



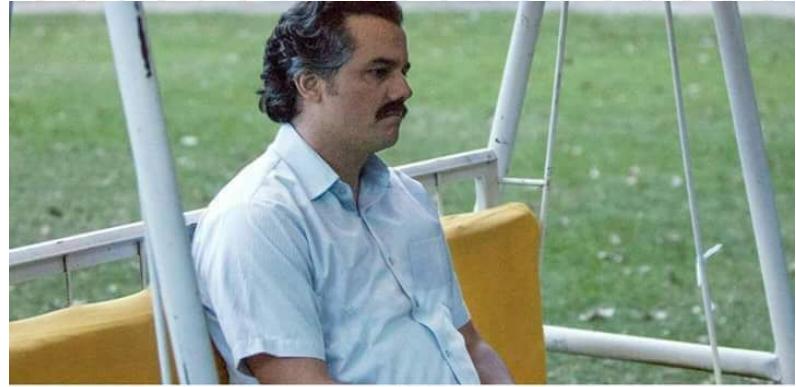
A large, stylized arrow graphic is positioned on the left side of the slide. It starts at the bottom left with a yellow-to-red gradient and points towards the top right, ending near the title text.

# Hands-on With Apache Cassandra® ACID Transactions

**Patrick McFadin**, DataStax, Apache Cassandra Committer

# ➤ Are We There Yet?

Over 5 years in the making, and...



# ➤ Time to Get Hands-On!

<https://github.com/pmcfadin/awesome-accord>





## Warning



- This is pre-merge code
- There will be bugs
- There will be missing features



## Why then?

### Feedback

We need to understand real world use cases and syntax issues.

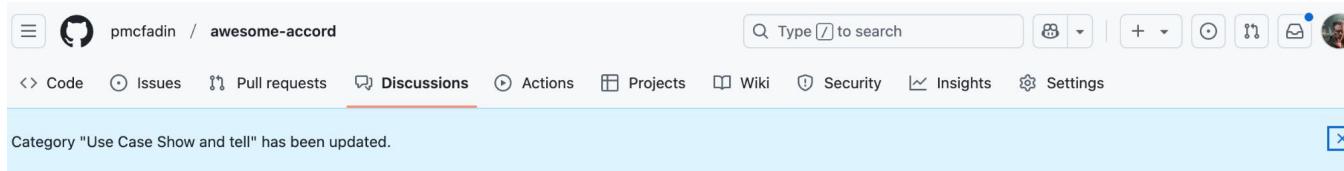
### Speed

More eyes. More Fixes. Faster time to release!

### Preparation

This will shape the next ten years of the project.

# ➤ Important Contributions



The screenshot shows a GitHub repository page for 'pmcfadin / awesome-accord'. The 'Discussions' tab is selected. A notification bar at the top says 'Category "Use Case Show and tell" has been updated.' Below it, the 'Manage discussion categories' section lists five categories:

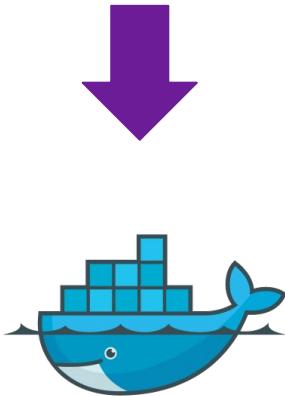
Category	Description	Actions
Announcements	Updates from maintainers	<a href="#">Edit</a> <a href="#">Delete</a>
Bug Reports	Found an error? Tell us about it.	<a href="#">Edit</a> <a href="#">Delete</a>
Enhancement Ideas	Share ideas for new features	<a href="#">Edit</a> <a href="#">Delete</a>
Questions about Accord	Ask the community for help	<a href="#">Edit</a> <a href="#">Delete</a>
Use Case Show and tell	Show off something you've made or your use case.	<a href="#">Edit</a> <a href="#">Delete</a>



What Do You  
Get?



# Local



/docker

# Cloud



/easy-cass-lab\*



```
# Pull the latest image
docker pull pmcfadin/cassandra-accord

# Start a container
docker run -d \
--name cassandra-accord \
-p 9042:9042 \
pmcfadin/cassandra-accord

# Connect using cqlsh
docker exec -it cassandra-accord ./bin/cqlsh
```

# › Examples

```
examples/
└── banking/          # Financial transaction examples
   ├── inventory/      # Inventory management examples
   └── user-management/ # User registration and
                         management
```

```
./setup.sh
```

› Use Cases Cassandra  
Can't Do...

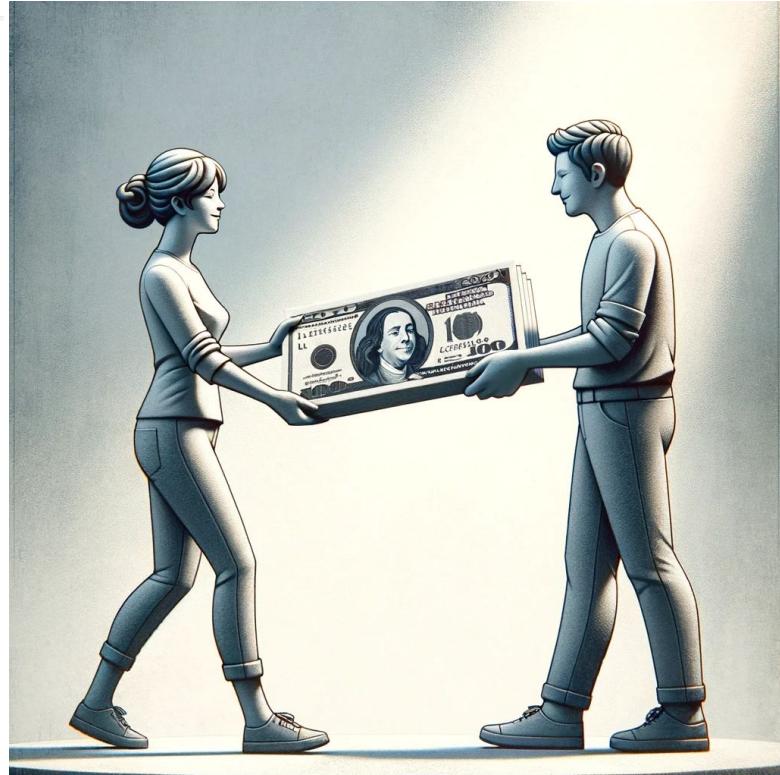
Yet



# ➤ Bank Transactions

- Moving data from one table to another. Exclusively
- Maintains order
- Maintains exclusivity

The classic “Alice to Bob” transaction



# ➤ Inventory Management

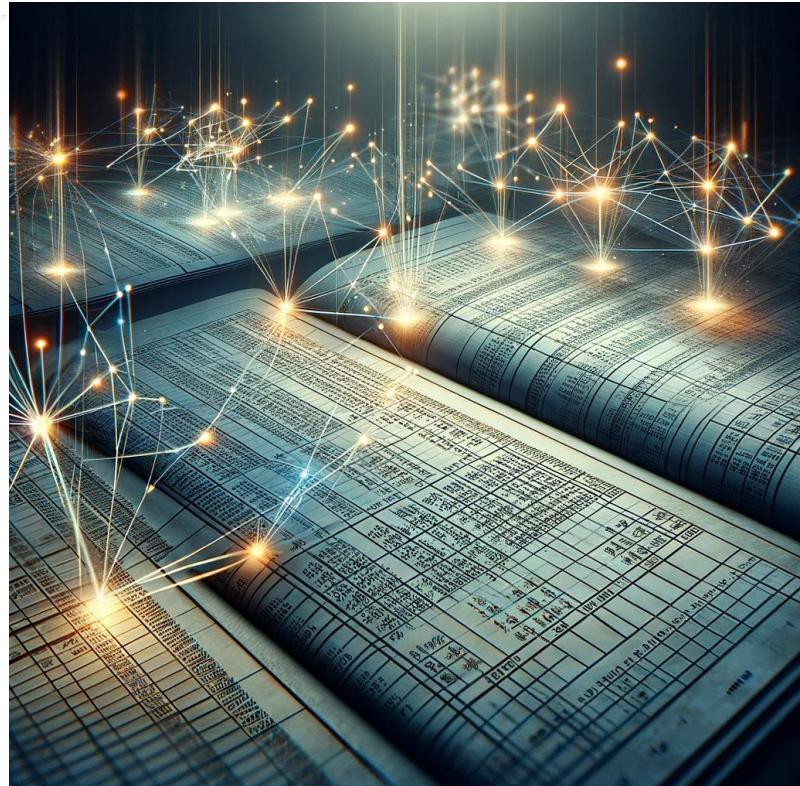
- Inventory is finite
- “Go to zero” problem

Bonus problem: High concurrency



# ➤ Batch consensus(user-management)

- Every change
  - Exclusive
  - In order
  - Adheres to timing



## ➤ Usage



# ➤ Transaction Syntax - Boundaries

```
BEGIN TRANSACTION
```

```
LET <tuple> = ( SELECT <column1>,<column2>.. FROM <table> WHERE <condition>);
```

```
SELECT <return_column> FROM <table> WHERE <condition>;
```

```
IF <tuple_condition> THEN
```

```
    UPDATE|INSERT|DELETE..
```

```
END IF
```

```
COMMIT TRANSACTION;
```

- Everything inside happens or it doesn't
- All statements are isolated
- All mutations are atomic
- No rollbacks (yet)

# ➤ Transaction Syntax - State collection

```
BEGIN TRANSACTION

LET <tuple> = (SELECT <column1>,<column2>.. FROM <table> WHERE <condition>);

SELECT <return_column> FROM <table> WHERE <condition>;

IF <tuple_condition> THEN
    UPDATE|INSERT|DELETE..
END IF

COMMIT TRANSACTION;
```

- Gather state for use in the conditional below
- Tuple stores one or more columns of data

# ➤ Transaction Syntax - Return value

```
BEGIN TRANSACTION

LET <tuple> = ( SELECT <column1>,<column2>.. FROM <table> WHERE <condition>);

SELECT <return_column> FROM <table> WHERE <condition>;

IF <tuple_condition> THEN
    UPDATE|INSERT|DELETE..
END IF

COMMIT TRANSACTION;
```

- Return state from before transaction

# ➤ Transaction Syntax - Conditional mutation

```
BEGIN TRANSACTION

LET <tuple> = ( SELECT <column1>,<column2>.. FROM <table> WHERE <condition>);

SELECT <return_column> FROM <table> WHERE <condition>;

IF <tuple_condition> THEN
    UPDATE | INSERT | DELETE ..
END IF

COMMIT TRANSACTION;
```

- Condition test (`=, !=, >, <, <=, >=`)
- Introduction of NULL, NOT NULL test
- False condition avoids updates

## ➤ Bank Transaction



Multi-Partition Exclusive Operation

# ➤ Bank Transaction - Setup

```
CREATE TABLE ks.accounts (
    account_holder text,
    account_balance decimal,
    PRIMARY KEY (account_holder)
);
```

```
INSERT INTO ks.accounts(account_holder, account_balance) VALUES ('bob', 100);
INSERT INTO ks.accounts(account_holder, account_balance) VALUES ('alice', 100);
```

# ➤ Bank Transaction - Begin

```
BEGIN TRANSACTION

// Get the balance from Alice's account and store as a Tuple
LET fromBalance = (SELECT account_balance FROM ks.accounts WHERE account_holder='alice');

// Return the balance before update after transaction complete
SELECT account_balance FROM ks.accounts WHERE account_holder='alice';

// If Alice's account balance is greater than $20, move $ 20 to Bob
IF fromBalance.account_balance >= 20 THEN
    UPDATE ks.accounts SET account_balance -= 20 WHERE account_holder='alice';
    UPDATE ks.accounts SET account_balance += 20 WHERE account_holder='bob';
END IF

COMMIT TRANSACTION;
```

# ➤ Bank Transaction - Pre-condition

```
BEGIN TRANSACTION

    // Get the balance from Alice's account and store as a Tuple
    LET fromBalance = (SELECT account_balance FROM ks.accounts WHERE account_holder='alice');

    // Return the balance before update after transaction complete
    SELECT account_balance FROM ks.accounts WHERE account_holder='alice';

    // If Alice's account balance is greater than zero, move $ 20 to Bob
    IF fromBalance.account_balance >= 20 THEN
        UPDATE ks.accounts SET account_balance -= 20 WHERE account_holder='alice';
        UPDATE ks.accounts SET account_balance += 20 WHERE account_holder='bob';
    END IF

    COMMIT TRANSACTION;
```

# ➤ Bank Transaction - Output previous state

```
BEGIN TRANSACTION

    // Get the balance from Alice's account and store as a Tuple
    LET fromBalance = ( SELECT account_balance FROM ks.accounts WHERE account_holder='alice');

    // Return the balance before update after transaction complete
    SELECT account_balance FROM ks.accounts WHERE account_holder='alice';

    // If Alice's account balance is greater than zero, move $ 20 to Bob
    IF fromBalance.account_balance >= 20 THEN
        UPDATE ks.accounts SET account_balance -= 20 WHERE account_holder='alice';
        UPDATE ks.accounts SET account_balance += 20 WHERE account_holder='bob';
    END IF

    COMMIT TRANSACTION;
```

# ➤ Bank Transaction - Conditional Statement

```
BEGIN TRANSACTION

    // Get the balance from Alice's account and store as a Tuple
    LET fromBalance = ( SELECT account_balance FROM ks.accounts WHERE account_holder='alice');

    // Return the balance before update after transaction complete
    SELECT account_balance FROM ks.accounts WHERE account_holder='alice';

    // If Alice's account balance is greater than zero, move $ 20 to Bob
    IF fromBalance.account_balance >= 20 THEN
        UPDATE ks.accounts SET account_balance -= 20 WHERE account_holder='alice';
        UPDATE ks.accounts SET account_balance +=20 WHERE account_holder='bob';
    END IF

    COMMIT TRANSACTION;
```

# ➤ Bank Transaction - Commit

```
BEGIN TRANSACTION

    // Get the balance from Alice's account and store as a Tuple
    LET fromBalance = ( SELECT account_balance FROM ks.accounts WHERE account_holder='alice');

    // Return the balance before update after transaction complete
    SELECT account_balance FROM ks.accounts WHERE account_holder='alice';

    // If Alice's account balance is greater than zero, move $ 20 to Bob
    IF fromBalance.account_balance >= 20 THEN
        UPDATE ks.accounts SET account_balance -= 20 WHERE account_holder='alice';
        UPDATE ks.accounts SET account_balance += 20 WHERE account_holder='bob';
    END IF

COMMIT TRANSACTION;
```

# Thanks and Stay In-touch!

Follow me on LinkedIn: <https://www.linkedin.com/in/patrick-mcfadin-53a8046/>

Looking to modernize your Cassandra deployments? Scan this link and find out what options you have.

