continuent

Preventing conflicts in Multi-master replication with Tungsten

Giuseppe Maxia, QA Director, Continuent

continuent

Introducing Continuent

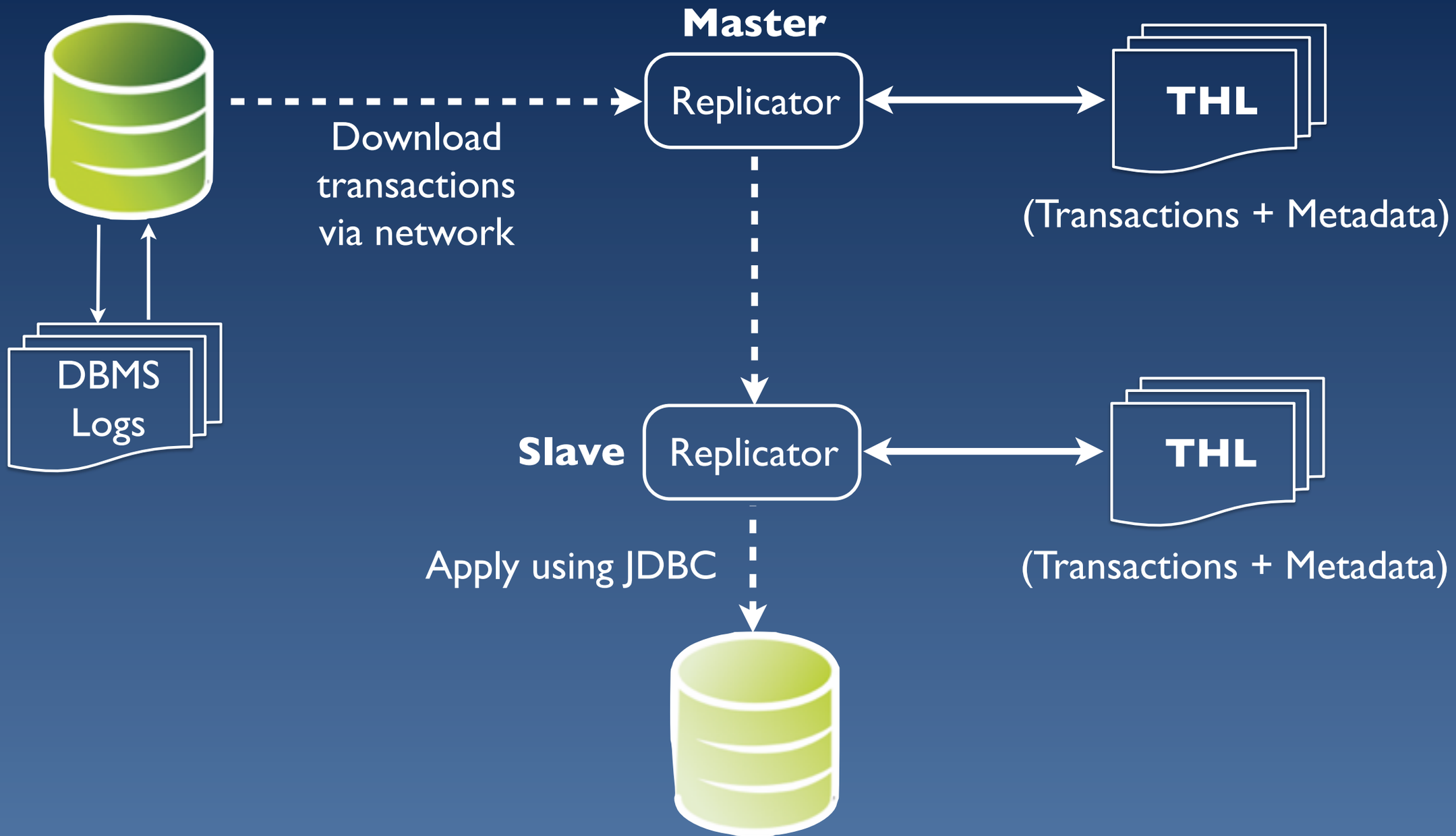
- The leading provider of clustering and replication for open source DBMS
- Our Product: **Continuent Tungsten**
  - Clustering - **Commercial**-grade HA, performance scaling and data management for MySQL
  - Replication - **Open source**, Flexible, high-performance data movement

What is Tungsten Replicator?

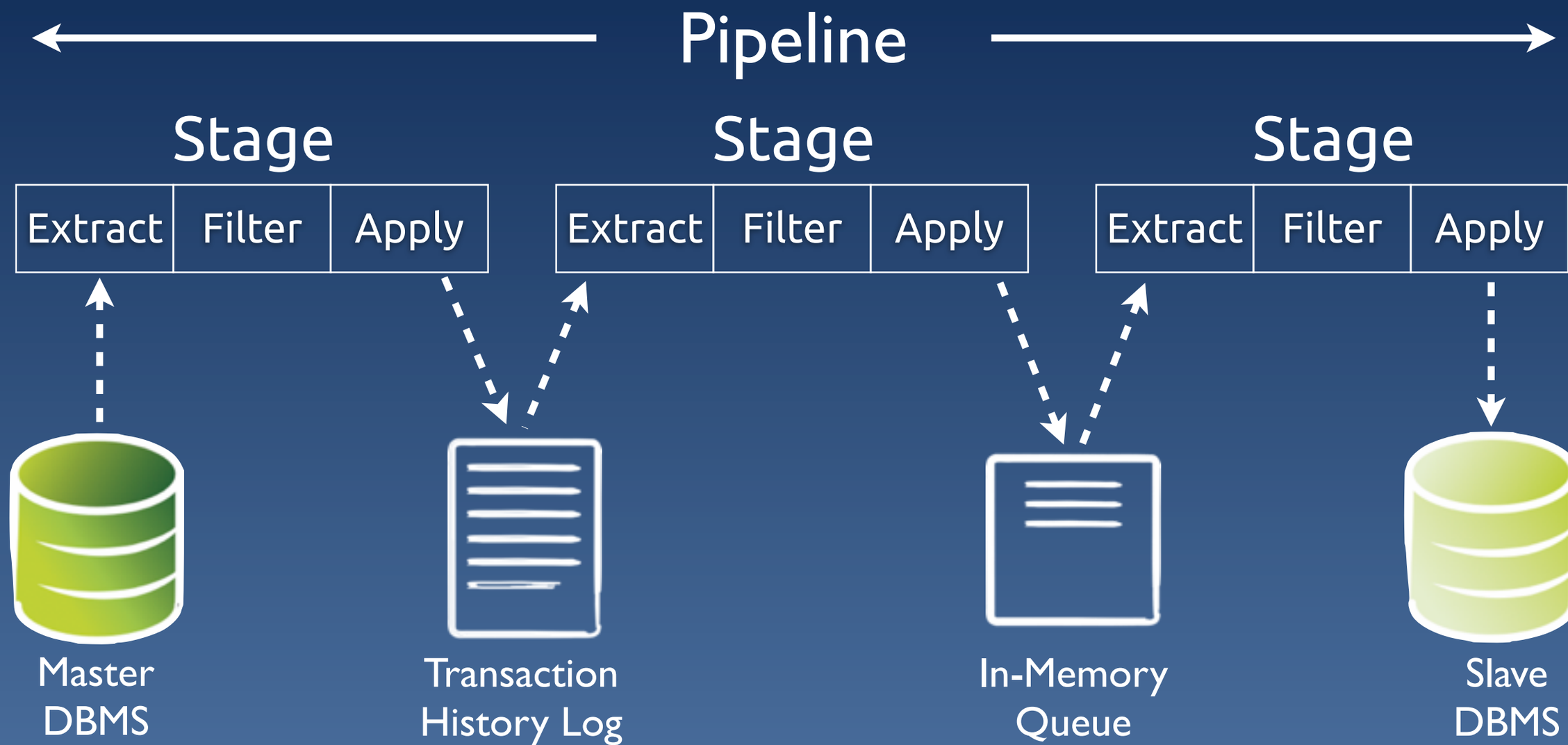
- Open source drop-in replacement for MySQL replication, providing:
  - Global transaction ID
  - Multiple masters
  - Multiple sources
  - Flexible topologies
  - Heterogeneous replication
  - Parallel replication
  - ... and more

continuent

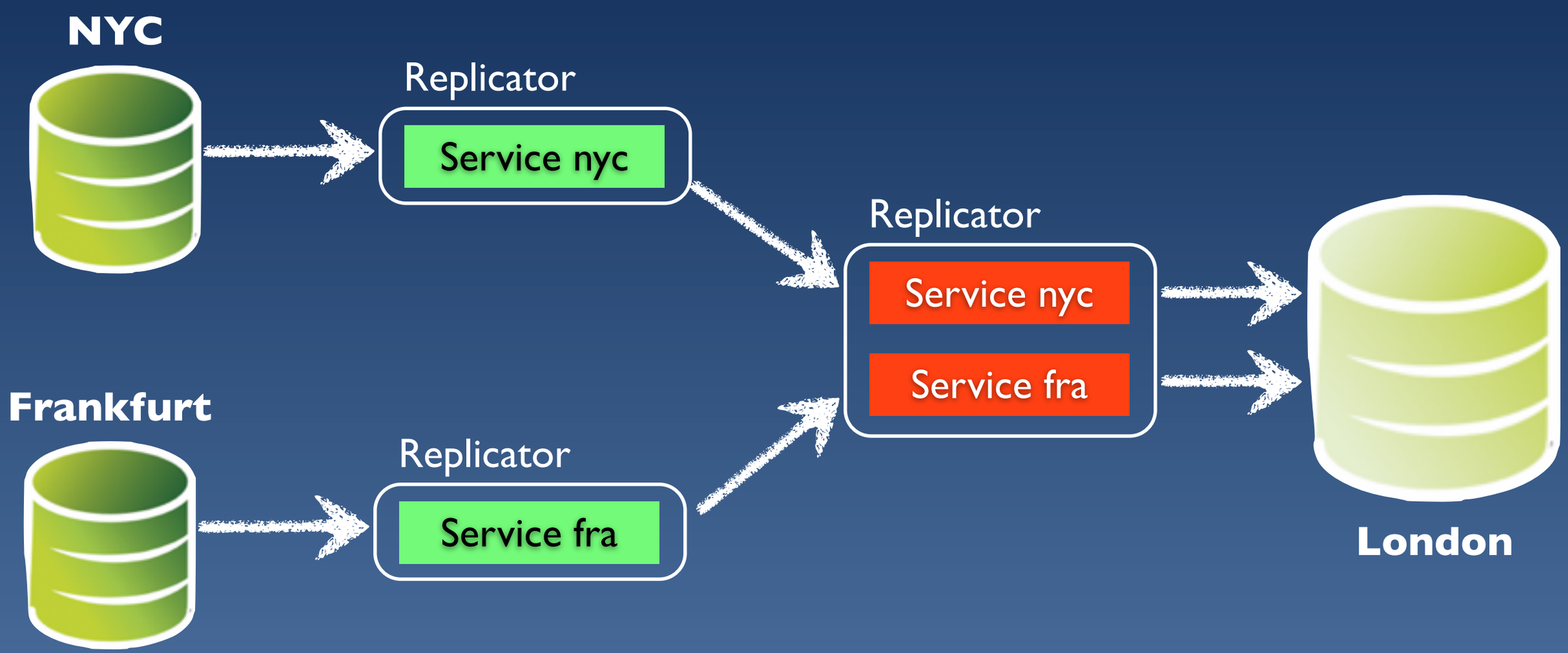
Tungsten Replicator Overview



Tungsten Replication Service



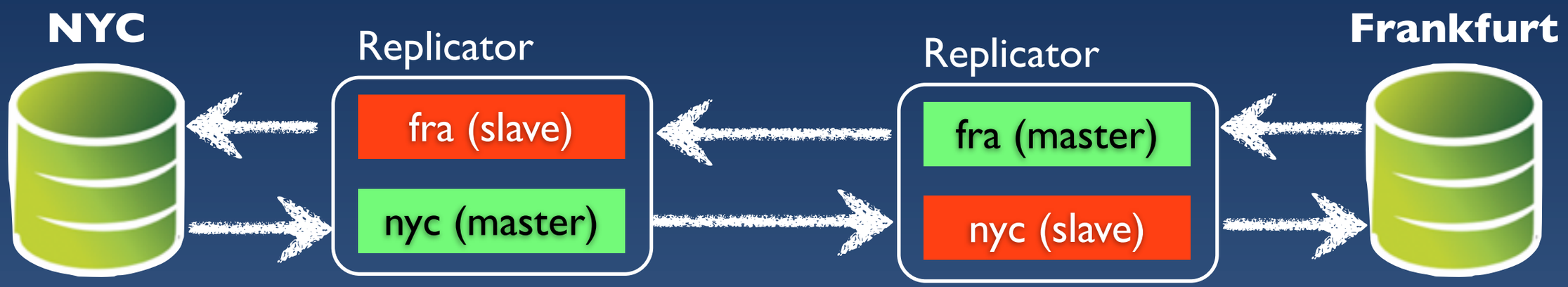
Multiple Services Per Replicator



Multi-Master Replication

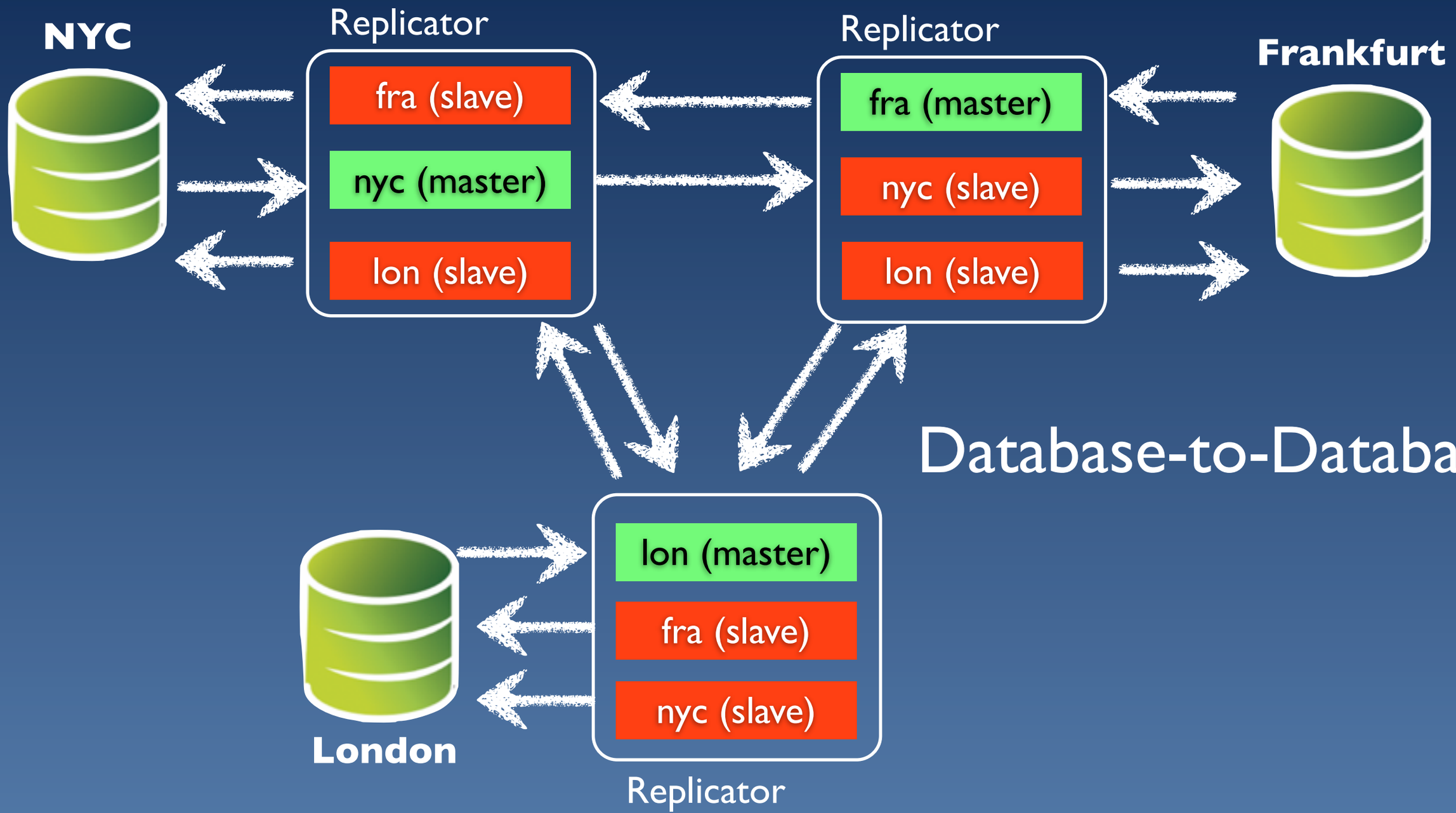
- Updates on 2+ sites (active-active mode)
- Enables geographic distribution of data
- No failover necessary if network fails or site becomes unavailable
- Not all applications can handle multi-master
  - Applications must avoid conflicts
  - Careful testing required
  - Restoration of broken systems may not be easy

Simple Multi-Master Configuration

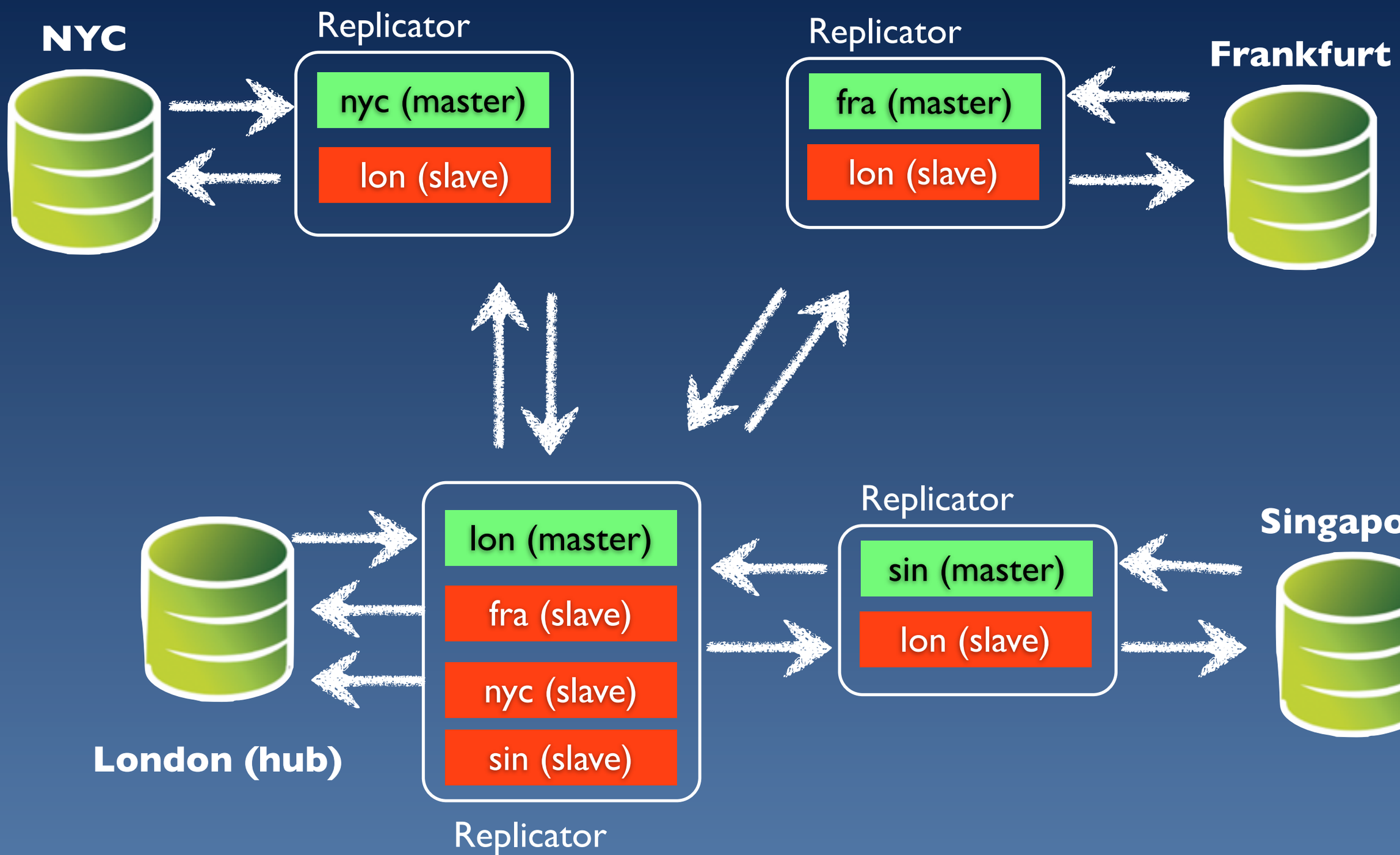


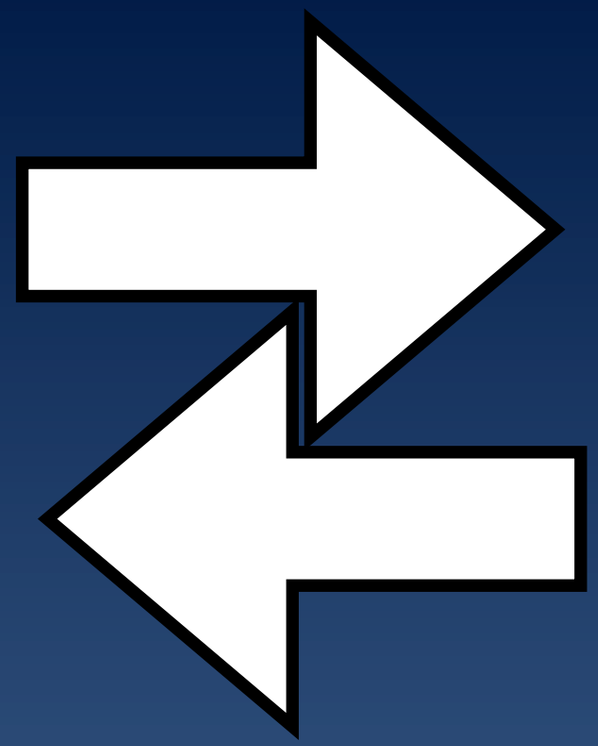
Database-to-Database

Three-node Multi-Master Configuration



multi-node star Configuration





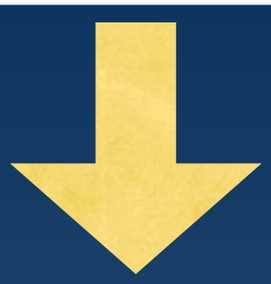
CONFLICTS

What's a conflict

- Data modified by several sources (masters)
- Creates one or more :
  - data loss (unwanted delete)
  - data inconsistency (unwanted update)
  - duplicated data (unwanted insert)
  - replication break

Data duplication

4	Matt	140



alpha

bravo

id	name	amount
1	Joe	100
2	Frank	110
3	Sue	100

charlie

4	Matt	130

BREAKS REPLICATION

auto_increment offsets are not a remedy

- A popular recipe
  - `auto_increment_increment + auto_increment_offset`
- They don't prevent conflicts
- They hide duplicates

Hidden data duplication

11	Matt	140

alpha
offset 1

INSERT

bravo
offset 2

id	name	amount
1	Joe	100
2	Frank	110
3	Sue	100

charlie
offset 3

INSERT

13	Matt	130

continuent

Data inconsistency

3	Sue	108

alpha

UPDATE

id	name	amount
1	Joe	100
2	Frank	110
3	Sue	100

bravo

charlie

UPDATE

3	Sue	105

continuent

Data loss

3	Sue	108

alpha

UPDATE

bravo

id	name	amount
1	Joe	100
2	Frank	110
3	Sue	100

charlie

DELETE

record #3

MAY BREAK REPLICATION

continuent

conflict handling strategies

- resolving
  - after the fact
  - Needs information that is missing in async replication
- avoiding
  - requires synchronous replication with 2pc
- preventing
  - setting and enforcing a split sources policy

used by Tungsten

- Transforming and resolving
  - all records are converted to INSERTs
  - conflicts are resolved within a given time window

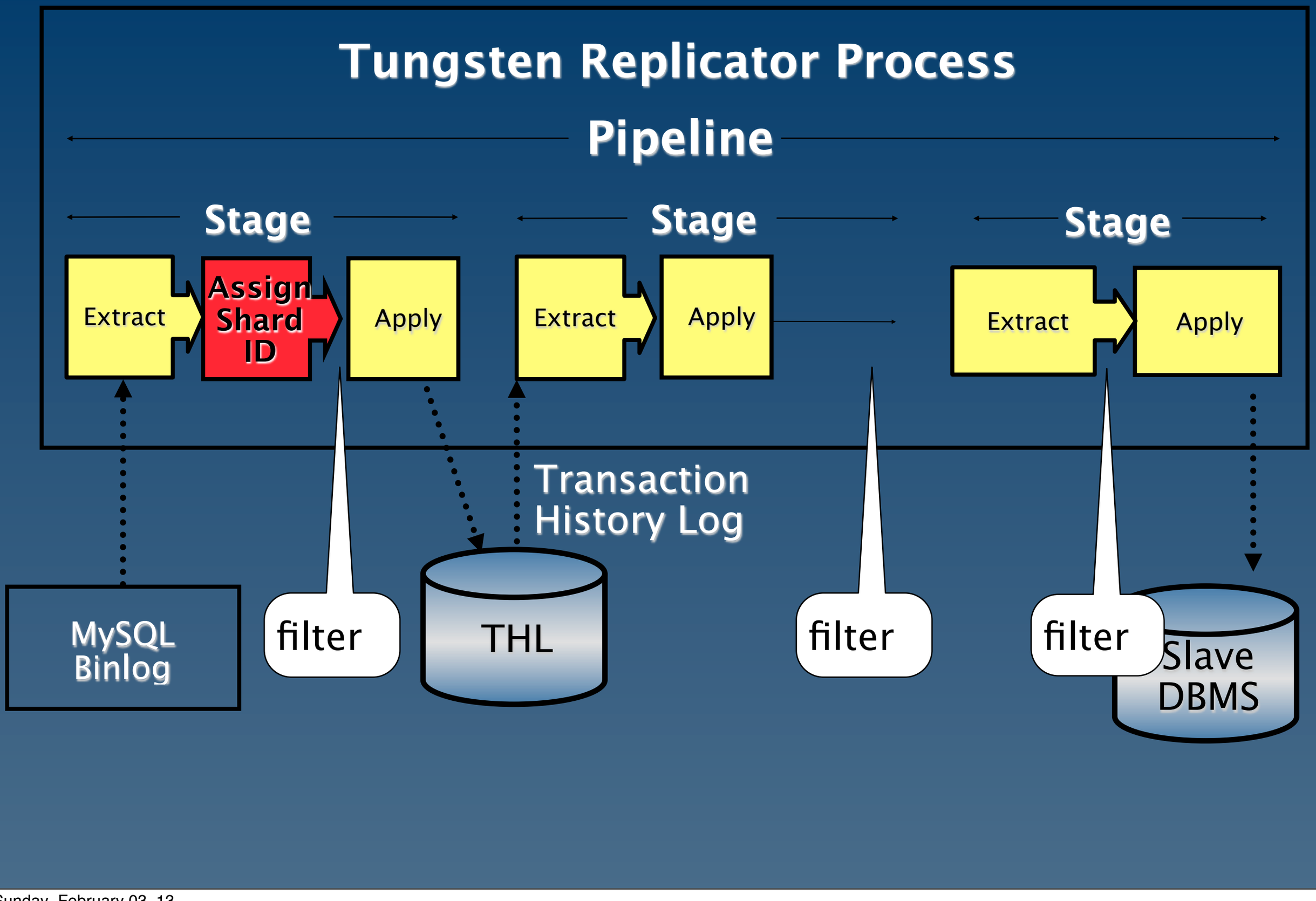
planned for future use

continuent

Tungsten Replicator Filters

continuent

Replicator Pipeline Architecture



Restrict replication to some schemas and tables

```
./tools/tungsten-installer \
   --master-slave -a \
  ...
  --svc-extractor-filters=replicate \
  "--property=replicator.filter.replicate.do=test,*.foo" \
  ...
  --start-and-report
```

```
# test="test.*"  -> same drawback as binlog-do-db in MySQL
# *.foo = table 'foo' in any database
# employees.dept_codes,employees.salaries => safest way
```


continuent

Multi-master:
Conflict prevention

continuent

Tungsten conflict prevention in a nutshell

1. define the rules

   (which master can update which database)

2. tell Tungsten the rules

3. define the policy

   (error, drop, warn, or accept)

4. Let Tungsten enforce your rules

continuent

Tungsten Conflict prevention facts

- Sharded by database
- Defined dynamically
- Applied on the slave services
- methods:
  - error: make replication fail
  - drop: drop silently
  - warn: drop with warning

Tungsten conflict prevention applicability

- unknown shards
  - The schema being updated is not planned
  - actions: accept, drop, warn, error
- unwanted shards
  - the schema is updated from the wrong master
  - actions: accept, drop, warn, error
- whitelisted shards
  - can be updated by any master

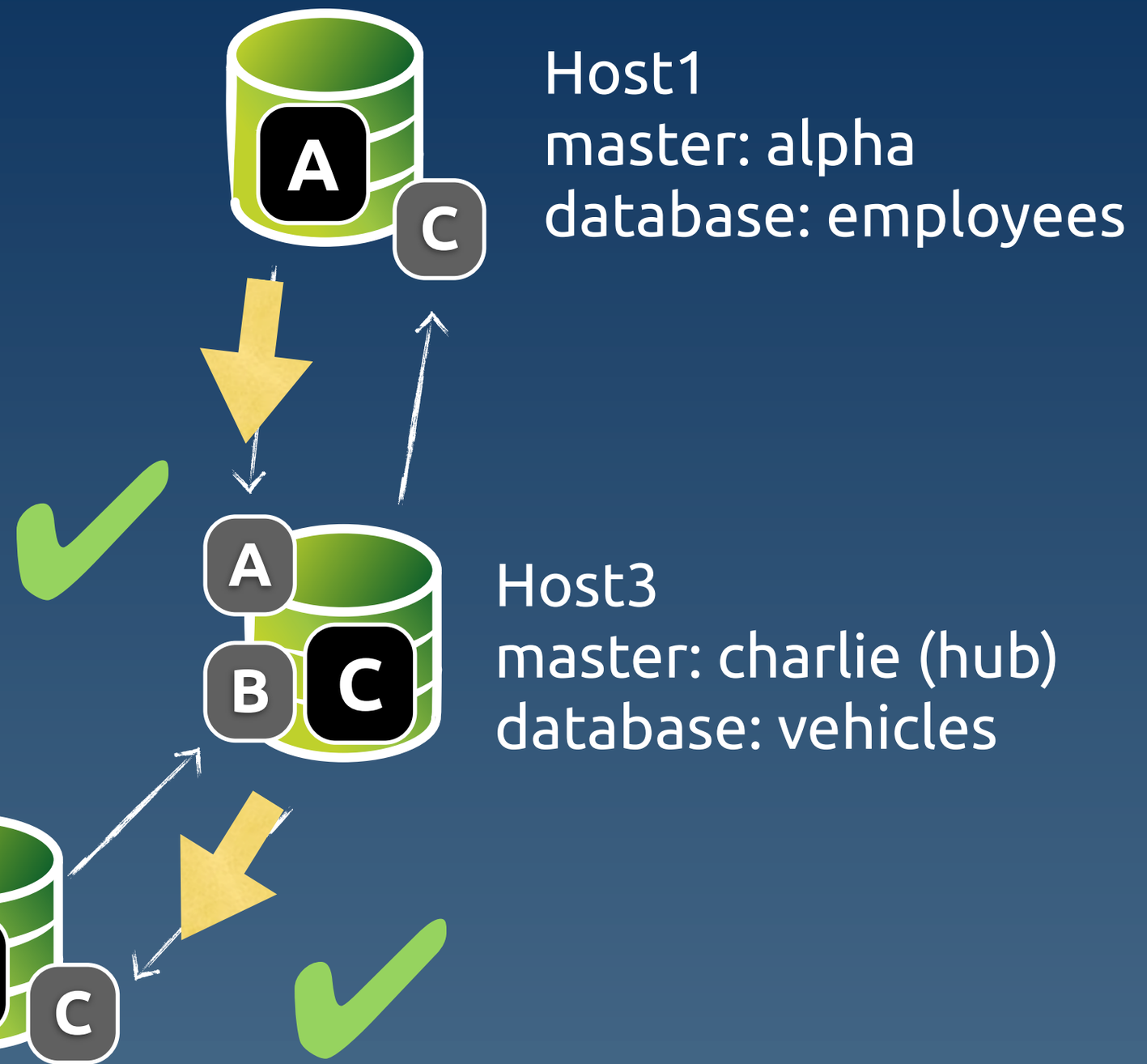
Conflict prevention directives

```
--svc-extractor-filters=shardfilter

replicator.filter.shardfilter.unknownShardPolicy=error

replicator.filter.shardfilter.unwantedShardPolicy=error

replicator.filter.shardfilter.enforceHomes=false

replicator.filter.shardfilter.allowWhitelisted=false
```


continuent

conflict prevention in a star topology

alpha updates employees



Host1
master: alpha
database: employees

Host3
master: charlie (hub)
database: vehicles

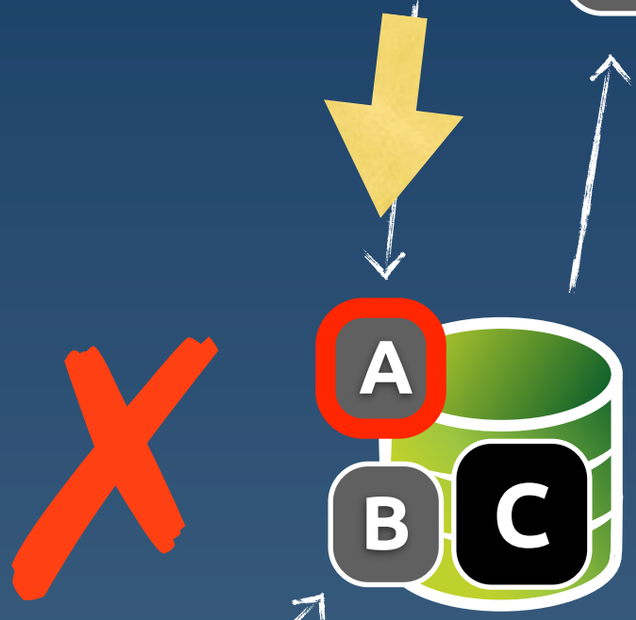
Host2
master: bravo
database: buildings

continuent

conflict prevention in a star topology

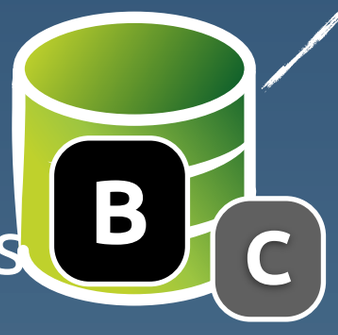
alpha updates
vehicles

Host1
master: alpha
database: employees



Host3
master: charlie (hub)
database: vehicles

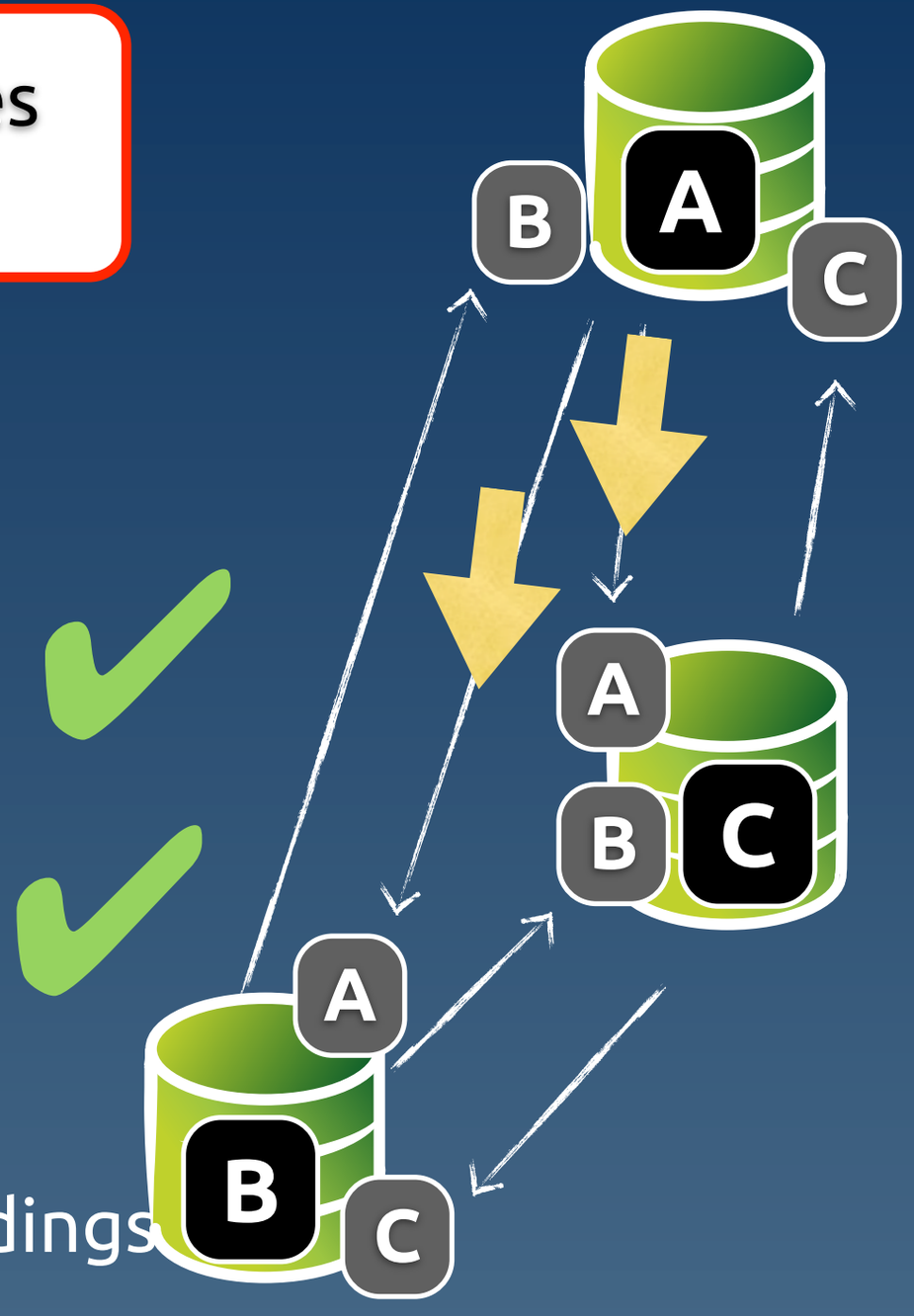
Host2
master: bravo
database: buildings



continuent

conflict prevention in a all-masters topology

alpha updates employees



Host1
master: alpha
database: employees

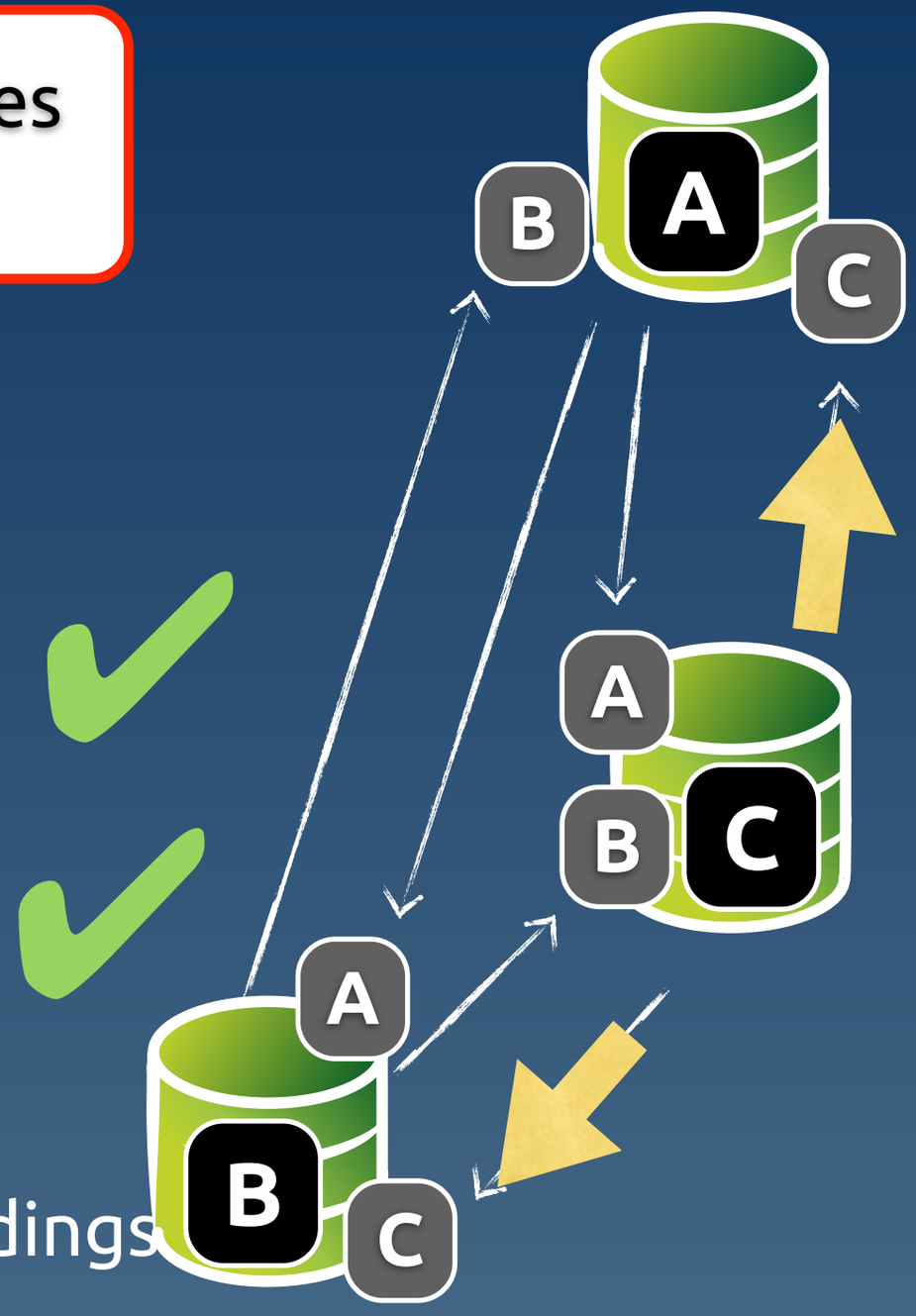
Host3
master: charlie (hub)
database: vehicles

Host2
master: bravo
database: buildings

continuent

conflict prevention in a all-masters topology

charlie updates vehicles



Host1
master: alpha
database: employees

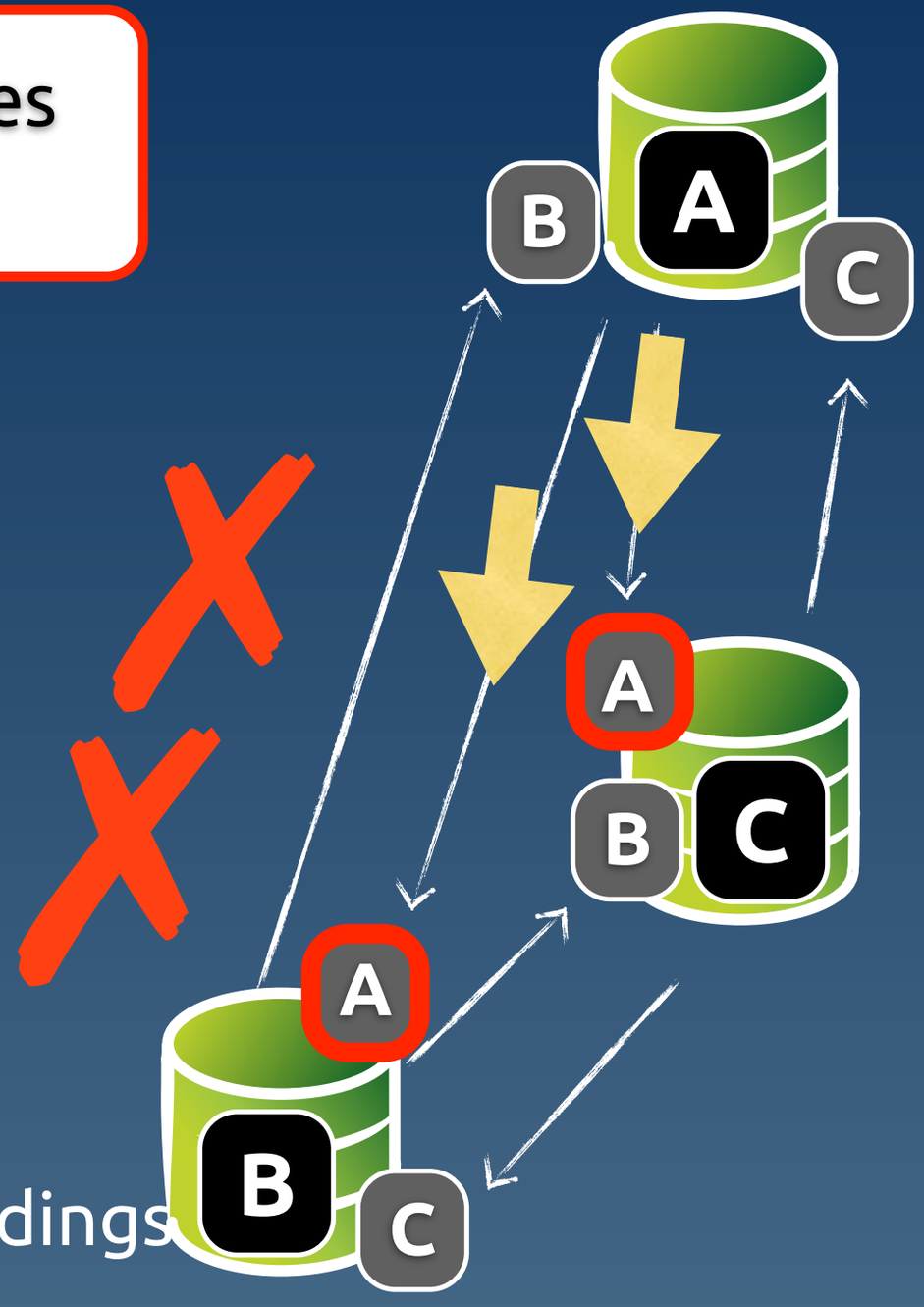
Host3
master: charlie (hub)
database: vehicles

Host2
master: bravo
database: buildings

continuent

conflict prevention in a all-masters topology

alpha updates buildings



Host1
master: alpha
database: employees

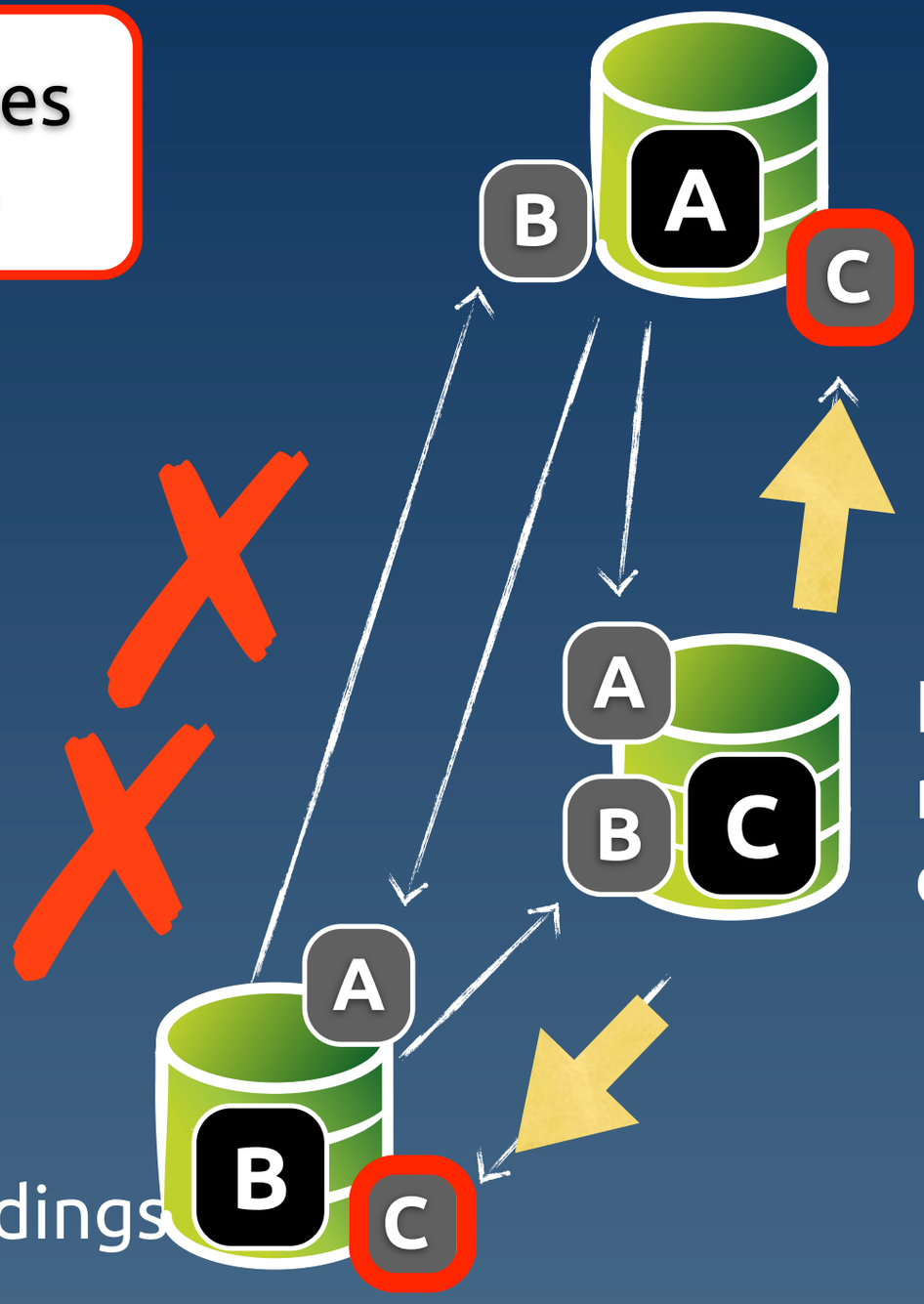
Host3
master: charlie (hub)
database: vehicles

Host2
master: bravo
database: buildings

continuent

conflict prevention in a all-masters topology

charlie updates employees



Host1
master: alpha
database: employees

Host3
master: charlie (hub)
database: vehicles

Host2
master: bravo
database: buildings

continuent

setting conflict prevention rules

```
trepctl -host host1 -service charlie \
  shard -insert < shards.map

cat shards.map
shard_id              master      critical
personnel             alpha       false
buildings             bravo       false
vehicles              charlie     false
test                  whitelisted false

# charlie is slave service in host 1
```

setting conflict prevention rules

```
trepctl -host host2 -service charlie \
  shard -insert < shards.map

cat shards.map
shard_id            master      critical
personnel           alpha       false
buildings           bravo       false
vehicles            charlie     false
test                whitelisted false

# charlie is slave service in host 2
```

setting conflict prevention rules

```
trepctl -host host3 -service alpha \
  shard -insert < shards.map
trepctl -host host3 -service bravo \
  shard -insert < shards.map

cat shards.map
shard_id              master       critical
personnel             alpha        false
buildings             bravo        false
vehicles              charlie      false
test                  whitelisted false

# alpha and bravo are slave services in host 3
```

Conflict prevention demo

- reminder
- Server #1 can update "employees"
- Server #2 can update "buildings"
- Server #3 can update "vehicles"

Sample correct operation (1)

```
mysql #1> create table employees.names( ... )


# all servers receive the table
# all servers keep working well
```

Sample correct operation (2)

```
mysql #2> create table buildings.homes( ... )


# all servers receive the table
# all servers keep working well
```


continuent

Sample incorrect operation (1)

```
mysql #2> create table employees.nicknames( ... )


# Only server #2 receives the table
# slave service in hub gets an error
# slave service in #1 does not receive anything
```

sample incorrect operation (2)

```
#3 $ trepct services | simple_services
alpha    [slave]
seqno:          7   - latency:   0.136 - ONLINE

bravo    [slave]
seqno:         -1   - latency:  -1.000 - OFFLINE:ERROR

charlie  [master]
seqno:         66   - latency:   0.440 - ONLINE
```


continuent

sample incorrect operation (3)

```
#3 $  trepct -service bravo status
NAME                          VALUE
----                          -----
appliedLastEventId      : NONE
appliedLastSeqno        : -1
appliedLatency          : -1.0
(...)
offlineRequests         : NONE
pendingError            : Stage task failed: q-to-dbms
pendingErrorCode        : NONE
pendingErrorEventId     : mysql-bin.000002:0000000000001241;0
pendingErrorSeqno       : 7
pendingExceptionMessage: Rejected event from wrong shard:
seqno=7 shard ID=employees shard master=alpha service=bravo
(...)
```

Fixing the issue

```
mysql #1> drop table if exists employees.nicknames;
mysql #1> create table if exists employees.nicknames ( ... ) ;

#3 $ trepct -service bravo online -skip-seqno 7
```

```
# all servers receive the new table
```


continuent

Sample whitelisted operation

```
mysql #2> create table test.hope4best( ... )

mysql #1> insert into test.hope4best values ( ... )


# REMEMBER: 'test' was explicitly whitelisted
# All servers get the new table and records
# But there is no protection against conflicts
```


continuent

Parting thoughts

We are hiring !

http://continuent.com/about/careers

continuent

blog: http://datacharmer.blogspot.com
twitter: @datacharmer

Continuent Website:
http://www.continuent.com

Tungsten Replicator 2.0:
http://tungsten-replicator.org