
AWS
re:Invent

DAT301

Deep dive on Amazon Aurora with PostgreSQL compatibility

Grant McAlister

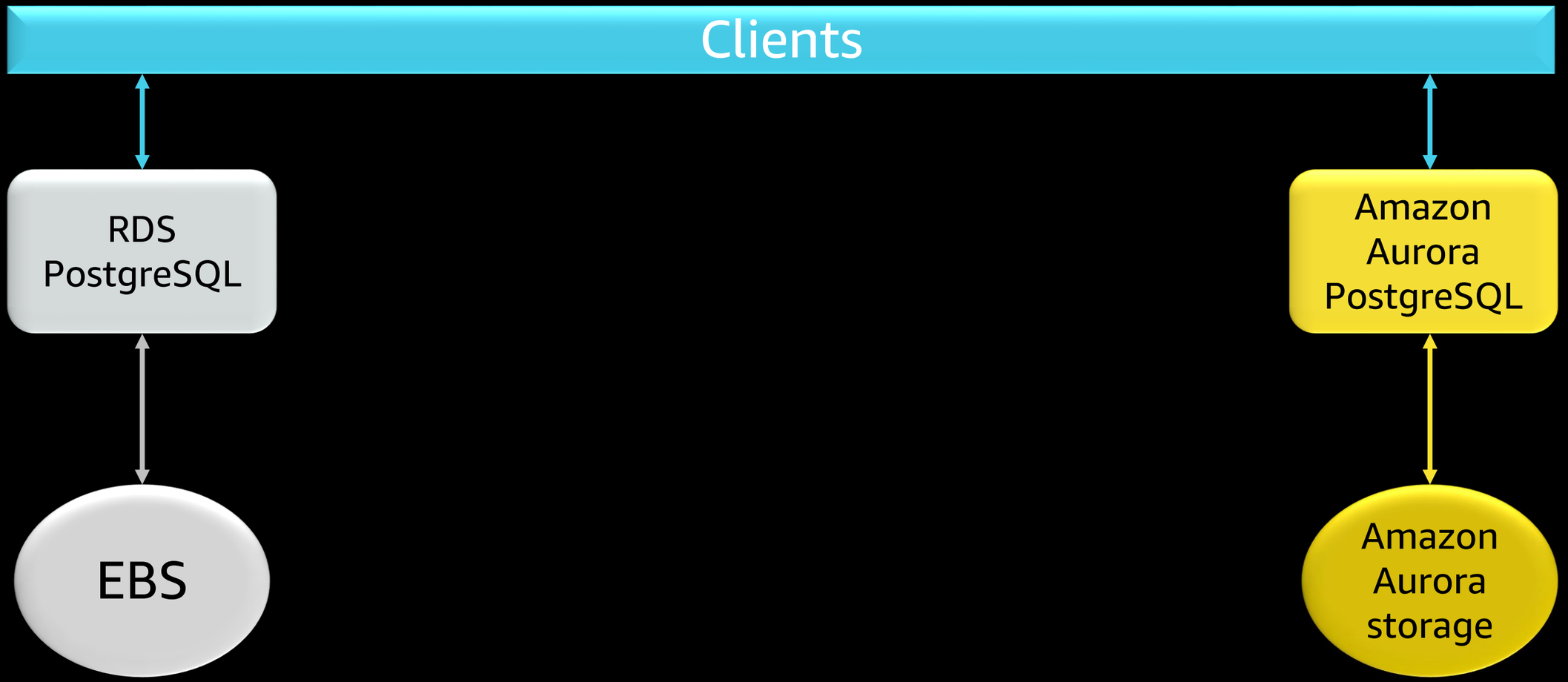
Senior Principal Engineer

AWS

AWS re:Invent

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

Amazon RDS PostgreSQL universe



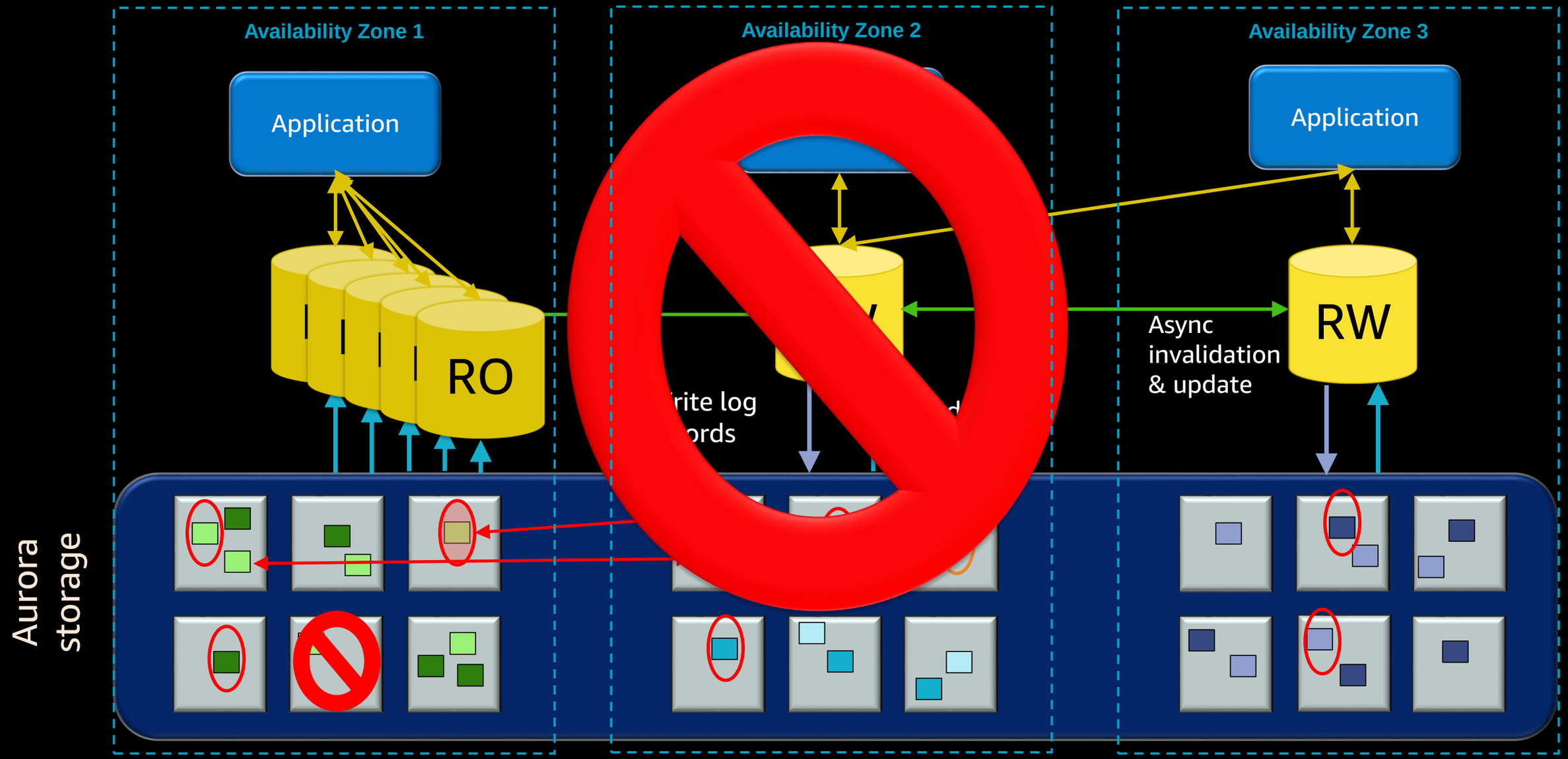
Base architecture



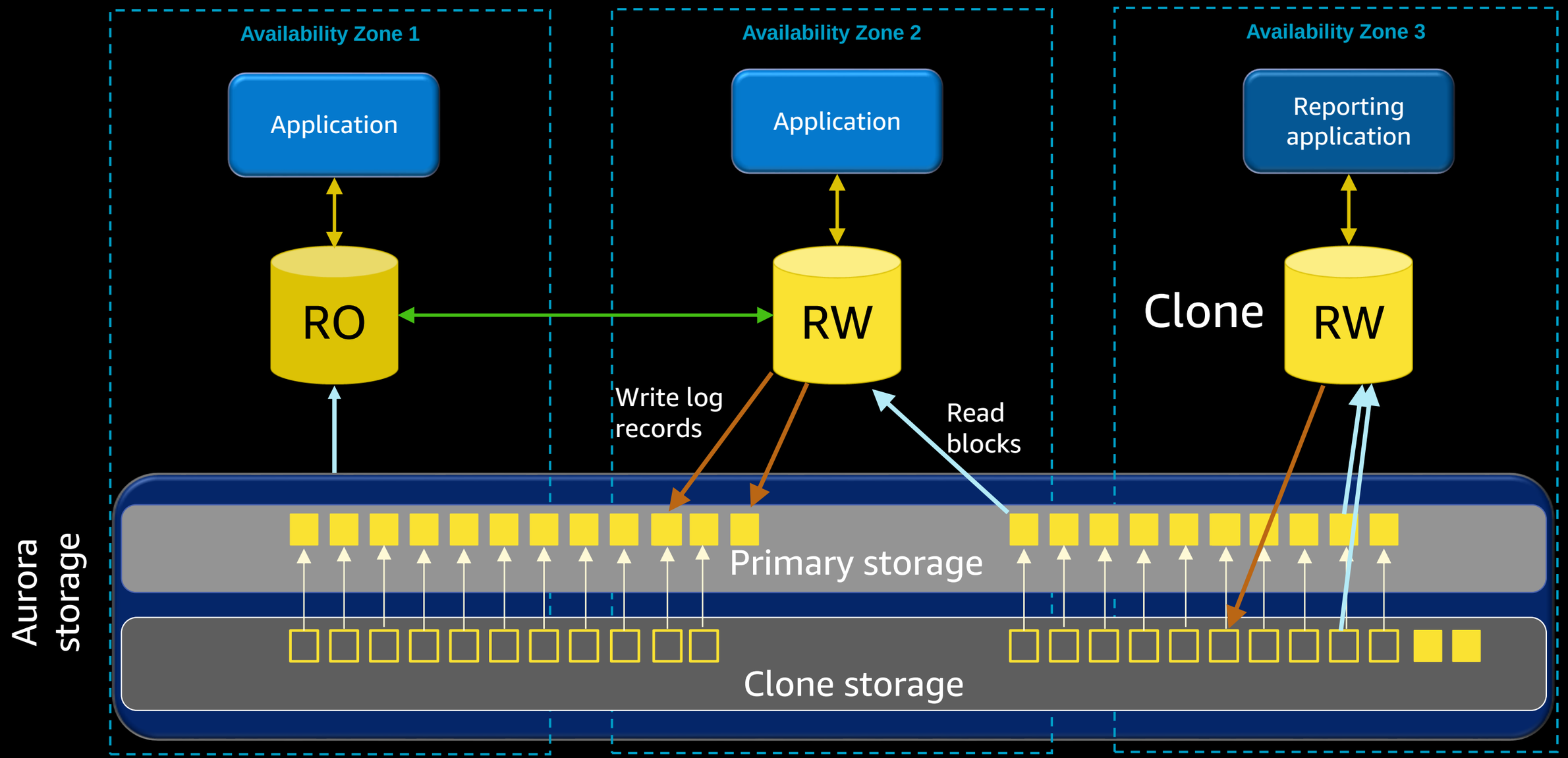
AWS re:Invent

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

Aurora storage and replicas



Fast clones

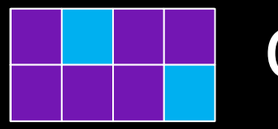

Availability Zone 1
Availability Zone 2
Availability Zone 3
Application
Application
Reporting application
RO
RW
Clone
RW
Write log records
Read blocks
Aurora storage
Primary storage
Clone storage

Log-based storage

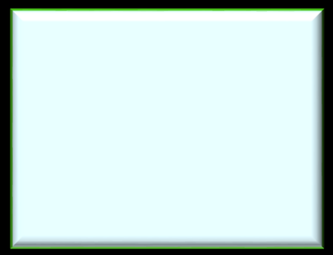


© 2020, Amazon Web Services, Inc. or its affiliates. All rights reserved.

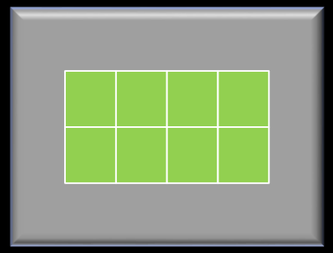
Concurrency: Remove log buffer



Queued work



Log buffer



Storage

PostgreSQL



Queued work

Durability tracking





Storage

Aurora PostgreSQL

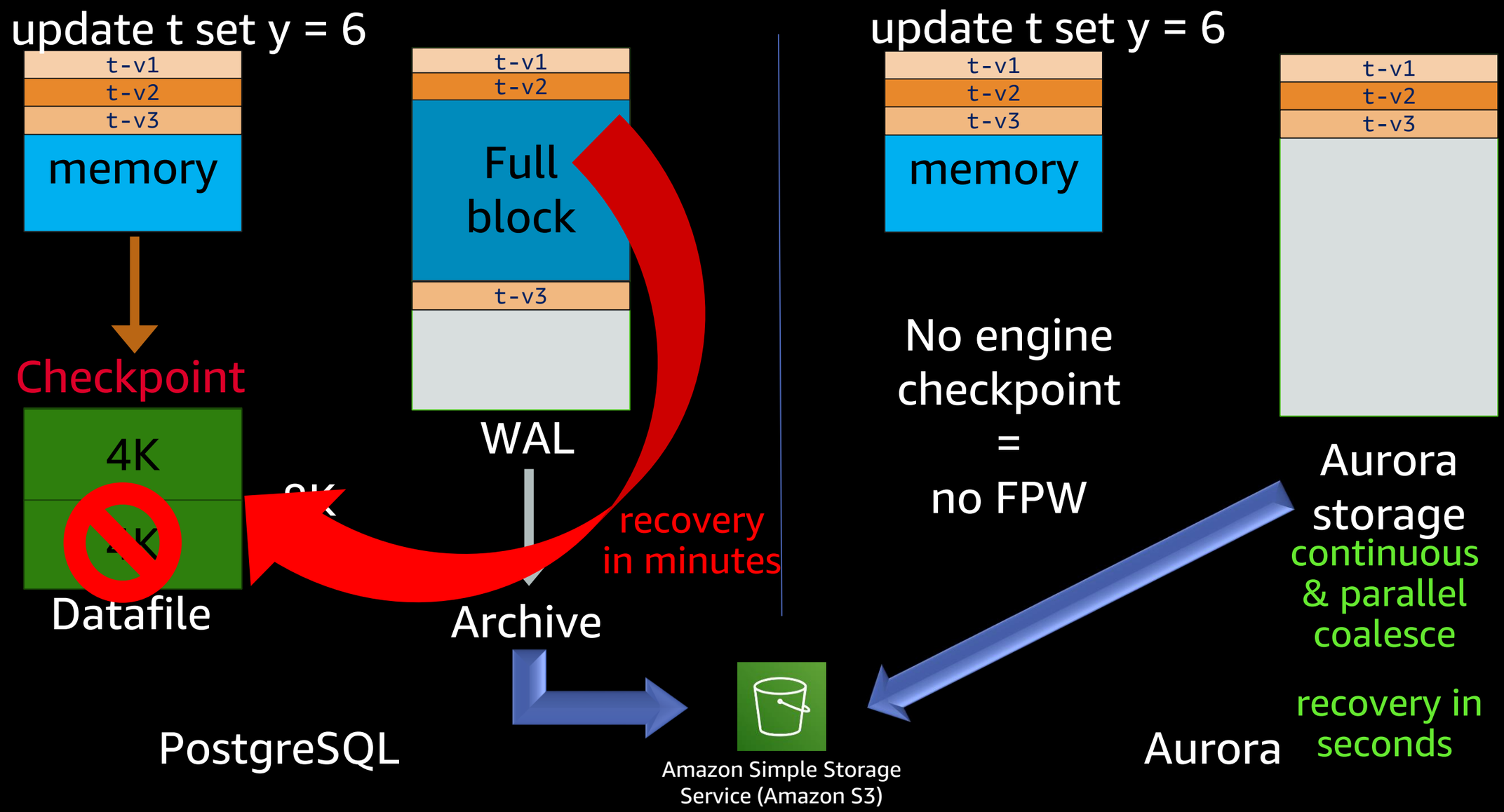


Transaction

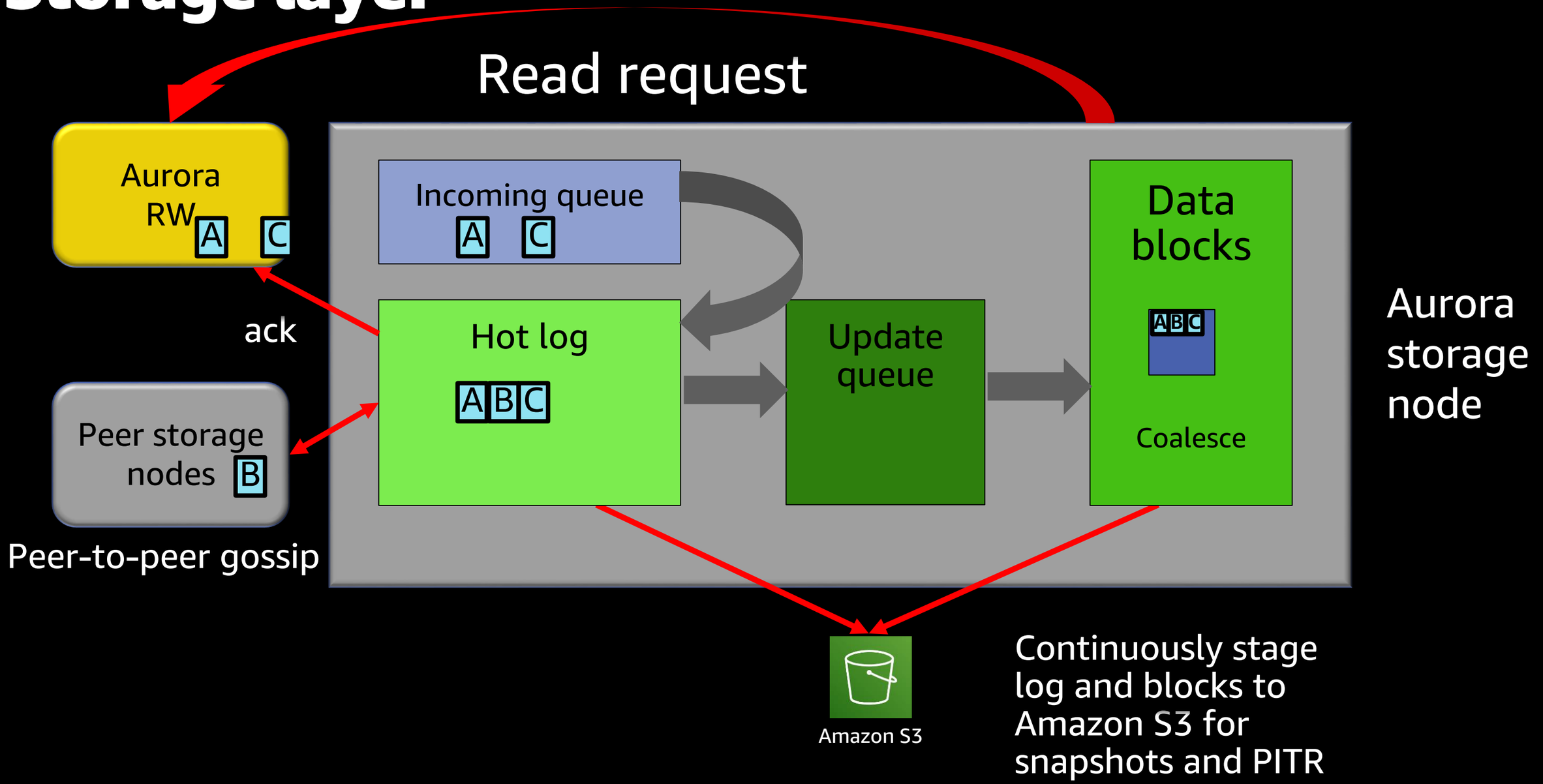


Write

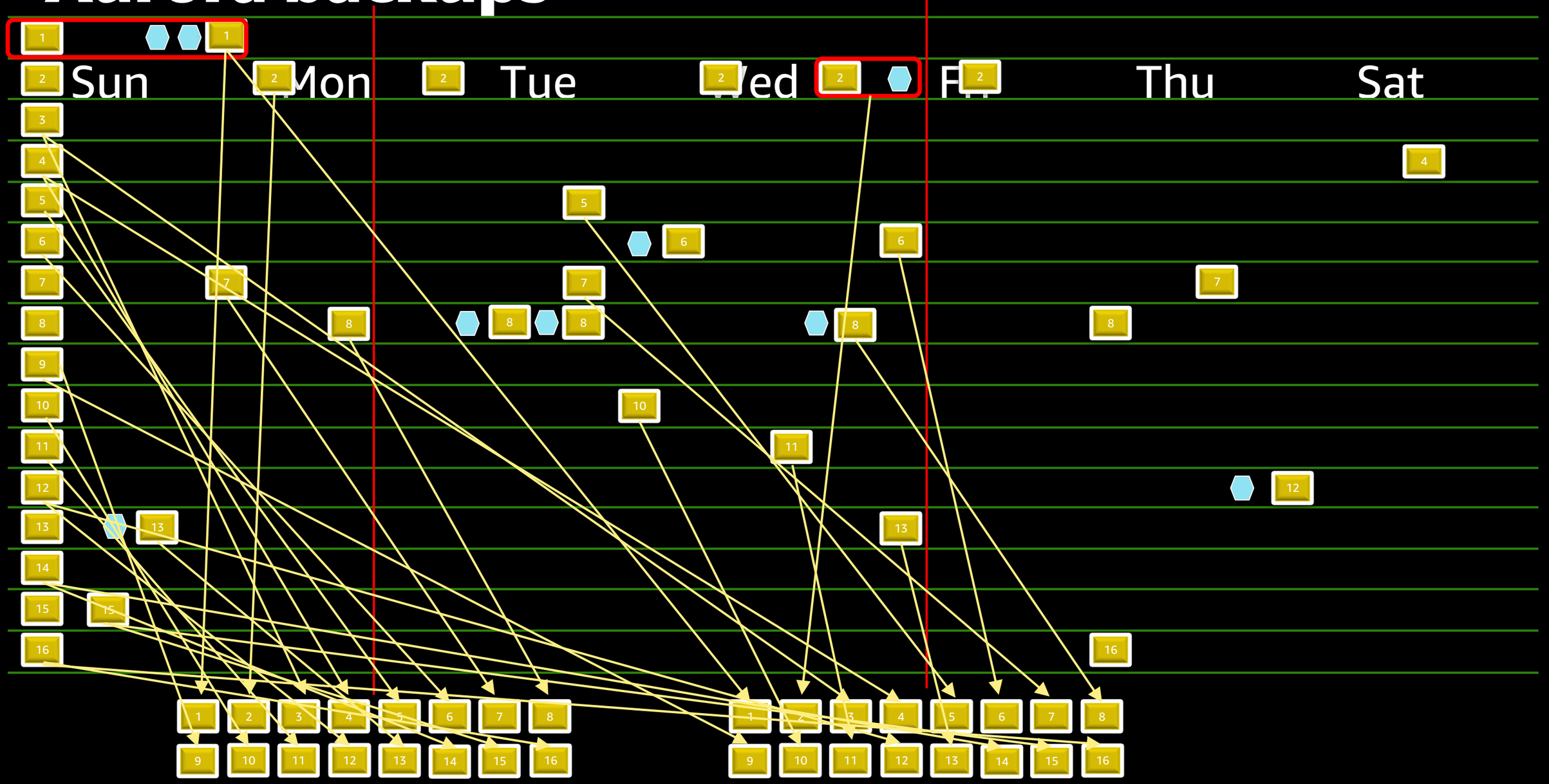
Aurora: Writing less



Storage layer



Aurora backups



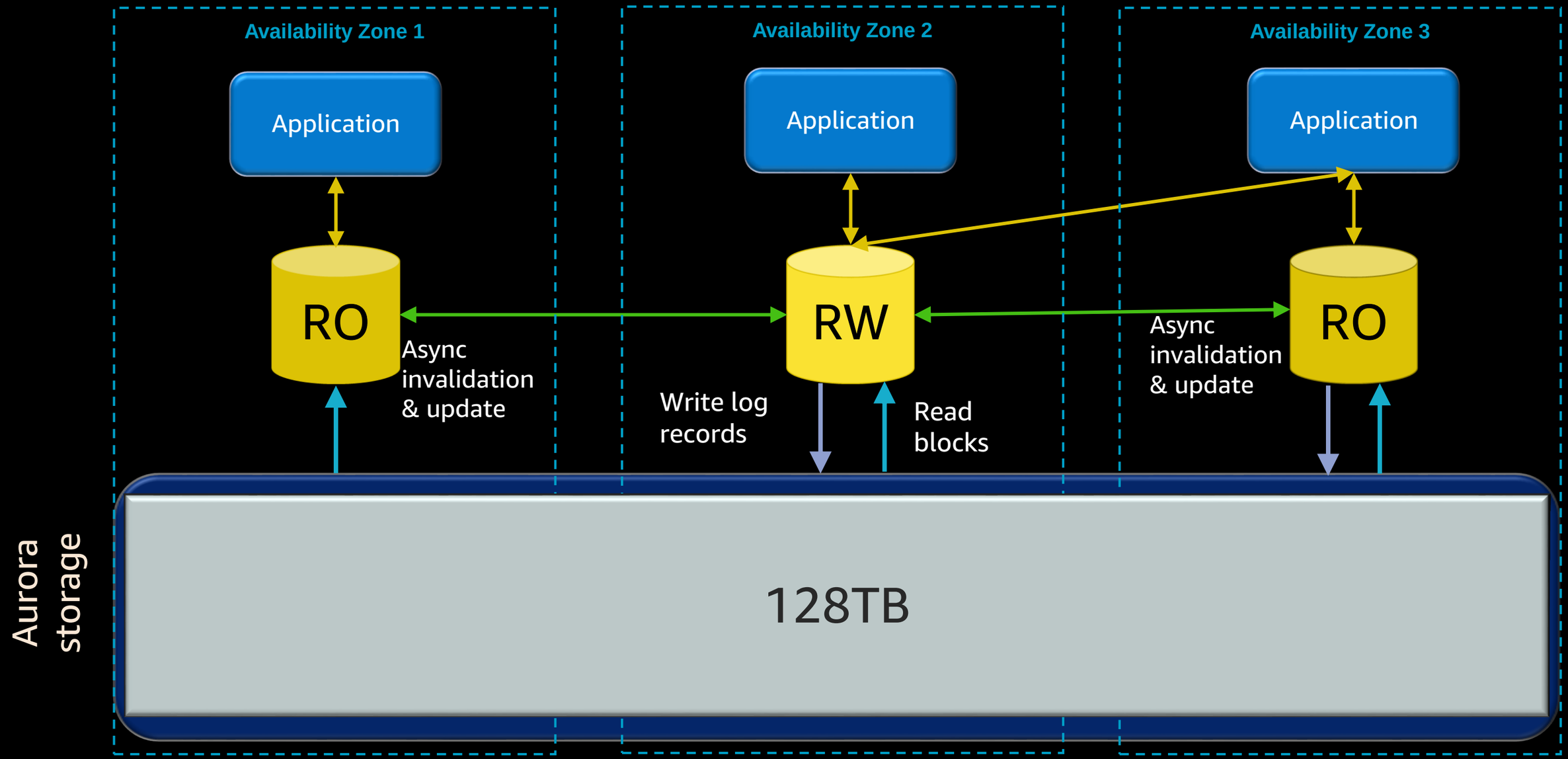
Scaling



AWS re:Invent

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

Aurora storage scaling



Partitions in PostgreSQL 12

```
CREATE TABLE door_knocks (
    id int not null,
    knock_date timestamp  not null,
    city varchar(255) not null,
    zipcode char(5) not null
) PARTITION BY RANGE (knock_date);

CREATE TABLE door_knock_date_y2020103001 PARTITION OF door_knocks
   FOR VALUES FROM ('2020-10-30 01:00:00') TO ('2020-10-30 02:00:00');

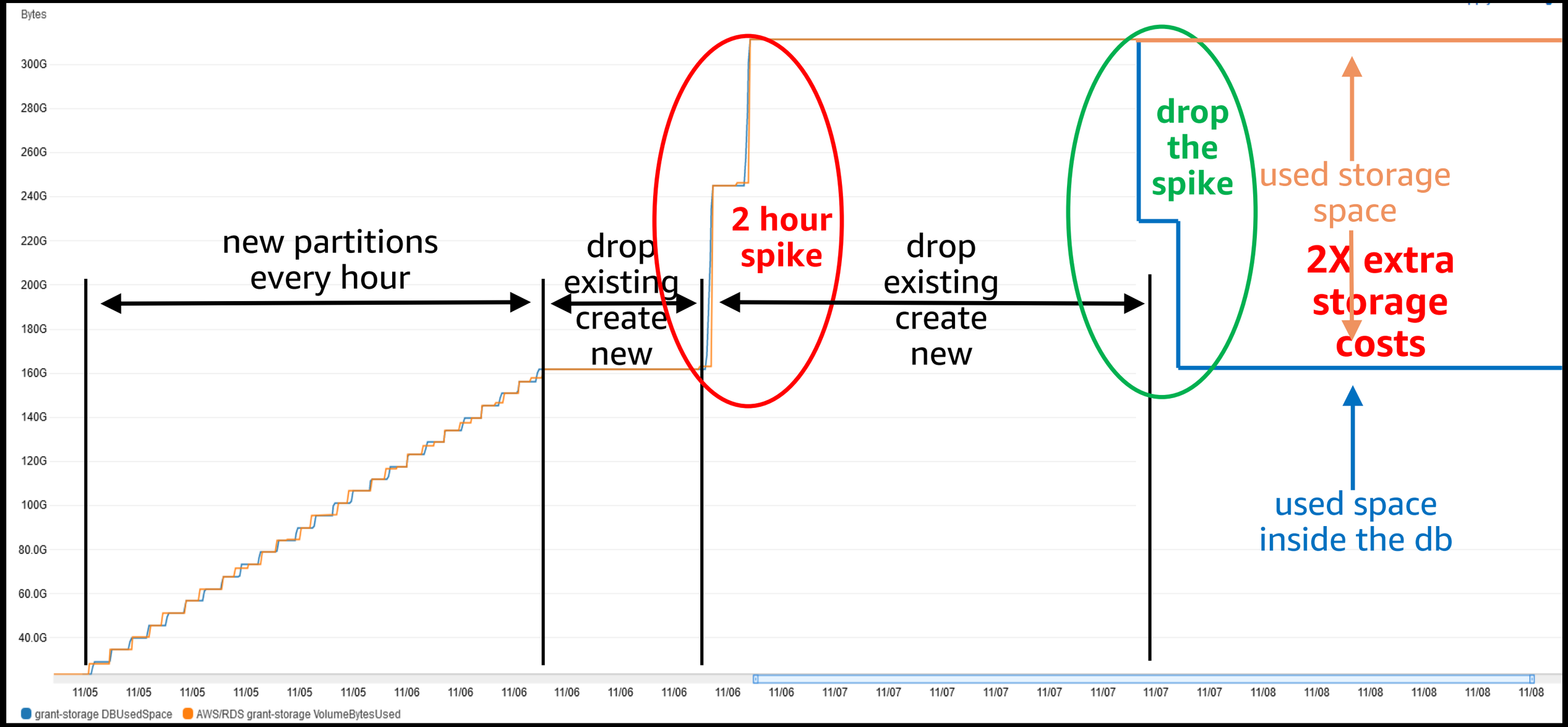
CREATE TABLE door_knock_date_y2020103002 PARTITION OF door_knocks
   FOR VALUES FROM ('2020-10-30 02:00:00') TO ('2020-10-30 03:00:00');

CREATE TABLE door_knock_date_y2020103003 PARTITION OF door_knocks
   FOR VALUES FROM ('2020-10-30 03:00:00') TO ('2020-10-30 04:00:00');
```

- FK support
- list, range, hash
- Multi-level
- Attach without lock

```
 Partitioned table "public.door_knocks"
   Column   |            Type             | Collation | Nullable | Default | Storage  | Stats target | Description
------------+-----------------------------+-----------+----------+---------+----------+--------------+-------------
 id         | integer                     |           | not null |         | plain    |              |
 knock_date | timestamp without time zone |           | not null |         | plain    |              |
 city       | character varying(255)      |           | not null |         | extended |              |
 zipcode    | character(5)                |           | not null |         | extended |              |
Partition key: RANGE (knock_date)
Partitions: door_knock_date_y2020103001 FOR VALUES FROM ('2020-10-30 01:00:00') TO ('2020-10-30 02:00:00'),
            door_knock_date_y2020103002 FOR VALUES FROM ('2020-10-30 02:00:00') TO ('2020-10-30 03:00:00'),
            door_knock_date_y2020103003 FOR VALUES FROM ('2020-10-30 03:00:00') TO ('2020-10-30 04:00:00')
```

Storage Management – dynamic resizing



Replication / Data flow



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

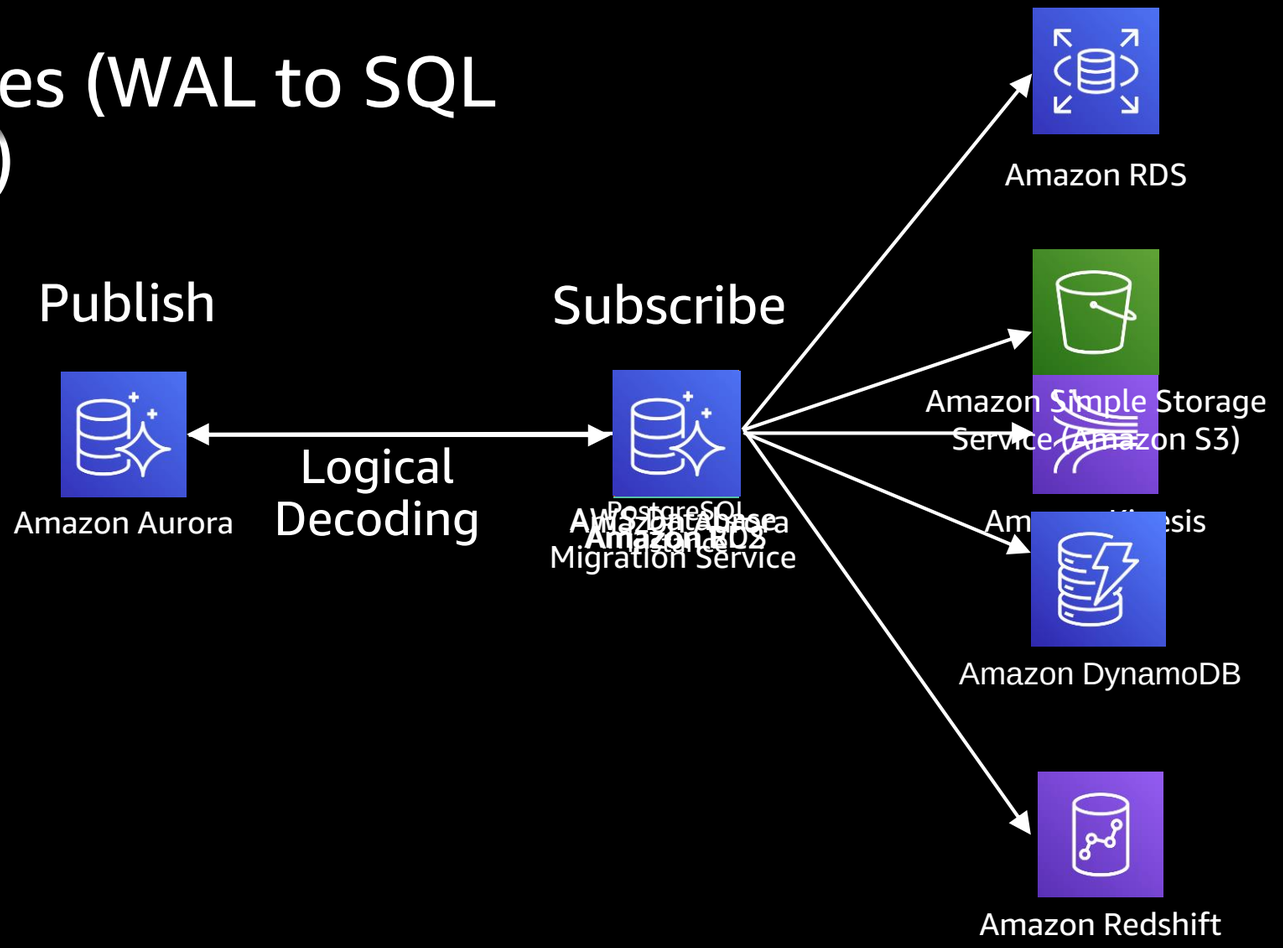
Logical replication support

Converts physical changes (WAL to SQL statements – i.e., logical)

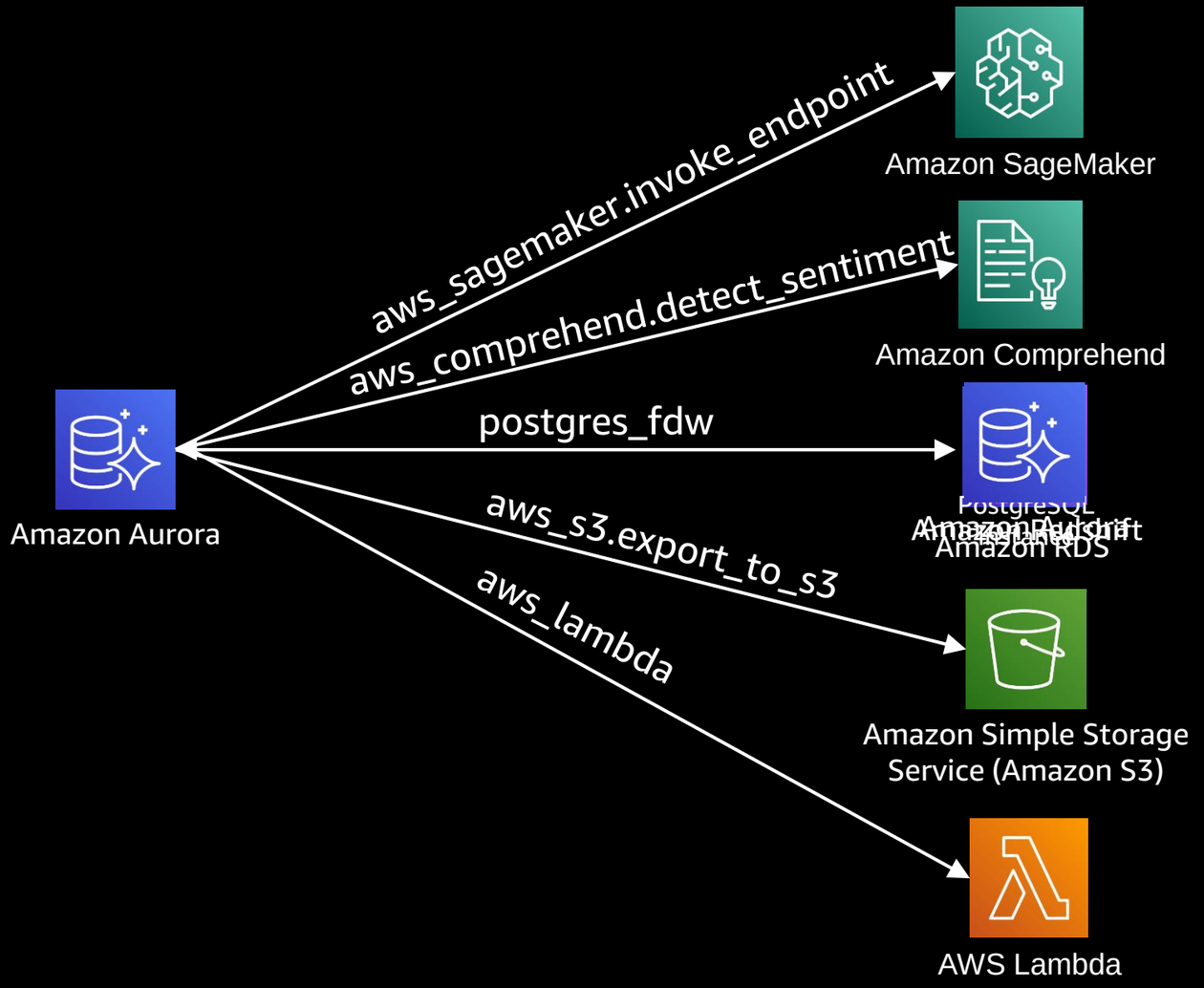
- Logical decoding plugin (test_decoder, decoder_raw, wal2jason)

- V10/11: Publish/subscribe to another PostgreSQL instance

- pglogical



Data flow – PostgreSQL extensions



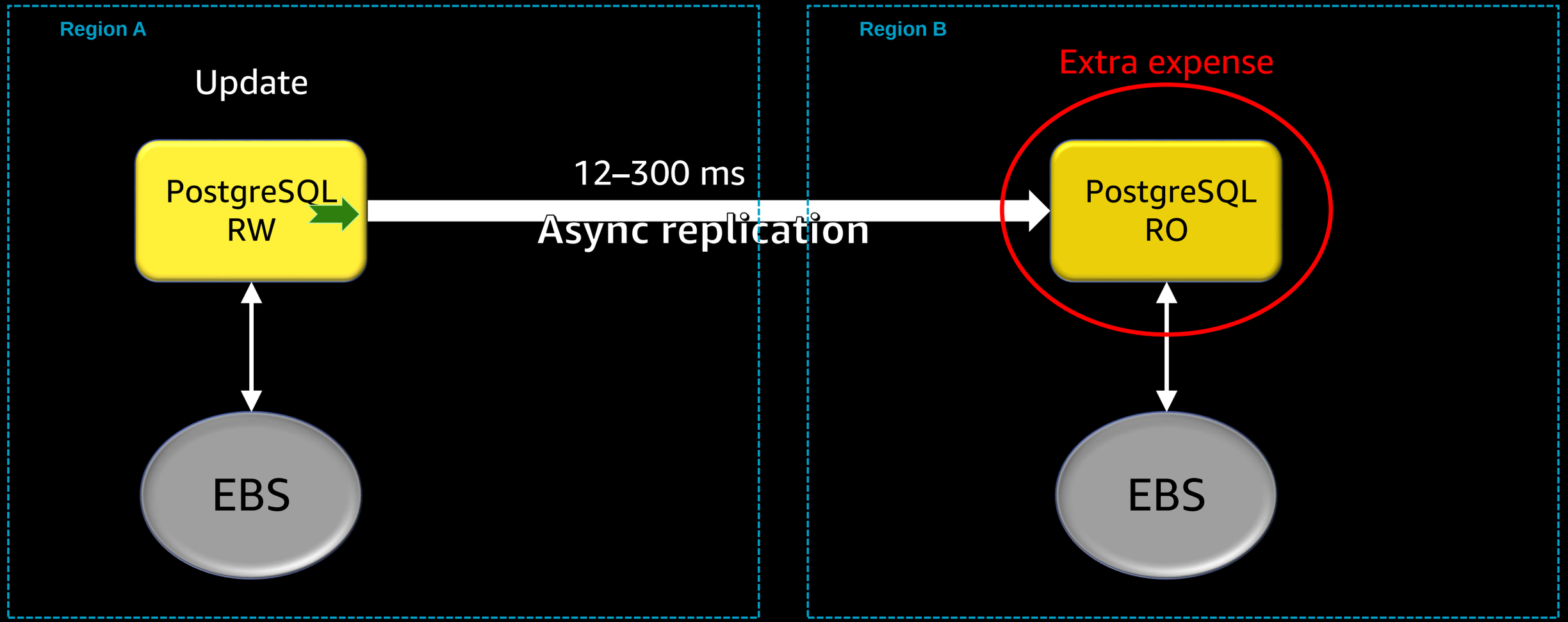
Global DB –
Cross region replication



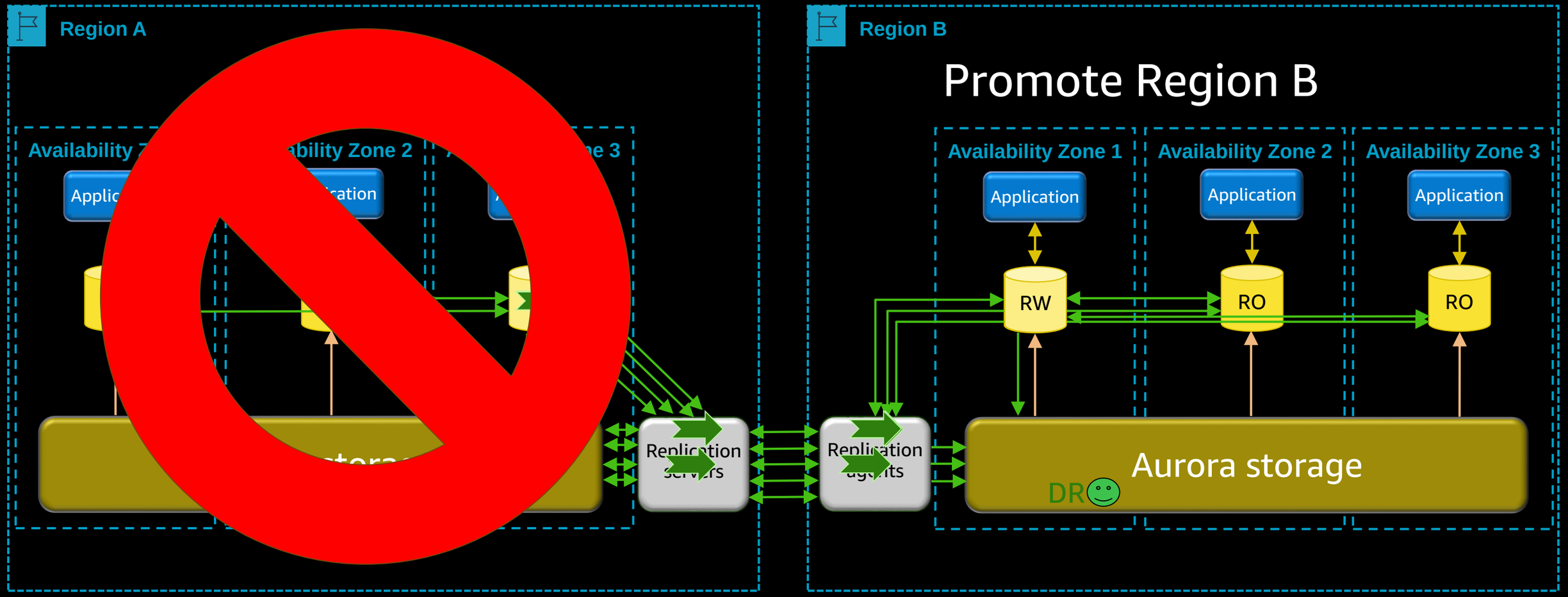
AWS re:Invent

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

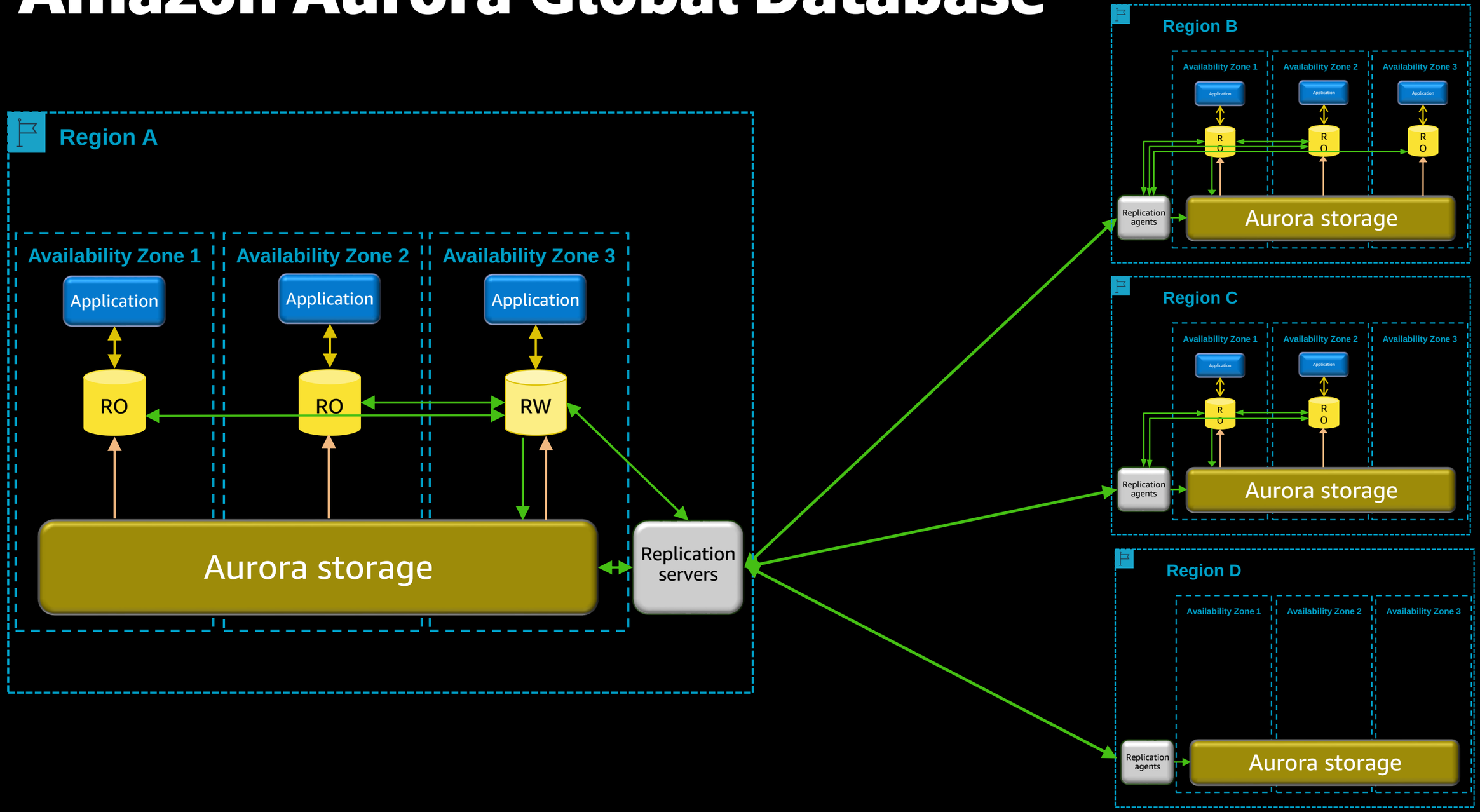
Cross region replicas: PostgreSQL



Amazon Aurora Global Database



Amazon Aurora Global Database

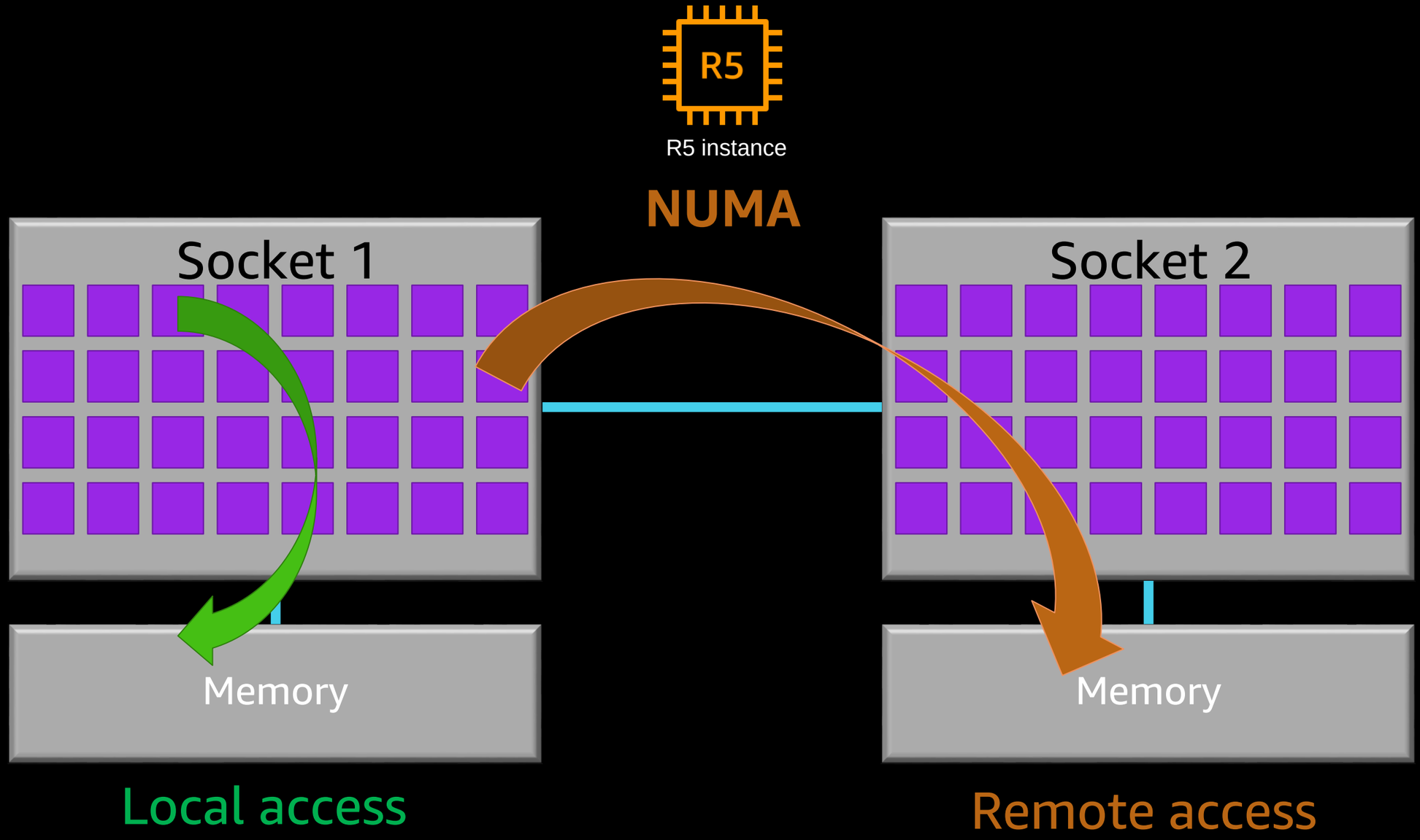


Performance



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

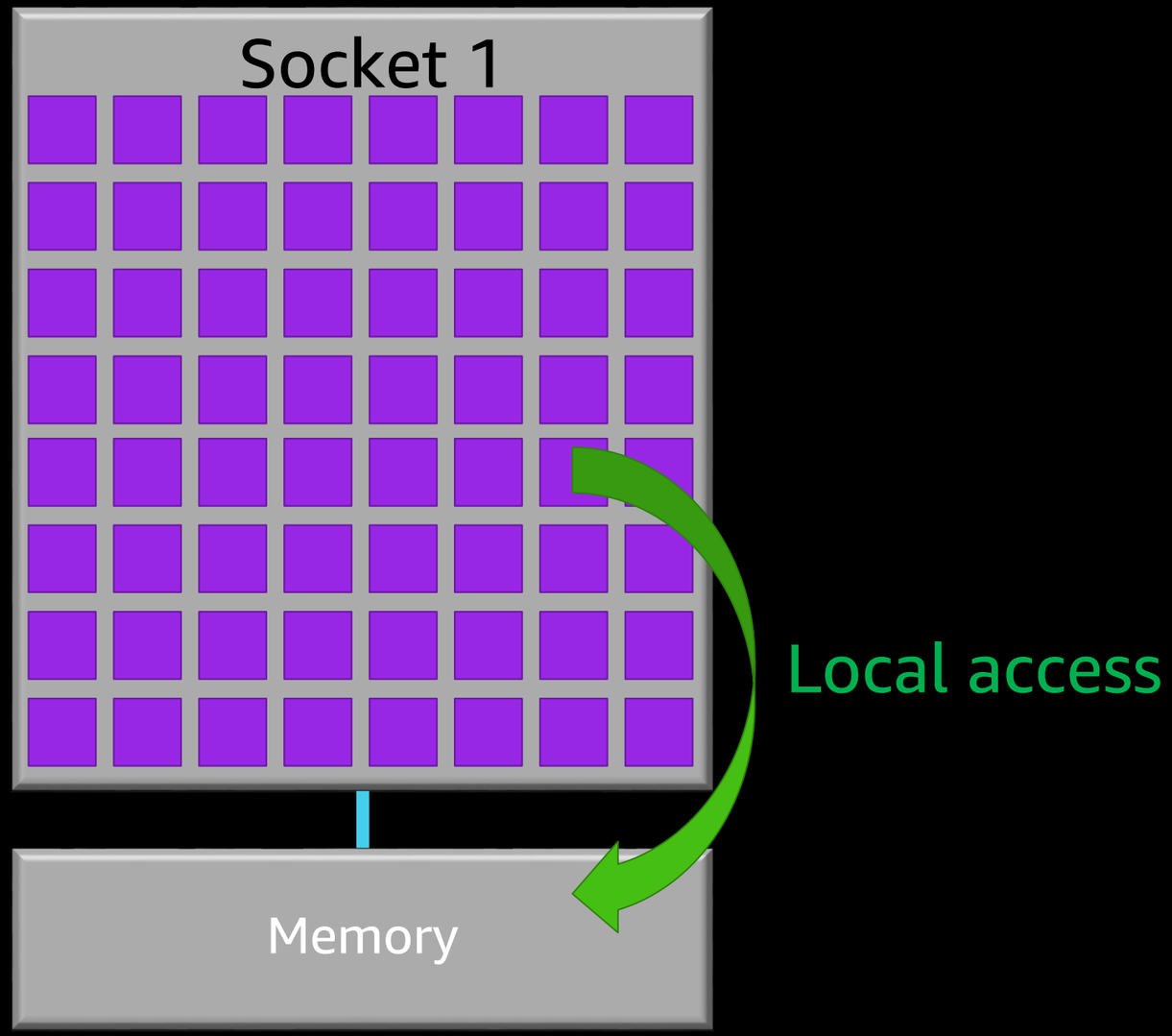
Non-uniform memory access



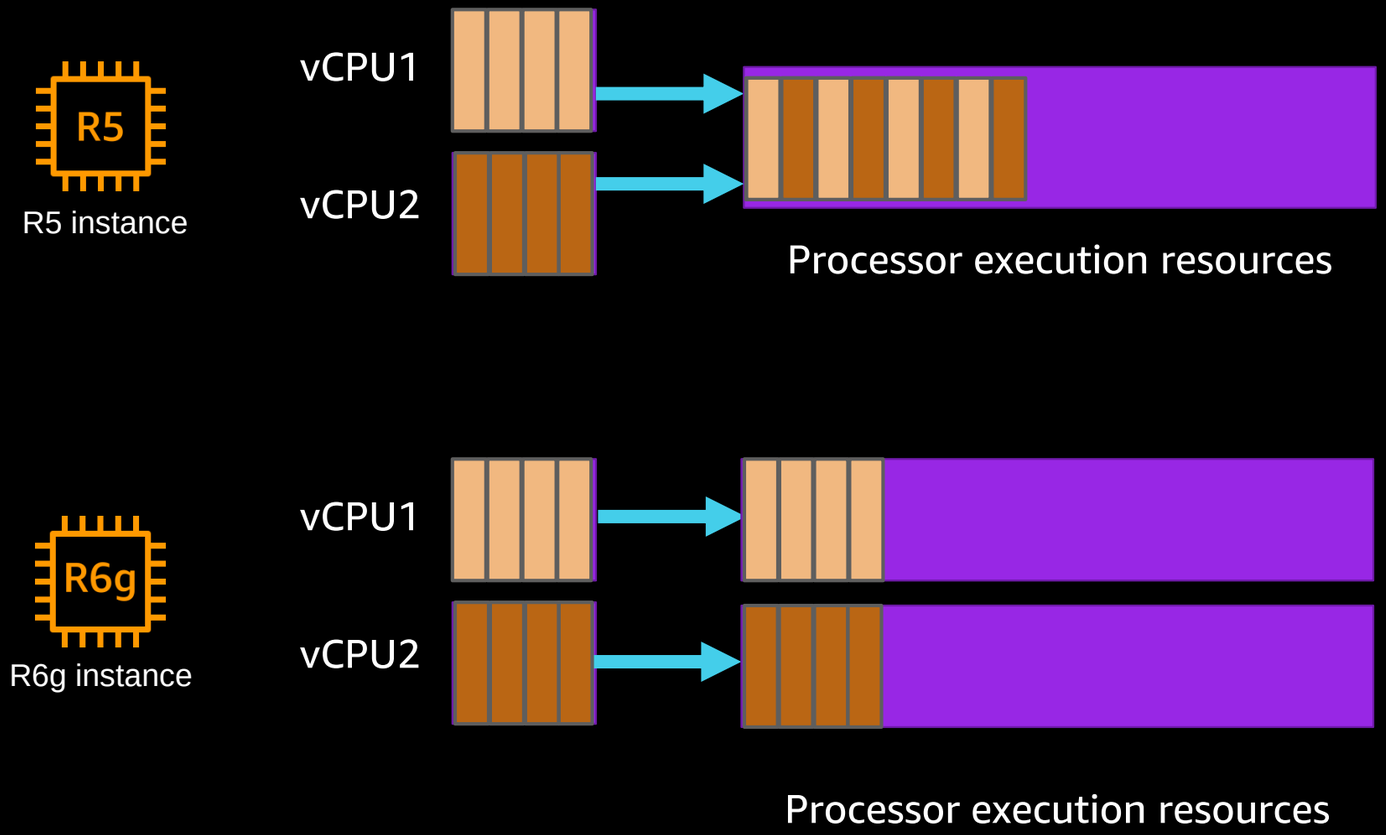
Graviton2 – Uniformity



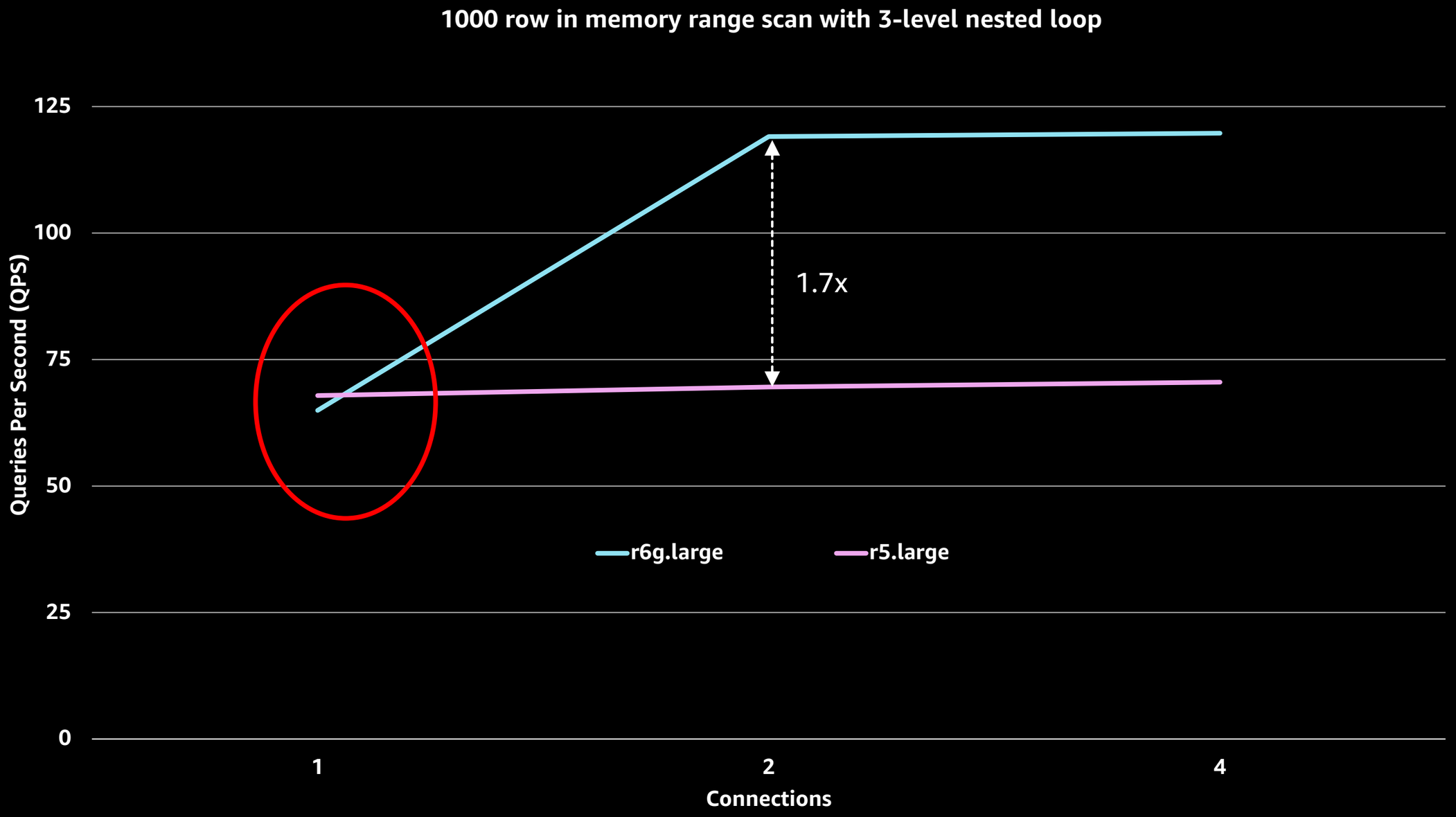
R6g instance



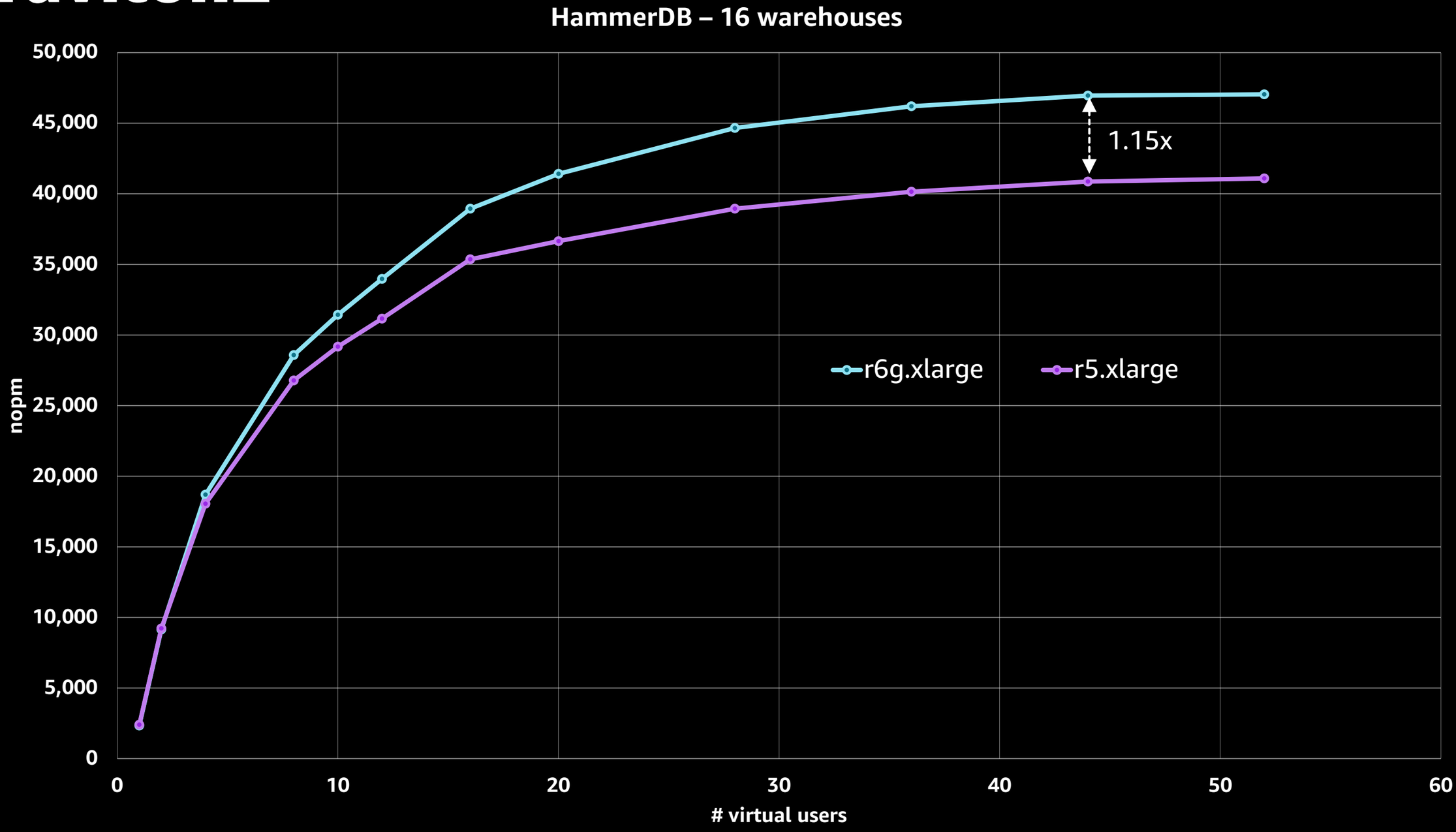
Graviton2 – vCPU



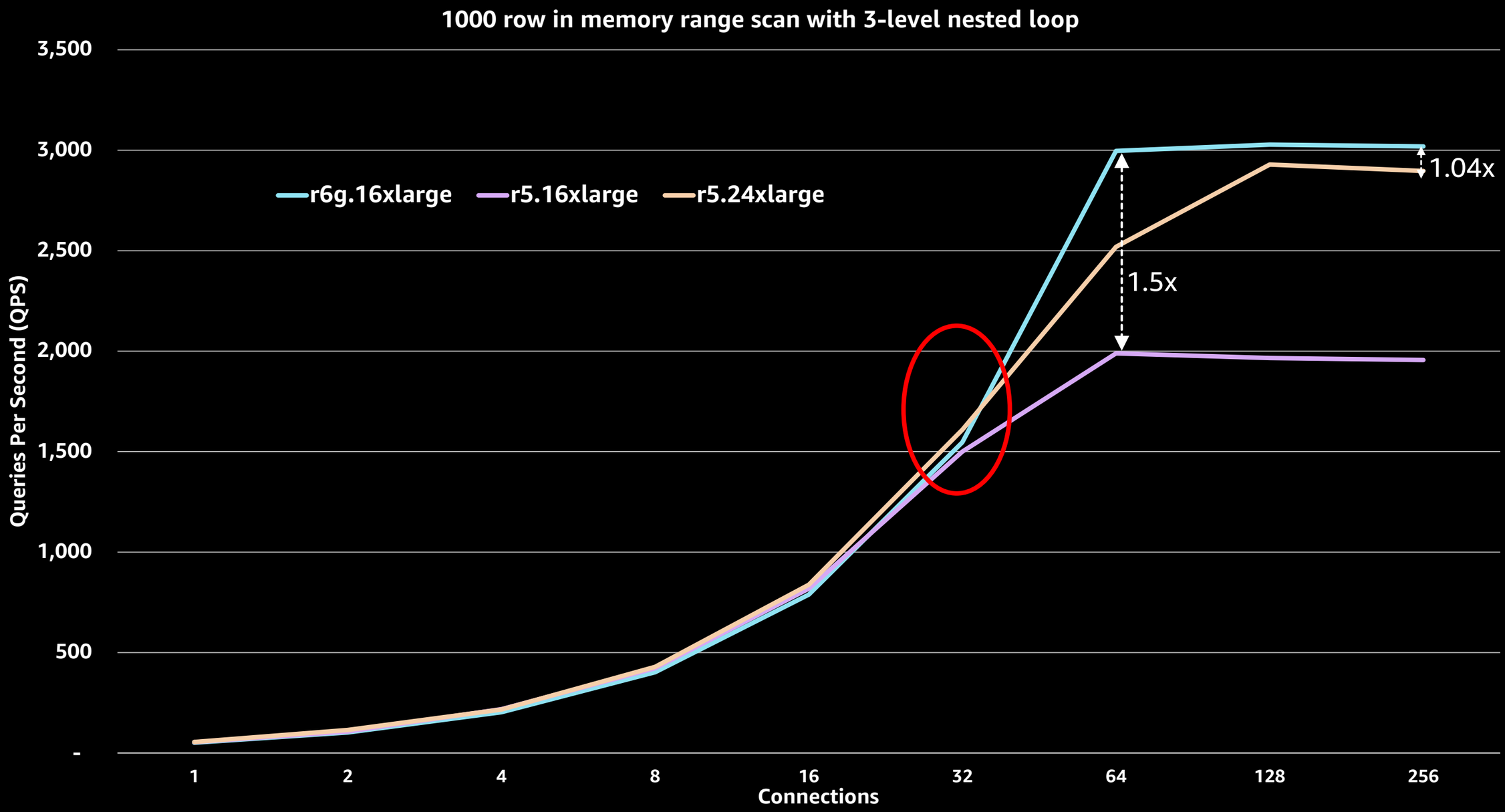
Graviton2


1000 row in memory range scan with 3-level nested loop
125
100
75
50
25
0
Queries Per Second (QPS)
1.7x
r6g.large
r5.large
1
2
4
Connections

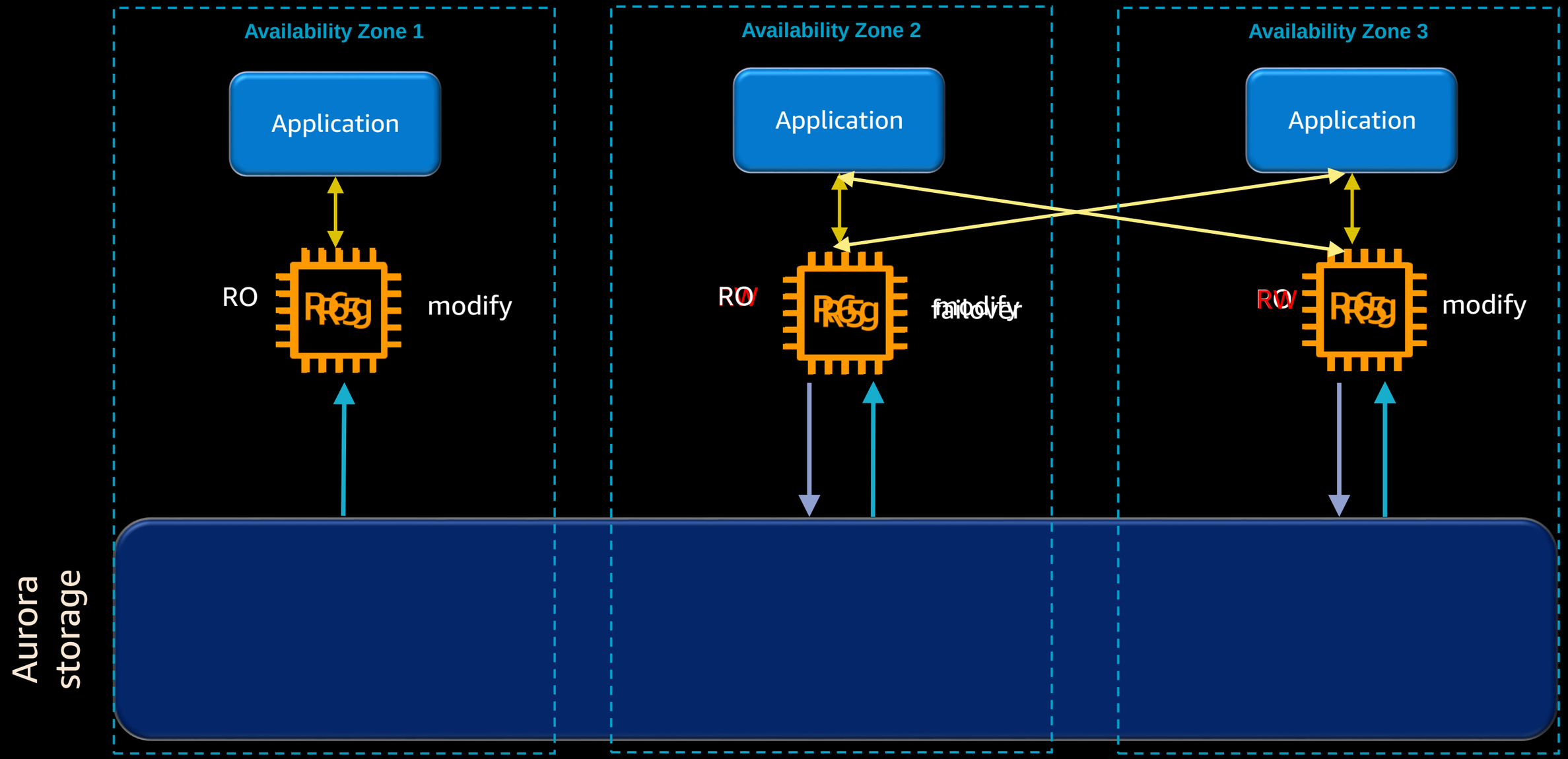
Graviton2


HammerDB – 16 warehouses
50,000
45,000
40,000
35,000
30,000
25,000
20,000
15,000
10,000
5,000
0
nopm
0
10
20
30
40
50
60
virtual users
r6g.xlarge
r5.xlarge
1.15x

Graviton2


1000 row in memory range scan with 3-level nested loop
3,500
3,000
2,500
2,000
1,500
1,000
500
-
Queries Per Second (QPS)
r6g.16xlarge
r5.16xlarge
r5.24xlarge
1.04x
1.5x
1
2
4
8
16
32
64
128
256
Connections

Aurora – Heterogeneous clusters



Migration



AWS re:Invent

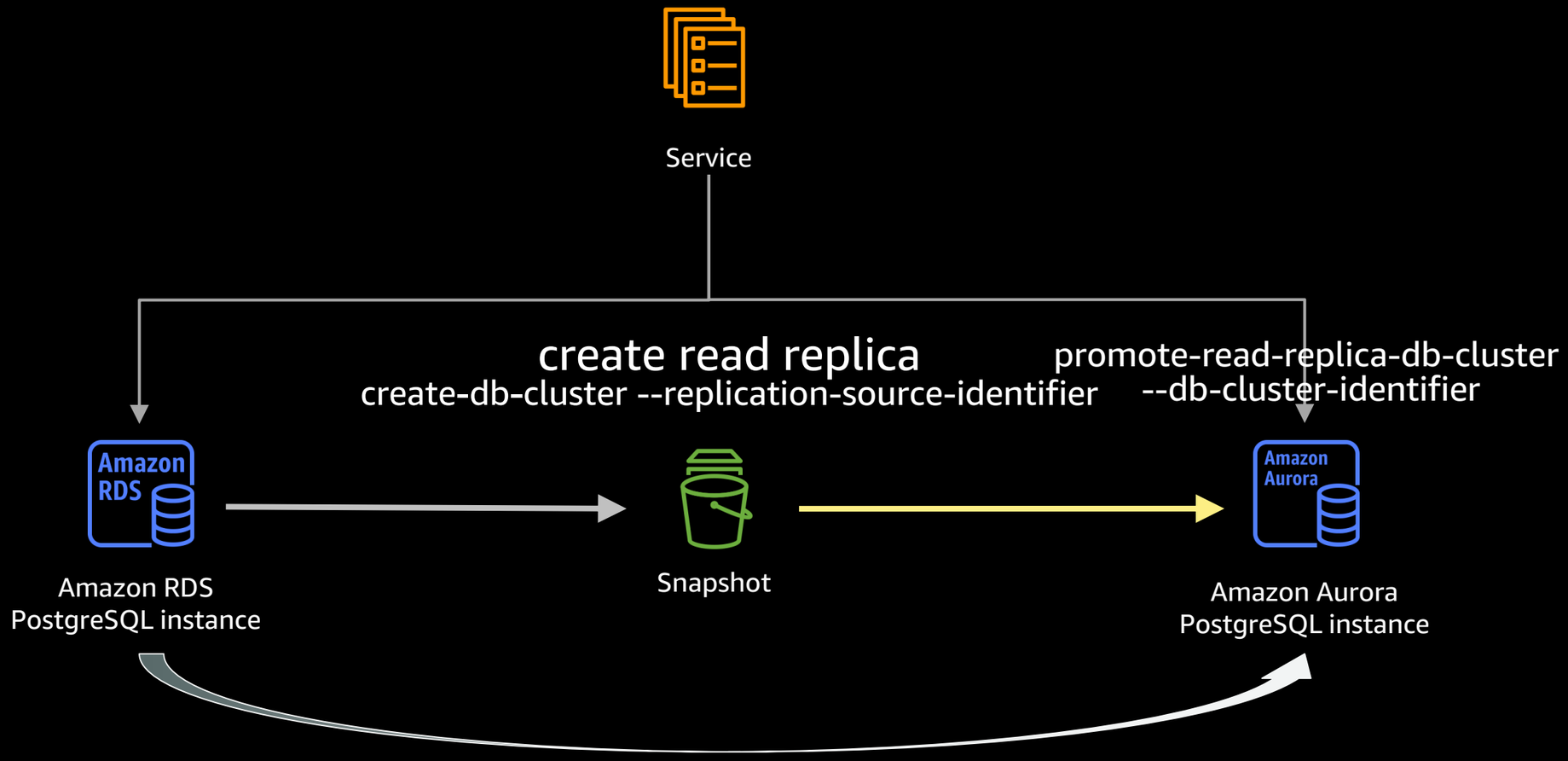
© 2020, Amazon Web Services, Inc. or its affiliates. All rights reserved.

Migration to Aurora

Methods

- AWS Database Migration Service (AWS DMS)
- PostgreSQL: pg_dump/pg_restore
- PostgreSQL Logical Replication (including pglogical)
- Amazon RDS PostgreSQL: Snapshot import
- Amazon RDS PostgreSQL: Read replica

Migration: Read replica



Catchup via PostgreSQL asynchronous replication

Related sessions

DAT302

Deep dive on Amazon Aurora with MySQL compatibility

DAT401

Amazon Aurora storage demystified

DAT404

Deep dive on global databases for Amazon Aurora

AWS re:Invent

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

Thank you!



AWS re:Invent

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



Please complete the session survey

AWS re:Invent

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