TROUBLE WITH TRANSACTIONS

Evan Jones
@epcjones

MITRO

https://www.mitro.co

Transactions make it easier to write correct applications

Transactions are slow
and hard to scale

PhD research:
Make transactions fast across many machines

MITRO

Password manager for organizations

Application using Postgres

We use transactions!

We use transactions!

Transactions *cause* bugs

Transactions
Cause bugs

CHANGE PASSWORD TITLE

```
begin transaction
if user does not have write permission:
    throw NoPermission
update password title
commit transaction
```

SYMPTOM

some requests block for nearly 60s

CHANGE PASSWORD TITLE

```
begin transaction
if user does not have write permission:
    throw NoPermission
update password title
commit transaction
```

SYMPTOM

some requests block for nearly 60s

If you begin, you must commit/abort

Open transactions hold locks, cause conflicts

Easy to screw up error paths

If you *don't* begin, never commit/abort

Deeply nested function calls `commit()`
Implicitly start new transaction
Result: partially committed data

ONE DAY, BROWSING THE LOG…

org.postgresql.util.PSQLException: ERROR: could not serialize access due to read/write dependencies among transactions

Detail: Reason code: Canceled on conflict out to pivot 31675867, during read.

Hint: The transaction might succeed if retried.

```
at org.postgresql.core.v3.QueryExecutorImpl.receiveErrorResponse(QueryExecutorImpl.java:2157)
at org.postgresql.core.v3.QueryExecutorImpl.processResults(QueryExecutorImpl.java:1886)
at org.postgresql.core.v3.QueryExecutorImpl.execute(QueryExecutorImpl.java:255)
at org.postgresql.jdbc2.AbstractJdbc2Statement.execute(AbstractJdbc2Statement.java:555)
at org.postgresql.jdbc2.AbstractJdbc2Statement.executeWithFlags(AbstractJdbc2Statement.java:417)
at org.postgresql.jdbc2.AbstractJdbc2Statement.executeQuery(AbstractJdbc2Statement.java:302)
```

ONE DAY, BROWSING THE LOG…

```
org.postgresql.util.PSQLException: ERROR:
could not serialize access due to read/
write dependencies among transactions
Detail: Reason code: Canceled on conflict
out to pivot 31675867, during read.
Hint: The transaction might succeed if
retried.
at org.postgresql.core.v3.QueryExecutorImpl.receiveErrorResponse(QueryExecutorImpl.java:2157)
at org.postgresql.core.v3.QueryExecutorImpl.processResults(QueryExecutorImpl.java:1886)
at org.postgresql.core.v3.QueryExecutorImpl.execute(QueryExecutorImpl.java:255)
at org.postgresql.jdbc2.AbstractJdbc2Statement.execute(AbstractJdbc2Statement.java:555)
at org.postgresql.jdbc2.AbstractJdbc2Statement.executeWithFlags(AbstractJdbc2Statement.java:417)
at org.postgresql.jdbc2.AbstractJdbc2Statement.executeQuery(AbstractJdbc2Statement.java:302)
```

ALWAYS RETRY

Concurrency errors can happen *anywhere*

Wrap transactions in a retry loop

ALWAYS RETRY

Concurrency errors can happen *anywhere*

Wrap transactions in a retry loop

```
while true:
  begin transaction
  try:
    doWork()
    commit transaction
    break
  except e:
      rollback transaction
      if not e.isConcurrencyError():
        throw e
```

STARTING A BACKGROUND TASK

```
begin transaction
insert data
start task
respond to user
commit transaction
```

SYMPTOM

not found from background task

COMMUNICATE AFTER COMMIT

Correctly handles aborts
Reliability? Your problem.

At least once:

- Record task in DB (original txn)
- Retry until success
- Remove task from DB (new txn)

APPLICATION "CONCURRENCY CONTROL"

Users edit the access list at the same time

APPLICATION "CONCURRENCY CONTROL"

Users edit the access list at the same time

DROPBOX

First write wins, others fail (conflicts)

TICKETMASTER

First access blocks others

EVERY APP EVER

Last write silently wins

TRANSACTIONS: NONE OF THE ABOVE

First write wins, others fail: Implement OCC

First access blocks: keep transaction open?

Last write wins: no transaction / weak isolation

KEEP TRANSACTION OPEN?

Limit to number of concurrent transactions

Client is slow: blocks others

Client disappears: blocks others

NOT A LONG TERM SOLUTION

TROUBLE WITH TRANSACTIONS

- If you begin, always commit/abort
- If you don’t begin, never commit/abort
- Retry concurrency aborts
- Communicate after commit
- Never hold txn across client round trips

WHY DID I MAKE THESE MISTAKES?

I didn’t read the documentation

I didn’t write my code carefully

Sam Madden should revoke my PhD

The interface makes it easy to make mistakes

OTHER INTERFACE PROBLEMS

- Schema evolution
- Testing
- Programming language integration
- Backups
- Reusing data with other systems
- Performance debugging
- Replication

LET'S MAKE USABLE DATABASES

LET'S MAKE USABLE DATABASES

Evan Jones
@epcjones

MITRO

https://www.mitro.co