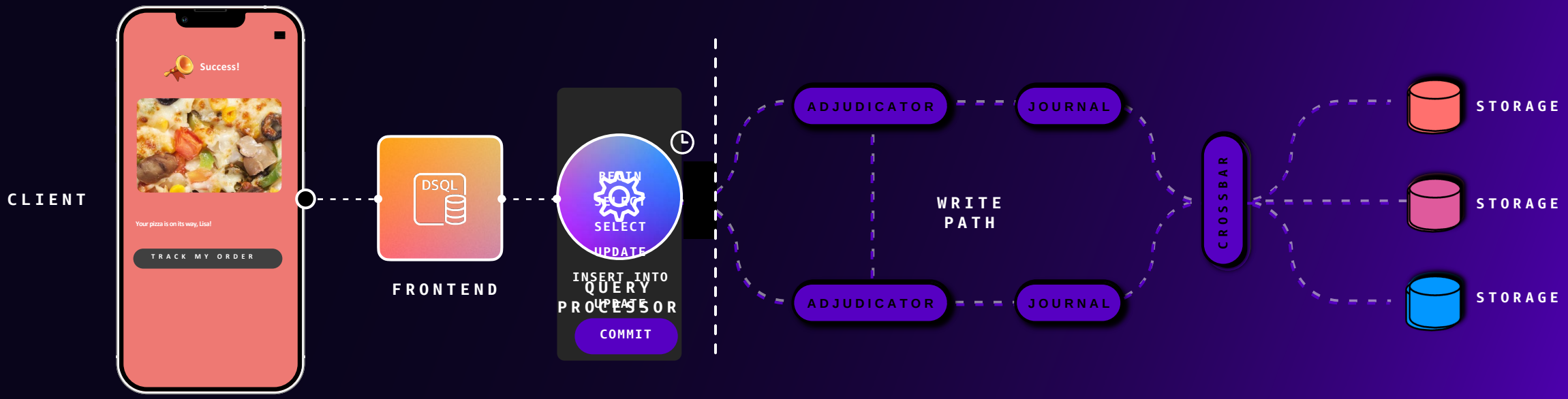


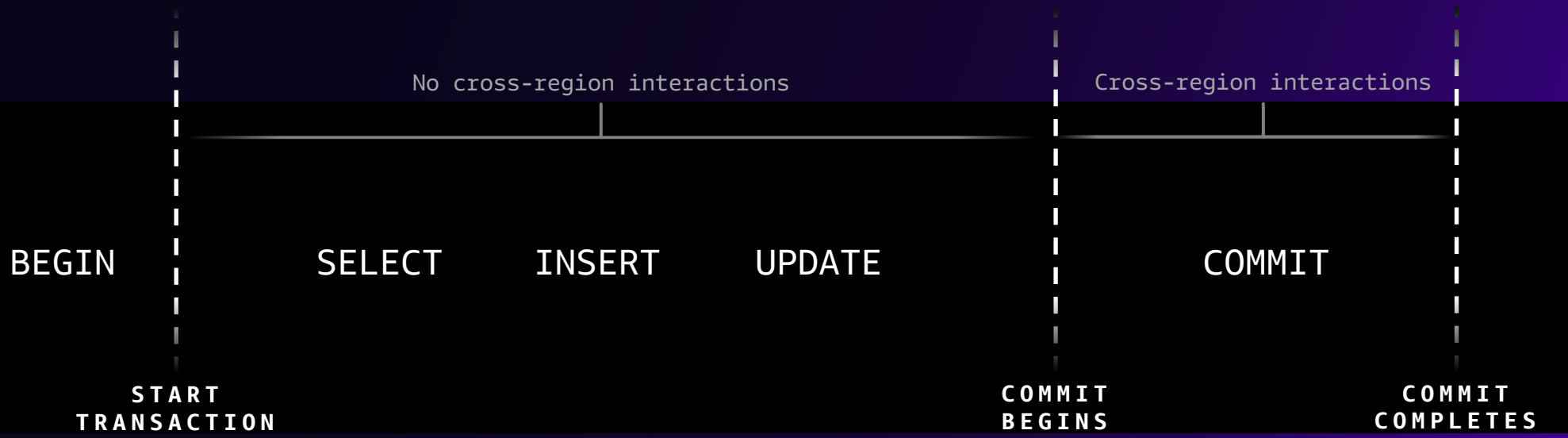
Amazon Aurora DSQL

Interactive transaction

```
BEGIN;
SELECT * FROM Restaurants WHERE rating >= 4.0;
SELECT * FROM Items WHERE restaurant_id = 1;
SELCT * FROM Items WHERE item_id = 3;
UPDATE Items SET qty = qty - 1 WHERE item_id = 3;
INSERT INTO Orders VALUES (NOW(), 3, 1, ...);
UPDATE Users SET last_order = NOW() WHERE id = 7;
COMMIT;
```

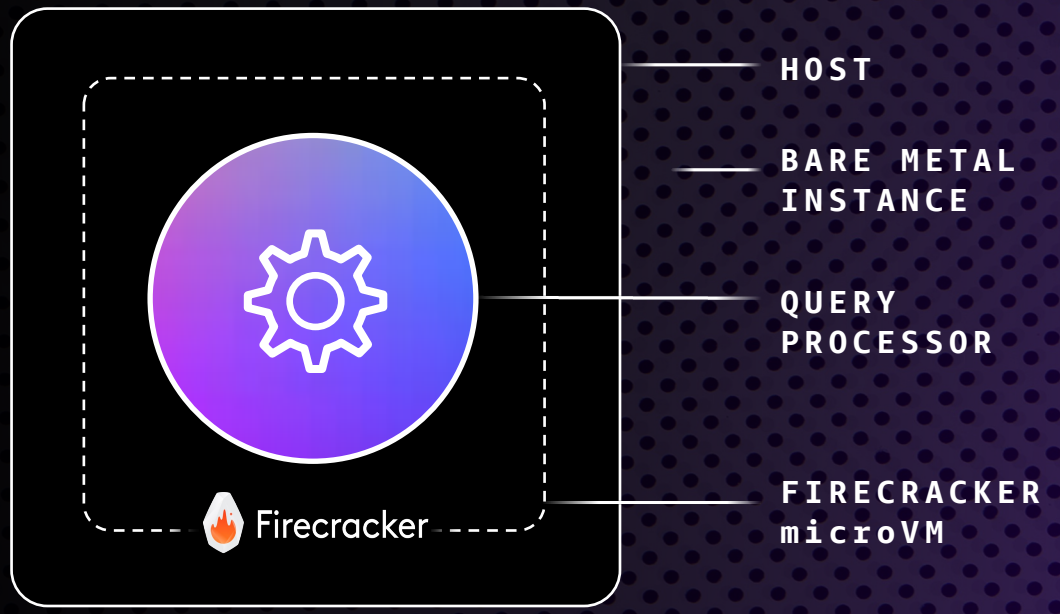




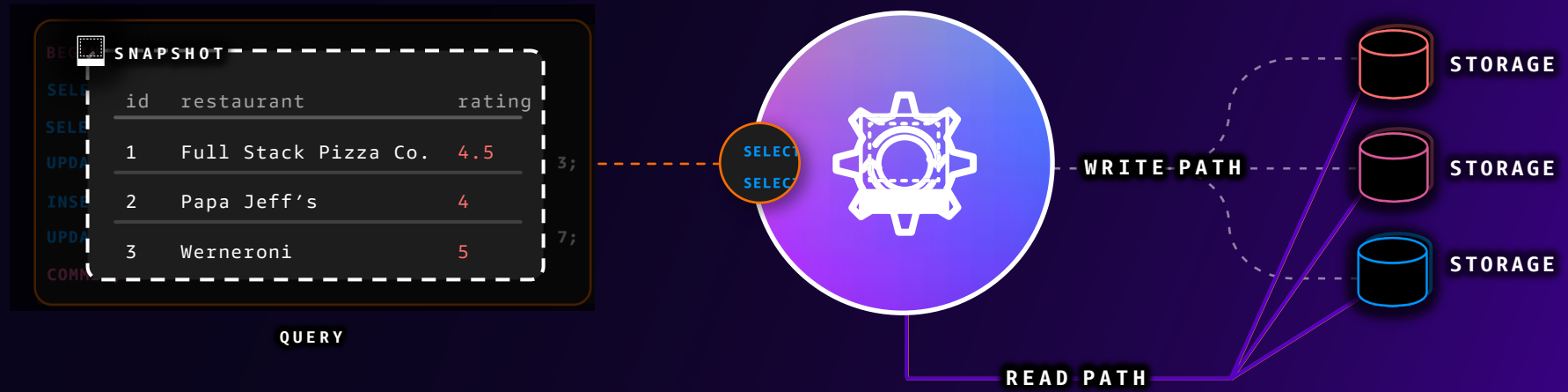


Amazon Aurora DSQL

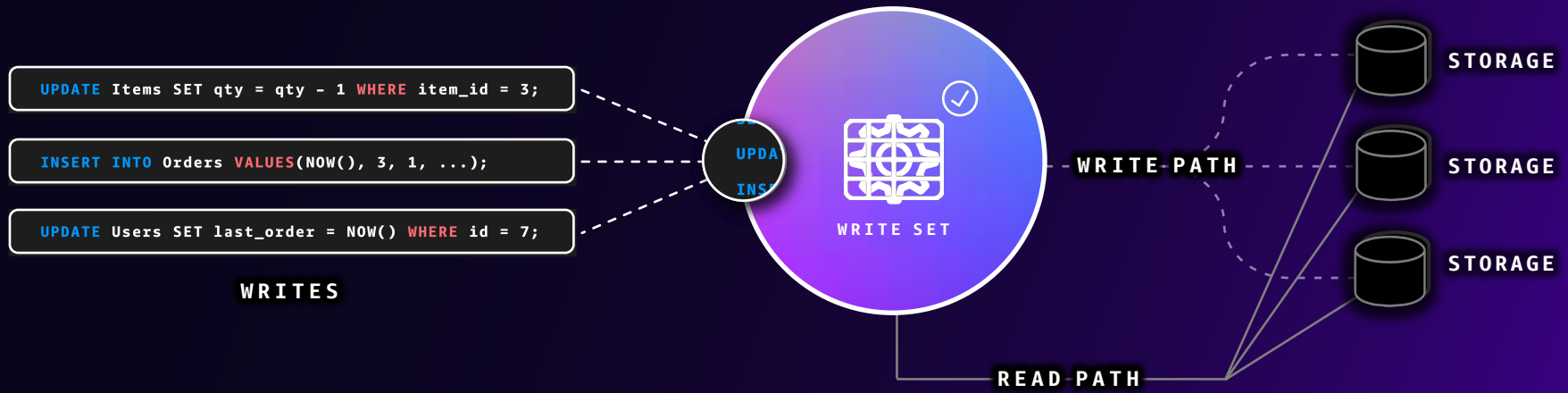
The Query Processor



01 Snapshot isolation

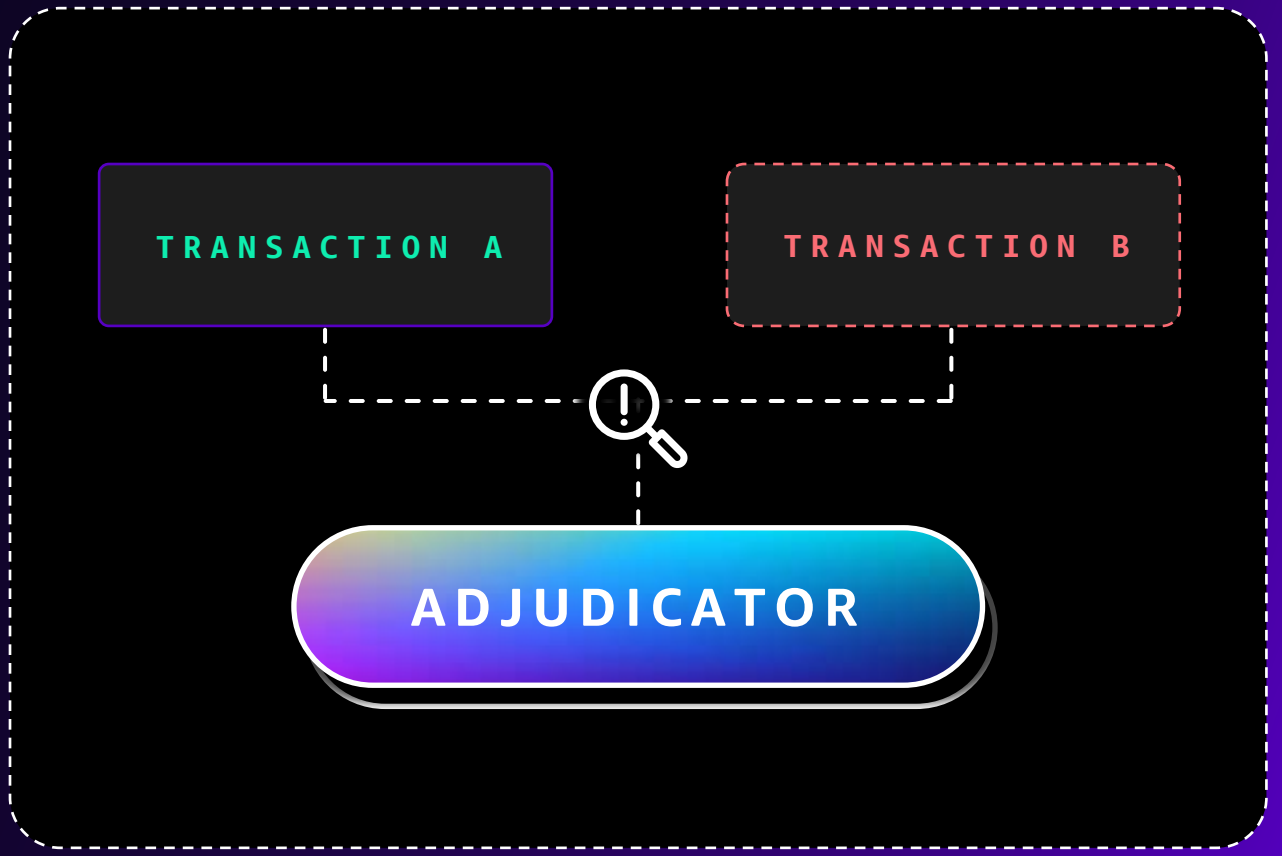


02 Spooling WRITES locally on the Query Processor

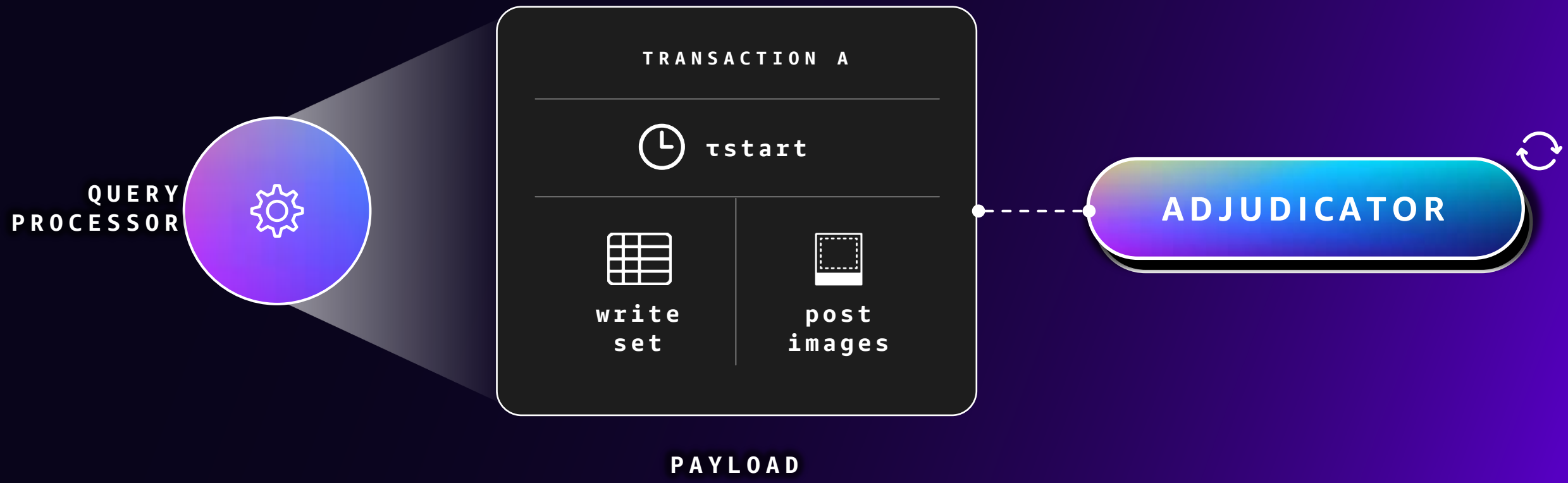


Amazon Aurora DSQL

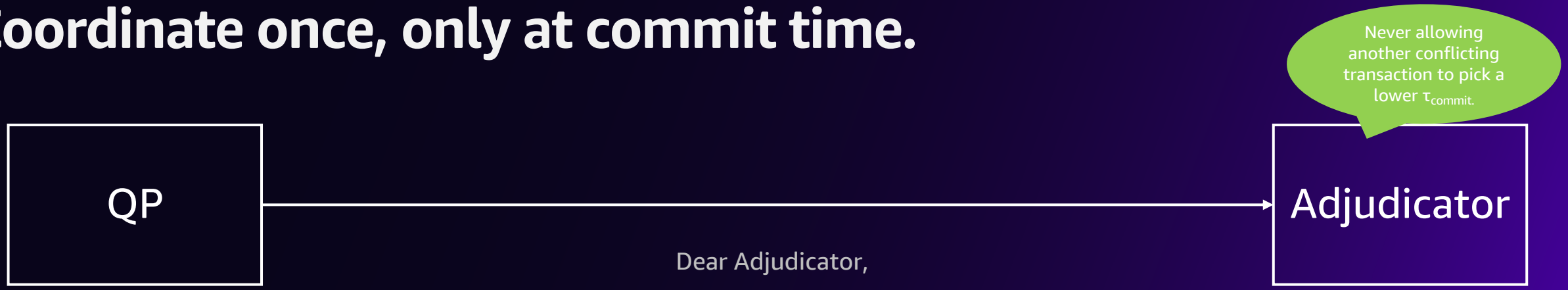
The Adjudicator



QP creates transaction payload

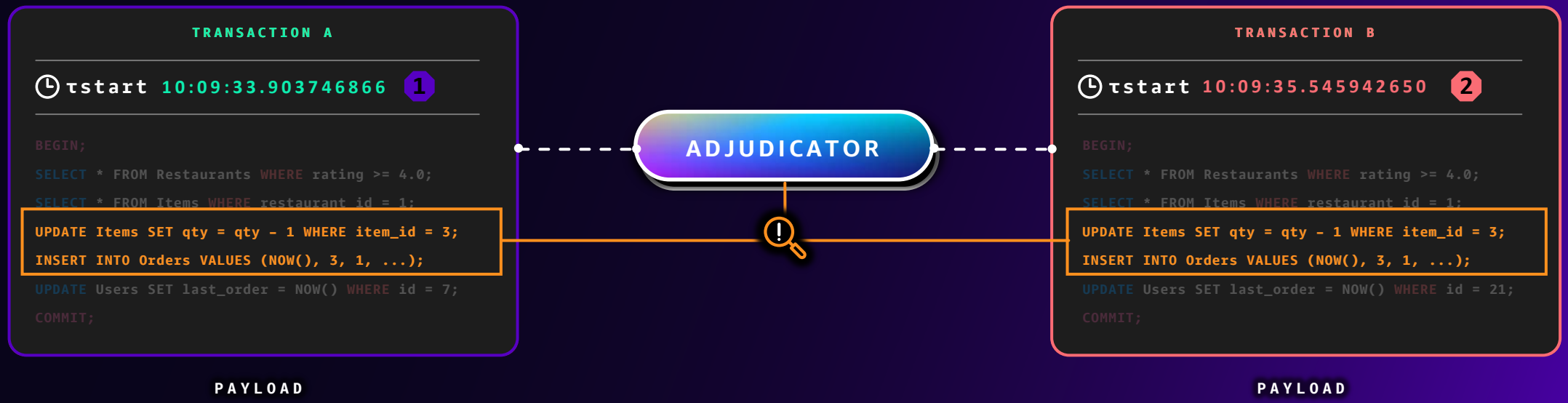


Coordinate once, only at commit time.



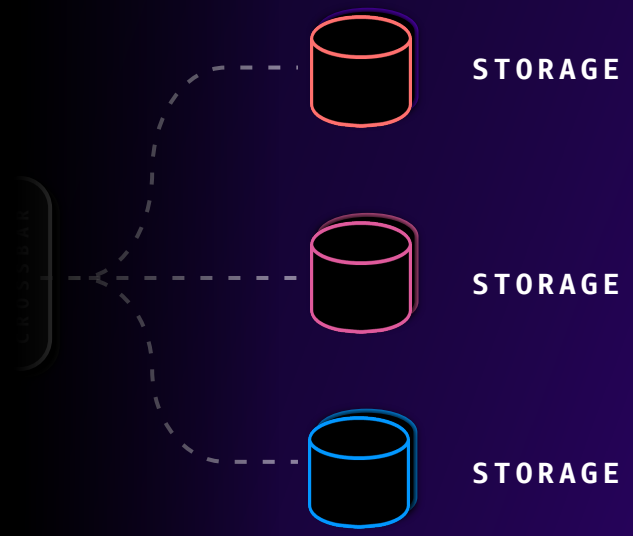
Concurrent transactions

Adjudicator discovers intersecting writes

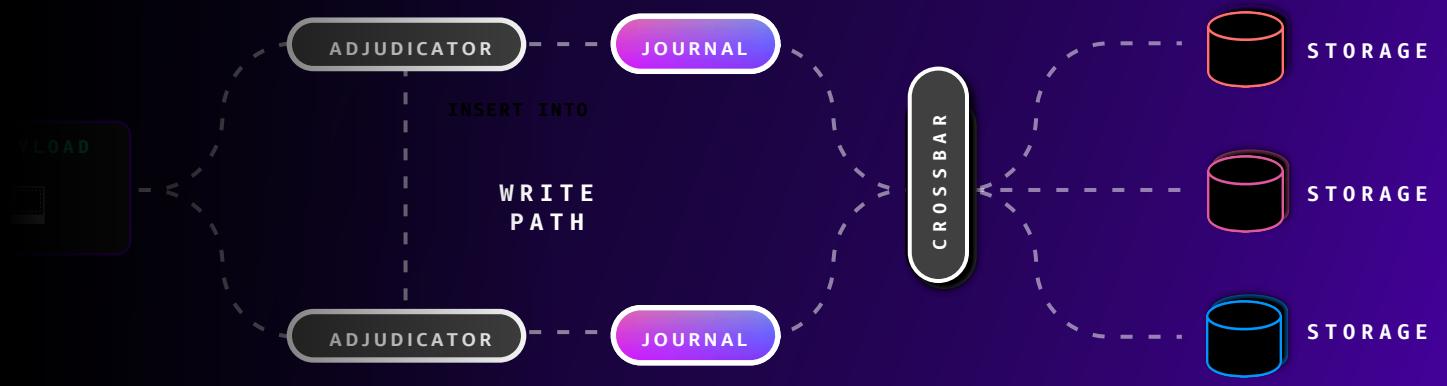


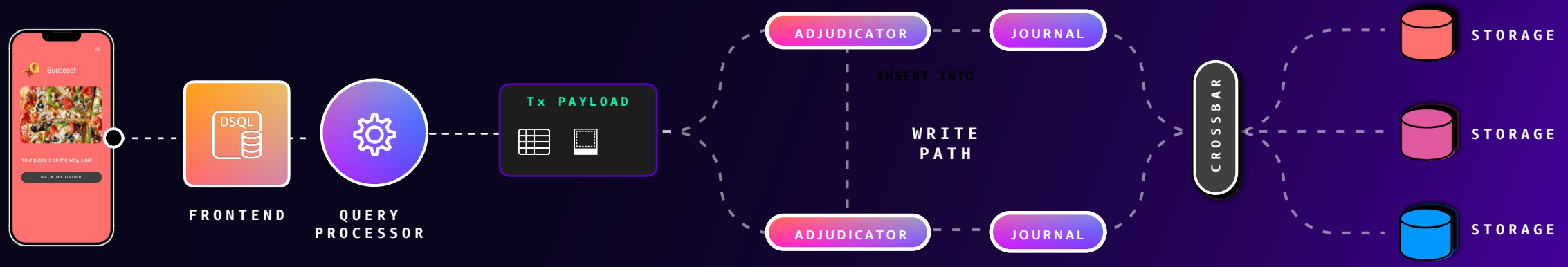
Transactions do not intersect; commit proceeds

Traditionally,
durability happens
in the **storage layer**



But DSQL makes the **journal** responsible for durability





Aurora DSQL

OPTIMISTIC CONCURRENCY CONTROL

No pessimistic locking

No deadlocks

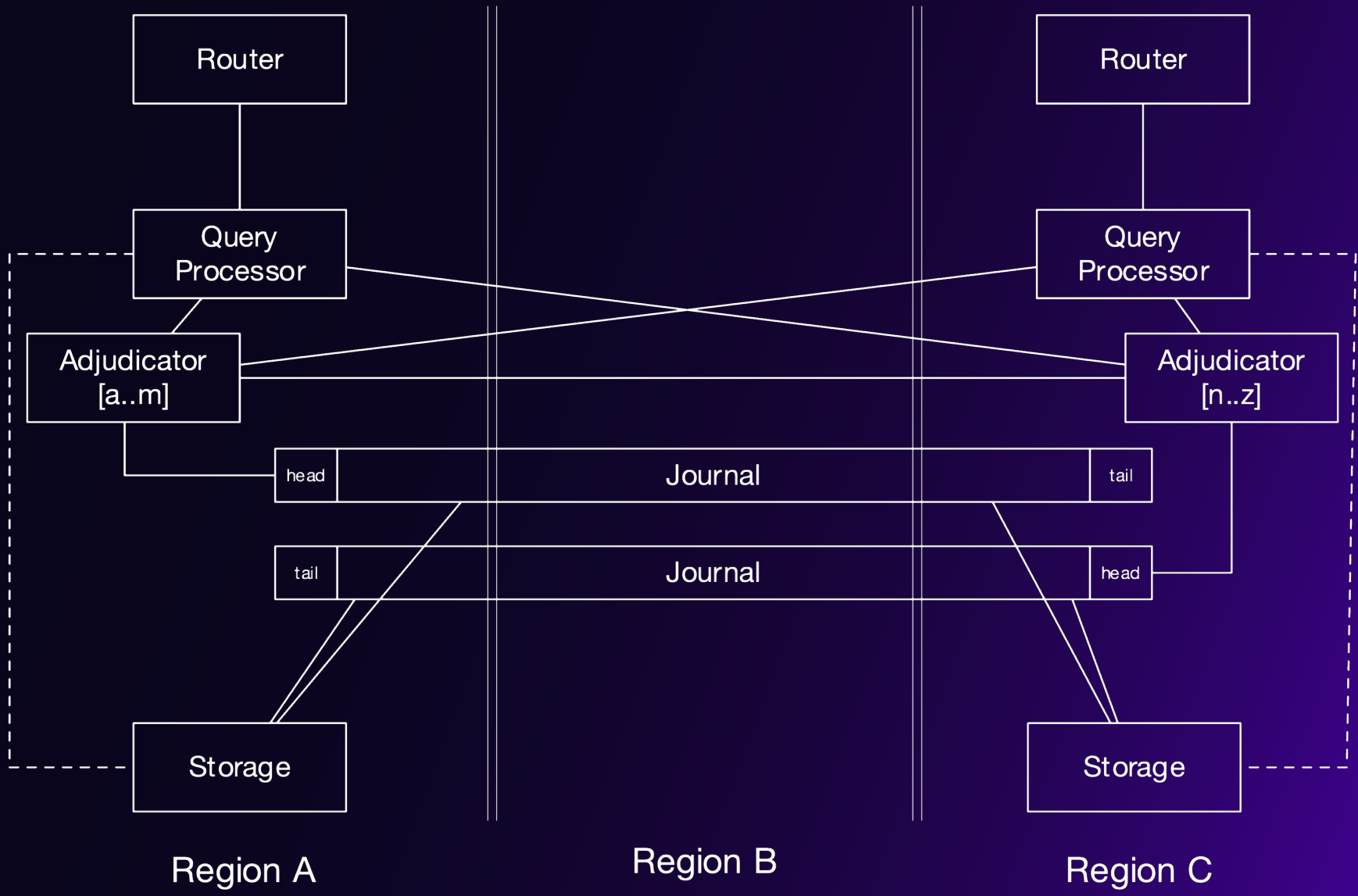
Conflicts detected at transaction commit

Multi-region concurrency protection

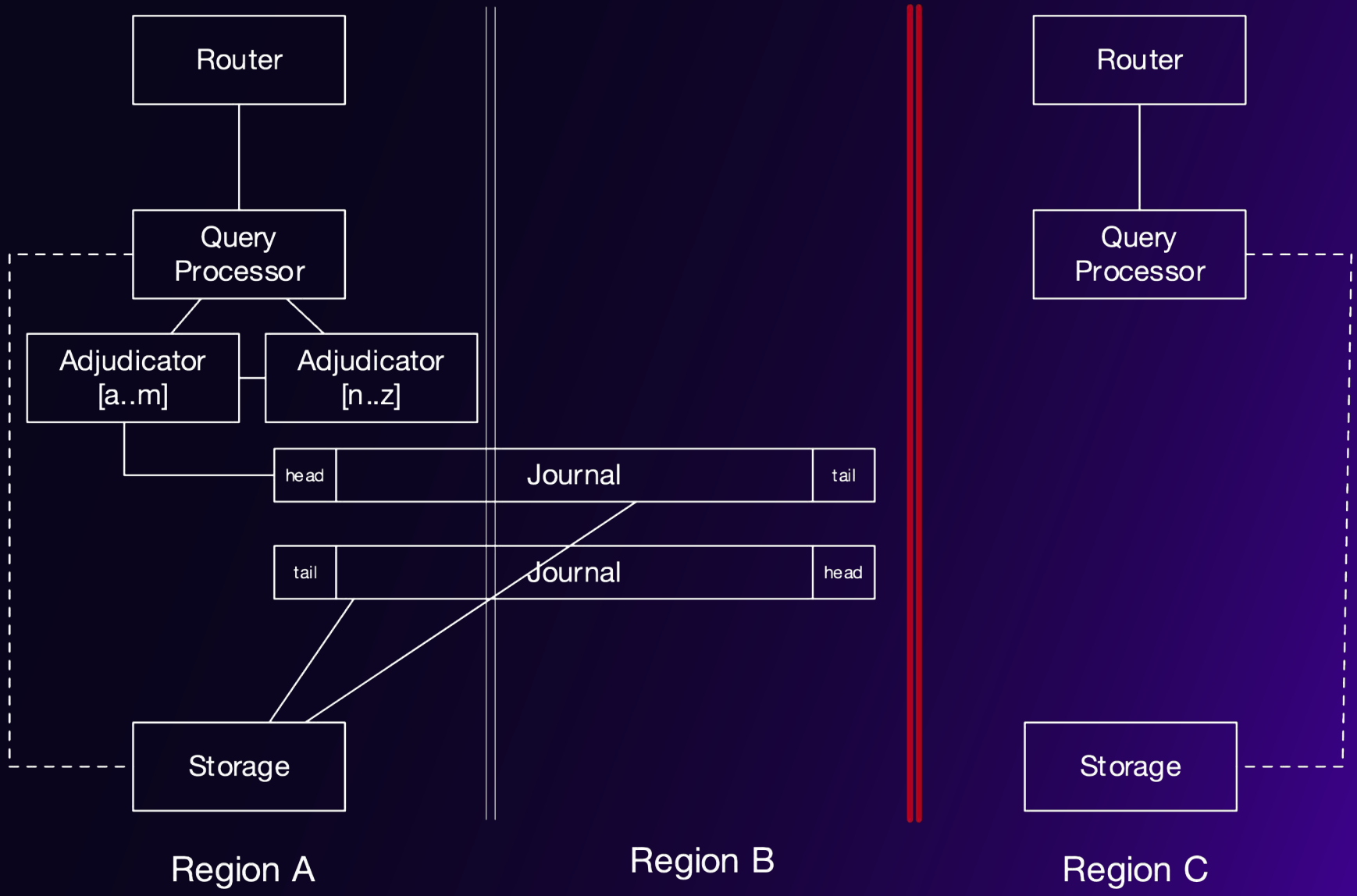
Efficient storage-layer synchronous replication

New transaction patterns for developers

Active Active architecture



Optimized for failover



Resources to Getting Started

For more information, visit:

Aurora DSQL resources

Aurora DSQL documentation

Aurora DSQL Useful Blogs

Thank you!