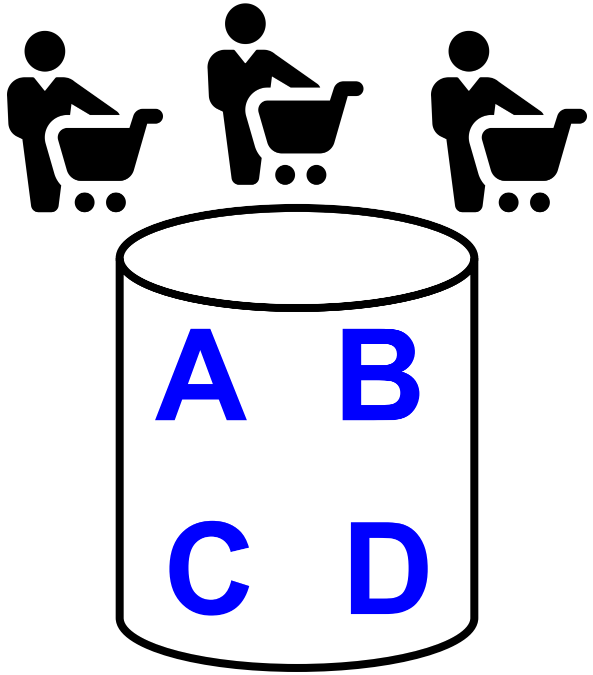
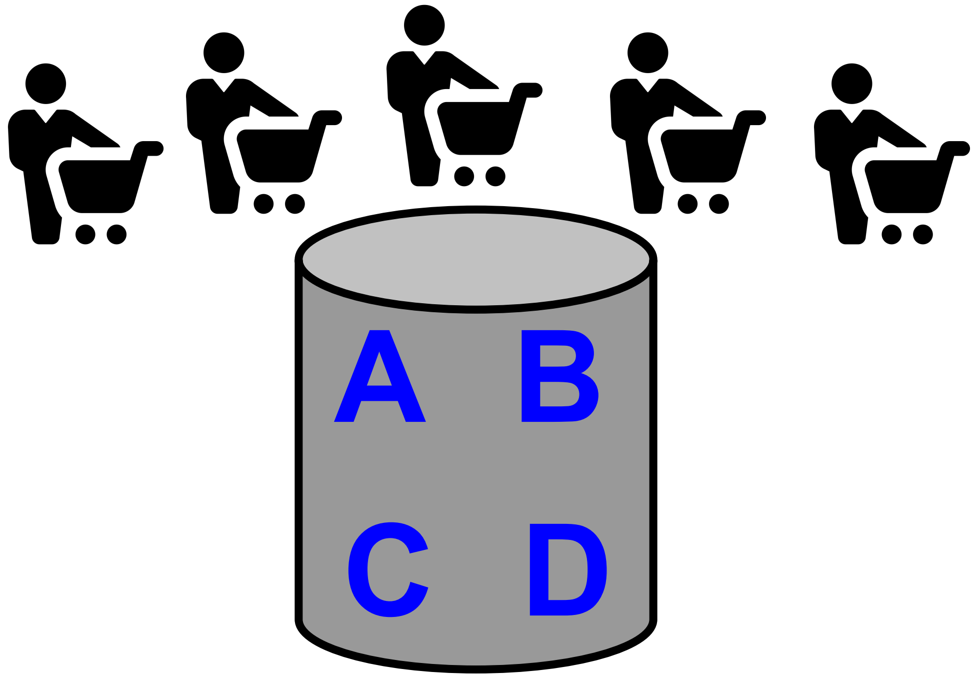
Dynamic Replication & Partitioning in Dynamically Mastered DBs

Brad Glasbergen
Michael Abebe
CS 848 (April 2019)

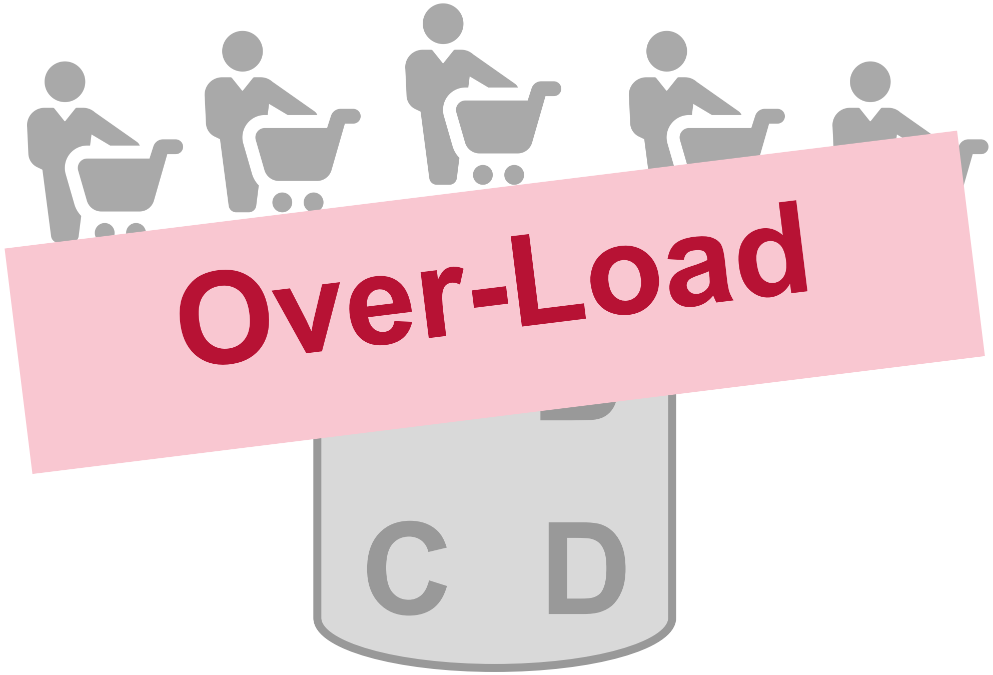
Single Database



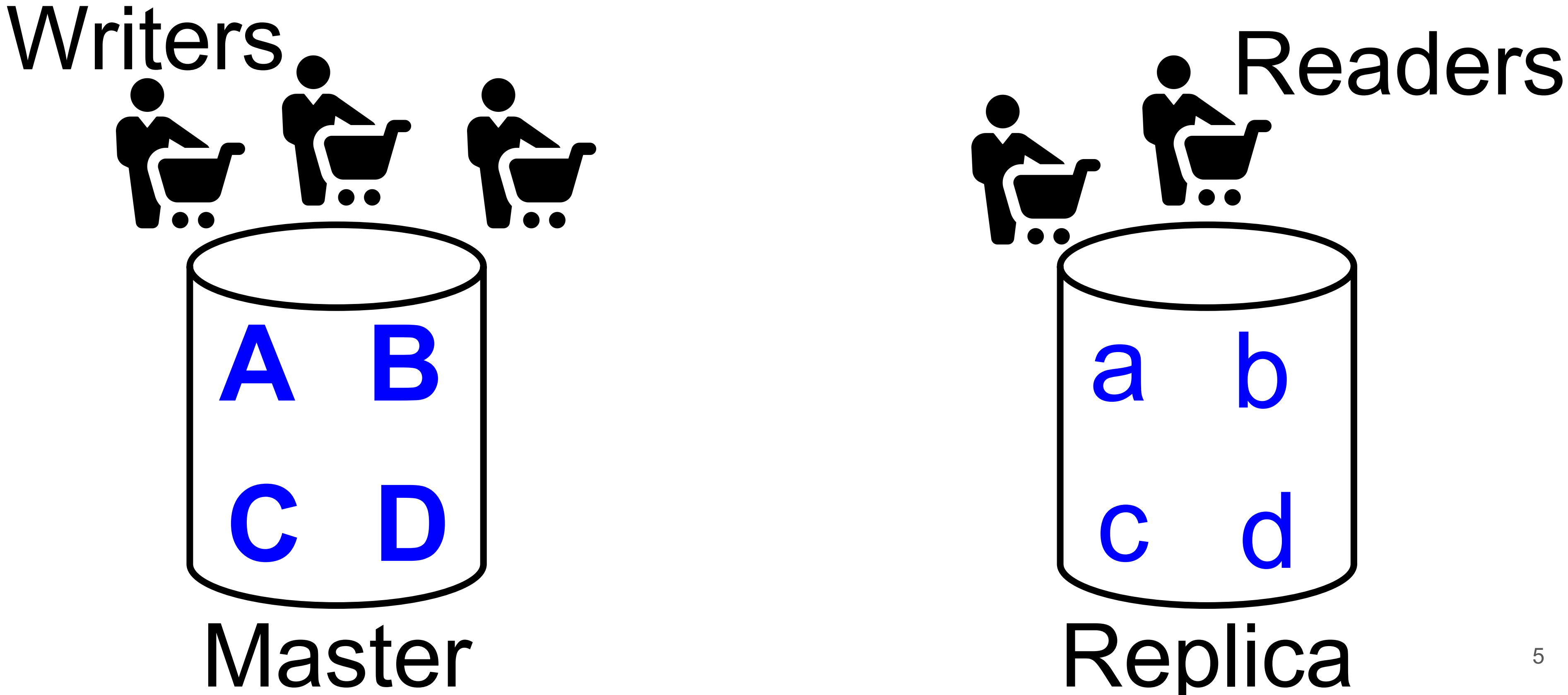
Single Database



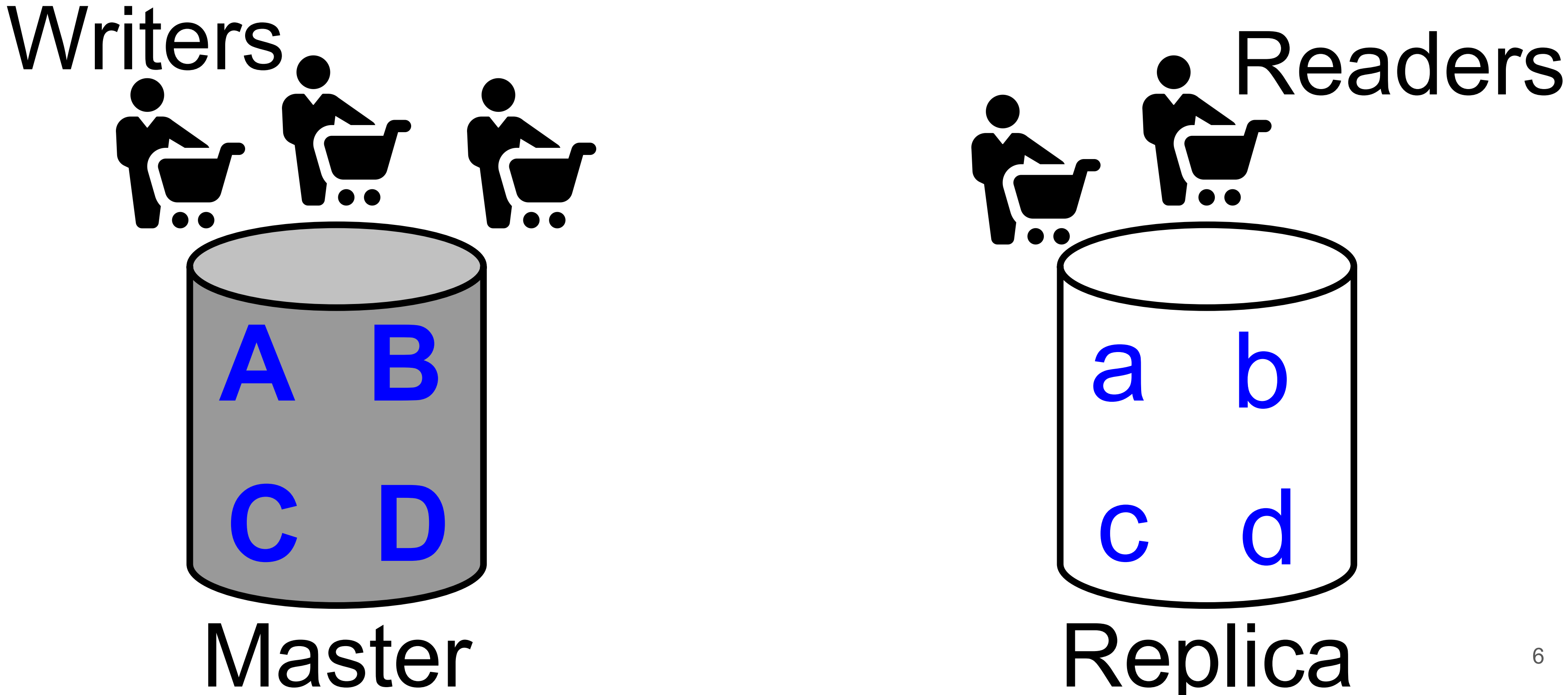
Single Database



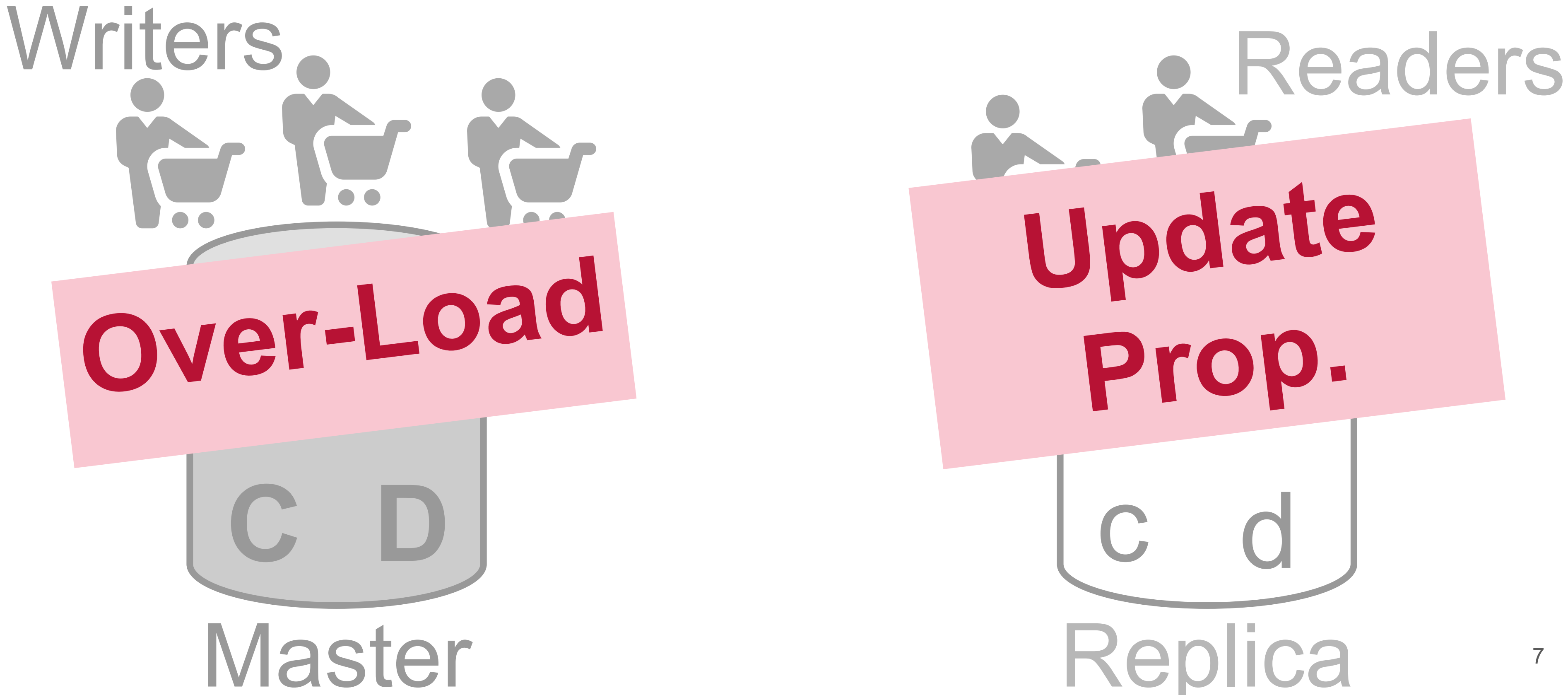
Replicated Databases



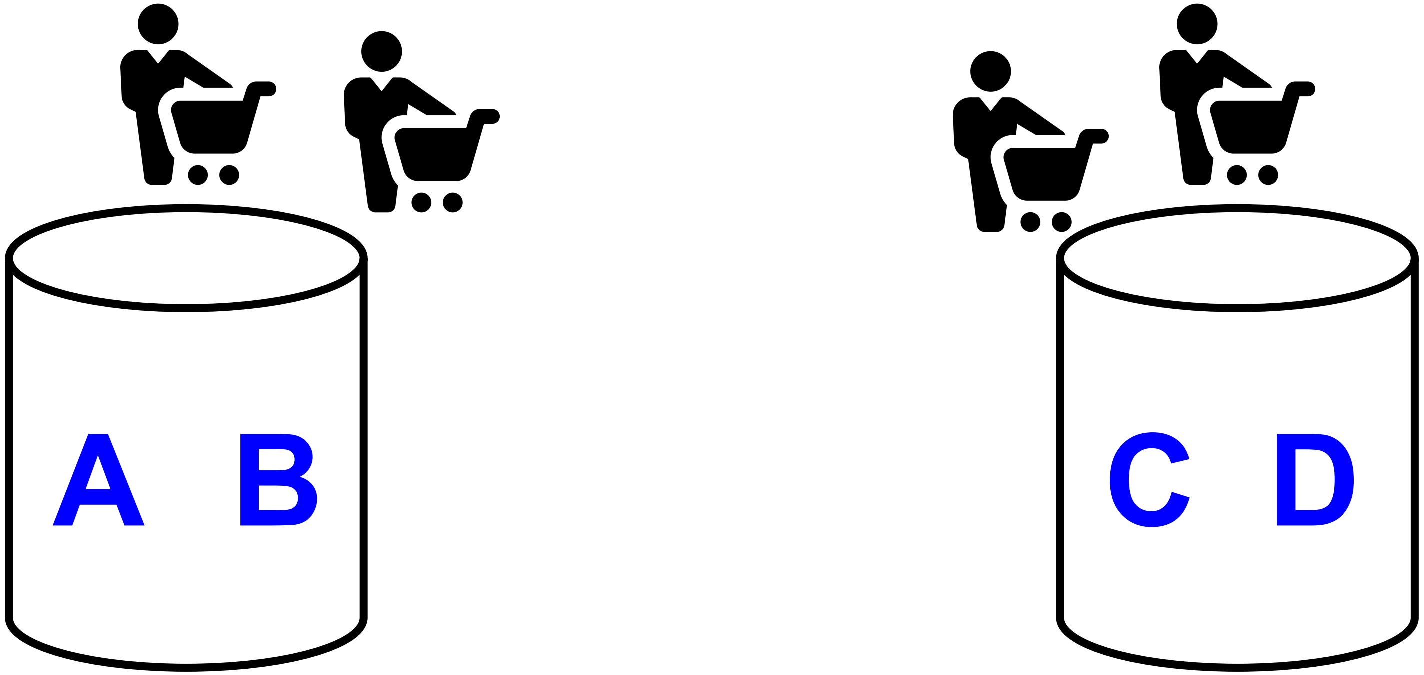
Replicated Databases



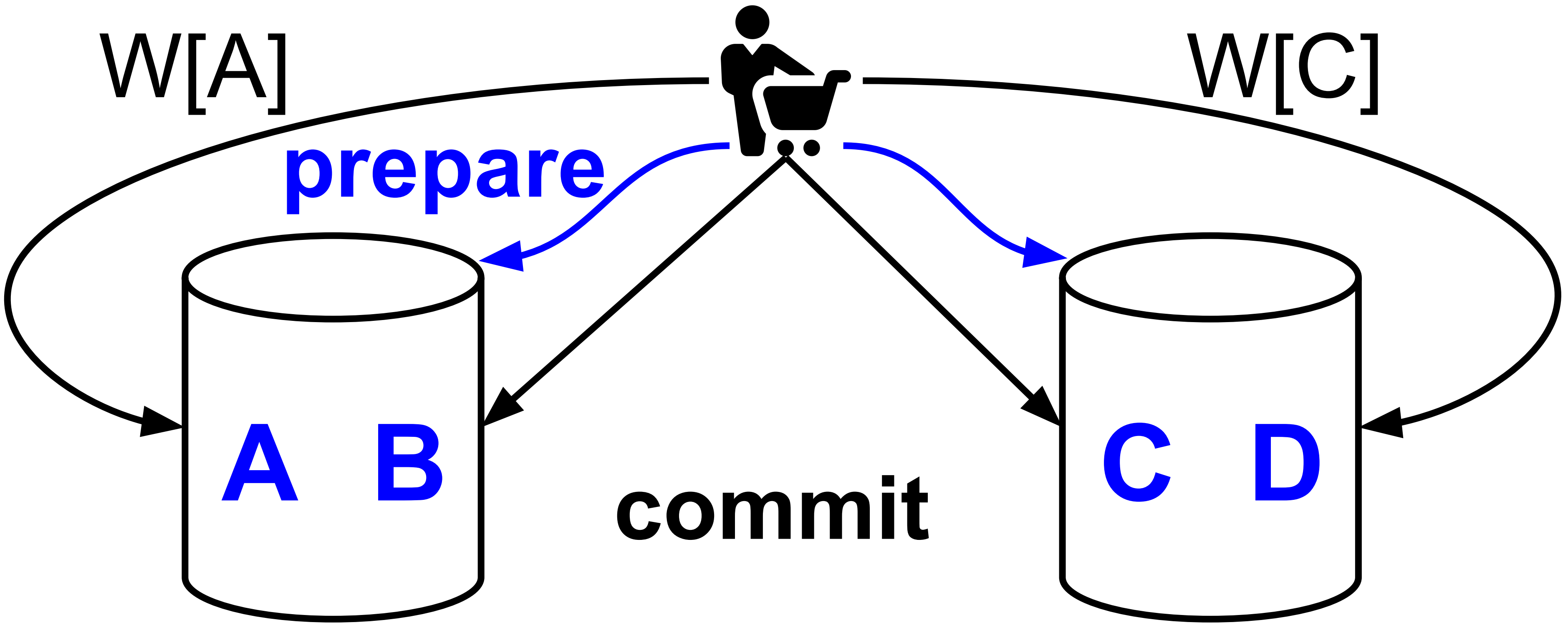
Replicated Databases



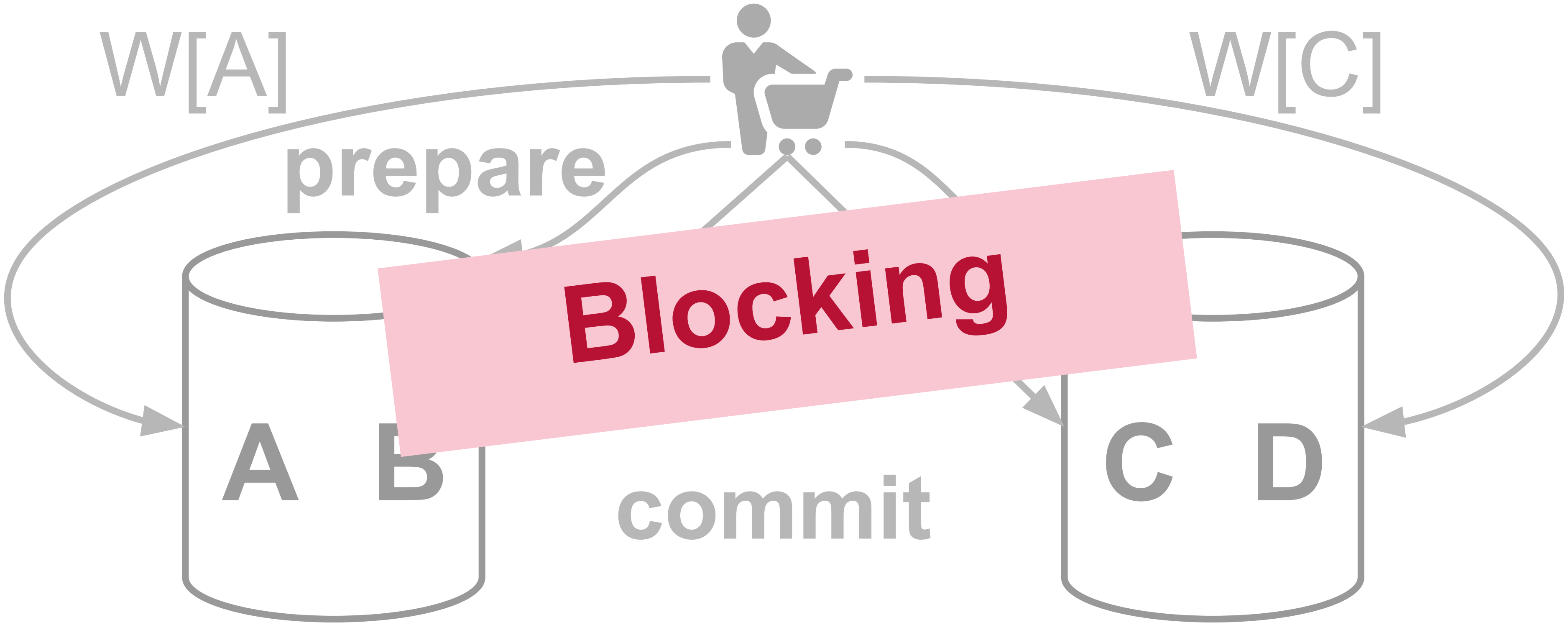
Partitioned Databases



Partitioned Databases

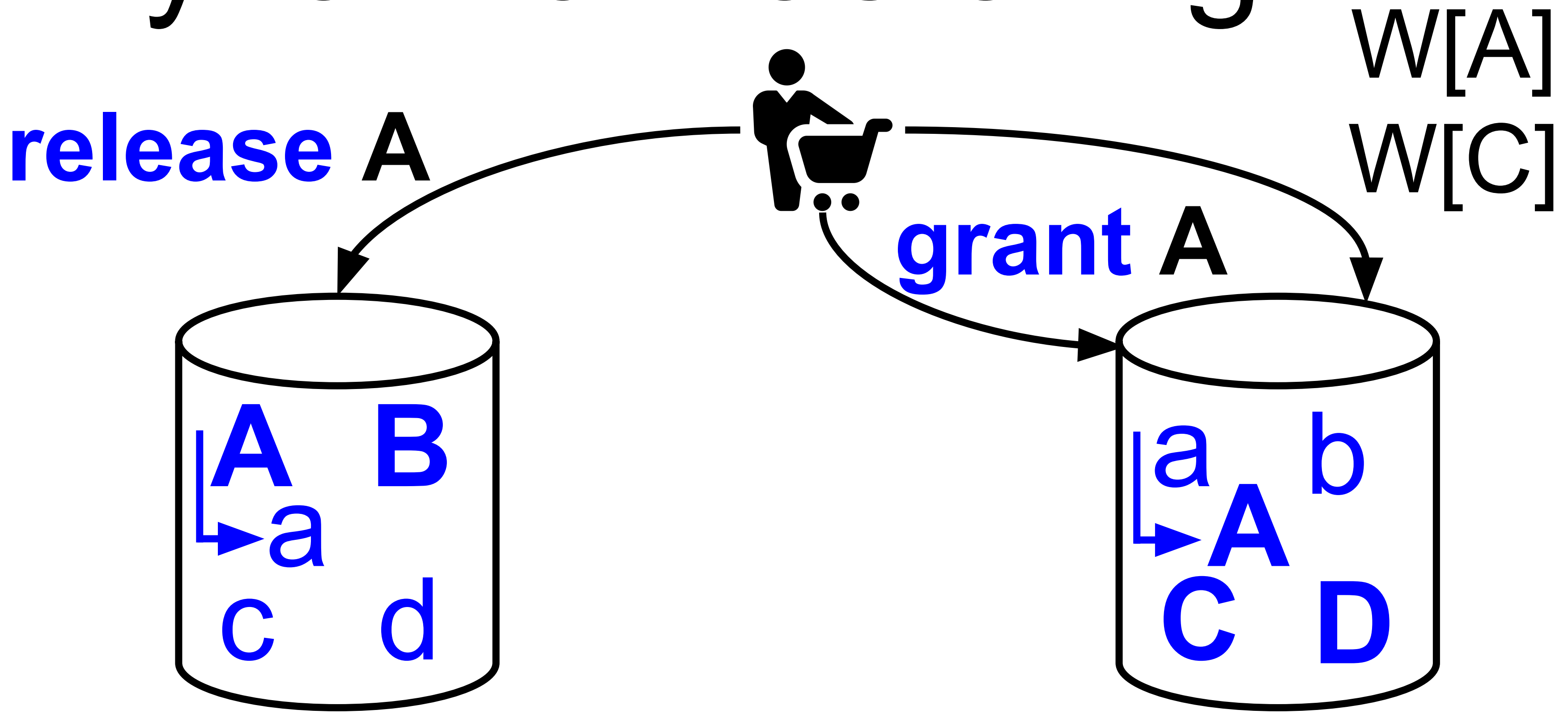


Partitioned Databases

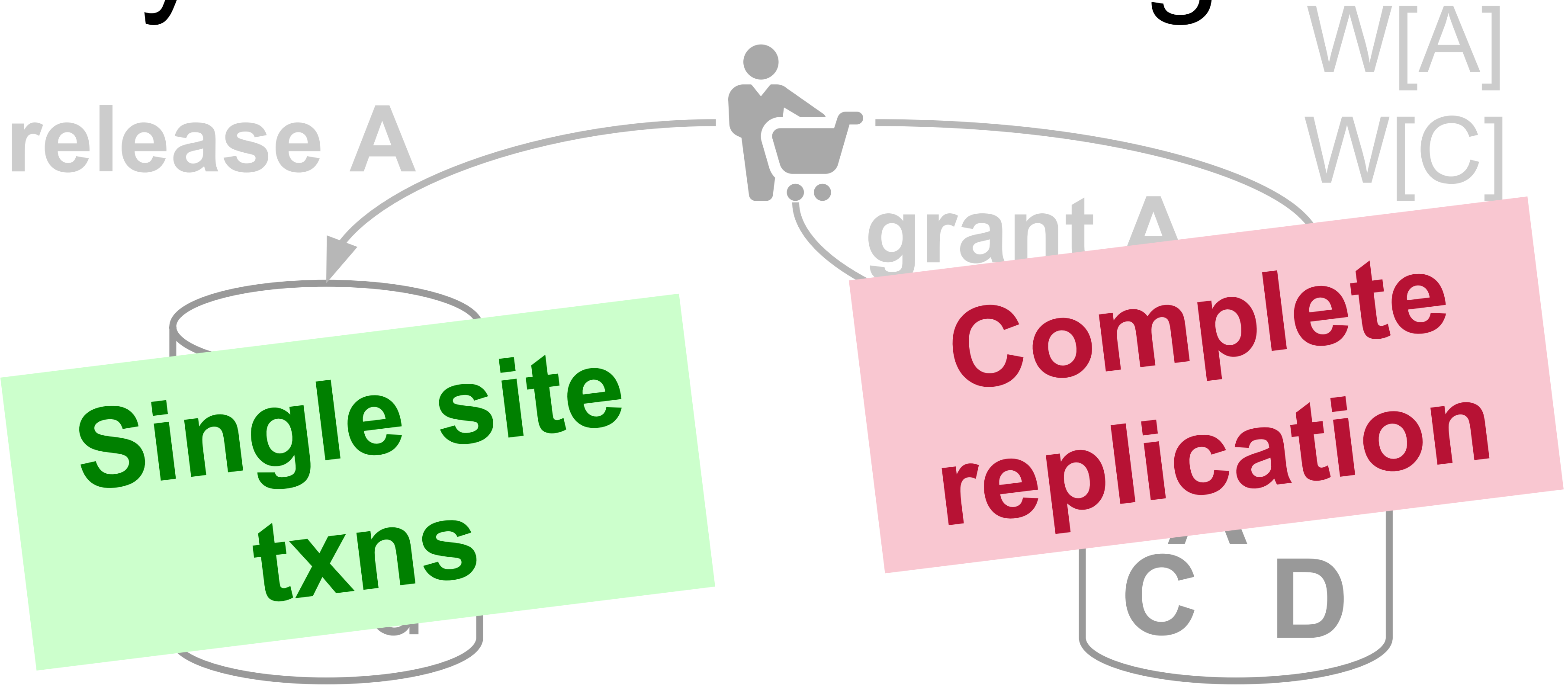


Dynamically Mastered DBs

Dynamic Mastering



Dynamic Mastering

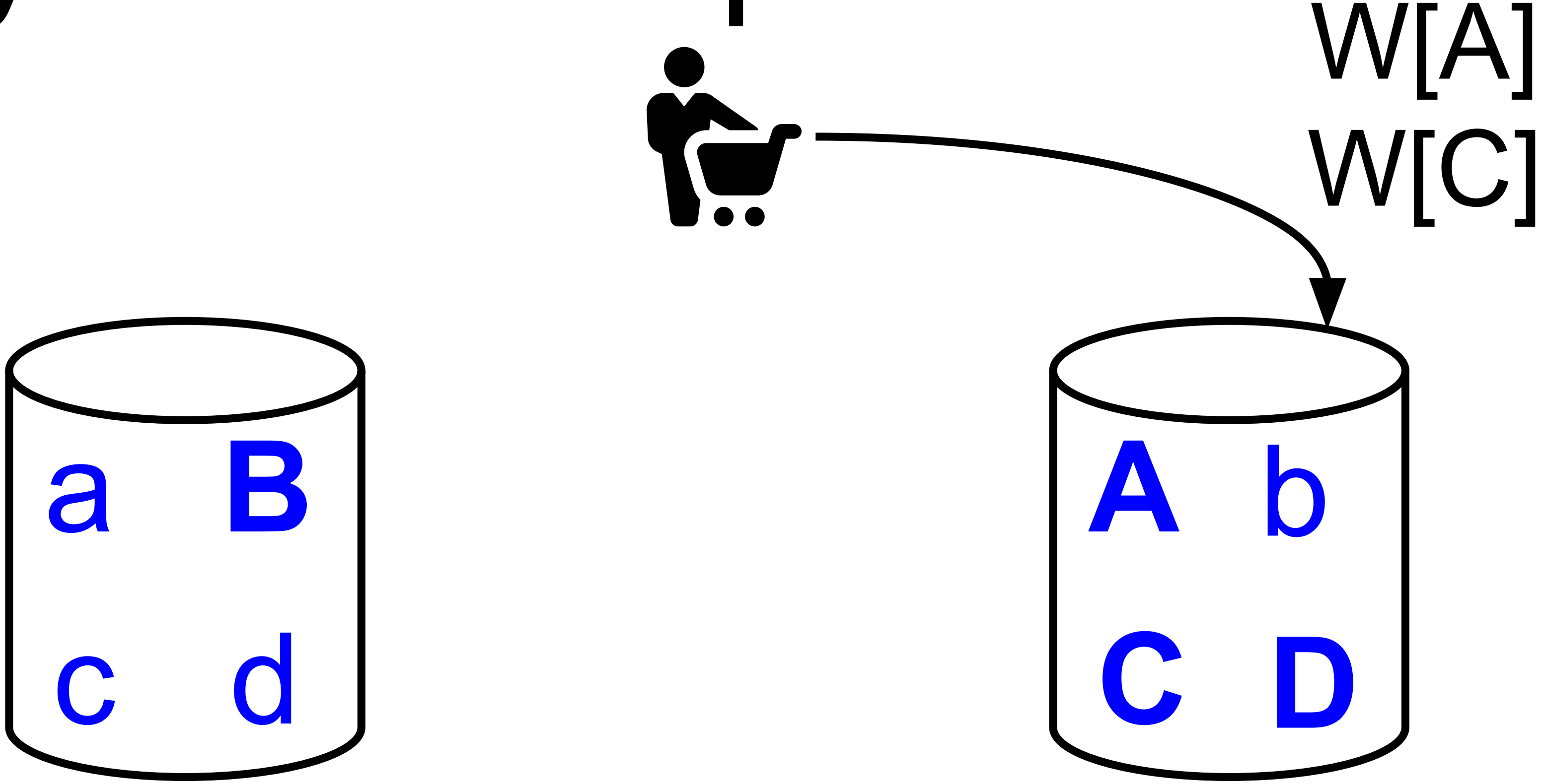


Single site txns

Complete replication

Dynamic Replication & Partitioning

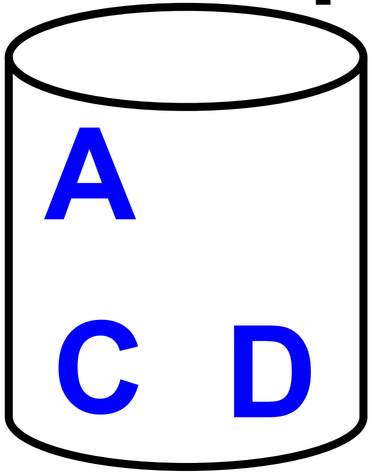
Dynamic Replication



Dynamic Replication

add replica

remove replica



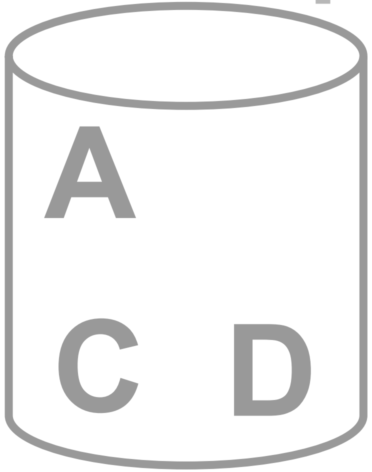
Dynamic Replication

Store more data

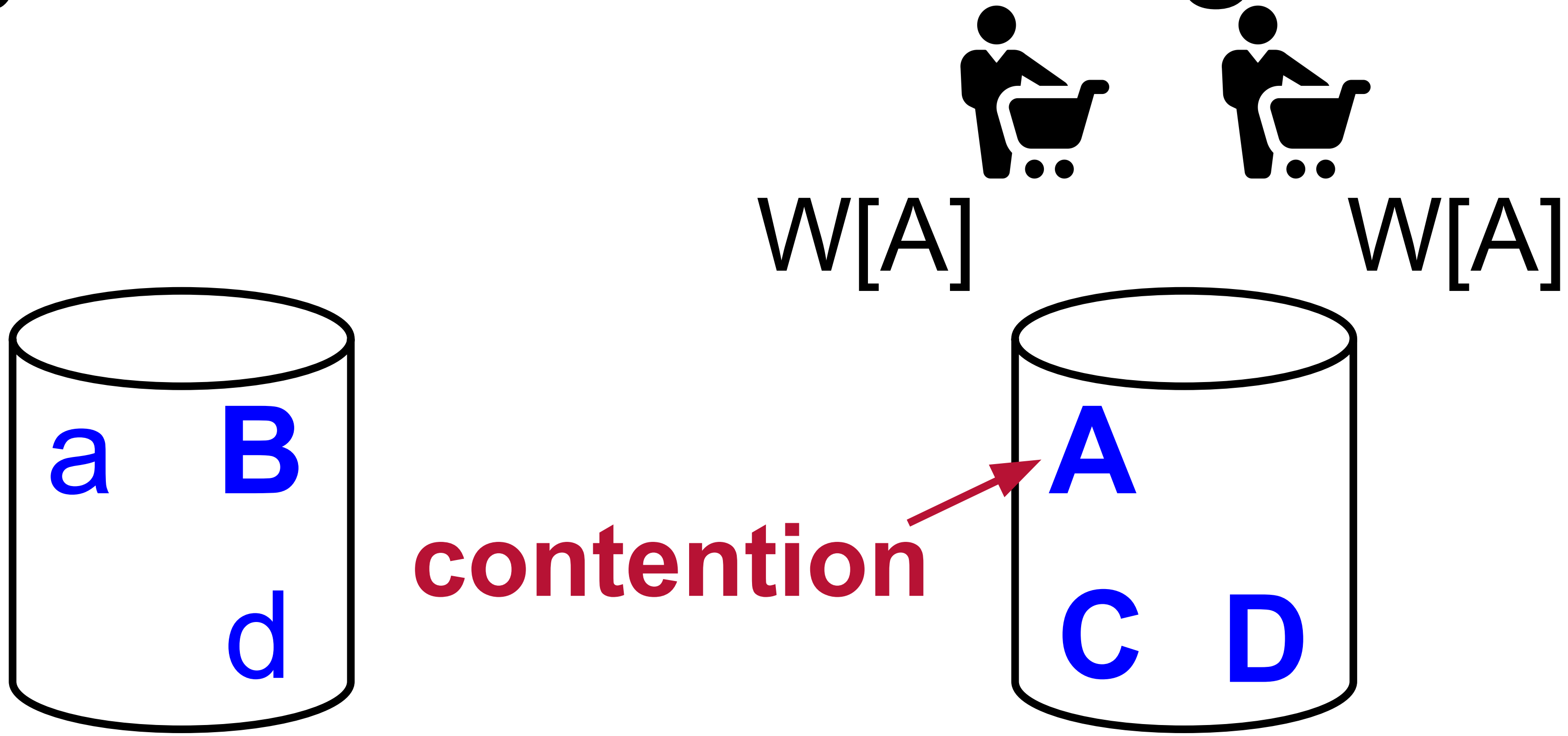
Distribute load

add replica

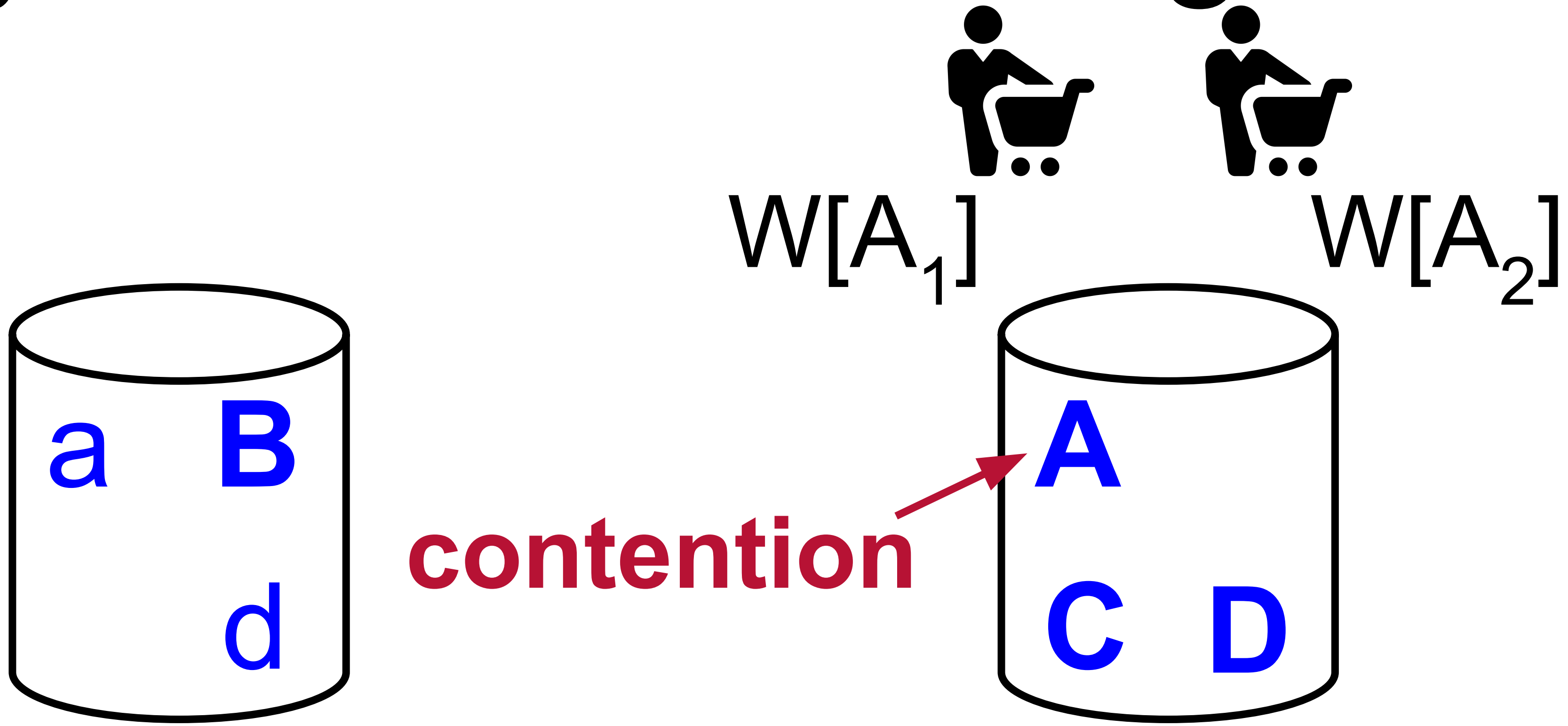
remove replica



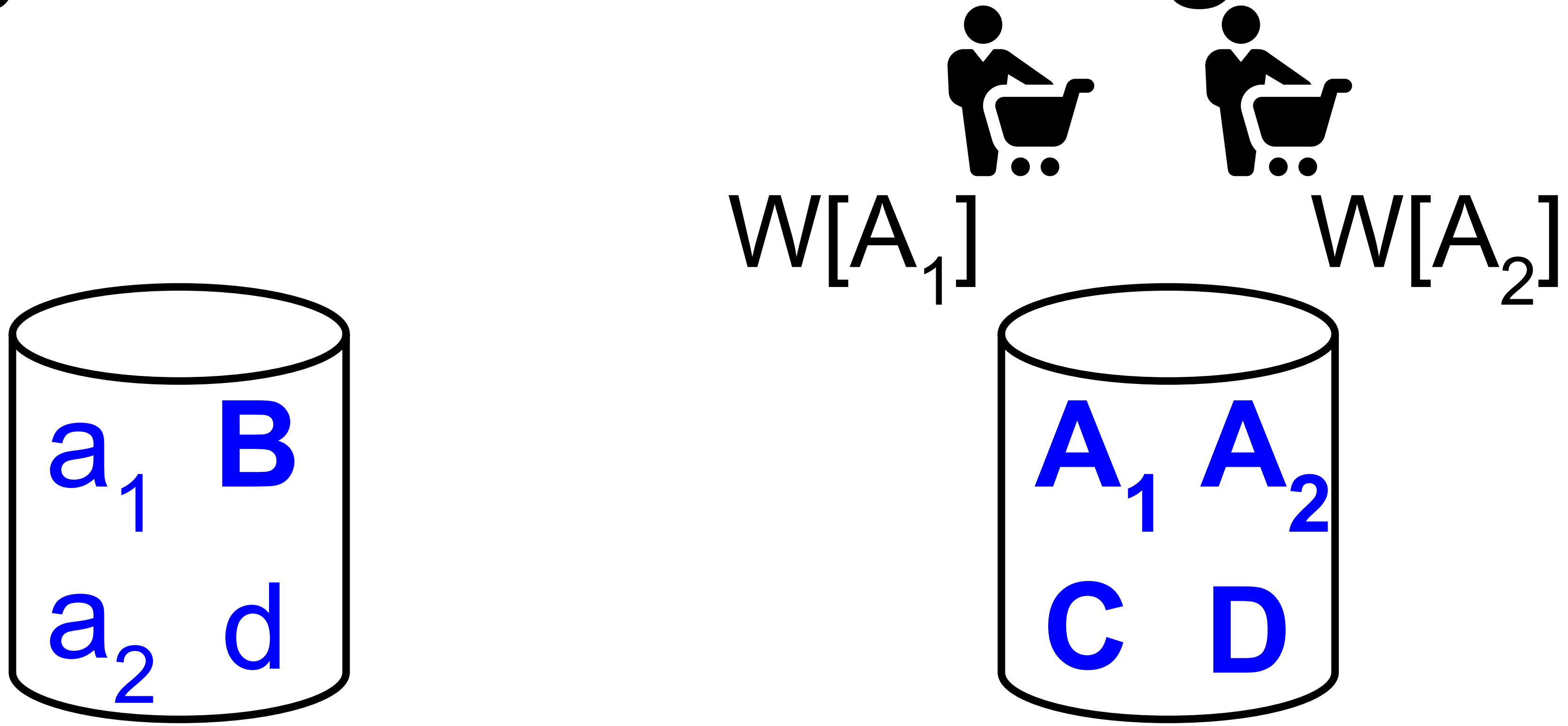
Dynamic Partitioning



Dynamic Partitioning



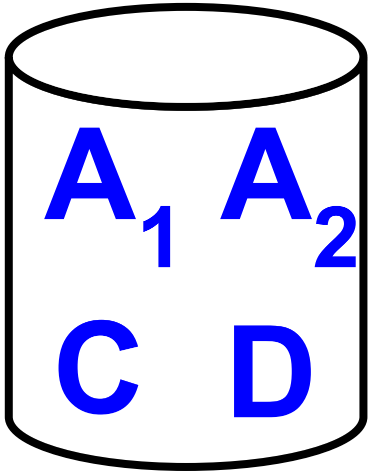
Dynamic Partitioning



Dynamic Partitioning

split partition

merge partition

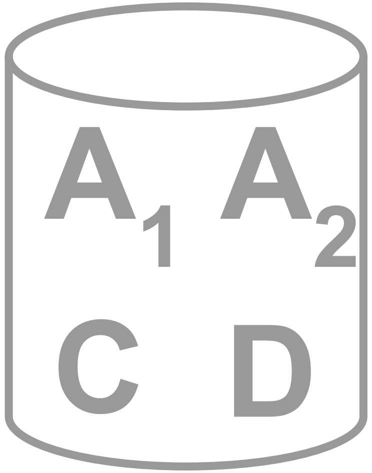


Dynamic Partitioning

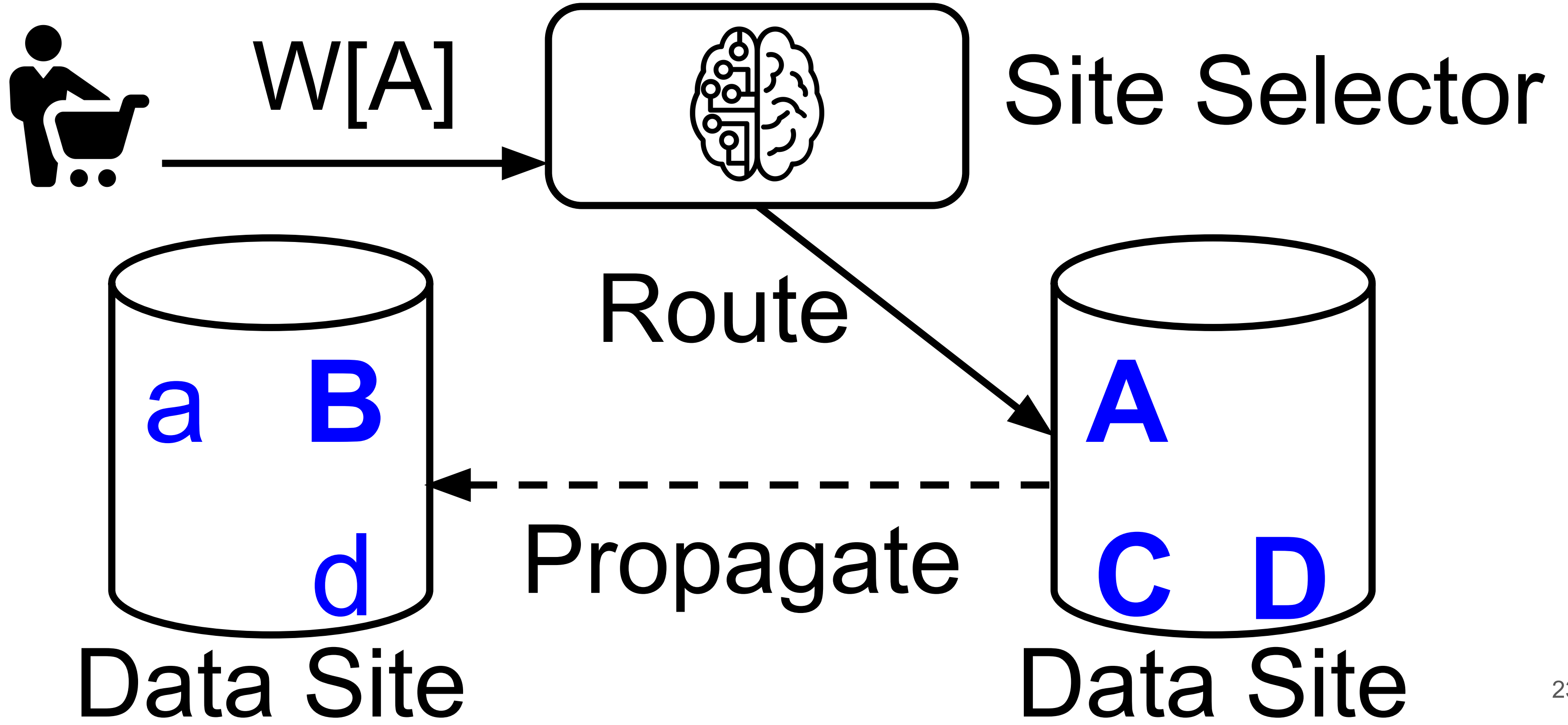
split partition

merge partition

Mitigate contention



Architecture



DRP Challenges

How to execute operations efficiently?

How to decide which operations to use?

DRP Challenges

How to execute operations efficiently?

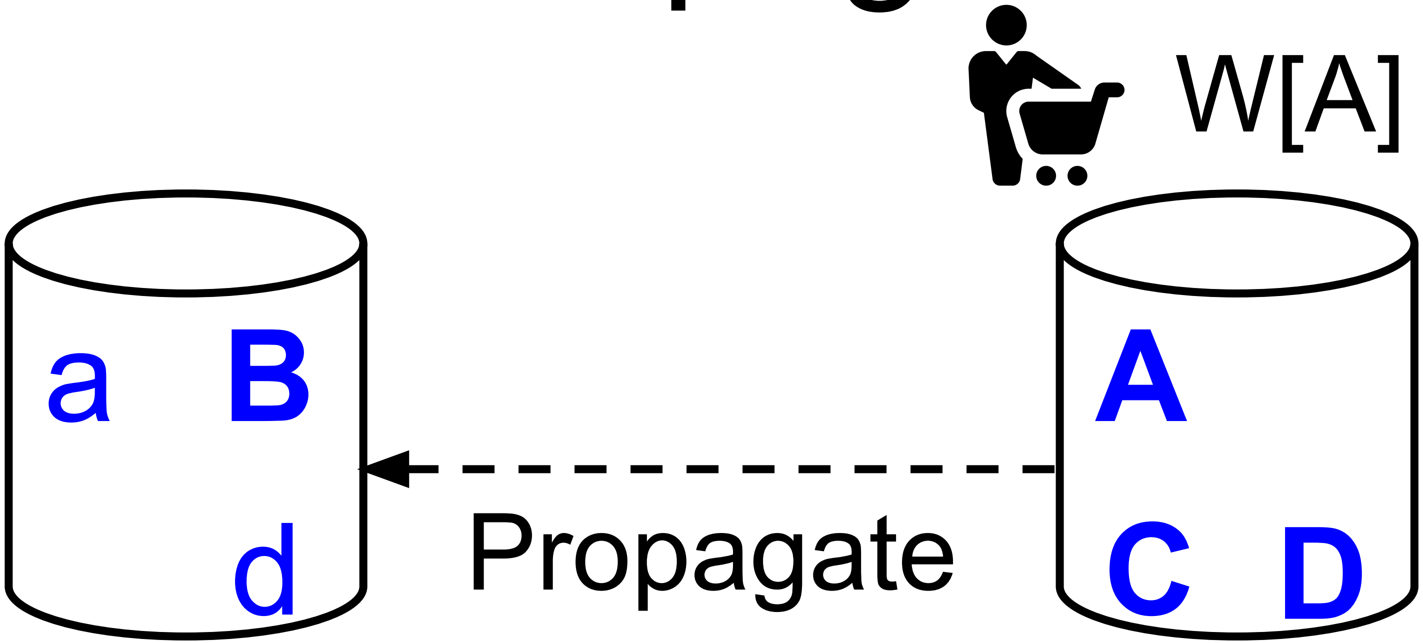
How to decide which operations to use?

Efficient execution

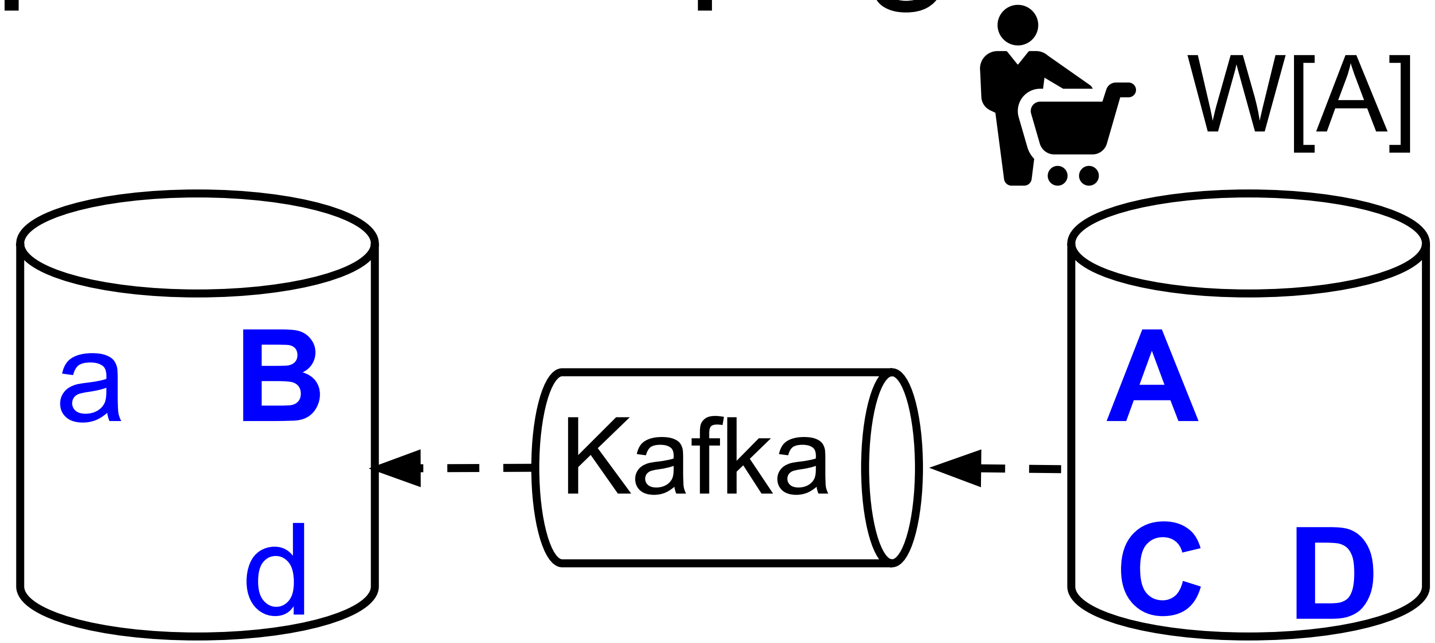
Decouple partition **reads** & **writes**

Partition based **multi-version** concurrency control

Update Propagation

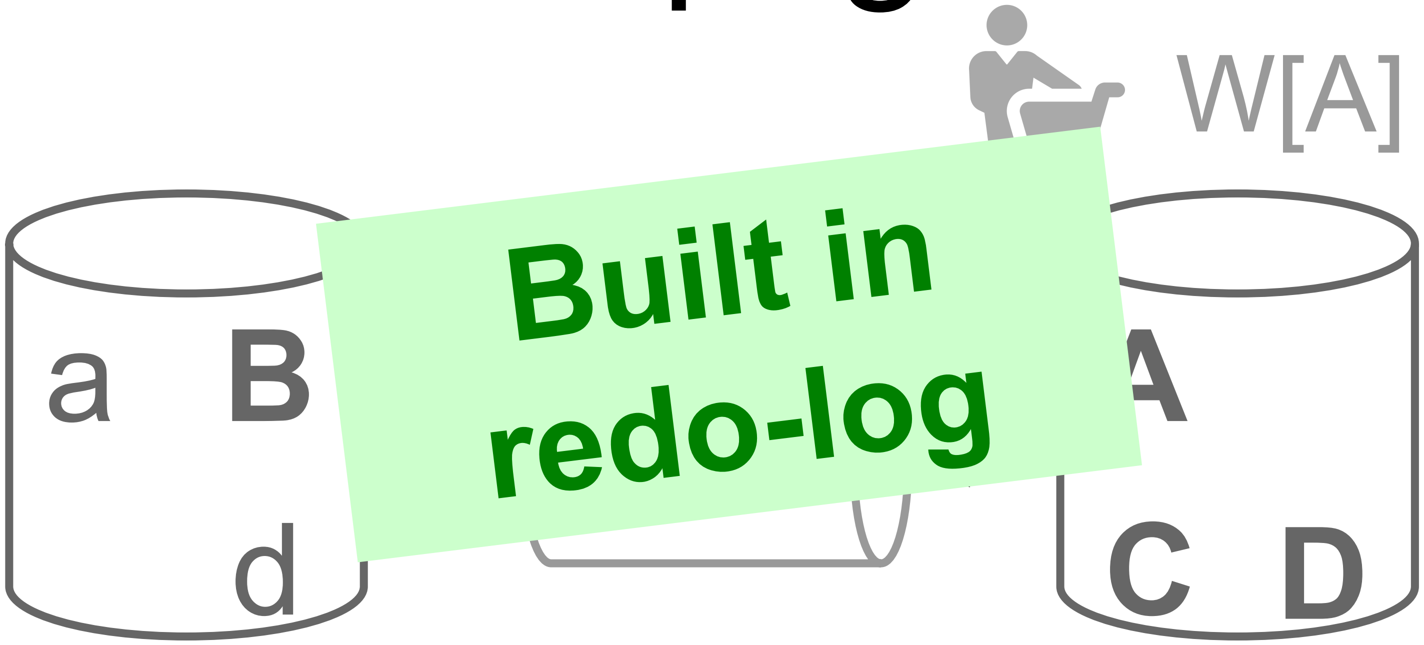


Update Propagation



Subscribe to partition updates

Update Propagation



Subscribe to partition updates

Update Propagation

Exploit multi-versioning to apply updates

Multiplex partition updates to Kafka

Remastering & repartitioning requires **changing subscriptions**

Adding replicas

Exploit multi-versioning & Kafka log

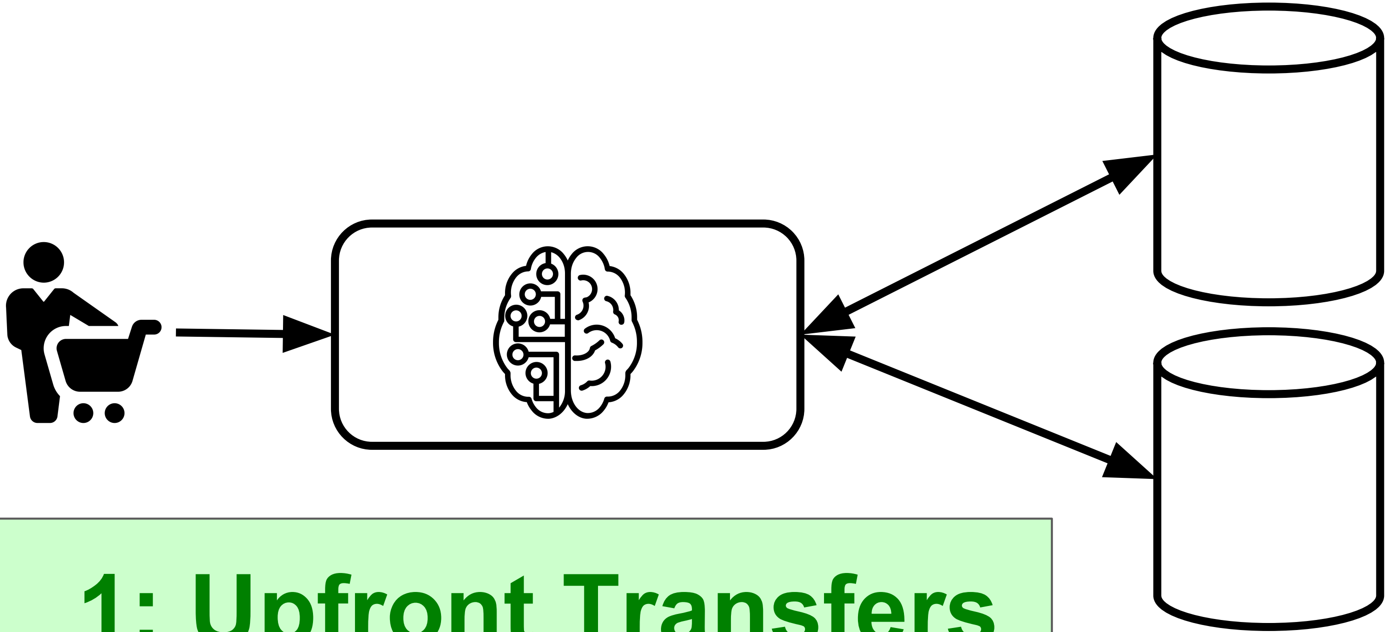
- Take a **read-only** partition **snapshot**
- Install snapshot & **subscribe** to Kafka

DRP Challenges

How to execute operations efficiently?

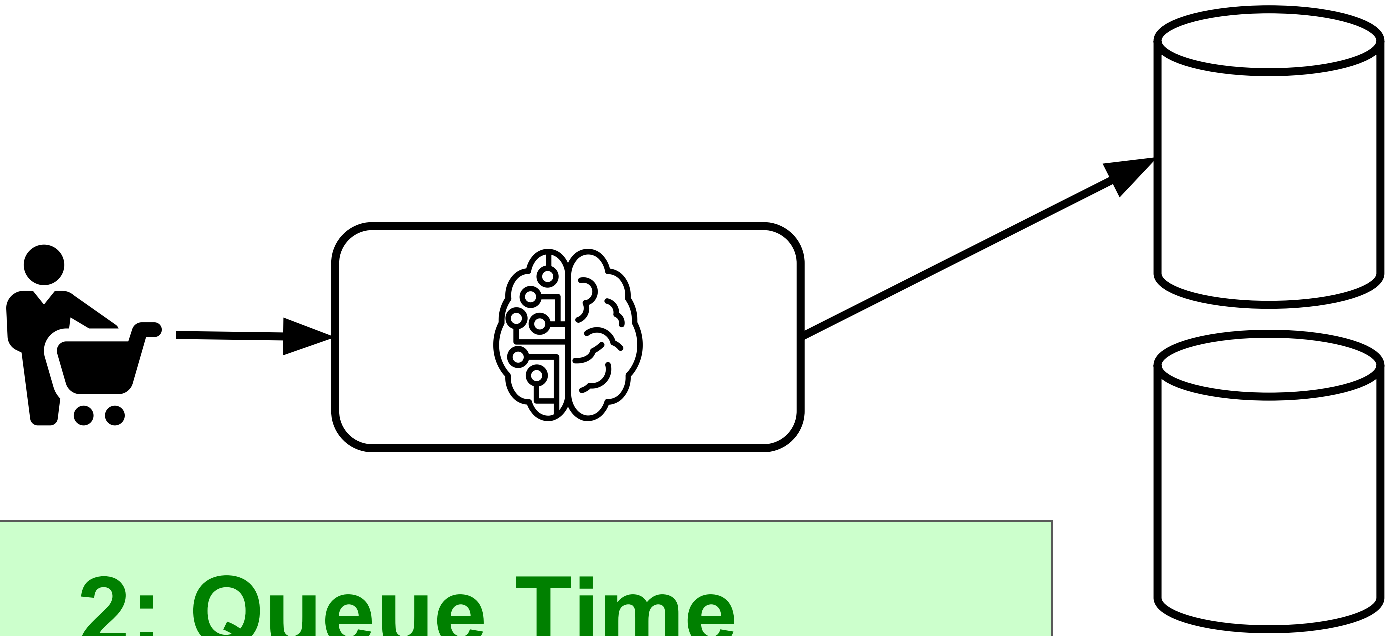
How to decide which operations to use?

The Cost Model



1: Upfront Transfers

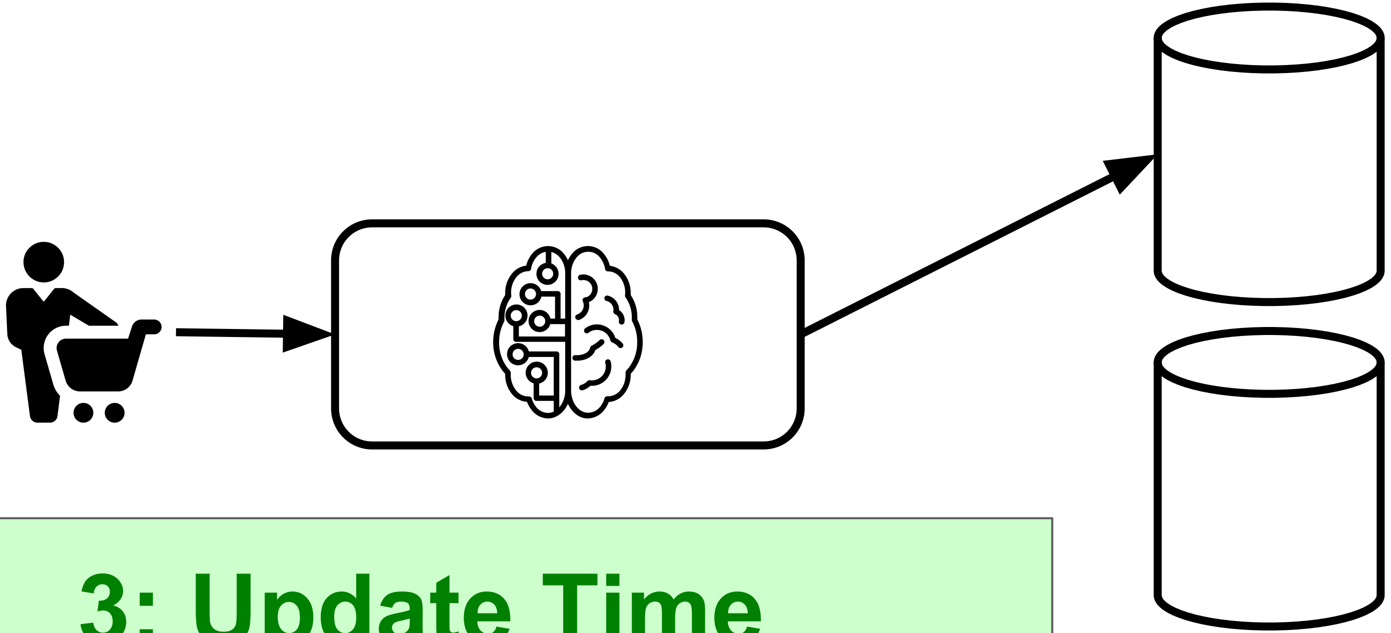
The Cost Model



<u>Queue</u>
T1
T2
…

2: Queue Time

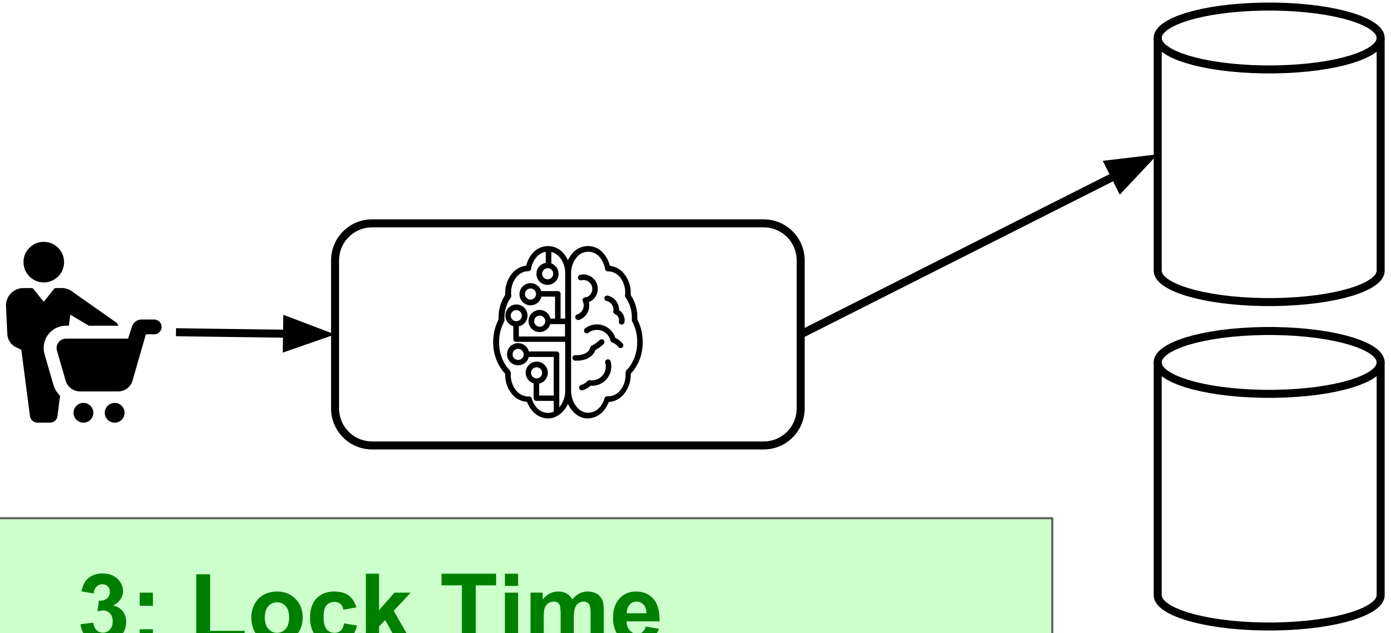
The Cost Model



Apply:
T15
T22

3: Update Time

The Cost Model



Waiters
A: **T1, T2, T3**
B:

3: Lock Time

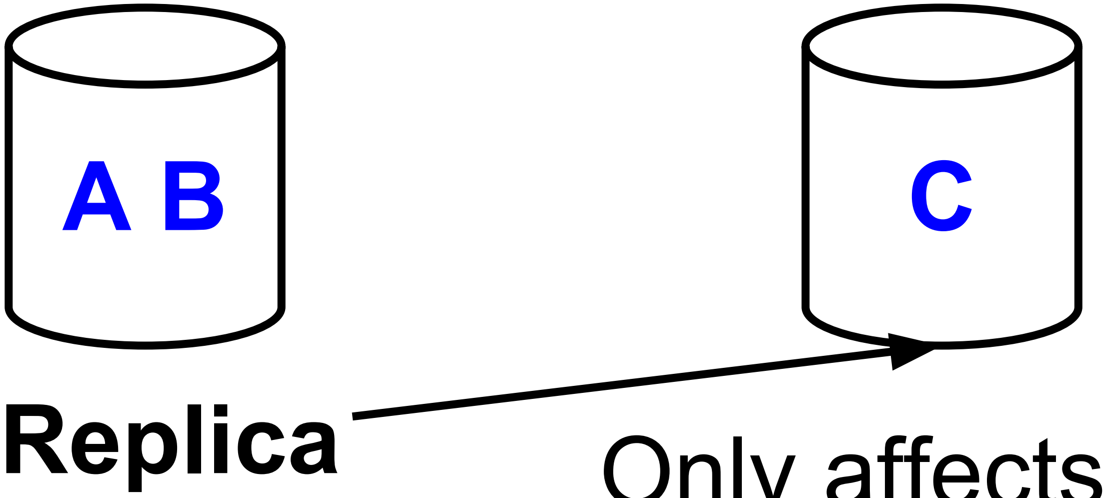
In Short:

$$arg\ \ min_D\left[\mathbb{E}_T\left[C_D(T)\right]\right]$$

ILP? Offline/Expensive

Online/Iterative Approach

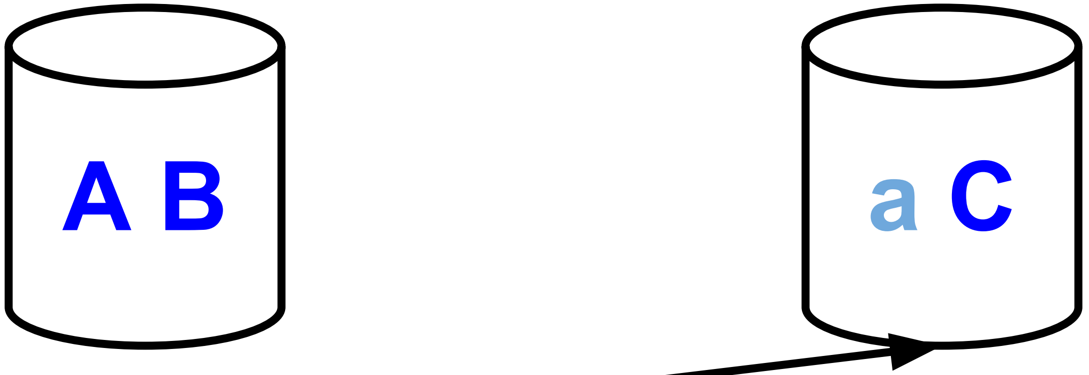
Ex: Adding a Replica



Add Replica of A?

Only affects reads!

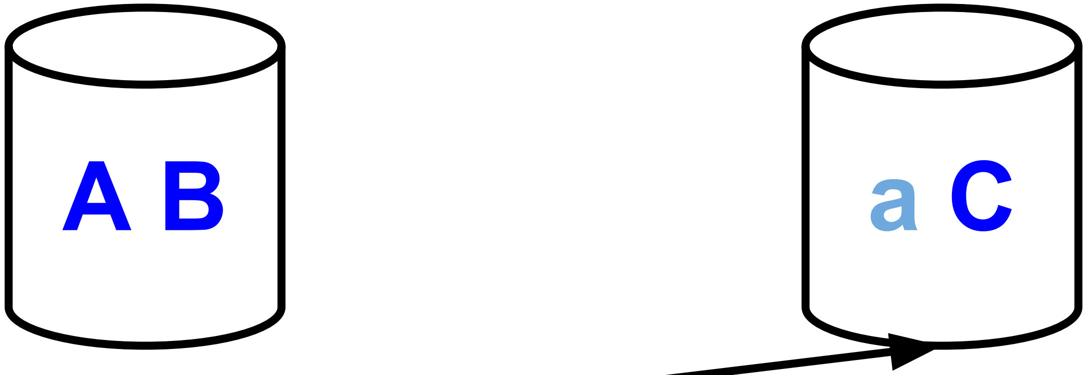
Ex: Adding a Replica



Add Replica of A?

Splits R[A] load

Ex: Adding a Replica

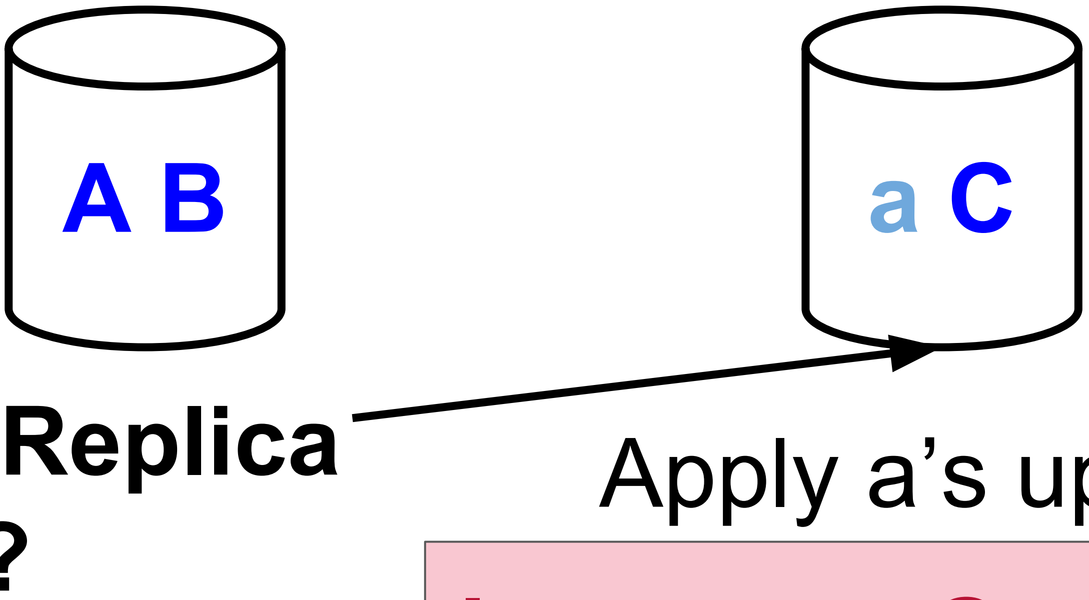


Add Replica of A?

Splits R[A] load

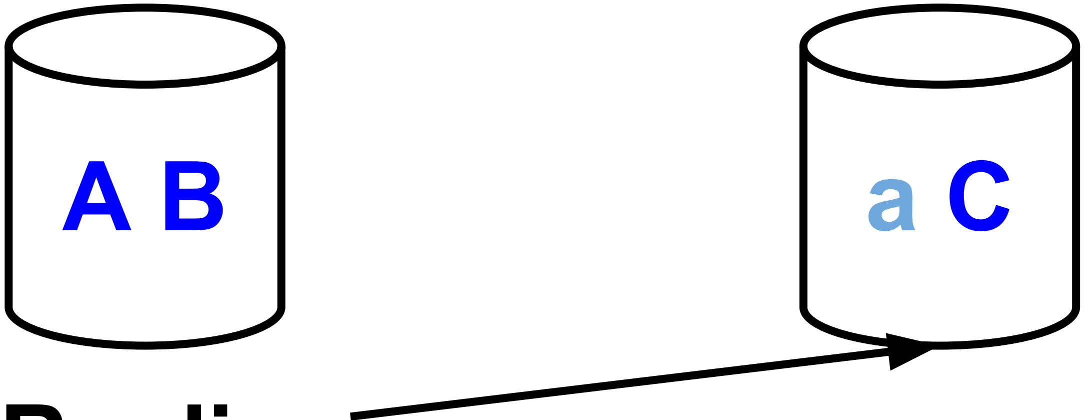
Reduces Queue Time

Ex: Adding a Replica



Increases Queue Time

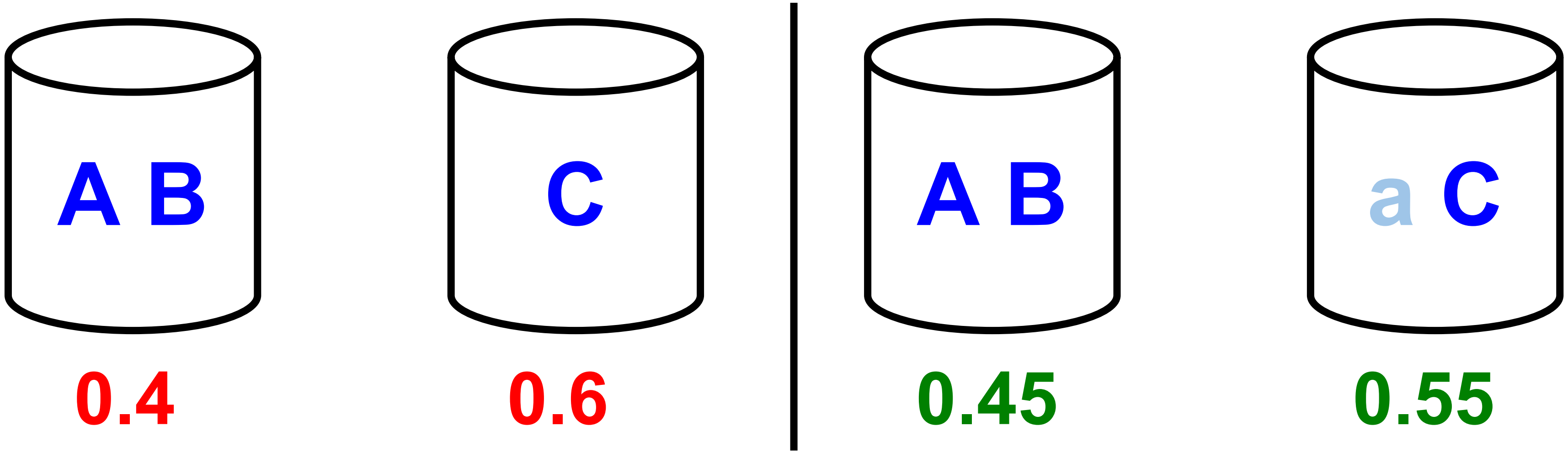
Ex: Adding a Replica



Add Replica of A?

Does not split R[A,B] load!

Add Replica Strategy



Compare estimated load balance before and after proposed replica placement

DRP Takeaways

Avoid distributed coordination

Dynamic replication and **partitioning**

Online **iterative physical design** adjustments

What’s Done:

- Update Propagation and Infrastructure Support
- Basic Underlying Cost Model
- Strategy Design for Split/Merge Partitions, Add/Remove Replicas, Remastering, Transaction Routing
- Statistics Support, Tracking, Sampling

What’s Left:

- Implementing Strategies into DRP
- Comparisons against alternative strategies/baselines
- Comprehensive Experimental Evaluation
- Beyonds this course: **Optimization**