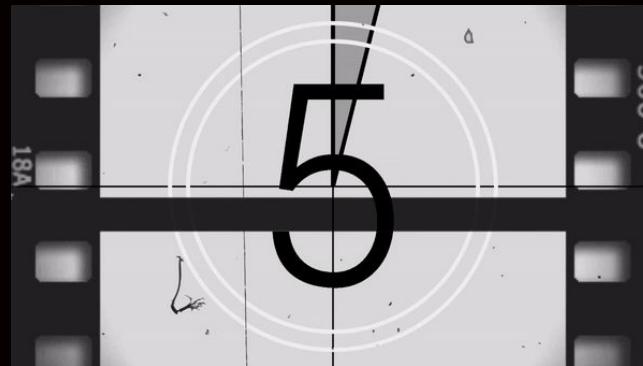


MCO2

Distributed Database

Cua, Lander Peter E.
Gaba, Jacob Bryan B.
Gonzales, Mark Edward M.
Lee, Hylene Jules G.



01

DISTRIBUTED DATABASE SETUP

Cloud platform, data sharding, remote database access

02

ALGORITHM

Flow, happy paths and unhappy paths

03

GLOBAL CONCURRENCY

System handling of concurrent transactions

04

GLOBAL RECOVERY

System response to crashing nodes

05

EVALUATION AND RESULTS

Concurrency and failure recovery validation

06

INSIGHTS AND CONCLUSION

Summary of findings and analyses



Cloud Platform AWS RDS

AWS Services Search for services, features, blogs, docs, and more [Alt+S]

RDS > Databases

Databases

Filter by databases

DB identifier	Role	Engine	Region & AZ	Size	Status	CPU	Current activity	Maintenance
central-node	Instance	MySQL Community	us-east-1a	db.t2.micro	Available	5.33%	7 Connections	none
node-2	Instance	MySQL Community	us-east-1f	db.t2.micro	Available	5.42%	4 Connections	none
node-3	Instance	MySQL Community	us-east-1f	db.t2.micro	Available	5.83%	4 Connections	none

Group resources Modify Actions ▾ Restore from S3 Create database

DB identifier Role Engine Region & AZ Size Status CPU Current activity Maintenance

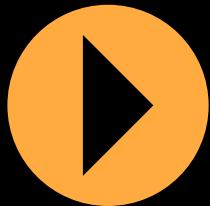
central-node Instance MySQL Community us-east-1a db.t2.micro Available 5.33% 7 Connections none

node-2 Instance MySQL Community us-east-1f db.t2.micro Available 5.42% 4 Connections none

node-3 Instance MySQL Community us-east-1f db.t2.micro Available 5.83% 4 Connections none

Dashboard Databases Query Editor Performance insights Snapshots Automated backups Reserved instances Proxies Subnet groups Parameter groups Option groups Custom Availability Zones Custom engine versions Events Event subscriptions Recommendations (3) Certificate update

Feedback English (US) © 2022, Amazon Web Services, Inc. or its affiliates. Privacy Terms Cookie preferences



Database Access

```
const centralPool = mySQL.createPool({  
  connectionLimit: 20,  
  host: process.env.CENTRAL_URL,  
  port: process.env.DB_PORT,  
  user: process.env.CENTRAL_USERNAME,  
  password: process.env.CENTRAL_PASSWORD,  
  database: DATABASE,  
  connectTimeout: 30000  
});  
  
const node2Pool = mySQL.createPool({  
  connectionLimit: 20,  
  host: process.env.NODE2_URL,  
  port: process.env.DB_PORT,  
  user: process.env.NODE2_USERNAME,  
  password: process.env.NODE2_PASSWORD,  
  database: DATABASE,  
  connectTimeout: 30000  
});  
  
const node3Pool = mySQL.createPool({  
  connectionLimit: 20,  
  host: process.env.NODE3_URL,  
  port: process.env.DB_PORT,  
  user: process.env.NODE3_USERNAME,  
  password: process.env.NODE3_PASSWORD,  
  database: DATABASE,  
  connectTimeout: 30000  
});
```

```
PORT = 3000  
DB_PORT = 3306  
HOSTNAME = localhost  
CENTRAL_URL = central-node.cb7q6f4hhftm.us-east-1.rds.amazonaws.com  
CENTRAL_USERNAME = centralAdmin  
CENTRAL_PASSWORD = centralPassword  
NODE2_URL = node-2.cb7q6f4hhftm.us-east-1.rds.amazonaws.com  
NODE2_USERNAME = node2Admin  
NODE2_PASSWORD = node2Password  
NODE3_URL = node-3.cb7q6f4hhftm.us-east-1.rds.amazonaws.com  
NODE3_USERNAME = node3Admin  
NODE3_PASSWORD = node3Password
```

CONNECTION POOL



Database Schema

Movies Table

Attribute	Data Type	Description
ID	int	ID of the movie (primary key)
title	varchar	Title of the movie
year	int	Year of release of the movie
genre	varchar	Genre of the movie
rank	float	Rating of the movie
director	varchar	Director of the movie
actor1	varchar	First actor in the movie
actor2	varchar	Second actor in the movie



Distributed Database Topology

CENTRAL NODE

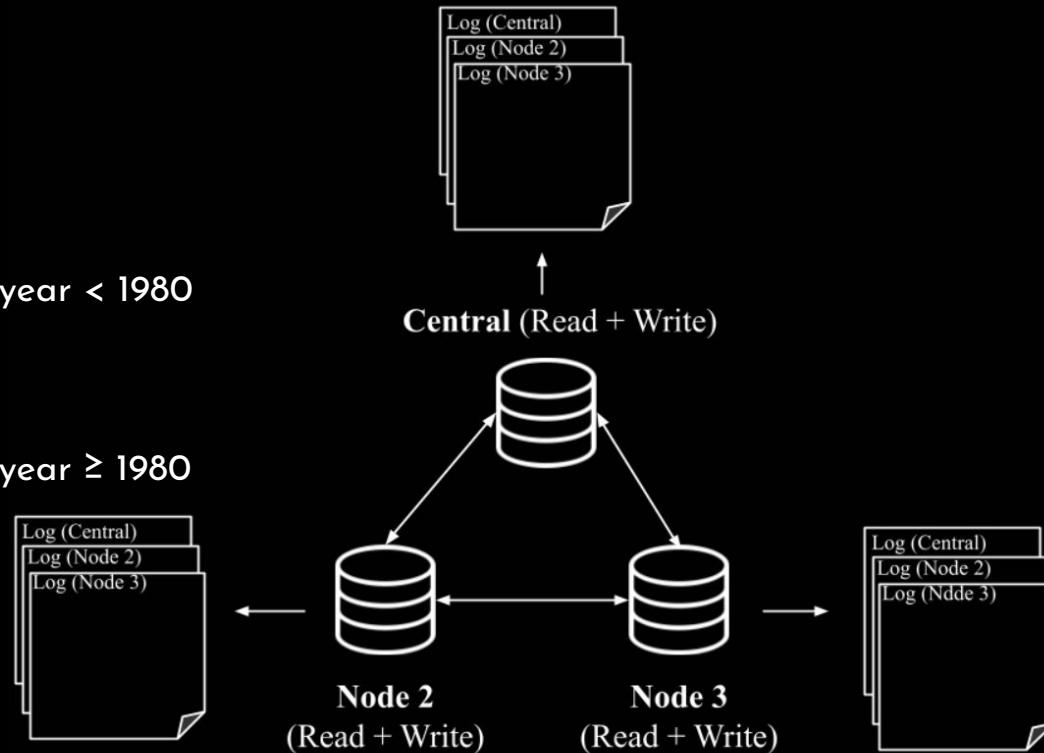
- Contains all movies
- 173394 entries

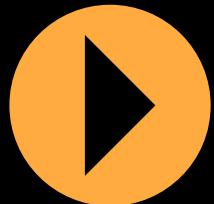
NODE 2

- Contains all movies with $\text{year} < 1980$
- 92003 entries

NODE 3

- Contains all movies with $\text{year} \geq 1980$
- 81391 entries





Web Application

<https://cqql-distributed-db.herokuapp.com/>

171404	Zeder	1983	Horror	5.8	Pupi Avati	Adolfo Belletti	Maria Teresa Tofano	
Showing 1 to 10 of 2,000 entries								
Previous	1	2	3	4	5	...	200	Next

Search

ID, Title, Genre, Director, or Actor
Leave blank to display all entries

Display Partial Results

Display All (May be slow)

Insert

Title	Director
Title of the movie	Lead director
Year	Actor 1
Year of release	Lead actor
Genre	Actor 2
Comedy, drama, etc.	Supporting actor

Update

ID	Director
Basis for selecting entry	Lead director
Title	Actor 1
Title of the movie	Lead actor
Genre	Actor 2
Comedy, drama, etc.	Supporting actor

Delete

ID
Movie ID

Delete

Insert

Rank
Movie rating

Insert

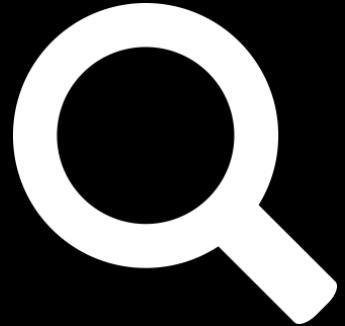
Update

Rank
Movie rating

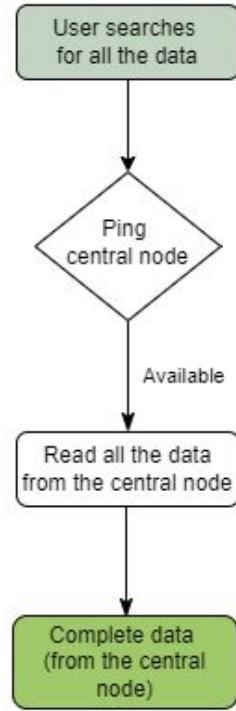
Update



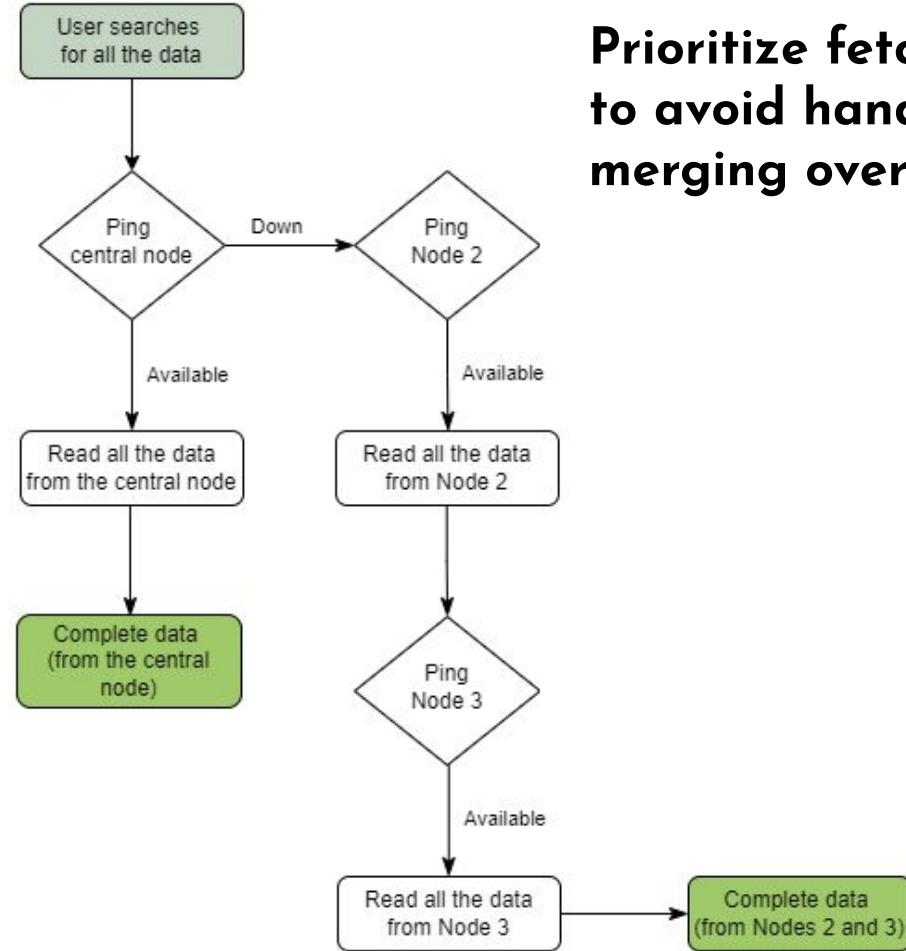
Web Application



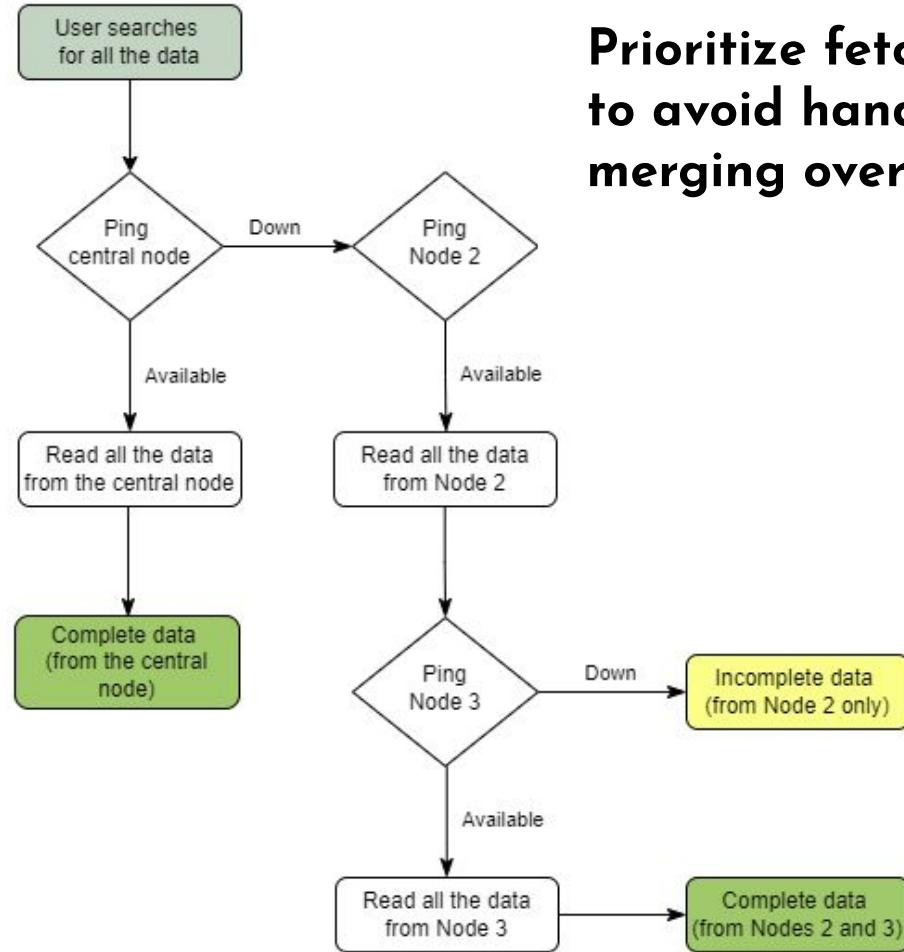
Retrieve All



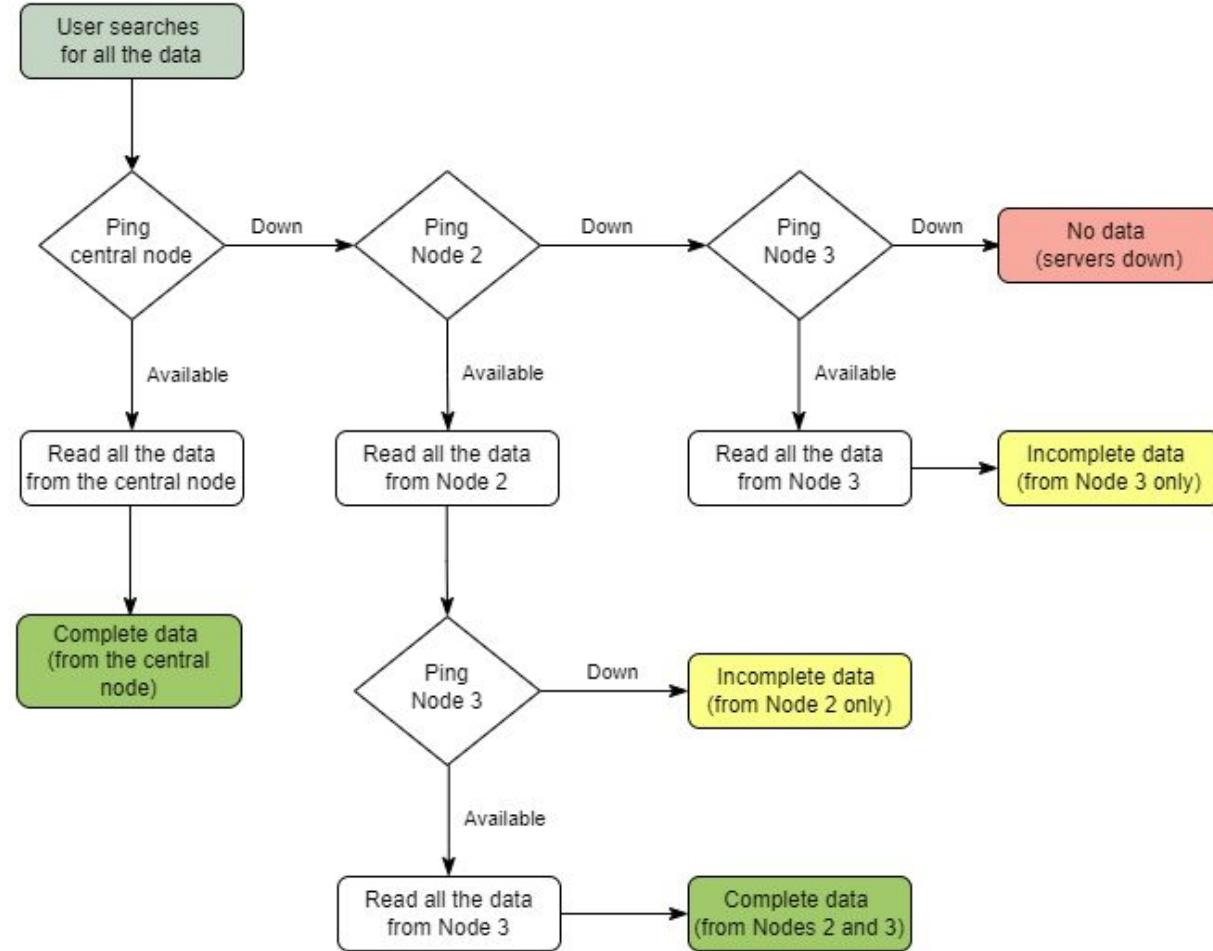
Prioritize fetching from central node to avoid handshaking and result set merging overhead



Prioritize fetching from central node to avoid handshaking and result set merging overhead



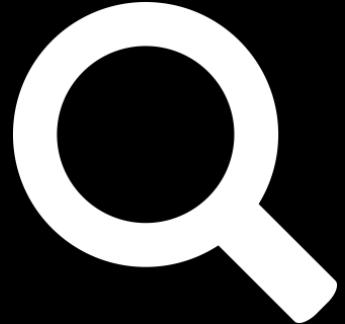
Prioritize fetching from central node to avoid handshaking and result set merging overhead



► ALGORITHM (RETRIEVE ALL)

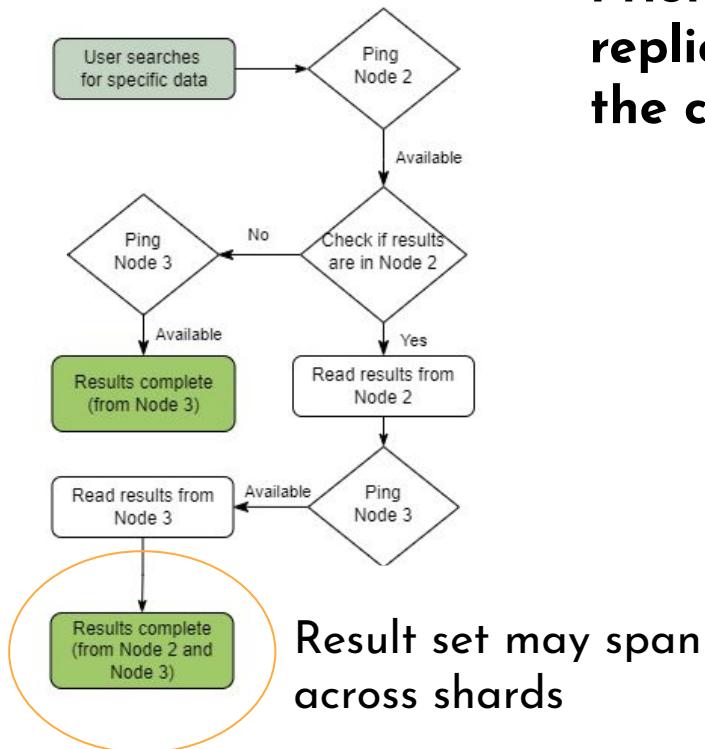


Web Application



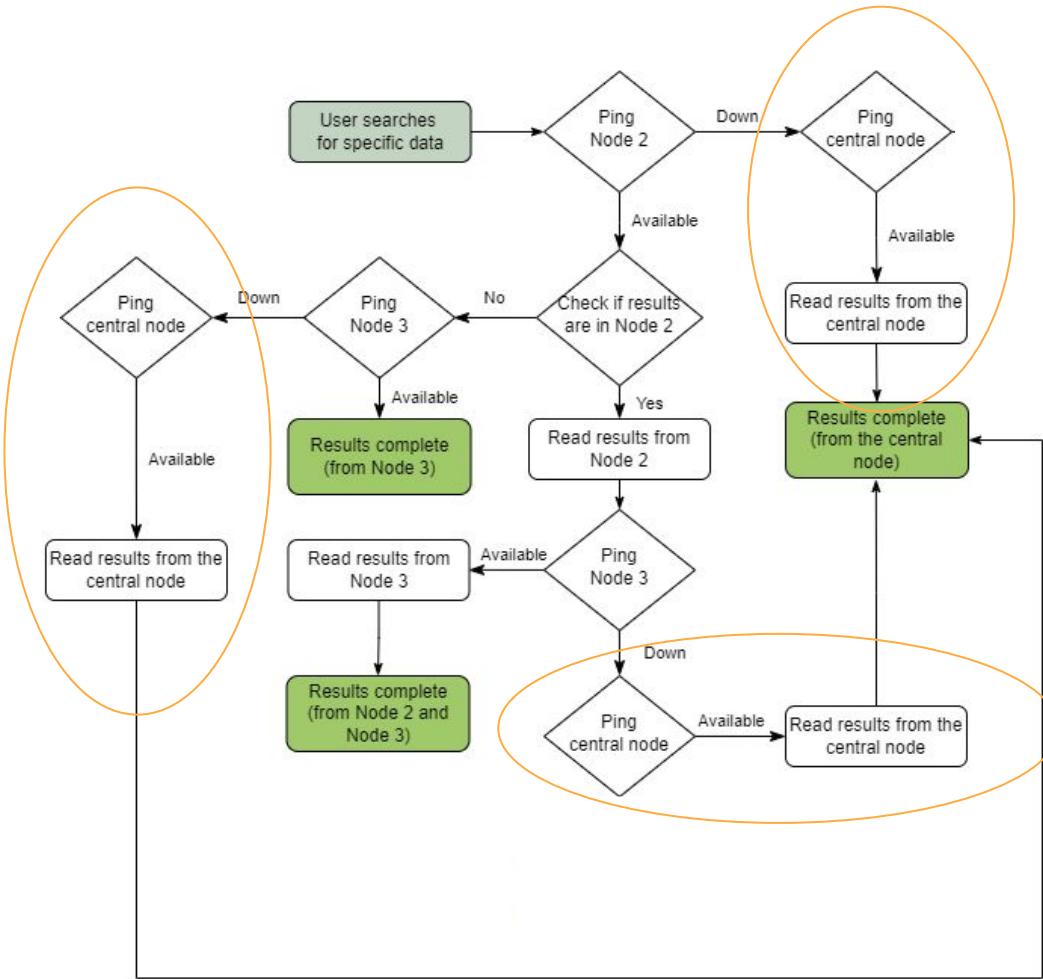
Retrieve All

Retrieve Based
on Search Criterion



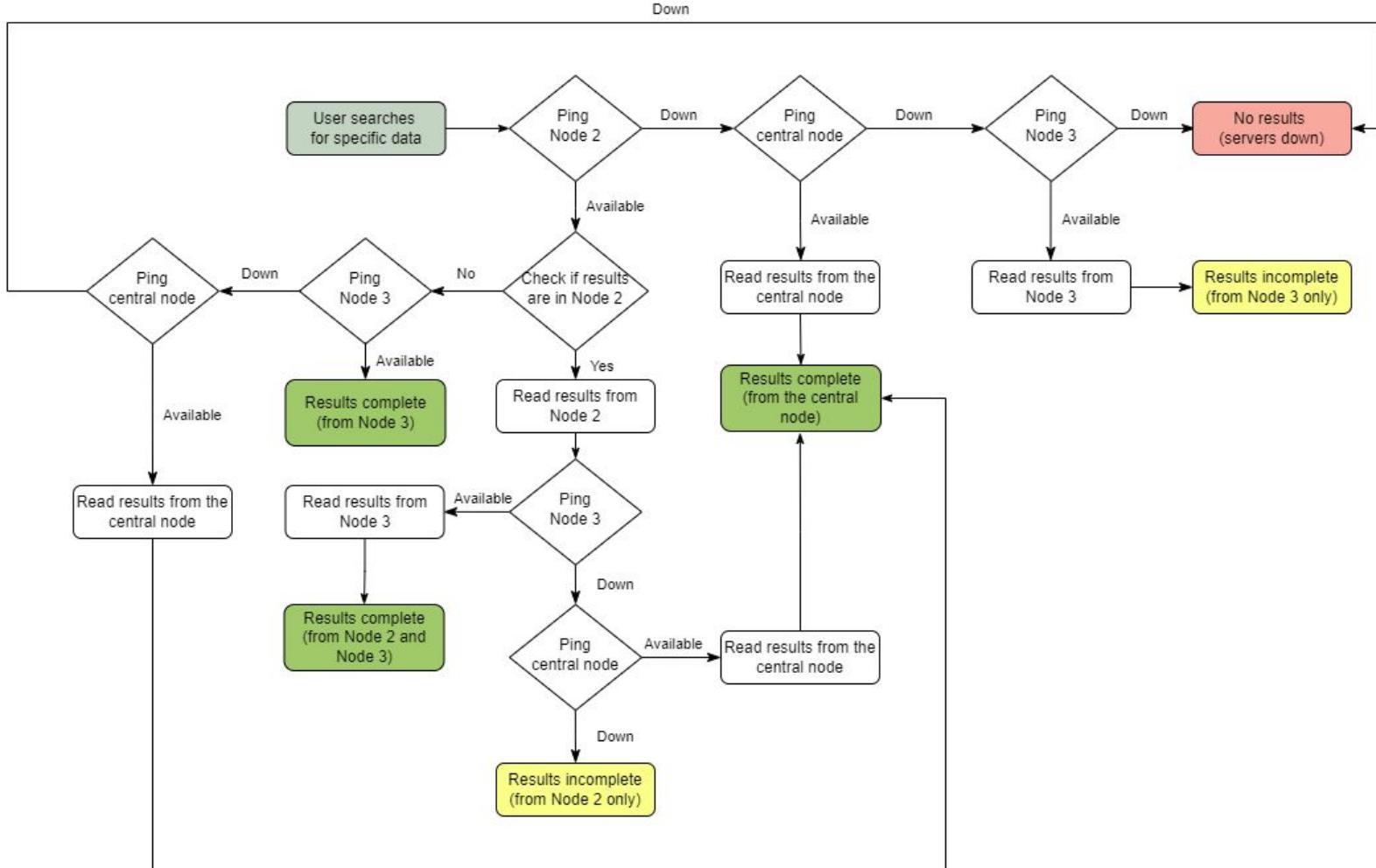
Prioritize fetching from partial replicas to avoid overloading the central node

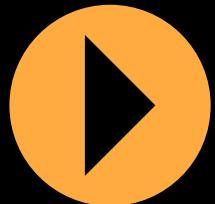
► ALGORITHM (RETRIEVE SPECIFIC)



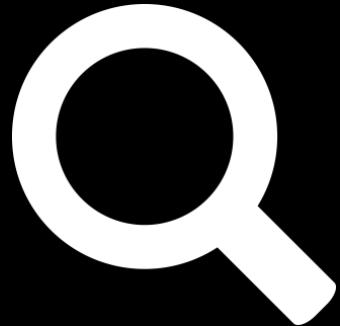
If one of the partial replicas is down, fetch from central

▶ ALGORITHM (RETRIEVE SPECIFIC)



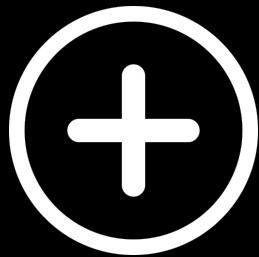


Web Application



Retrieve All

Retrieve Based
on Search Criterion



Insert



Update



Delete

ALGORITHM



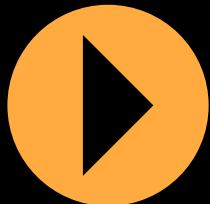
Log Table

Partly based on MySQL's GTID
logging technique

Attribute	Description
node_id	Numerical identifier of the node (1 for the central node, 2 for Node 2, and 3 for Node 3)
lock_status	Binary value indicating whether the node is locked (0 for not locked, 1 for locked)
next_trans_record	Identifier of the next transaction to be recorded in the log
next_trans_commit	Identifier of the next transaction to be committed
id_new_entry	ID (primary key) of the most recently inserted entry
statements	SQL queries executed (or to be executed) in the node, prefixed by a zero-based auto-incrementing identifier and delimited by three vertical pipes

ALGORITHM

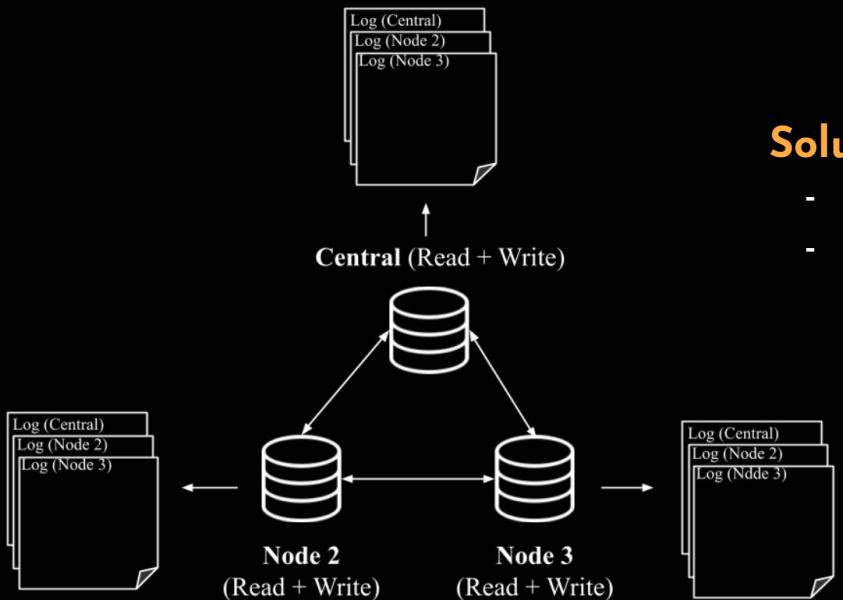




Log Table

Scenario

- Central node fails and the user inserts a movie entry before 1980
- Committed in Node 2 but not in central node
- Central node recovers, but Node 2 is now down
- **Problem:** Central node cannot refer to Node 2's logs to catch up

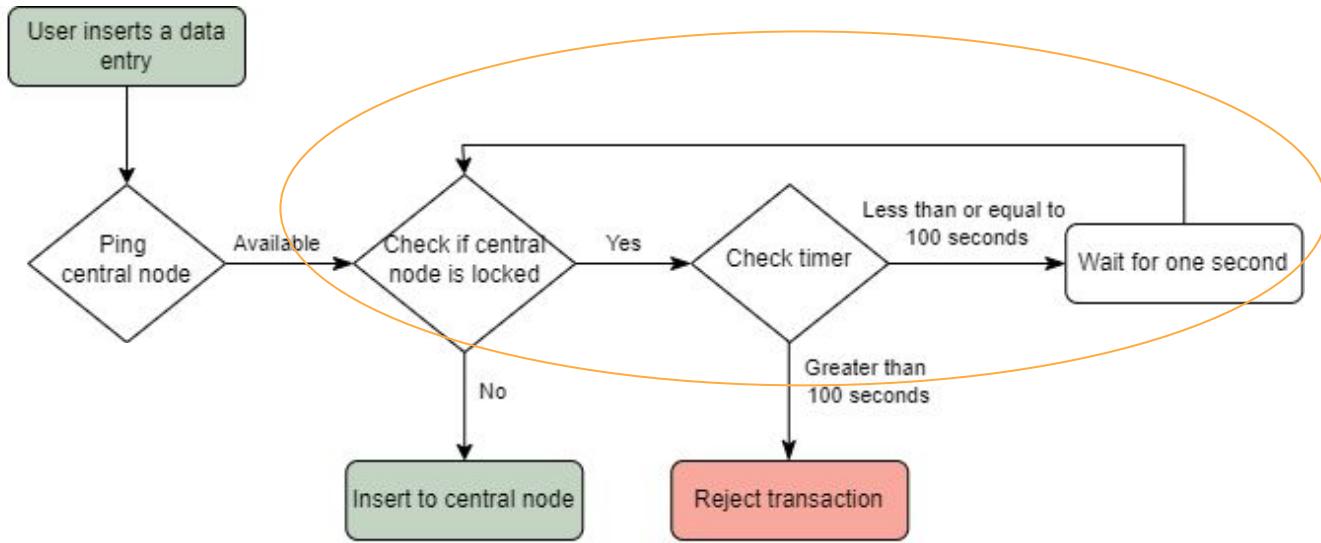


Solution

- Each node has a copy of all the nodes' logs
- Motivated by decentralized strategy of blockchain (Crosby et al., 2015)

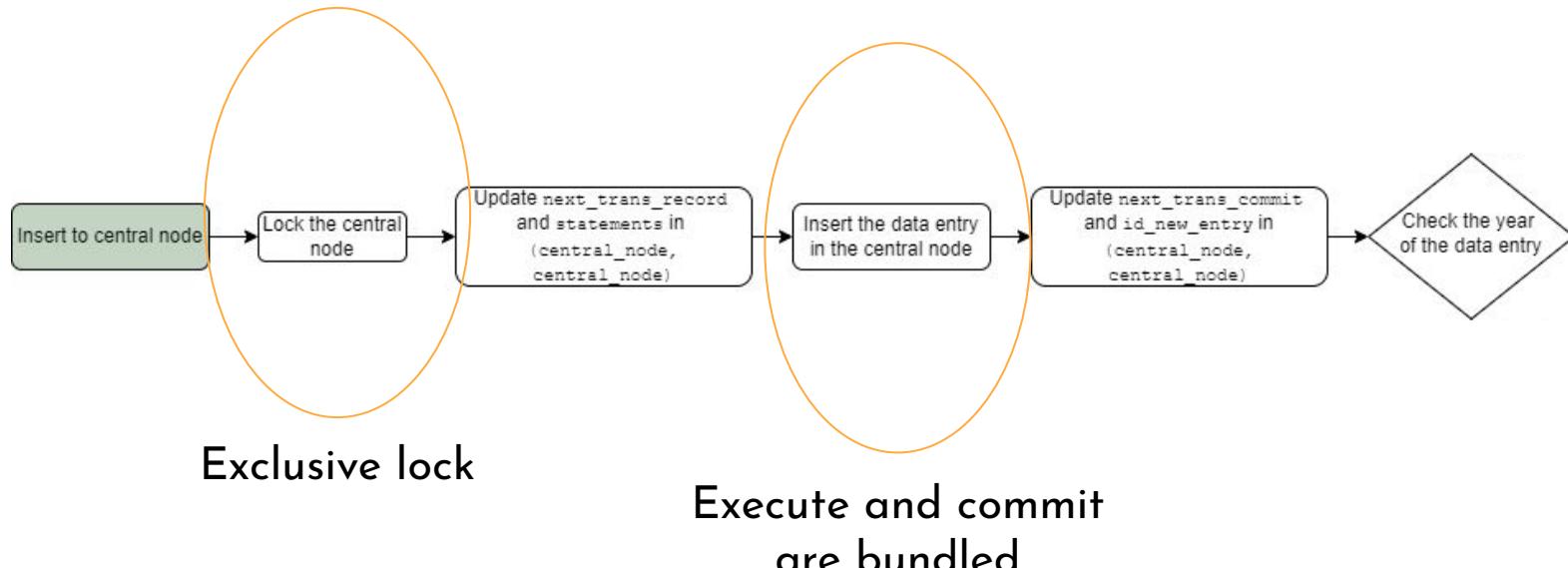
► ALGORITHM (INSERT)

Lock wait timeout



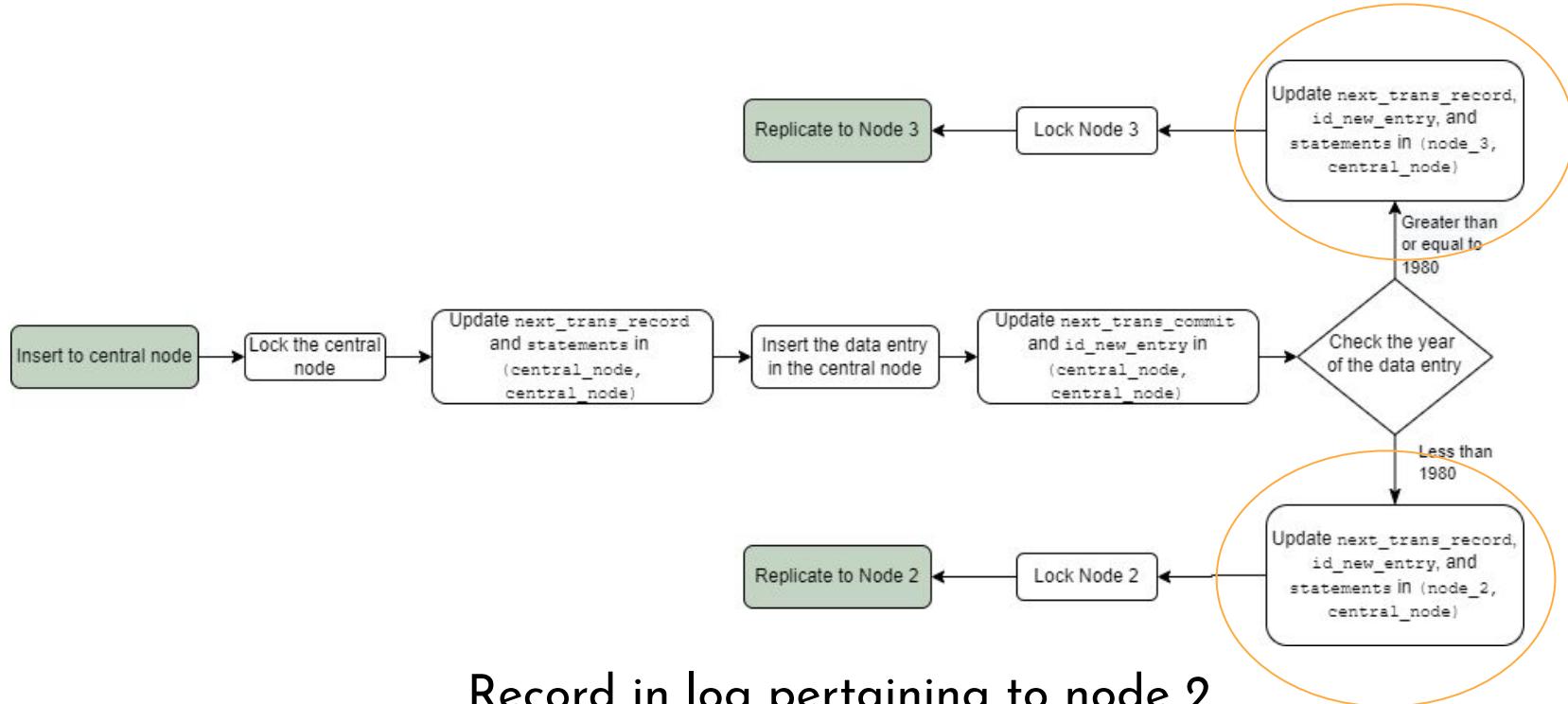
Prioritize inserting to the central node

For reference, MySQL follows an execute-log-commit pipeline



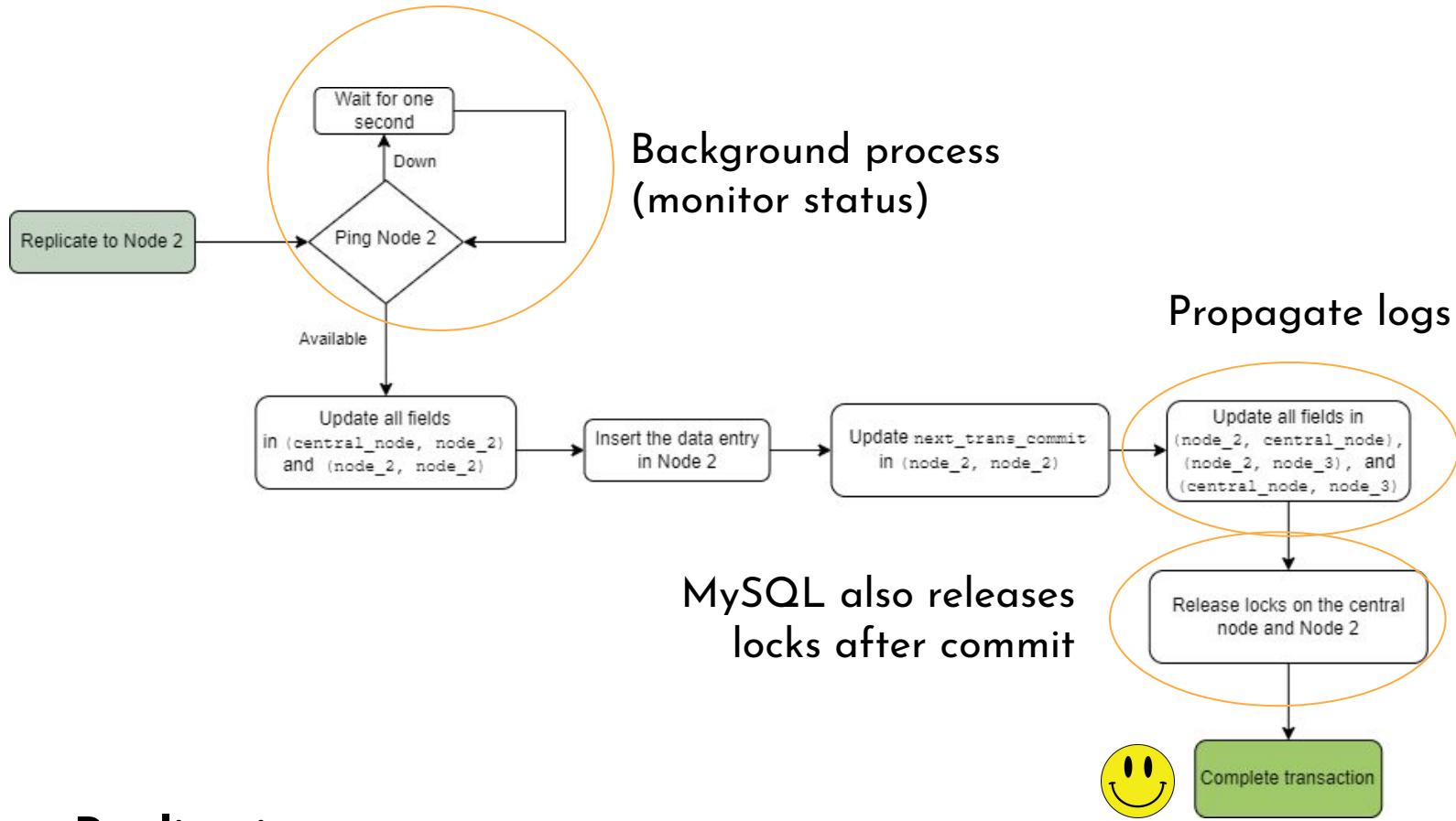
In our system, logging takes precedence

ALGORITHM (INSERT)

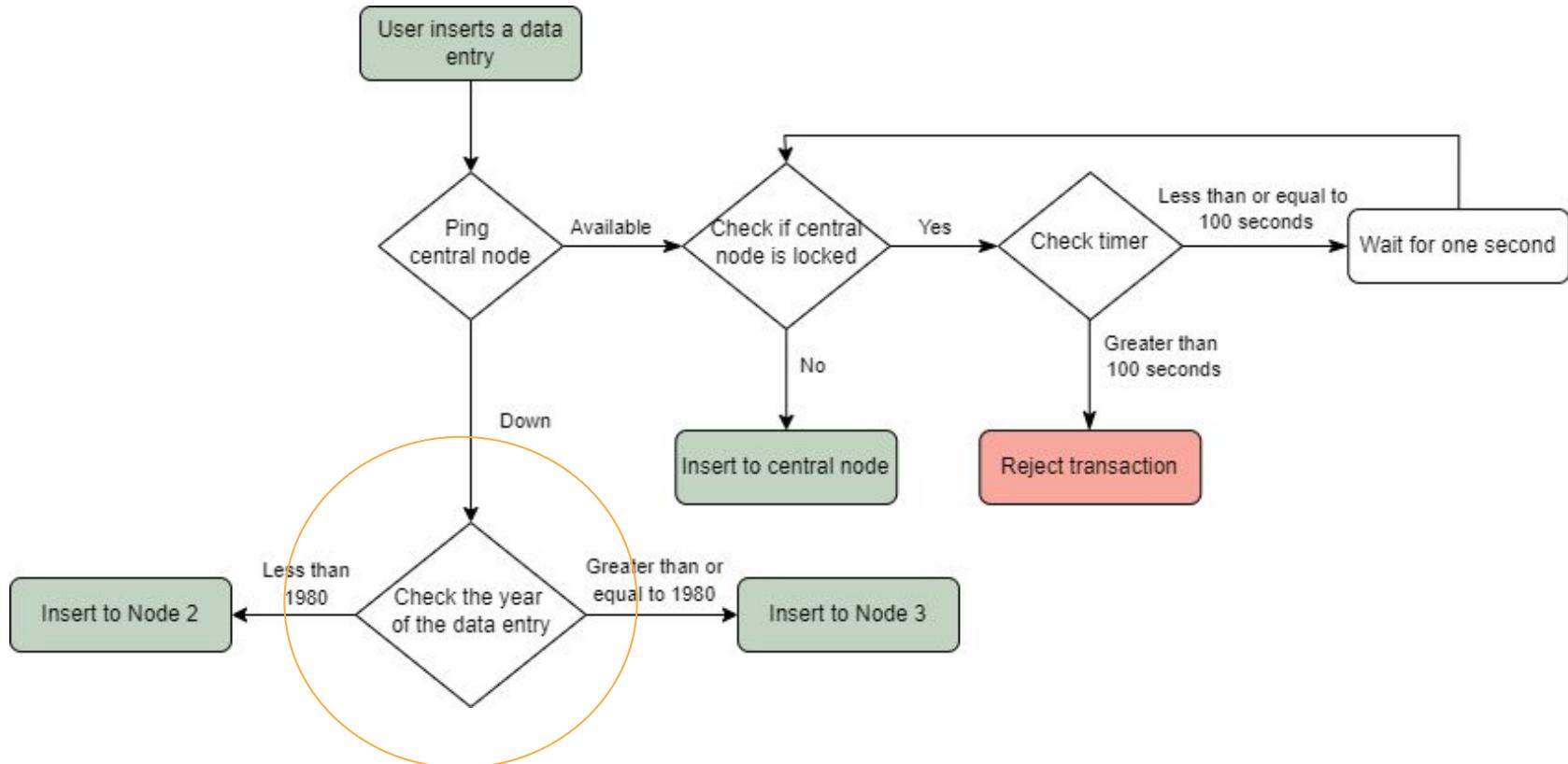


Record in log pertaining to node 2
(stored in the central node)

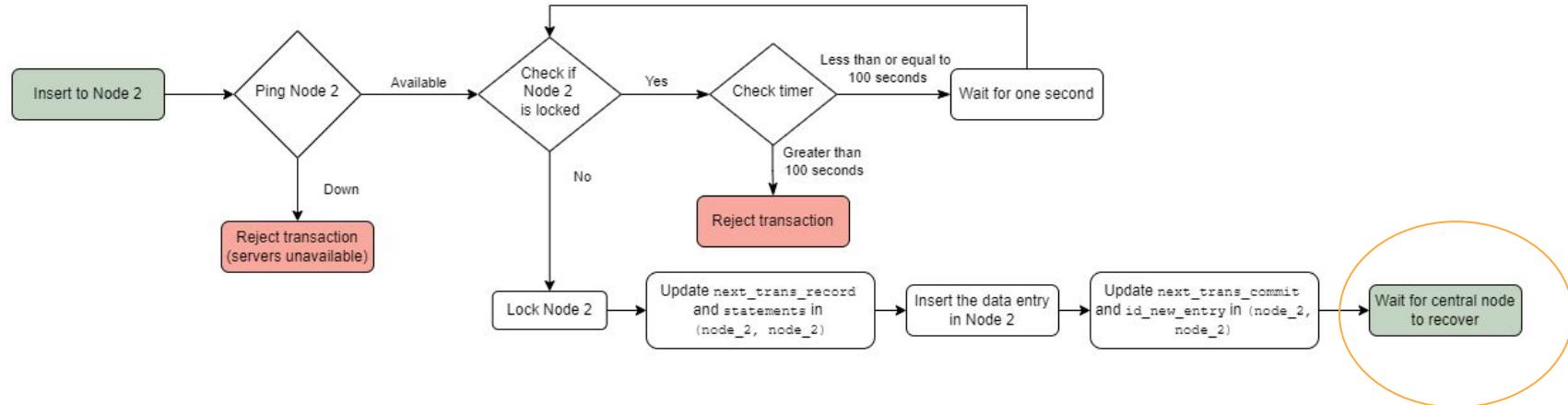
Replication



If central node is down, redirect write to replica

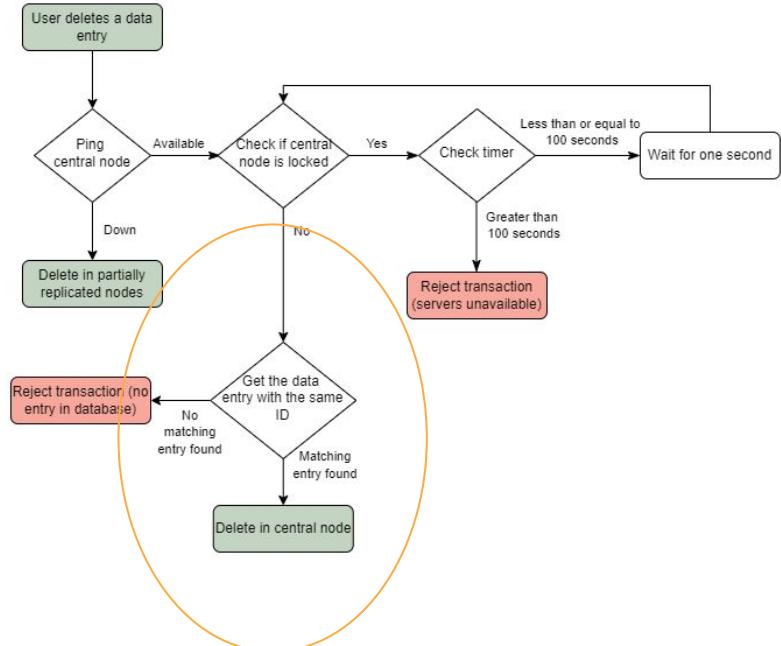
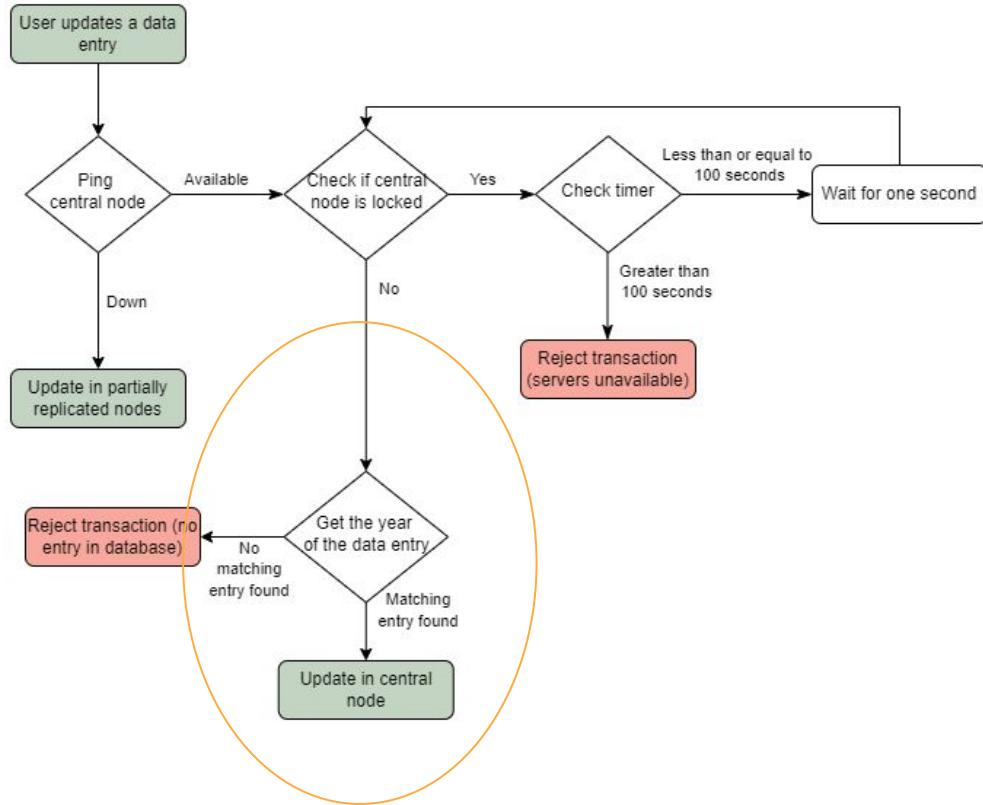


► ALGORITHM (INSERT)



Background
process

If central node is down, redirect write to replica



► ALGORITHM (UPDATE AND DELETE)

Challenges & Considerations



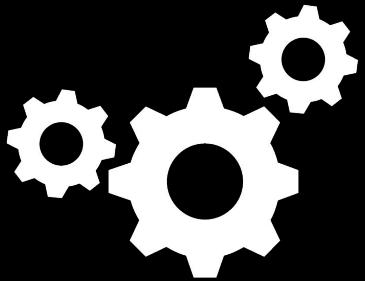
Callback Hell



String Parsing

Log files have to undergo string processing, especially since SQL queries have been parameterized to prevent SQL injection.

Challenges & Considerations



Background Process

```
function monitor() {  
  if (c != undefined)  
    c.ping(func(err) {  
      t = setTimeout(function(monitor, 1000);  
      if (!err)  
        connected();  
    }) ;  
  
  else  
    setTimeout(function(monitor, 1000));  
}
```



Global Concurrency

Case 1: All Transactions are Reading

User 1	User 2	User 3
<pre>START TRANSACTION; SELECT * FROM movies WHERE id = 174202;</pre>		
<pre>DO SLEEP(20); COMMIT;</pre>	<pre>START TRANSACTION; SELECT * FROM movies WHERE id = 174202; DO SLEEP(20); COMMIT;</pre>	<pre>START TRANSACTION; SELECT * FROM movies WHERE id = 174202; DO SLEEP(20); COMMIT;</pre>

CONCURRENCY





Global Concurrency

Case 1: All Transactions are Reading

Reads are **non locking**. They **do not request locks** as well.

Concurrent reads are **allowed**, and they **do not change** the database state.



Global Concurrency

Case 2: 1 Transaction is Writing, 2 Transactions are Reading



The application layer ensures that a transaction contains only a single SELECT statement.



This prevents the possibility of non-repeatable and phantom reads.



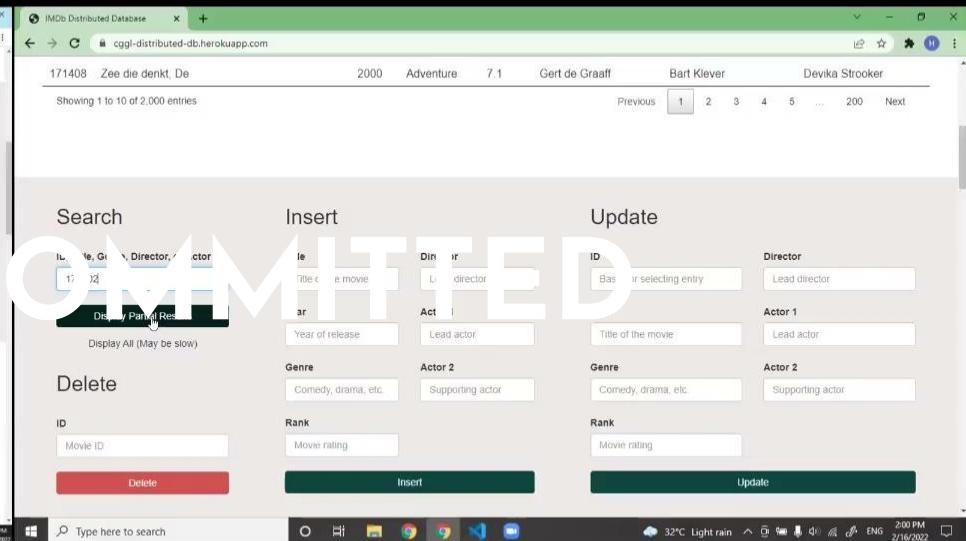
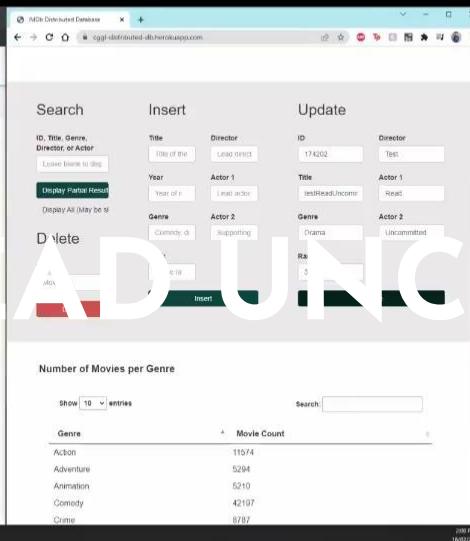
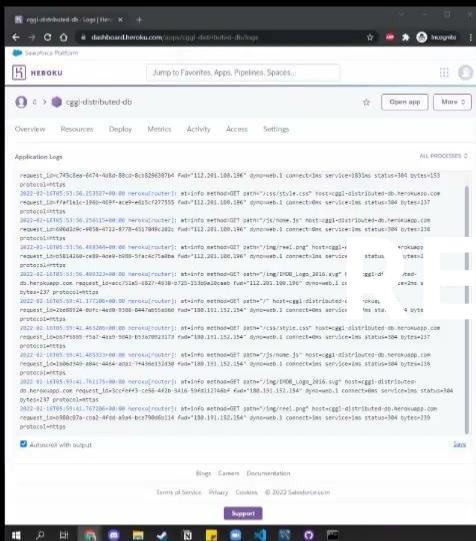


Global Concurrency

Case 2: 1 Transaction is Writing, 2 Transactions are Reading

User 1	User 2	User 3
<pre>START TRANSACTION; UPDATE movies SET title = 'testReadCommitted', genre = 'Drama', `rank` = 2.5, director = 'Test', actor1 = 'Read', actor2 = 'Committed' WHERE id = 174202;</pre>		
<pre>DO SLEEP(20);</pre>	<pre>START TRANSACTION; SELECT * FROM movies WHERE id = 174202; COMMIT;</pre>	<pre>START TRANSACTION; SELECT * FROM movies WHERE id = 174202; COMMIT;</pre>
<pre>COMMIT;</pre>		

▲ CONCURRENCY



The screenshot shows the Heroku dashboard for the 'cglib-distributed-dbs' application. Key details include:

- Overview:** CPU 1%, Memory 100MB, Dynos 0.
- Resources:** heroku-1 (PostgreSQL) and heroku-2 (PostgreSQL).
- Deploy:** Last deployment was successful at 2022-01-18T09:07:15Z.
- Metrics:** CPU usage is ~1% and ~100MB memory usage.
- Activity:** Recent activity includes a POST request to /api/v1/deployments.
- Access:** Deployment history from 2022-01-18T09:07:15Z to 2022-01-18T09:07:15Z.
- Settings:** Environment variables include PORT=3000, DB_HOST=heroku-1, DB_NAME=heroku-1, DB_USER=heroku-1, DB_PASSWORD=heroku-1, and DB_PORT=5432.
- Logs:** Logs show errors related to connection refused and port 5432 being closed.

IMDb Distributed Database										
Lander Peter E. Cus, Jacob Bryan B. Gaba, Mark Edward M. Gonzales, Hylenne G. Lee							IMDb			
STAUWDB Major Course Output 2, De La Salle University										
Show	10	▼ entries								
ID	Title	Year	Genre	Rank	Director	Actor 1	Actor 2			
171399	Zebrahead	1992	Drama	5.9	Anthony Draxton Marek Lozinski	Abdul Hassan Sharif Adriana Curreli	Nicola Wright Zeca Pagodinho			
	... nie bolito	1996	D		Max Piel Naroush Kurokawa					
	Zeca Pagodinho MTV	'93	D	1		Moenir Ezek				
02	Zecchino di Natale		D		Peter Greenaway	Agnieszka Holland				
171403	Così fan tutte	1985	Drama	7	Dimitre Osvanovski	Natalia (I) Stepanovic	Maria Teresa Staneska			
171404	Zeder	1983	Horror	5.8	Pupi Avati	Adriano Bellini	Snezana Staneska			
171405	Zedj	1971	Drama		Nikola (I) Stepanovic	Rados Bajic	Stivo Zigan			
171406	Zedj	1981	Short		Brian G. Hutton	Elizabeth (I) Taylor	Susan York			
171407	Zee and Co.	1972	Drama	4.3	Gert de Greef	Bart Kever	Dirkje Strooker			
171408	Die denkt, die	2000	Adventure	7.1						

IMDb Distributed Database

cggl-distributed-db.herokuapp.com

Search

ID, Title, Genre, Director, or Actor
174202

Insert

Title: Title of the movie
Director: Lead director
Year of release: Year of release
Lead actor: Lead actor
Genre: Comedy, drama, etc.
Actor 2: Supporting actor
Rank: Movie rating

Update

ID: Basis for selecting entry
Title: Title of the movie
Genre: Comedy, drama, etc.
Rank: Movie rating

Director

Actor 1: Lead actor
Actor 2: Supporting actor

Buttons

Delete

Insert

Update

Number of Movies per Genre

REPEATABLE READ

This screenshot shows the Heroku application interface. At the top, there's a navigation bar with links for Overview, Resources, Deploy, Metrics, Activity, Access, and Settings. Below this is a table titled "IMDb Distributed Database" showing movie details like title, year, genre, and cast. A search, insert, and update form is overlaid on the table. The update form includes fields for ID, Director, Actor 1, Actor 2, and Rank. The delete form has fields for ID and Rank. The bottom section displays a chart titled "Number of Movies per Genre" with a dropdown for entries and a search bar.

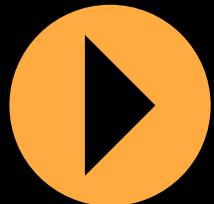
This screenshot shows the Heroku application interface. It features a table of movie data with columns for ID, Title, Year, Director, and Actor 1. An "Update" button is visible next to each row. Below the table is a search form with fields for ID, Title, Director, or Actor. It also includes "Display Partial Results" and "Display All (May be slow)" buttons. A "Delete" section with ID and Rank fields is also present.

This screenshot shows the Heroku application interface. It displays a table of movie data with columns for ID, Title, Year, Director, and Actor 1. The interface includes "Previous" and "Next" navigation buttons at the bottom. Below the table is a search form with fields for ID, Title, Director, or Actor. It also includes "Display Partial Results" and "Display All (May be slow)" buttons. A "Delete" section with ID and Rank fields is also present.

The screenshot shows a web browser window with the URL cggl-distributed-db.herokuapp.com. The page is divided into three main sections:

- Search:** A form for searching by ID, Title, Genre, Director, or Actor. It includes a text input field with value "174202" and a button labeled "Display Particulars".
- Insert:** A form for inserting movie data. It includes fields for Title, Director, Actor 1, Actor 2, Rank, and Genre. The Director field has "Lead director" selected. The Actor 1 field has "Lead actor" selected. The Actor 2 field has "Supporting actor" selected. The Rank field has "Movie rating" selected.
- Update:** A form for updating movie data. It includes fields for ID, Director, Title, Actor 1, Actor 2, Genre, Rank, and an "Update" button.

At the bottom, there is a chart titled "Number of Movies per Genre" and a system tray with various icons and status information.



Global Concurrency

Case 2: 1 Transaction is Writing, 2 Transactions are Reading

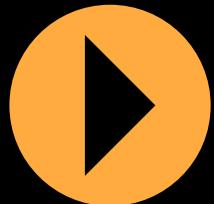
Concurrency is **supported**.

Dirty reads are blocked by using the **READ COMMITTED** isolation level.

The other read anomalies are prevented by the **design of the application layer**.



READ COMMITTED is also the default in major DBMS, such as Microsoft SQL Server (Microsoft, 2021) and PostgreSQL (PostgreSQL, n.d.).

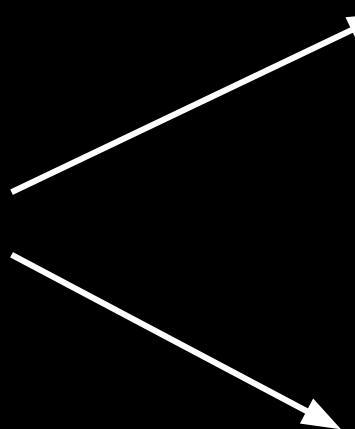


Global Concurrency

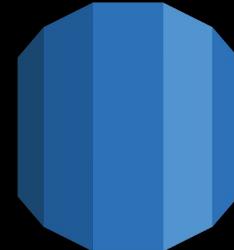
Case 2: 1 Transaction is Writing, 2 Transactions are Reading



Challenge:
Changing the isolation
level globally



Create a
parameter
group in
AWS RDS



Disable
autocommit
attribute





Global Concurrency

Case 3: All Transactions are Writing

User 1	User 2	User 3
<pre>START TRANSACTION; INSERT INTO movies (title, year, genre, `rank`, director, actor1, actor2) VALUES ('First entry', 1976, 'Comedy', 6.6, 'Cua', 'Gaba', 'Gonzales');</pre>		
<pre>DO SLEEP(20); COMMIT;</pre>	<pre>START TRANSACTION; INSERT INTO movies (title, year, genre, `rank`, director, actor1, actor2) VALUES ('Second entry', 1980, 'Drama', 7.7, 'Gaba', 'Gonzales', 'Lee'); DO SLEEP(20); COMMIT;</pre>	<pre>START TRANSACTION; INSERT INTO movies (title, year, genre, `rank`, director, actor1, actor2) VALUES ('Third entry', 1985, 'Action', 8.8, 'Gonzales', 'Lee', 'Cua'); DO SLEEP(20); COMMIT;</pre>

► CONCURRENCY



Global Concurrency

Case 3: All Transactions are Writing

User 1	User 2	User 3
<pre>START TRANSACTION; UPDATE movies SET title = 'First Edit' genre = 'Comedy' `rank` = 6.6 director = 'Gaba' actor1 = 'Gonzales' actor2 = 'Lee'; WHERE id = 174202;</pre>		
<pre>DO SLEEP(20); COMMIT;</pre>	<pre>START TRANSACTION; UPDATE movies SET title = 'Second Edit' genre = 'Comedy' `rank` = 6.6 director = 'Gaba' actor1 = 'Gonzales' actor2 = 'Lee'; WHERE id = 174202; DO SLEEP(20); COMMIT;</pre>	<pre>START TRANSACTION; UPDATE movies SET title = 'Third Edit' genre = 'Comedy' `rank` = 6.6 director = 'Gaba' actor1 = 'Gonzales' actor2 = 'Lee'; WHERE id = 174202; DO SLEEP(20); COMMIT;</pre>

▲ CONCURRENCY

UPDATING

Search

ID, Title, Genre, Director, or Actor

Leave blank to display all entries

Display	Partial Result
Display	(May be w)

Insert

Title	Director	Up	te
<input type="text" value="Title of the movie"/>	<input type="text" value="Lead director"/>	<input type="button" value="Insert"/>	<input type="button" value="Update"/>
<input type="text" value="Year"/>	<input type="text" value="Actor 1"/>	<input type="button" value="ID"/>	<input type="button" value="Dir"/>
<input type="text" value="Genre"/>	<input type="text" value="Actor 2"/>	<input type="button" value="First Edit"/>	<input type="button" value="Edit"/>
<input type="text" value="Comedy, drama, etc."/>	<input type="text" value="Genre"/>	<input type="button" value="Actor 1"/>	<input type="button" value="Actor 2"/>
<input type="text" value="Movie ID"/>	<input type="text" value="Rank"/>	<input type="button" value="Delete"/>	<input type="button" value="Insert"/>

Delete

ID	Rank
<input type="text" value="Movie ID"/>	<input type="text" value="Rank"/>

Update

ID	Title	Director
<input type="text" value="174202"/>	<input type="text" value="Title of the movie"/>	<input type="text" value="Lead director"/>
<input type="text" value="Title"/>	<input type="text" value="Year of release"/>	<input type="text" value="Actor 1"/>
<input type="text" value="Actor 1"/>	<input type="text" value="Actor 2"/>	<input type="text" value="Genre"/>
<input type="text" value="Actor 2"/>	<input type="text" value="Supporting actor"/>	<input type="text" value="Comedy, drama, etc."/>
<input type="text" value="Movie rating"/>	<input type="text" value="Rank"/>	<input type="text" value="Movie ID"/>

Insert

ID, Title, Genre, Director, or Actor

Leave blank to display all entries

Display	Partial Result
Display	(May be w)

Insert

Title	Director	Up	te
<input type="text" value="Title of the movie"/>	<input type="text" value="Lead director"/>	<input type="button" value="Insert"/>	<input type="button" value="Update"/>
<input type="text" value="Year"/>	<input type="text" value="Actor 1"/>	<input type="button" value="ID"/>	<input type="button" value="Dir"/>
<input type="text" value="Genre"/>	<input type="text" value="Actor 2"/>	<input type="button" value="First Edit"/>	<input type="button" value="Edit"/>
<input type="text" value="Comedy, drama, etc."/>	<input type="text" value="Genre"/>	<input type="button" value="Actor 1"/>	<input type="button" value="Actor 2"/>
<input type="text" value="Movie ID"/>	<input type="text" value="Rank"/>	<input type="button" value="Delete"/>	<input type="button" value="Insert"/>

Delete

ID	Rank
<input type="text" value="Movie ID"/>	<input type="text" value="Rank"/>

Update

ID	Title	Director
<input type="text" value="174202"/>	<input type="text" value="Title of the movie"/>	<input type="text" value="Lead director"/>
<input type="text" value="Title"/>	<input type="text" value="Year of release"/>	<input type="text" value="Actor 1"/>
<input type="text" value="Actor 1"/>	<input type="text" value="Actor 2"/>	<input type="text" value="Genre"/>
<input type="text" value="Actor 2"/>	<input type="text" value="Supporting actor"/>	<input type="text" value="Comedy, drama, etc."/>
<input type="text" value="Movie rating"/>	<input type="text" value="Rank"/>	<input type="text" value="Movie ID"/>

Update

ID, Title, Genre, Director, or Actor

Leave blank to display all entries

Display	Partial Result
Display	(May be w)

Update

ID	Title	Director
<input type="text" value="174202"/>	<input type="text" value="Title of the movie"/>	<input type="text" value="Lead director"/>
<input type="text" value="Title"/>	<input type="text" value="Year of release"/>	<input type="text" value="Actor 1"/>
<input type="text" value="Actor 1"/>	<input type="text" value="Actor 2"/>	<input type="text" value="Genre"/>
<input type="text" value="Actor 2"/>	<input type="text" value="Supporting actor"/>	<input type="text" value="Comedy, drama, etc."/>
<input type="text" value="Movie rating"/>	<input type="text" value="Rank"/>	<input type="text" value="Movie ID"/>

IMDb Distributed Database

http://csg-distributed-db.herokuapp.com

Serving 1 to 10 of 2,000 entries

Prev 1 2 3 4 5 ... 200 Next

Search

ID, Title, Genre, Director, or Actor
Leave blank to display all entries

Display Partial Results

Display All (May be slow)

elete

Movie ID:
Delete

Insert

Title: Director:
Year: Actor 1:
Year of release: Lead actor:
Genre: Actor 2:
CC: C: Drama:
Rank: Movie Rating:
Insert

Update

ID: Director:
Title: Actor 1:
Actor 2: Director:
Genre: Drama:
Rank: Timeout:
Update

Number of Movies per Genre

Show 10 entries

Genre: Movie Count:
Search:

IMDb Distributed Database

Not secure | cgg1-distributed-db.herokuapp.com

Search

ID, Title, Genre, Director, or Actor
Leave blank to display all entries

Display results
Display All (may be slow)

Insert

Title: Title of the movie
Director: Lead director
Genre: Yes
of re --
Lead actor
Genre: Comedy, drama, etc.
Supporting actor

Update

ID: 174202
Director: Lee
Actor 1: Gonzales
Actor 2: Gabe

Rank: 5.6

Update

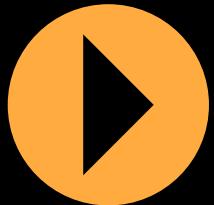


Global Concurrency

Case 3: All Transactions are Writing

Write transactions are handled via **locking**.

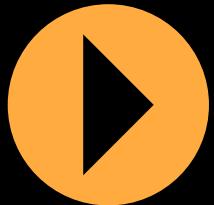
A lock wait timeout mechanism was implemented to prevent **lock contention issues**.



Global Recovery

Recovery occurs by establishing a new connection that is directed towards the crashed node.

For the experiments, three down-time windows were tested:
5, 60, and 300 seconds.

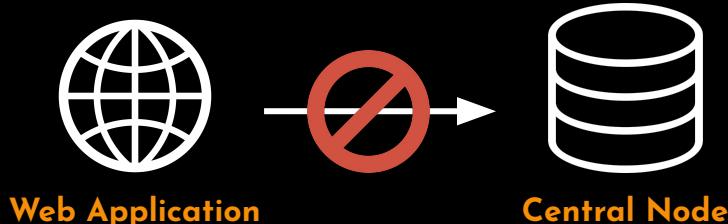


Global Recovery

SIMULATING CASE 1

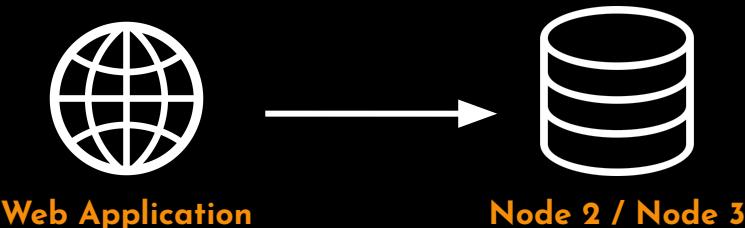
- Force Central Node crash **before transaction** to route algorithm to Node 2 / 3
- Central Node is crashed again during **replication** from Node 2 / 3

1. Initial Transaction

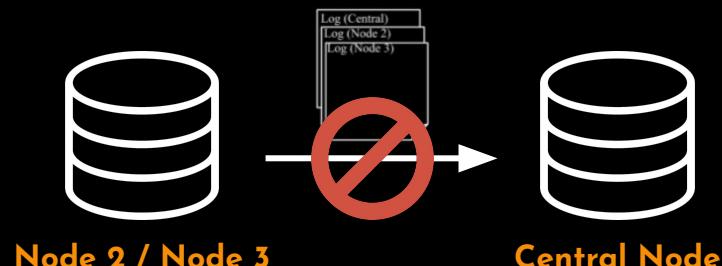


Case 1: Failure in replicating from Node 2 or Node 3 to Central Node

2. Redirection



3. Replication



cggl-distributed-db - Logs | Hero

dashboard.herokuapp.com/apps/cggl-distributed-db/logs

Incognito

Salesforce Platform

HEROKU

Jump to Favorites, Apps, Pipelines, Spaces...

Overview Resources Deploy Metrics Activity Access Settings

Application Logs ALL PROCESSES

```

2022-02-16T19:13:26.180417+00:00 app[web.1]: Commit successful!
2022-02-16T19:13:26.181435+00:00 app[web.1]: Executing query to update next transaction commit count for central node log file in node 3
2022-02-16T19:13:26.181435+00:00 app[web.1]: Commit successful!
2022-02-16T19:13:26.189649+00:00 app[web.1]: Executing query to update next transaction commit count for node 2 log file in node 3
2022-02-16T19:13:26.194886+00:00 app[web.1]: Commit successful!
2022-02-16T19:13:26.196115+00:00 app[web.1]: Executing query to unlock central node
2022-02-16T19:13:26.204659+00:00 app[web.1]: Unlock committed
2022-02-16T19:13:40.563725+00:00 heroku[router]: at=info method=GET path="/searchEntry?searchCriteria=1&isolationLevel=READONLYCOMMITTED" host=cggl-distributed-db.herokuapp.com request_id=5bf6d68f... 38bd-4d9c-bf... 54e96a4657cb fwd="112.201.100.196" dyno=web.1 connect=0ms service=422ms status=200 bytes=335 protocol=https
2022-02-16T19:13:40.565605+00:00 app[web.1]: Results complete -> from node 2 + node 3
2022-02-16T19:14:49.998817+00:00 heroku[router]: at=info method=GET path="/js/home.js" host=cggl-distributed-db.herokuapp.com request_id=6fe6f26-b95a-e442-a155-1742de2ec395 fwd="112.201.100.196" dyno=web.1 connect=0ms service=1ms status=304 bytes=238 protocol=https
2022-02-16T19:14:49.237550+00:00 heroku[router]: at=info method=GET path="/img/reel.png" host=cggl-distributed-db.herokuapp.com request_id=d338e5ae-97b2-4acb-b53d-935ead0187bf fwd="112.201.100.196" dyno=web.1 connect=0ms service=1ms status=304 bytes=239 protocol=https
2022-02-16T19:14:49.230134+00:00 heroku[router]: at=info method=GET path="/img/IMDB_Logo_2016.svg" host=cggl-distributed-db.herokuapp.com request_id=18c886d6-8f35-4df4-80a7-a5b7d4c75bd9 fwd="112.201.100.196" dyno=web.1 connect=0ms service=304 bytes=237 protocol=https
2022-02-16T19:14:49.741748+00:00 heroku[router]: at=info method=GET path="/" host=cggl-distributed-db.herokuapp.com request_id=1ec4a1e9-6c9f-409a-8d42-0d36dc17e295 fwd="112.201.100.196" dyno=web.1 connect=0ms service=1597ms status=200 bytes=830830 protocol=https
2022-02-16T19:14:49.001845+00:00 heroku[router]: at=info method=GET path="/css/style.css" host=cggl-distributed-db.herokuapp.com request_id=d6079431-ce23-4fe7-9ab9-8d53e86f605d fwd="112.201.100.196" dyno=web.1 connect=0ms service=1ms status=304 bytes=237 protocol=https

```

Autoscroll with output

Save

Blogs Careers Documentation

Terms of Service Privacy Cookies © 2022 Salesforce.com

Support

Type here to search

IMDb Distributed Database

cggl-distributed-db.herokuapp.com

Apps M Gmail F Twitt... YouTube N Google... D A MC Pipol - Google... Reading list

Technic... raw vid... IMDb D... cggl-di... Lounge... STADV... Task List deleteC... GitHub... Zoom Meetin... Sticky ... 26°C ENG 3:15 AM 17/02/2022

Search	Insert	Update
ID, Title, Genre, Director, or Actor Leave blank to disp:	Title Title of the	Director Lead direct
Case 3 (Central Node)	Year Year of r	Actor 1 Lead actor
Case 4a (Node 2 Una	Genre Comedy, di	Actor 2 Supporting
Case 4b (Node 3 Una	Rank Movie ra	Genre Drama
Extra (Central Node +	Rank 5.5	Actor 2 1a
Extra (Central Node +	Case 1 (Failure Replicating to Ce	Case 1 (Failure While Replicating to Central N
Extra (Node 2 + Node	Case 2a (Failure Replicating to N	Case 2a (Failure While Replicating to Node 2
Extra (Node 3 + Node	Case 2b (Failure Replicating to N	Case 2b (Failure While Replicating to Node 3
Delete	Case 3 (Central Node Unavailable	Case 3 (Central Node Unavailable)
ID Movie ID	Case 4a (Node 2 Unavailable)	Case 4a (Node 2 Unavailable)
Case 1 (Failure Rep	Case 4b (Node 3 Unavailable)	Case 4b (Node 3 Unavailable)
Case 2a (Failure Rep		
Case 2b (Failure Rep		
Case 3 (Central Node		
Case 4a (Node 2 Una		

cggl-distributed-db - Logs | Hero x +

dashboard.herokuapp.com/apps/cggl-distributed-db/logs

Salesforce Platform

HEROKU Jump to Favorites. Apps, Pipelines, Spaces...

cggl-distributed-db

Open app More

Overview Resources Deploy Metrics Activity Access Settings

Application Logs ALL PROCESSES

```
protocol=https
2022-02-16T19:15:21.254593+00:00 app[web.1]: connected
2022-02-16T19:15:21.256490+00:00 app[web.1]: Extracting log files from central node
2022-02-16T19:15:21.272212+00:00 app[web.1]: Executing query for central node update of central log file
2022-02-16T19:15:21.273318+00:00 app[web.1]: Committing changes; central node update for central log file
2022-02-16T19:15:21.277756+00:00 app[web.1]: Commit success
2022-02-16T19:15:21.278428+00:00 app[web.1]: Executing query for central node update of node 2 log file
2022-02-16T19:15:21.283621+00:00 app[web.1]: Commit success
2022-02-16T19:15:21.283647+00:00 app[web.1]: Beginning statement extraction from node 2
2022-02-16T19:15:21.283699+00:00 app[web.1]: Executing query for central node update of movies table
2022-02-16T19:15:21.288776+00:00 app[web.1]: Commit successful!
2022-02-16T19:15:21.289476+00:00 app[web.1]: Executing query to update next transaction commit count of node id 1
file
2022-02-16T19:15:21.294743+00:00 app[web.1]: Commit successful!
2022-02-16T19:15:21.295706+00:00 app[web.1]: Executing query to update next transaction commit count of node id 1 in node 2 log file
2022-02-16T19:15:21.302388+00:00 app[web.1]: Commit successful!
2022-02-16T19:15:21.303367+00:00 app[web.1]: Executing query to update next transaction commit count for central node log file in node 3
2022-02-16T19:15:21.310970+00:00 app[web.1]: Commit successful!
2022-02-16T19:15:21.312450+00:00 app[web.1]: Executing query to update next transaction commit count for node 2 log file in node 3
2022-02-16T19:15:21.319779+00:00 app[web.1]: Commit successful!
2022-02-16T19:15:21.328808+00:00 app[web.1]: Executing query to unlock central node
2022-02-16T19:15:21.329227+00:00 app[web.1]: Unlock committed
2022-02-16T19:15:31.156032+00:00 app[web.1]: Results complete -> from node 2 + node 3
2022-02-16T19:15:31.157836+00:00 heroku[router]: at=info method=GET path="/searchEntry?searchCriteria=t174202&solutionLevel=READ20COMMITTED" host=cggl-distributed-db.herokuapp.com request_id=5b319ff6-d019-4ac4-8ba7-b5ddaa06565d fwd="112.201.100.196" dyno=web.1 connect=0ms service=326ms status=200 bytes=335 protocol=https

 Autoscroll with output
```

Blogs Careers Documentation

Terms of Service Privacy Cookies © 2022 Salesforce.com

Support

Incognito

Type here to search

The screenshot shows a web-based application for managing a distributed database, likely for the IMDb dataset. The interface is divided into several sections:

- Search:** A section for querying the database using various filters like ID, Title, Genre, Director, or Actor. It includes a note to "Leave blank to disp." and a list of test cases:
 - Case 3 (Central Node)
 - Case 4a (Node 2 Unavailable)
 - Case 4b (Node 3 Unavailable)
 - Extra (Central Node +)
 - Extra (Node 2 + Node 3)
- Insert:** A section for adding new data records. It includes fields for Title, Director, Year, Actor 1, Actor 2, Genre, Rank, and Movie ID. It lists test cases:
 - Case 1 (Failure Replicating to Central Node)
 - Case 2a (Failure Replicating to Node 2)
 - Case 2b (Failure Replicating to Node 3)
 - Case 3 (Central Node Unavailable)
 - Case 4a (Node 2 Unavailable)
 - Case 4b (Node 3 Unavailable)
- Update:** A section for modifying existing data. It includes fields for ID, Director, Title, Actor 1, Actor 2, Genre, and Rank. It lists test cases:
 - Case 1 (Failure While Replicating to Central Node)
 - Case 2a (Failure While Replicating to Node 2)
 - Case 2b (Failure While Replicating to Node 3)
 - Case 3 (Central Node Unavailable)
 - Case 4a (Node 2 Unavailable)
 - Case 4b (Node 3 Unavailable)
- Delete:** A section for removing data. It includes a field for ID (Movie ID) and lists test cases:
 - Case 1 (Failure Replicating to Central Node)
 - Case 2a (Failure Replicating to Node 2)
 - Case 2b (Failure Replicating to Node 3)

The application is running on a Heroku server, as indicated by the URL in the browser bar: `cggl-distributed-db.herokuapp.com`. The browser interface shows various tabs and extensions.



Global Recovery

Case 2: Failure in replicating from Central Node to Node 2 or Node 3

SIMULATING CASE 2

- Crash Node 2 / 3 during replication by destroying connection to Web App

1. Initial Transaction



Web Application



Central Node

2. Replication



Central Node



Node 2 / Node 3

CASE 2A

CASE 2B

cggl-distributed-db · Logs | Herox +

dashboard.herokuapp.com/apps/cggl-distributed-db/logs

Incognito

Reading list

Salesforce Platform

HEROKU

Jump to Favorites: Apps, Pipelines, Spaces...

Open app More

Overview Resources Deploy Metrics Activity Access Settings

Application Logs ALL PROCESSES 0

```
2022-02-16T19:22:11.372680+00:00 app[web.1]: connected
2022-02-16T19:22:11.374446+00:00 app[web.1]: Extracting log files from central node
2022-02-16T19:22:11.386334+00:00 app[web.1]: Node 3 available!
2022-02-16T19:22:11.396169+00:00 app[web.1]: Executing query for node 2 update of central log file
2022-02-16T19:22:11.399565+00:00 app[web.1]: Committing node 2 update for central log file
2022-02-16T19:22:11.402954+00:00 app[web.1]: Commit success
2022-02-16T19:22:11.404049+00:00 app[web.1]: Committing node 2 update for node 2 log file
2022-02-16T19:22:11.414815+00:00 app[web.1]: Commit success
2022-02-16T19:22:11.416853+00:00 app[web.1]: Beginning statement extraction from node 2
2022-02-16T19:22:11.539927+00:00 app[web.1]: Update successful!
2022-02-16T19:22:11.543398+00:00 app[web.1]: Commit successful!
2022-02-16T19:22:11.546751+00:00 app[web.1]: Executing query to update next transaction commit count of node id 1
2022-02-16T19:22:11.552931+00:00 app[web.1]: Commit successful!
2022-02-16T19:22:11.554697+00:00 app[web.1]: Executing query to update next transaction commit count of node id 2 in central node log file
2022-02-16T19:22:11.560414+00:00 app[web.1]: Commit successful!
2022-02-16T19:22:11.562453+00:00 app[web.1]: Executing query to update next transaction commit count for central node log file in node 3
2022-02-16T19:22:11.568722+00:00 app[web.1]: Commit successful!
2022-02-16T19:22:11.569821+00:00 app[web.1]: Executing query to update next transaction commit count for node 2 log file in node 3
2022-02-16T19:22:11.576230+00:00 app[web.1]: Commit successful!
2022-02-16T19:22:11.579252+00:00 app[web.1]: Executing query to unlock central node
2022-02-16T19:22:11.586113+00:00 app[web.1]: Unlock committed
2022-02-16T19:22:21.279484+00:00 app[web.1]: Results complete -> from node 2 + node 3
2022-02-16T19:22:21.280770+00:00 heroku[router]: at:info method=GET path="/searchEntry?searchCriteria=17420&isolationLevel=READ20COMMITTED" host=cggl-distributed-db.herokuapp.com request_id=3299d642-b51b-496e-9fc6-30a568e2cce4 fwd="112.201.180.196" dyno=web.1 connect=0ms service=342ms status=200 bytes=335 protocol=https
```

Autoscroll with output

Save

Blogs Careers Documentation

Terms of Service Privacy Cookies © 2022 Salesforce.com

Support

IMDb Distributed Database

cggl-distributed-db.herokuapp.com

Apps M Facebook Twitter LinkedIn YouTube Netflix D A MC Pipol - Google... Reading list

Task List deleteC... GitHub... Zoom Meetin... Sticky... 26°C ENG 3:21 AM 17/02/2022

Search Insert Update

ID, Title, Genre, Director, or Actor	Title	Director	ID	Director
Leave blank to disp	Title of the	Lead direct	174203	Test
Case 3 (Central Node)	Year	Actor 1	testCase2b	Case
Case 4a (Node 2 Unav)	Genre	Actor 2	testCase2c	Actor 2
Case 4b (Node 3 Unav)	Comedy, di	Supporting	Thriller	2b
Extra (Central Node)	Rank	Movie ra	Rank	7.7
Extra (Central Node)	Case 1 (Failure Replicating to Ce	Case 1 (Failure While Replicating to Central N		
Extra (Node 2 + Node	Case 2a (Failure Replicating to N	Case 2a (Failure While Replicating to Node 2		
Extra (Node 2 + Node	Case 2b (Failure Replicating to N	Case 2b (Failure While Replicating to Node 3		
Delete	Case 3 (Central Node Unavailable)	Case 3 (Central Node Unavailable)		
ID	Case 4a (Node 2 Unavailable)	Case 4a (Node 2 Unavailable)		
Movie ID	Case 4b (Node 3 Unavailable)	Case 4b (Node 3 Unavailable)		
Case 1 (Failure Rep	Case 2a (Failure Rep	Case 2b (Failure Rep		
Case 2a (Failure Rep	Case 3 (Central Node Unavailable)	Case 4a (Node 2 Unavailable)		
Case 2b (Failure Rep	Case 4a (Node 2 Unavailable)	Case 4b (Node 3 Unavailable)		



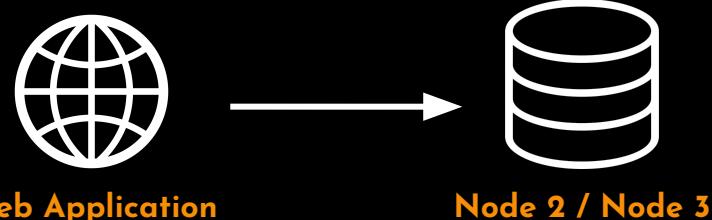
Global Recovery

SIMULATING CASE 3

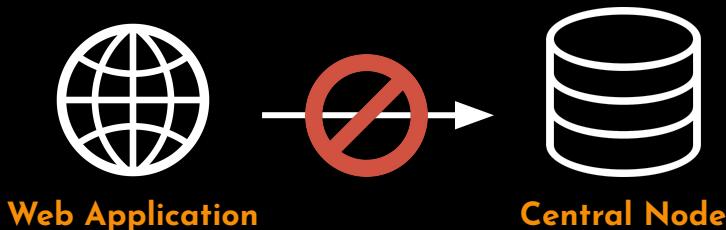
- Central Node is crashed **before initial transaction** by destroying connection of Web App

Case 3: Central Node is not available

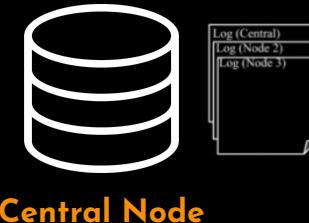
2. Redirection



1. Initial Transaction



3. Catching Up



cgg-distributed-db · Logs | Heroku

Jump to Favorite

Open app More

Application Logs ALL PROCESSES

```
2022-02-16T19:22:09.193078+00:00 app[web.1]: Commit successful!
2022-02-16T19:22:09.194225+00:00 app[web.1]: Executing query to unlock central node
2022-02-16T19:22:09.199766+00:00 app[web.1]: Unlock committed
2022-02-16T19:22:09.200823+00:00 heroku[router]: at=info method=POST path="/case2Update2" host=cgg-distributed-db.herokuapp.com request_id=1d19792496-3485-4884-9b95-3ed3ffff18e fwd="112.201.100.196" dyno=web.1 connect=0ms service=6367ms status=200 bytes=232 protocol=https
2022-02-16T19:22:17.198347+00:00 app[web.1]: Node 3 available!
2022-02-16T19:22:17.343478+00:00 app[web.1]: Results complete -> from Node 3
2022-02-16T19:22:17.344783+00:00 heroku[router]: at=info method=GET path="/searchEntry?searchCriteria=174203&isolationLevel=READONLYCOMMITTED" host=cgg-distributed-db.herokuapp.com request_id=b739f5c31-b9ba-49aa-a23e-6ddc7f1cc340 fwd="112.201.100.196" dyno=web.1 connect=0ms service=346ms status=200 bytes=338 protocol=https
2022-02-16T19:26:04.417327+00:00 heroku[router]: at=info method=GET path="/" host=cgg-distributed-db.herokuapp.com request_id=1d118e47a-f56b-40c8-a3e2-7b51a9e2afe fwd="112.201.100.196" dyno=web.1 connect=0ms service=1615ms status=200 bytes=830826 protocol=https
2022-02-16T19:26:04.654065+00:00 heroku[router]: at=info method=GET path="/css/style.css" host=cgg-distributed-db.herokuapp.com request_id=83595jal6-claf-42b4-9631-95e8e237391a fwd="112.201.100.196" dyno=web.1 connect=0ms service=1ms status=304 bytes=237 protocol=https
2022-02-16T19:26:04.883697+00:00 heroku[router]: at=info method=GET path="/js/home.js" host=cgg-distributed-db.herokuapp.com request_id=bd3ce5b1-3ae0-45c7-86ab-85ec27c649eb fwd="112.201.100.196" dyno=web.1 connect=0ms service=1ms status=304 bytes=238 protocol=https
2022-02-16T19:26:05.116170+00:00 heroku[router]: at=info method=GET path="/img/IMDB_Logo_2016.svg" host=cgg-distributed-db.herokuapp.com request_id=4fffd6ca6-0297-46d1-93c4-149e5079a9e fwd="112.201.100.196" dyno=web.1 connect=0ms service=1ms status=304 bytes=237 protocol=https
2022-02-16T19:26:13.279208+00:00 heroku[router]: at=info method=GET path="/img/reel.png" host=cgg-distributed-db.herokuapp.com request_id=2488861c-3a8c-4725-9f94-98b6715508de fwd="112.201.100.196" dyno=web.1 connect=0ms service=4ms status=304 bytes=239 protocol=https
```

Autoscroll with output

Save

Blogs Careers Documentation

Terms of Service Privacy Cookies © 2022 Salesforce.com

Support

IMDb Distributed Database

Search Insert Update

ID, Title, Genre, Director, or Actor

Title Director

Leave blank to display all entries

Case 3 (Central Node Unavailable)

Year Actor 1

1920 Case

Case 4a (Node 2 Unavailable)

Genre Actor 2

Drama 3

Case 4b (Node 3 Unavailable)

Rank

5.5

Extra (Central Node + Node 2)

Case 1 (Failure Replicating to Central Node from Node 2)

Case 2a (Failure Replicating to Node 2 from Central Node)

Case 2b (Failure Replicating to Node 3 from Central Node)

Case 3 (Central Node Unavailable)

Case 4a (Node 2 Unavailable)

Case 4b (Node 3 Unavailable)

Delete

ID

Movie ID

Case 1 (Failure Replicating to Central Node from Node 2)

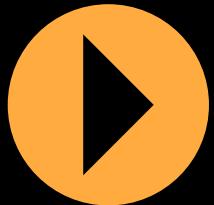
Case 2a (Failure Replicating to Node 2 from Central Node)

Case 2b (Failure Replicating to Node 3 from Central Node)

Case 3 (Central Node Unavailable)

Case 4a (Node 2 Unavailable)

Case 4b (Node 3 Unavailable)



Global Recovery

Case 4: Node 2 or Node 3 is not available

SIMULATING CASE 4

- Crash Node 2 / 3 before the start of the transaction by destroying connection to Web App

1. Initial Transaction

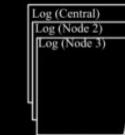


Web Application



Central Node

2. Catching Up



Node 2 / Node 3

cggl-distributed-db · Logs | Hero + dashboard.herokuapp.com/apps/cggl-distributed-d... Incognito

HEROKU Jump to Favorite Open app More

cggl-distributed-db Application Logs ALL PROCESSES

```
dyno=web.1 connect=0ms service=0ms status=304 bytes=238 protocol=https
2022-02-16T19:29:36.742639+00:00 heroku[router]: at=info method=GET path="/img/IMDb_Logo_2016.svg"
host=cggl-distributed-db.herokuapp.com request_id=ab67b810c-185b-45ad-9ae2-d3e086a7edb
fwd="112.201.100.196" dyno=web.1 connect=0ms service=1ms status=304 bytes=237 protocol=https
2022-02-16T19:29:36.734124+00:00 heroku[router]: at=info method=GET path="/img/reel.png" host=cggl-distributed-db.herokuapp.com request_id=ae05e2f7-b9c4-4a97-baf5-ac47486a72a fwd="112.201.100.196"
dyno=web.1 connect=0ms service=1ms status=304 bytes=239 protocol=https
2022-02-16T19:29:44.435105+00:00 heroku[router]: at=info method=GET path="/searchEntry?
searchCriteria=insertCase3isolationLevel=READNOCOMMITTED" host=cggl-distributed-db.herokuapp.com
request_id=4a29c856-ab5a-4ff8-e1edc0581d3b fwd="112.201.100.196" dyno=web.1 connect=0ms
service=357ms status=304 bytes=159 protocol=https
2022-02-16T19:29:44.720828+00:00 heroku[router]: at=info method=GET path="/" host=cggl-distributed-db.herokuapp.com request_id=9b0c12d-5119-414d-9839-474de6fe108 fwd="112.201.100.196" dyno=web.1
connect=0ms service=169ms status=304 bytes=153 protocol=https
2022-02-16T19:29:45.988491+00:00 heroku[router]: at=info method=GET path="/css/style.css"
host=cggl-distributed-db.herokuapp.com request_id=48be5a1b-2653-4775-ac92-a665a1bb37cc
fwd="112.201.100.196" dyno=web.1 connect=0ms service=1ms status=304 bytes=237 protocol=https
2022-02-16T19:29:45.994079+00:00 heroku[router]: at=info method=GET path="/js/home.js" host=cggl-distributed-db.herokuapp.com request_id=4eba9293-6362-49b0-adce-2ee11828d8d fwd="112.201.100.196"
dyno=web.1 connect=3ms service=1ms status=304 bytes=238 protocol=https
2022-02-16T19:29:46.229948+00:00 heroku[router]: at=info method=GET path="/img/IMDb_Logo_2016.svg"
host=cggl-distributed-db.herokuapp.com request_id=ae05e2f7-b9c4-4a97-baf5-ac47486a72a fwd="112.201.100.196"
dyno=web.1 connect=0ms service=1ms status=304 bytes=239 protocol=https
2022-02-16T19:29:46.219146+00:00 heroku[router]: at=info method=GET path="/img/reel.png" host=cggl-distributed-db.herokuapp.com request_id=7c9ffca9-f7fd-4247-a651-f68623371eac fwd="112.201.100.196"
dyno=web.1 connect=0ms service=1ms status=304 bytes=239 protocol=https
```

Autoscroll with output Save

Blogs Careers Documentation

Terms of Service Privacy Cookies © 2022 Salesforce.com Support

Technic... raw vid... IMDb D... cggl-di... Lounge... STADV... Task List deleteC... GitHub... Zoom Meetin... Sticky ... 26°C ENG 3:29 AM 17/02/2022

IMDb Distributed Database IMDb Distributed Database IMDb Distributed Database IMDb Distributed Database

cggl-distributed-db.herokuapp.com

Showing 1 to 10 of 2,000 entries

ID	Title, Genre, Director, or Actor	Title	Director	ID	Director		
171408	Zee die denkt, De	2000	Adventure	7.1	Gert de Graaff	Bart Klever	Devika Strooker
171409	Zeemeerman, De	1996	Comedy	2.5	Frank Herrebout	Angelique de Brujne	Marjolein Sligte
171410	Zefir v shokolade	1993	Comedy		Aleksandr Pavlovsky	Aleksandr Pankratov-Chyomry	Olga Solodovnikova

Search Insert Update

Display All (May be slow)

Delete

ID: Movie ID

Genre: Comedy, drama, Supporting actor

Rank: Movie rating

Insert

Actor 1: Lead actor

Actor 2: Supporting actor

Genre: Comedy, drama, etc.

Rank: Movie rating

Update

Actor 1: Lead director

Actor 2: Supporting actor

Genre: Title of the movie

Rank: Title

Actor 1: Title of the movie

Actor 2: Lead actor

Genre: Title

Rank: Title

Actor 1: Lead director

Actor 2: Lead actor

Genre: Basis for selecting ent

Rank: Basis for selecting ent

Actor 1: Lead director

Actor 2: Lead actor

Genre: Display Preferences

Number of Movies per Genre

Show 10 entries Search:

cggl-distributed-db · Logs | Heroku

dashboard.herokuapp.com/apps/cggl-distributed-db... Incognito

Salesforce Platform

HEROKU Jump to Favorite

Open app More

Application Logs ALL PROCESSES

```
2022-02-16T19:27:03.738Z 2022-02-16T19:27:03.738Z [0] heroku[router]: at=info method=GET path="/js/home.js" host=cggl-distributed-db.herokuapp.com request_id=32f4f576-10e7-4f62-9108-4102bd043a53 fwd="112.201.100.196" dyno=web.1 connect=0ms service=1ms status=204 bytes=238 protocol=https
2022-02-16T19:27:03.856Z 2022-02-16T19:27:03.856Z [0] heroku[router]: at=info method=GET path="/css/style.css" host=cggl-distributed-db.herokuapp.com request_id=3e1d4c01-f842-4fb5-adc7-a56a20865815 fwd="112.201.100.196" dyno=web.1 connect=0ms service=1ms status=204 bytes=237 protocol=https
2022-02-16T19:27:03.889Z 2022-02-16T19:27:03.889Z [0] heroku[router]: at=info method=GET path="/img/reel.png" host=cggl-distributed-db.herokuapp.com request_id=20852748-811f-49d4-87fe-ce5e82bd5fef fwd="112.201.100.196" dyno=web.1 connect=0ms service=1ms status=204 bytes=239 protocol=https
2022-02-16T19:27:03.966Z 2022-02-16T19:27:03.966Z [0] heroku[router]: at=info method=GET path="/img/IMDB_Logo_2016.svg" host=cggl-distributed-db.herokuapp.com request_id=d42f673e7-82a5-47c4-9e98-370c68a4f2ac fwd="112.201.100.196" dyno=web.1 connect=0ms service=1ms status=204 bytes=237 protocol=https
2022-02-16T19:27:12.605Z 2022-02-16T19:27:12.605Z [0] heroku[router]: at=info method=GET path="/getNode2" host=cggl-distributed-db.herokuapp.com request_id=e02d6435-1ba1-4d08-9cd0-3130c0791b0a fwd="112.201.100.196" dyno=web.1 connect=0ms service=293ms status=200 bytes=740246 protocol=https
2022-02-16T19:27:13.069Z 2022-02-16T19:27:13.069Z [0] heroku[router]: at=info method=GET path="/js/home.js" host=cggl-distributed-db.herokuapp.com request_id=63f9a6f2-8b31-48c3-990b-6fc8c8396853 fwd="112.201.100.196" dyno=web.1 connect=0ms service=1ms status=204 bytes=238 protocol=https
2022-02-16T19:27:13.325Z 2022-02-16T19:27:13.325Z [0] heroku[router]: at=info method=GET path="/img/reel.png" host=cggl-distributed-db.herokuapp.com request_id=0e02bfccea-4b01-4a8a-a171-0fcf0f5fce400 fwd="112.201.100.196" dyno=web.1 connect=7ms service=204 bytes=239 protocol=https
2022-02-16T19:27:13.898Z 2022-02-16T19:27:13.898Z [0] heroku[router]: at=info method=GET path="/img/IMDB_Logo_2016.svg" host=cggl-distributed-db.herokuapp.com request_id=de59c342-508f-4d7b-96a6-5eba7bf011aa fwd="112.201.100.196" dyno=web.1 connect=0ms service=1ms status=204 bytes=237 protocol=https
2022-02-16T19:27:13.898Z 2022-02-16T19:27:13.898Z [0] heroku[router]: at=info method=GET path="/css/style.css" host=cggl-distributed-db.herokuapp.com request_id=43ce8133-cccb-42e3-9ef6-6de02a628f2c fwd="112.201.100.196" dyno=web.1 connect=0ms service=9ms status=204 bytes=237 protocol=https
```

Autoscroll with output

Save

Blogs Careers Documentation

Terms of Service Privacy Cookies © 2022 Salesforce.com

Support

Type here to search

Technic... raw vid... IMDb D... cgg... Lounge... STADV... Task List deleteC... GitHub... Zoom Meetin... Sticky ... 26°C ENG 3:28 AM 17/02/2022

IMDb Distributed Database

IMDb Distributed Database

IMDb Distributed Database

Jump to Favorite

Open app More

Search Insert Update

ID, Title, Genre, Director, or Actor

Title Director

Leave blank to display all entries

Case 3 (Central Node Unavailable)

Year Actor 1

2000 Case

Genre Actor 2

Romance 4b

Rank

Movie rating

CASE 4B

Extra (Central Node + Node 3)

Case 1 (Failure Replicating to Central Node from Node 2)

Case 2a (Failure Replicating to Node 2 from Central Node)

Case 2b (Failure Replicating to Node 3 from Central Node)

Case 3 (Central Node Unavailable)

Case 4a (Node 2 Unavailable)

Case 4b (Node 3 Unavailable)

Delete

ID

Movie ID

Case 1 (Failure Replicating to Central Node from Node 2)

Case 2a (Failure Replicating to Node 2 from Central Node)

Case 2b (Failure Replicating to Node 3 from Central Node)

Case 3 (Central Node Unavailable)

Case 4a (Node 2 Unavailable)

Case 4b (Node 3 Unavailable)

Concurrency

Results of the transactions were recorded and compared against the expected results as enumerated in the test scripts.

▲ EVALUATION METHOD

Concurrency

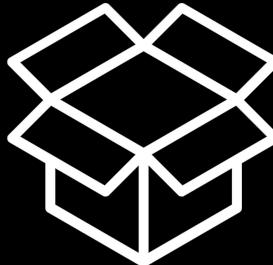
EVALUATION RESULTS

Case	Scenario	Behavior
1 - All Reads		Reads returned the same data
2 - One Write Two Reads	Read Uncommitted	Dirty reads occurred
	Read Committed	Reads return the original transaction values before committing
	Repeatable Read	Reads return the original transaction values before committing
	Serializable	Reads return the updated transaction values after committing
3 - All Writes	Lock wait timeout not exceeded	Transactions are executed serially
	Lock wait timeout exceeded	First transaction is executed, other transactions are aborted

Global Failure Recovery

White-box examinations of the **logs** were performed.

Black-box tests were performed on **attributes related to the expected output** or the **database states** at each of the nodes.



Global Failure Recovery

Read Transactions

Transaction Type	Attributes
Select (returns single entry)	Attribute values of the entry
	Cardinality and attributes of the results
Select (returns multiple entries)	Attribute values of the entry with the lowest ID
	Attribute values of the entry with the highest ID

EVALUATION METHOD



Global Failure Recovery

Write Transactions

Transaction Type	Attributes
Insert	Attribute values of the added entry
	Cardinality of the movies table in each node
	Mutual exclusivity of data in Nodes 2 and 3
Update	Attribute values of the updated entry
	Presence of updated entry in the nodes
Delete	Absence of deleted entry in the application layer
	Absence of deleted entry in the nodes

EVALUATION METHOD



Case 1: Failure in Replicating from Node 2 or Node 3 to the Central Node

The transaction is directed to the proper partial replica based on the **year**.

The transaction is **executed** and **committed**, and its **logs are updated**.

A record of the transaction persists in the **(central node, Node x)** log.

While the central node is down, **next_trans_commit** is **behind next_trans_record**.



Case 2: Failure in Replicating from the Central Node to Node 2 or Node 3

The transaction is directed to the **central node**.

The transaction is **executed** and **committed**, and its **logs are updated**.

A record of the transaction persists in the **(Node x, central node)** log.

While Node x is down, **next_trans_commit** is **behind next_trans_record**.



Case 3: Central Node is Not Available

The same recovery strategy as Case 1 is applied.

The **id_new_entry** log field informs the partial replicas of the **ID of the next entry** to be inserted.

Users can still **perform transactions** on **Nodes 2 and 3**.

After the central node recovers it **performs a catch up** based on the **logs**.



Case 4: Node 2 or Node 3 is Not Available

The same recovery strategy as Case 2 is applied.

Users can still **perform transactions** on the **central node**.

After a partial replica recovers it **performs a catch up** based on the **logs**.

Global Failure Recovery

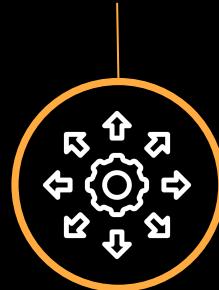
Unavailable Node	Cardinality of Result Set	Description
Central Node	11574	Full result set
Node 2	11574	Full result set
Node 3	11574	Full result set
Central Node and Node 2	7900	Partial result set
Central Node and Node 3	3674	Partial result set
Nodes 2 and 3	11574	Full result set
Central Node, Node 2, and Node 3	0	Empty result set

INSIGHTS AND CONCLUSIONS



Implementation independent of DBMS for concurrency and recover control protocols highlighted complexities of distributed database system maintenance

Decentralized statement-based logs facilitated robust concurrency and failure management



Combination of the READ COMMITTED isolation level and write locks prevents transaction anomalies

INSIGHTS AND CONCLUSIONS

Reliable point of access
for the IMDb dataset
and proof of concept for
distributed database
management techniques



Maintaining replicated
logs allows system
operation to continue
even when the central
node is down



Use of `async` and `await`
functions and threading
can improve the quality
of the implementation

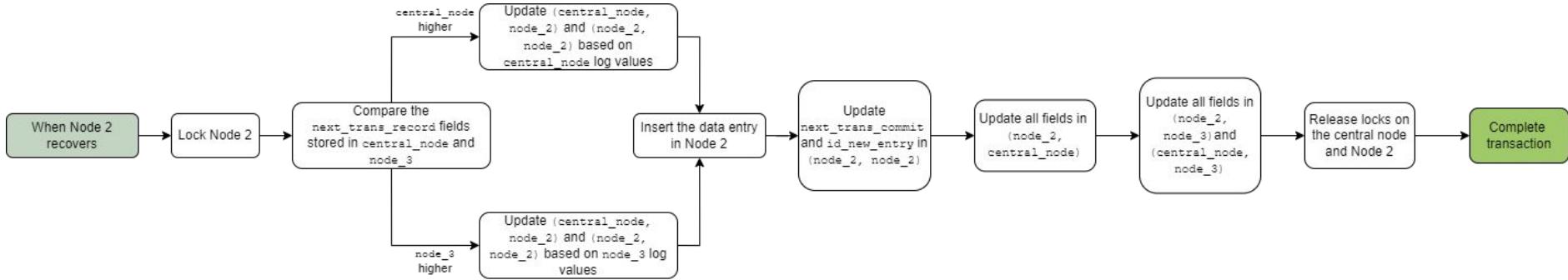


THANK YOU!

Group 13

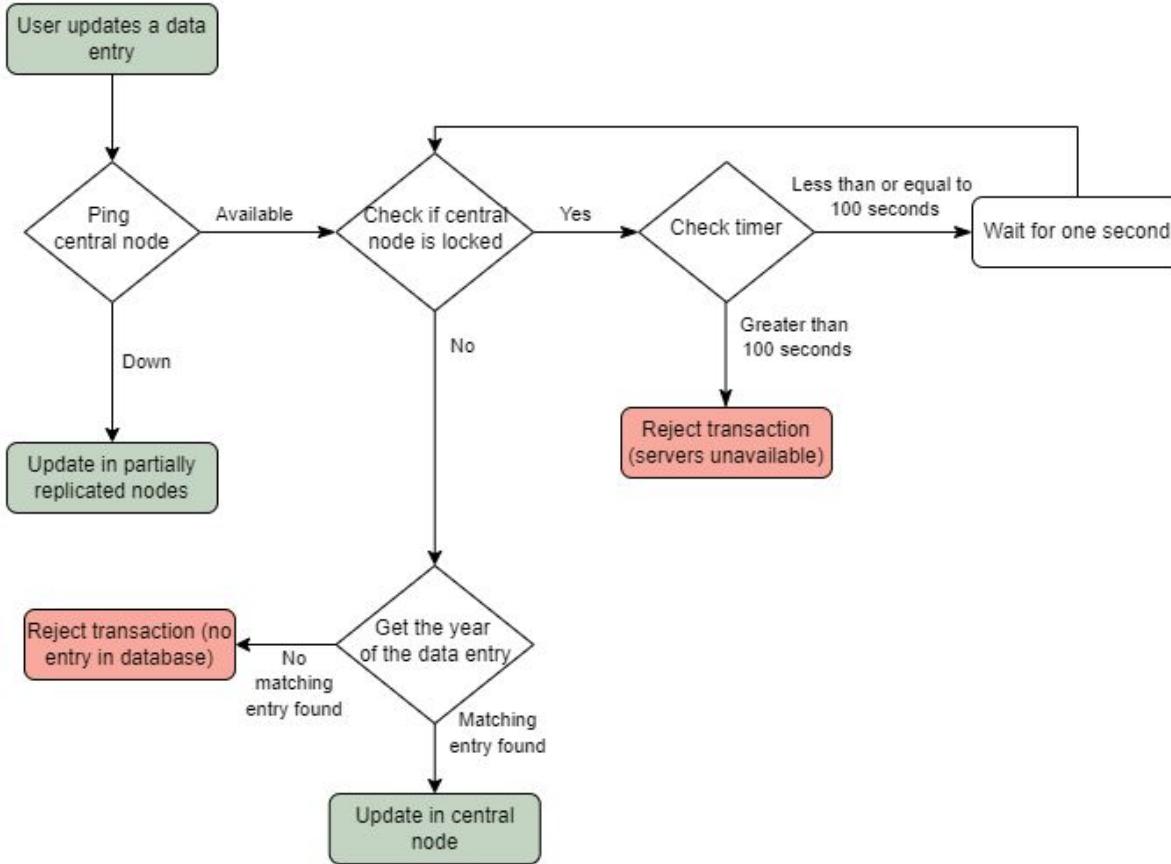
Cua, Lander Peter E
Gaba, Jacob Bryan B.
Gonzales, Mark Edward M.
Lee, Hylene Jules G.

CREDITS: This presentation template was created by Slidesgo, including icons by Flaticon, and infographics & images by Freepik.



► CATCHING UP

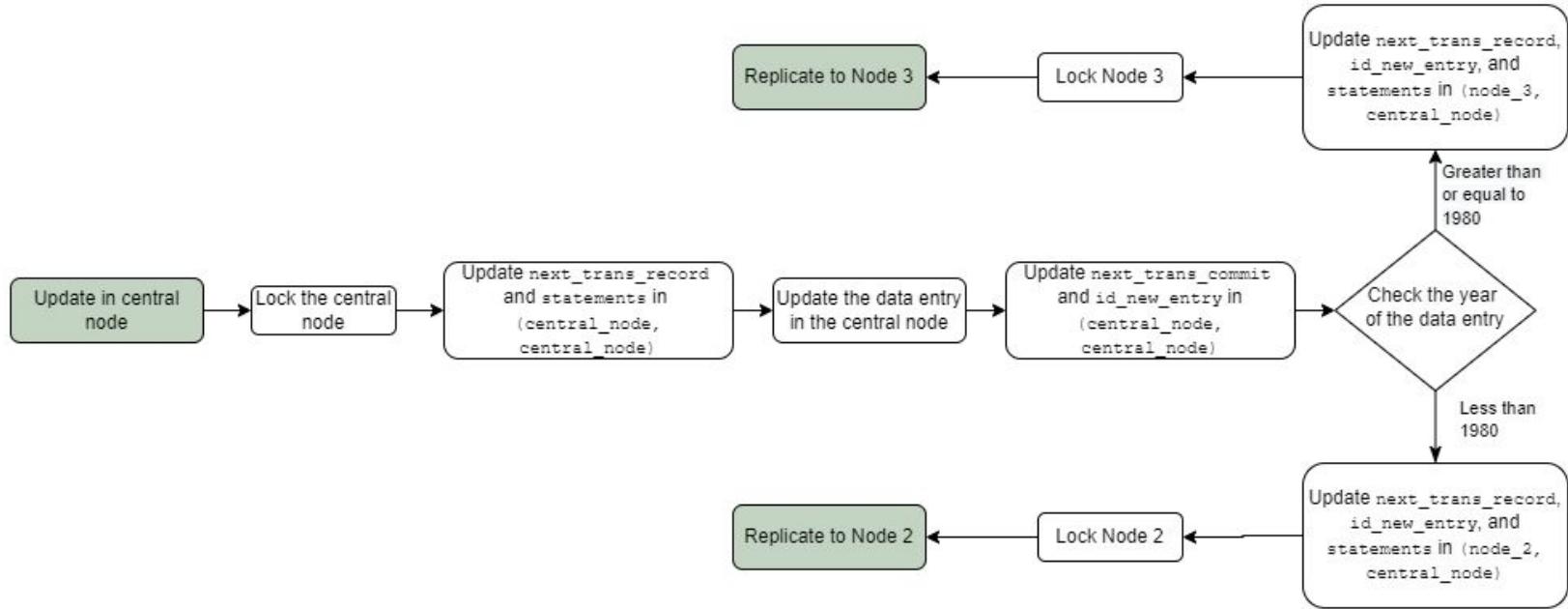
ALGORITHM (UPDATE)



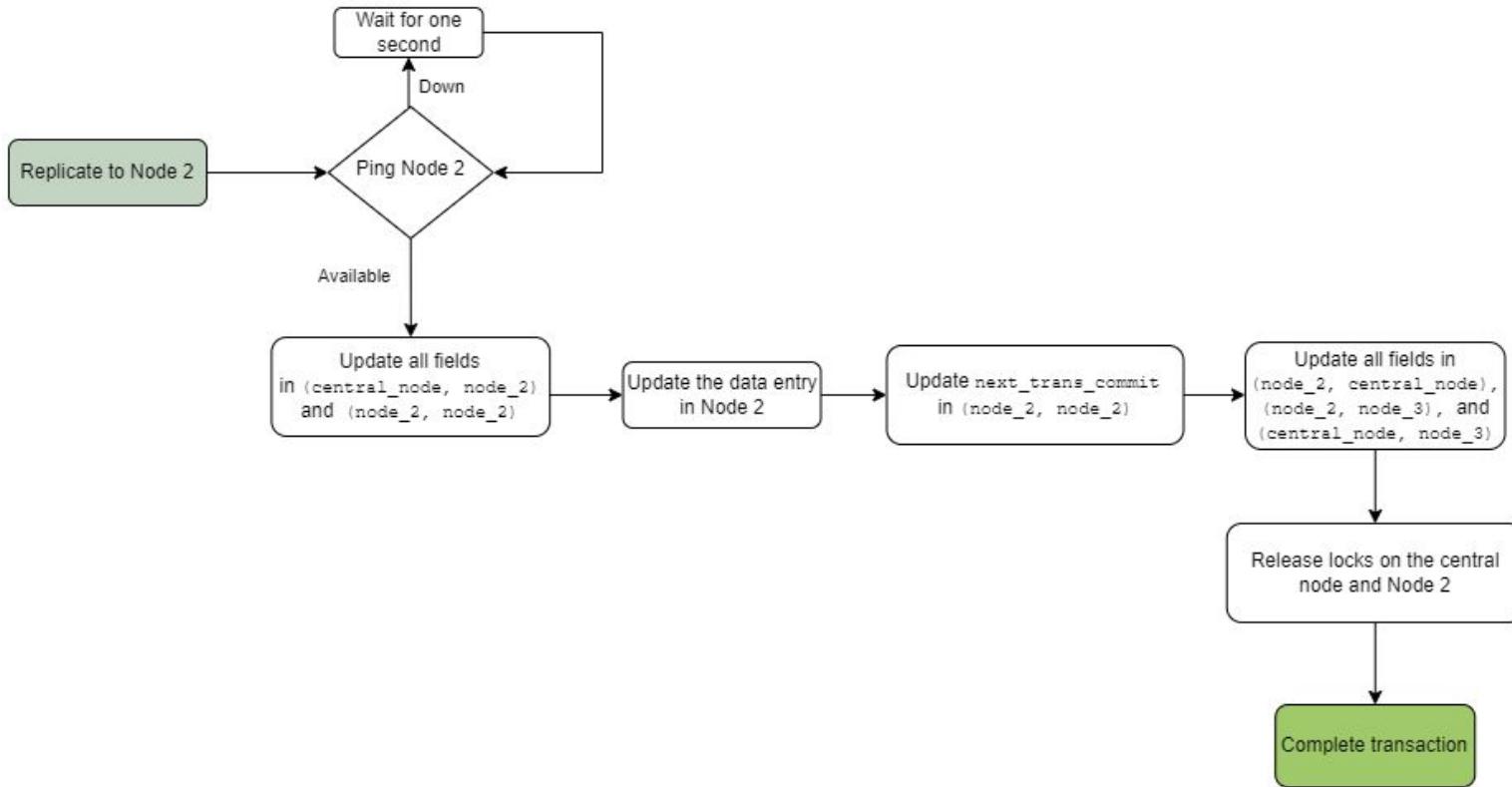
ALGORITHM (UPDATE)



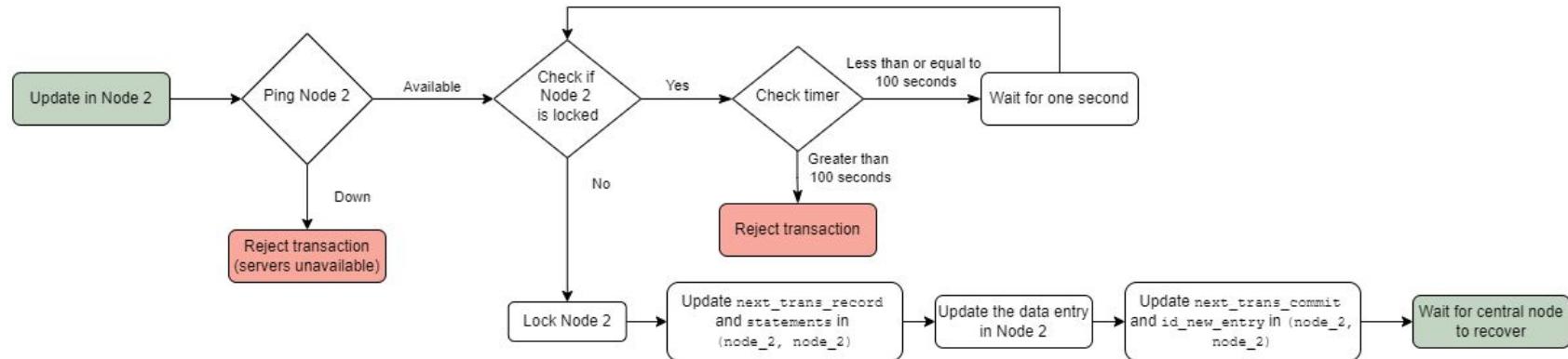
ALGORITHM (UPDATE)



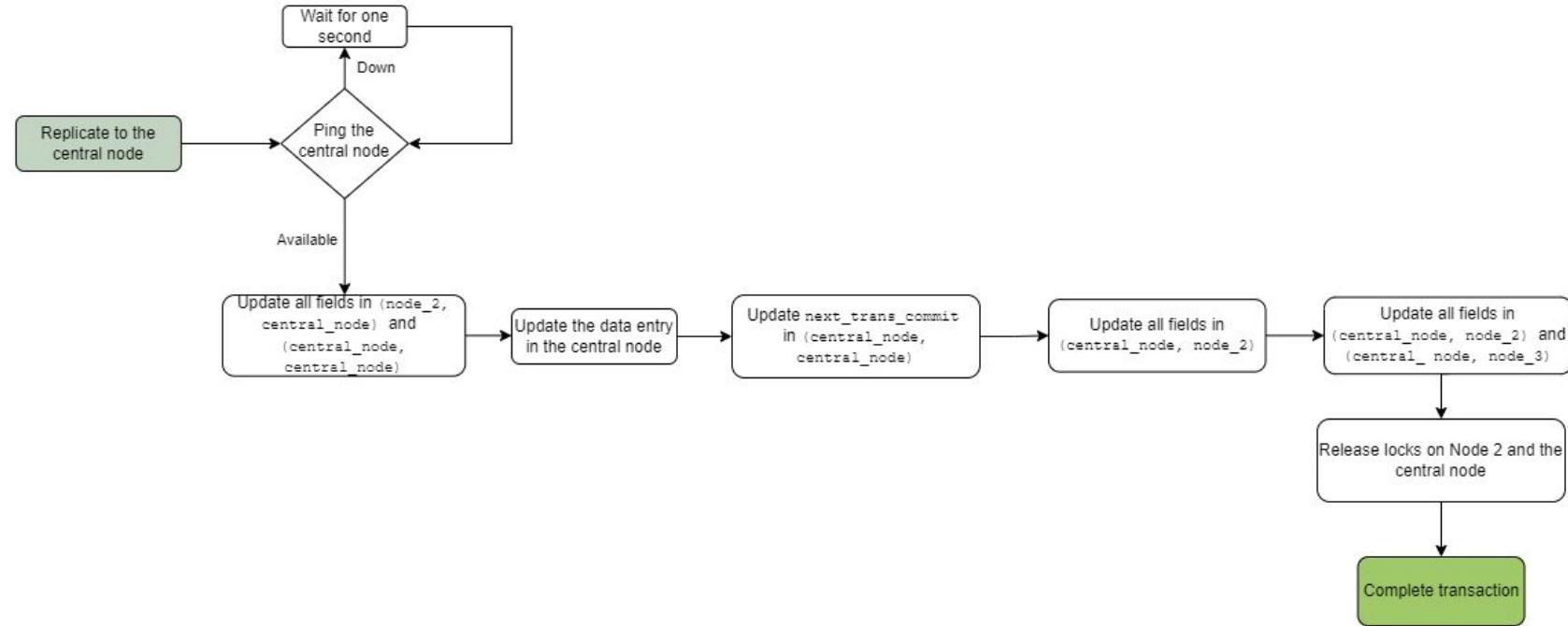
► ALGORITHM (UPDATE)



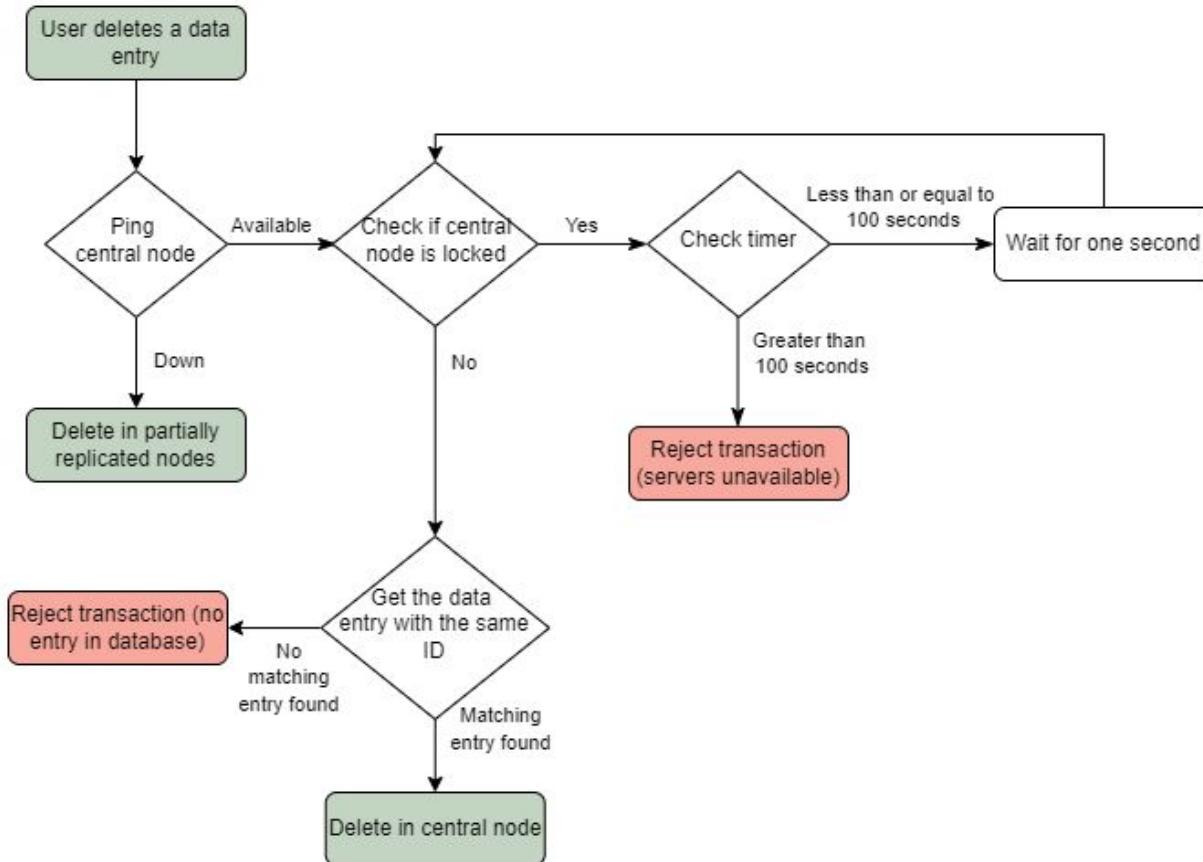
ALGORITHM (UPDATE)



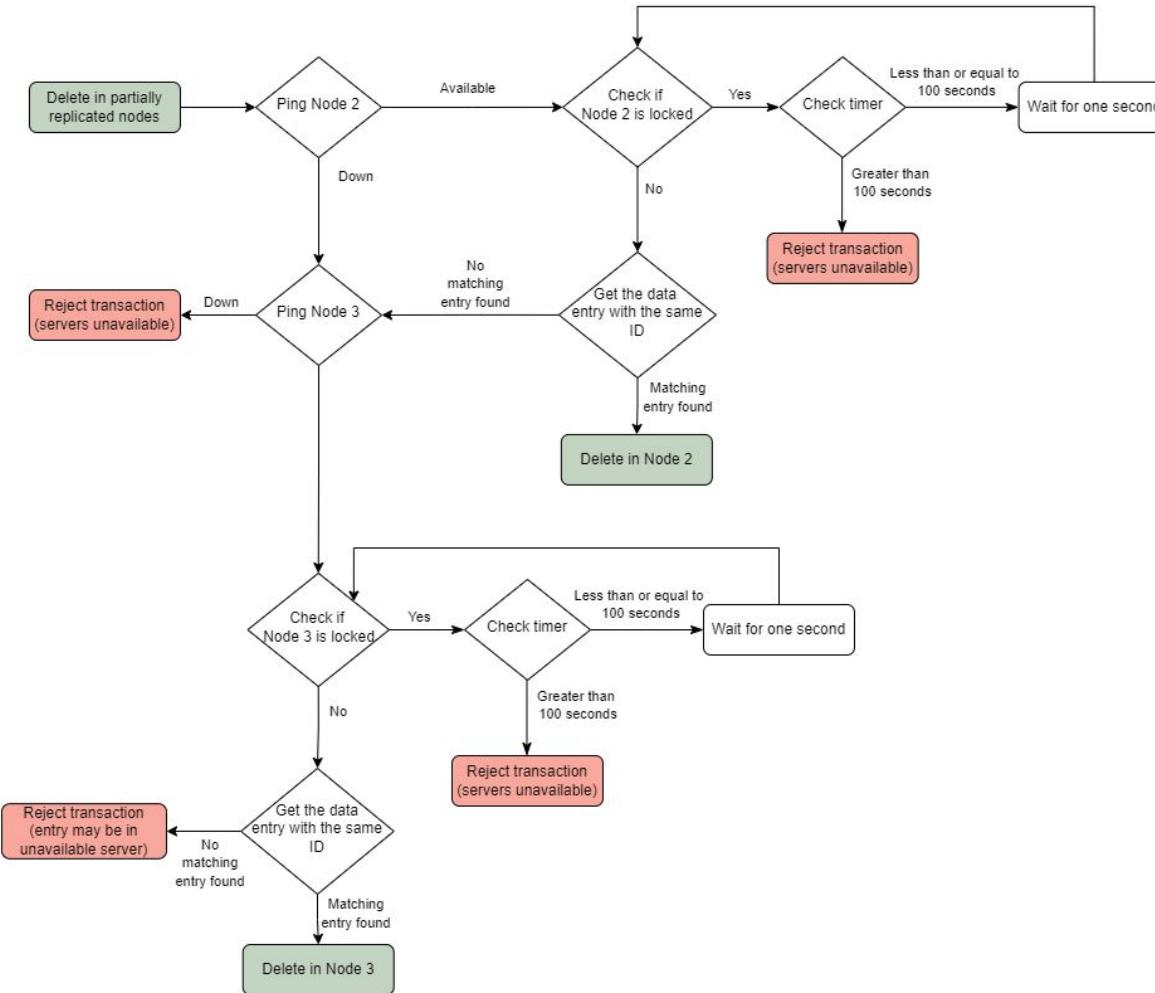
► ALGORITHM (UPDATE)



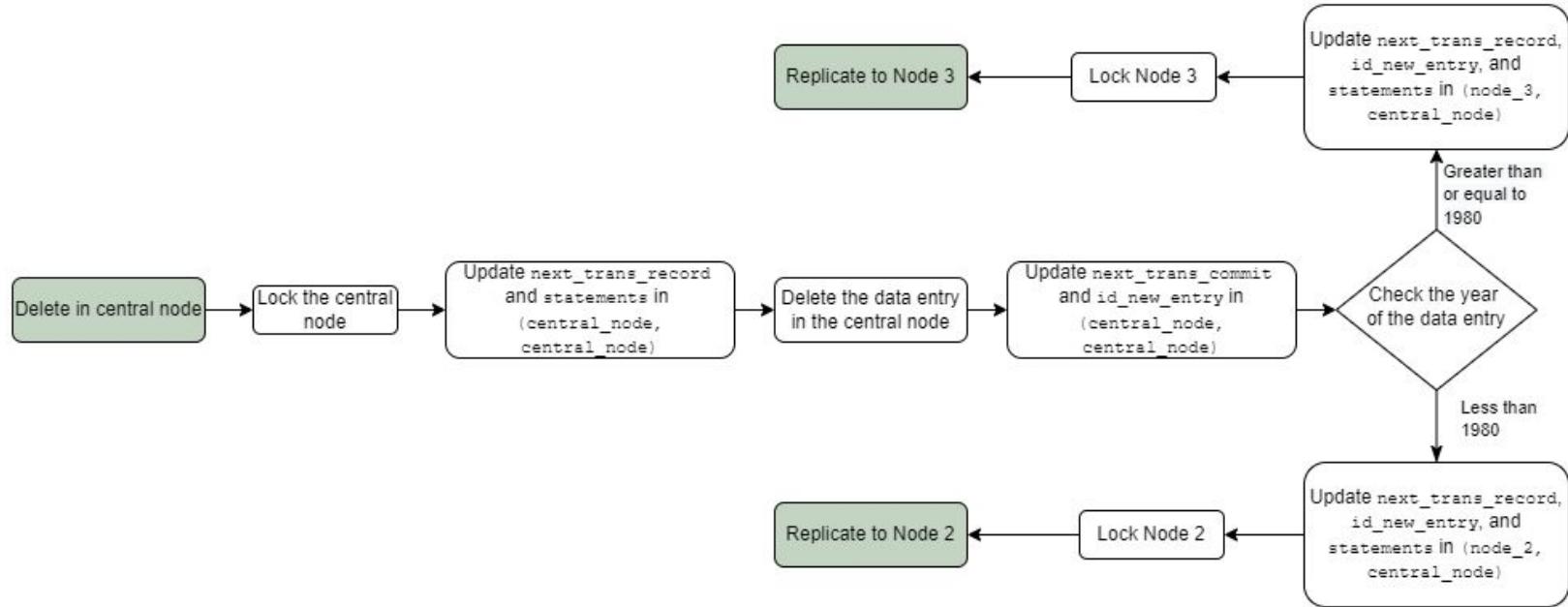
▲ ALGORITHM (DELETE)



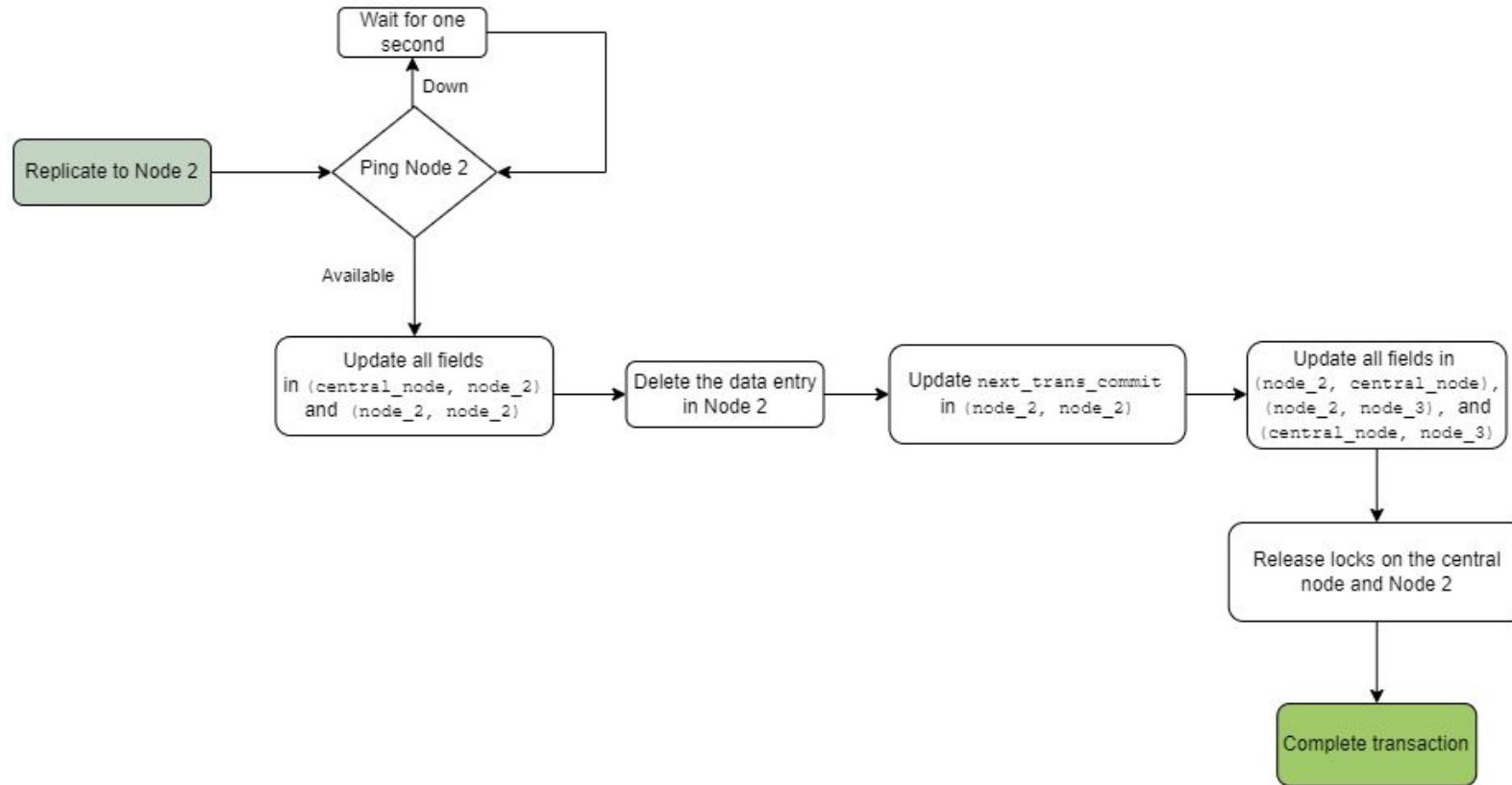
ALGORITHM (DELETE)



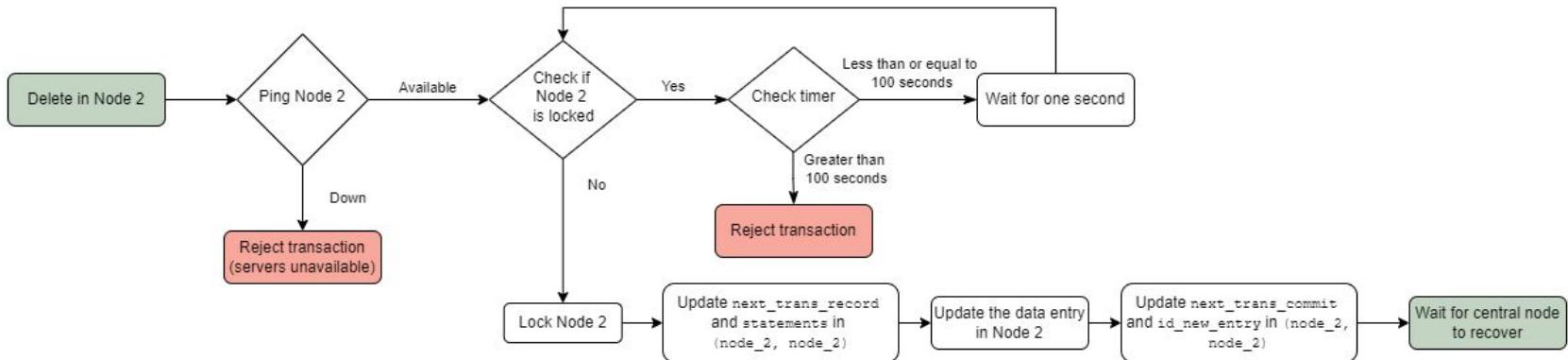
ALGORITHM (DELETE)



ALGORITHM (DELETE)



▲ ALGORITHM (DELETE)



ALGORITHM (DELETE)

