DBConnection
Ecto’s SQL Framework

SQL Adapter Issues in Ecto v1.1

- Transactions can get out of sync with database
- Pooling does not isolate errors
- Sandboxing is serial and transactions can get out of sync with database
- 4 SQL adapters
  - PostgreSQL
  - MySQL / MariaDB
  - MSSQL
  - SQLites

SQL Adapter Issues in Ecto v1.1

- Transactions are buggy
- Pooling is buggy
- Sandboxing is buggy

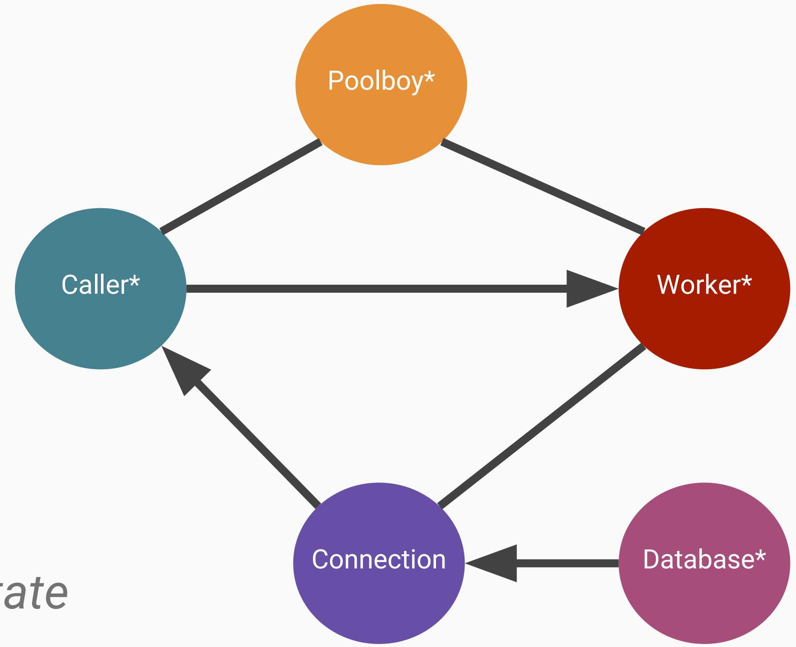
SQL Adapter Issues in Ecto v1.1

- Transactions are hard
- Pooling is hard
- Sandboxing is hard
- Generic abstractions are hard

Transactions in Ecto v1.1

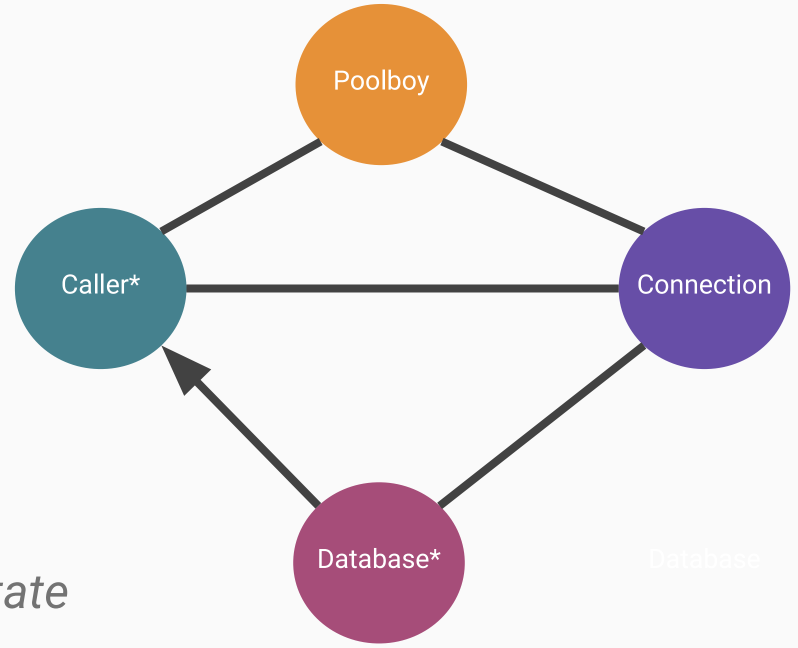
- 3 processes store transaction state:
  - Poolboy
  - Worker
  - Caller
- 3 processes assume their state is correct
- Connection does not store transaction state
- Only the database knows the true transaction state
- Errors can happen at any time

Transactions in Ecto v1.1



** Stores transaction state*

Transactions in DBConnection



** Stores transaction state*

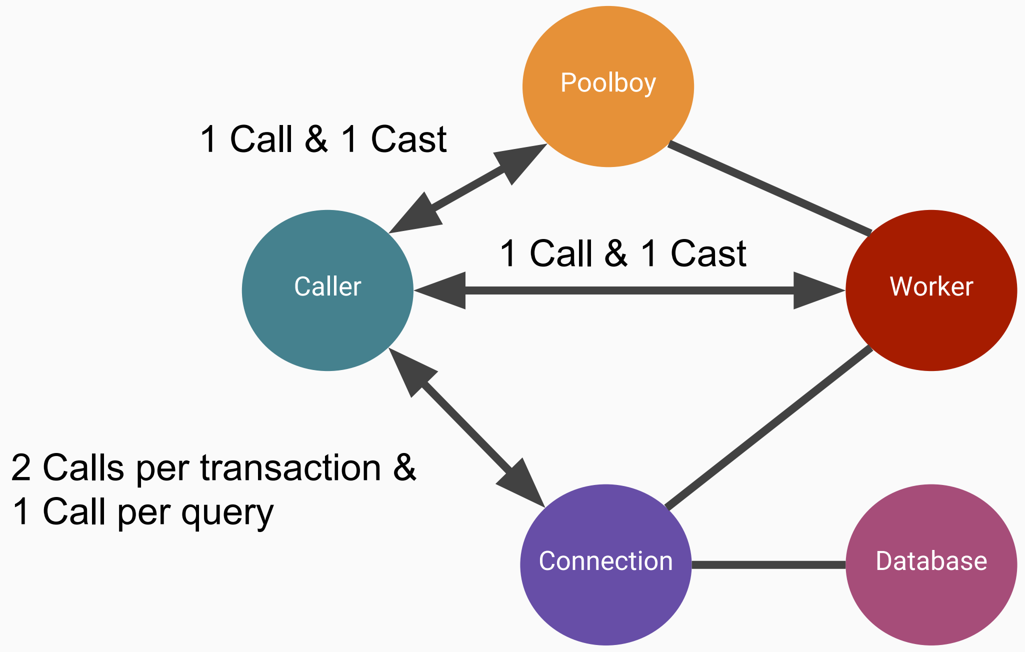
Transactions in DBConnection

- Caller stores transaction state
- Verify transaction state from database (transactions: :strict with Postgrex)
- Only the database knows the true transaction state
- Errors can happen at any time

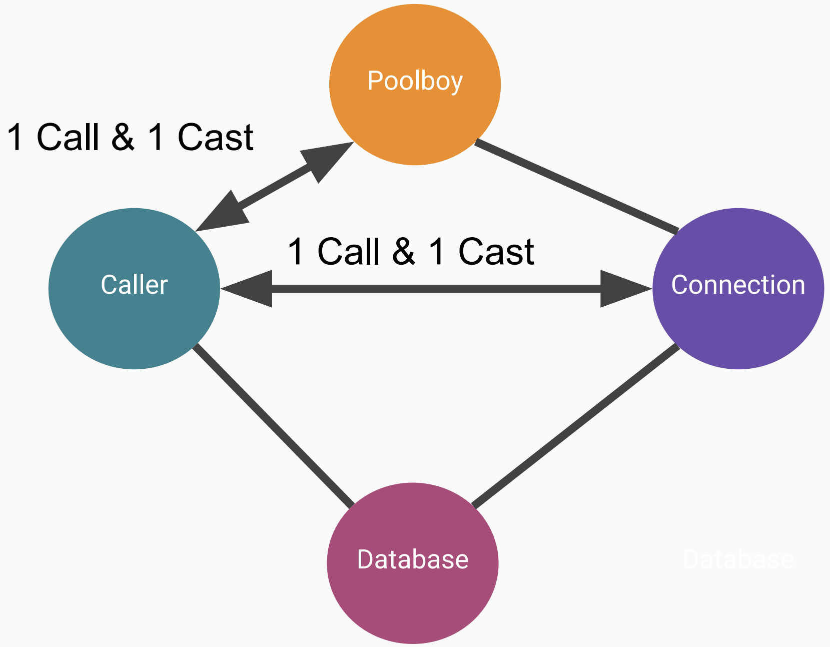
Pooling in Ecto v1.1

- Worker is nested pool of single Connection
- Timeout leaves Connection process in uncertain state
- Lazy connect blocks Caller
- Errors can happen at any time

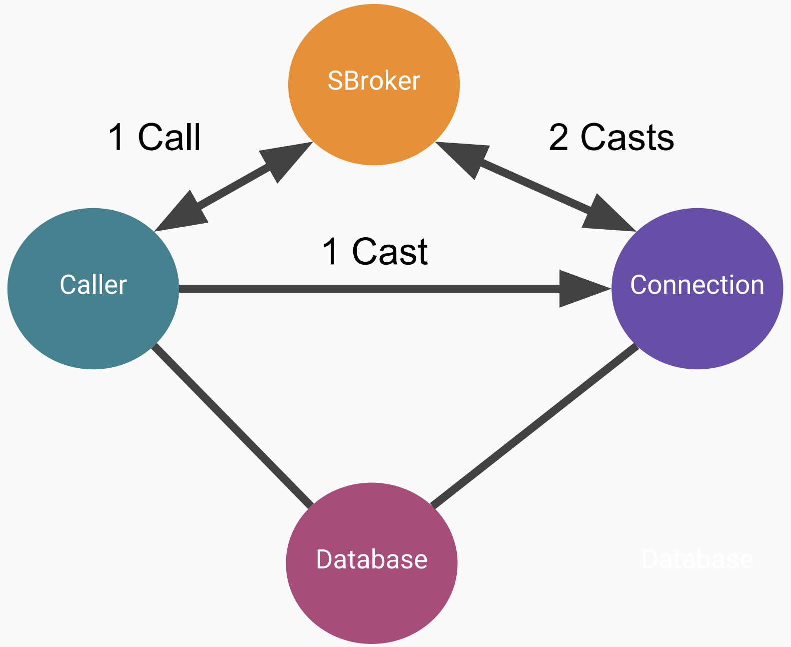
Pooling in Ecto v1.1



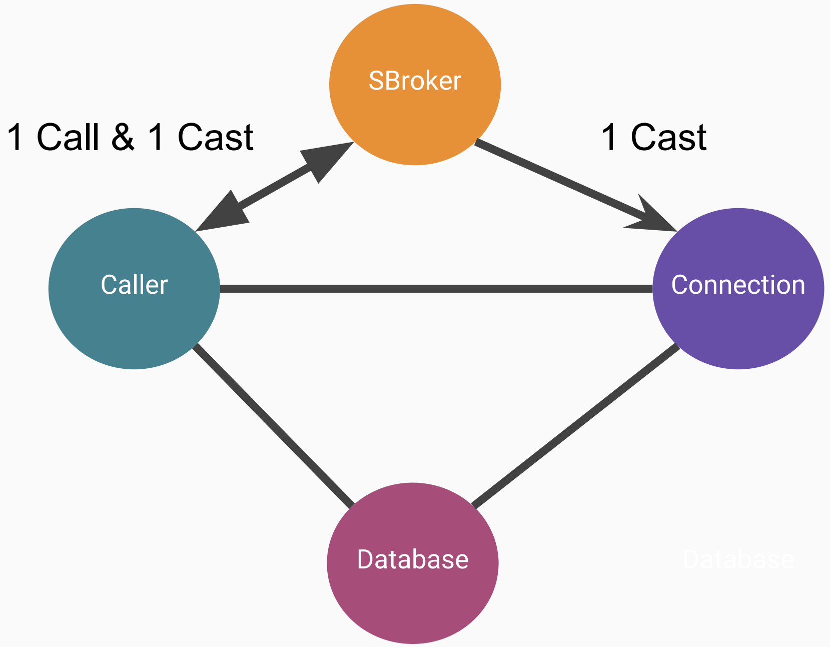
Pooling in DBConnection



Pooling in DBConnection



Pooling in DBConnection v1.2 (maybe)



Pooling in DBConnection

- Connection is nested pool of single Socket
- Caller interacts directly with Socket
- On timeout Connection closes Socket
- No lazy connect
- Errors are isolated by customising Connection for each Pool
  - Poolboy: Caller does synchronous call to Connection until Socket closed
  - SBroker: Connection only enqueues Socket when idle

Pooling Lessons

- Poolboy overflow mechanism can cause connection churn
  - pool_overflow: 0
- LIFO Connection queues can leave “bad” Connection at front of queue
  - idle_out: :out
- Setting socket active on check in and passive on check out is slow
  - idle: :passive

The Process Dictionary

- Process dictionary can lead to cleaner or more maintainable code
- Don’t store side effects in user accessible immutable data
- Store side effects in a mutable data type
  - Pid
  - ETS
  - Process Dictionary
- Make process dictionary abstraction explicit
  - put_info(%DBConnection{conn_ref: reference}, status, state)
  - get_info(%DBConnection{conn_ref: reference})

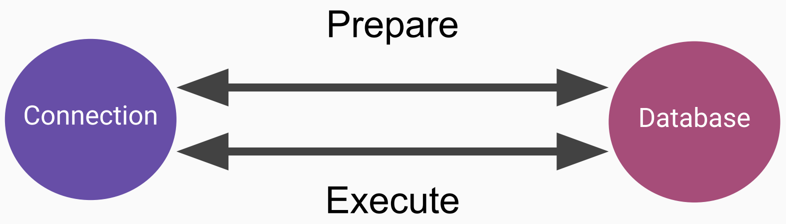
Socket Lessons

- Decode binary data with single/continuous binary match context
    - Mariaex.RowParser
    - Postgrex.TypeModule
- Default :gen_tcp buffer (1460) causes many socket calls if packet: :raw
    - buffer: max(snd_buf, rec_buf)
- Doing many :gen_tcp calls is slow
    - :gen_tcp.recv(socket, 0, timeout)
- Blocking socket code is clearer than non-blocking socket code
    - active: false

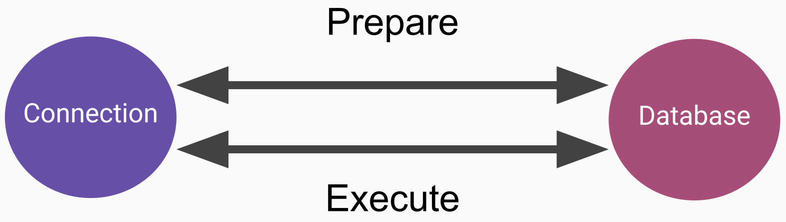
Extended Queries Ecto v1.1

- Prevents SQL injection
  - Mariaex.query(connection, statement, parameters)
  - Postgrex.query(connection, statement, parameters)
- 2 round trips to prepare and execute

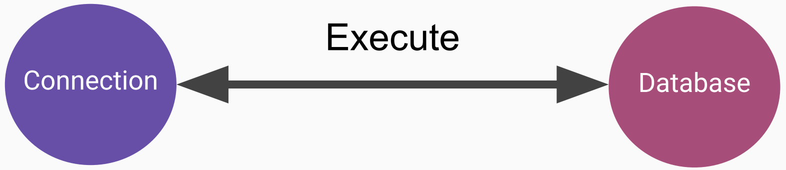
Postgrex Query in Ecto v1.1



(First) Postgrex Query in Ecto v2.0



(Next) Postgrex Query in Ecto v2.0



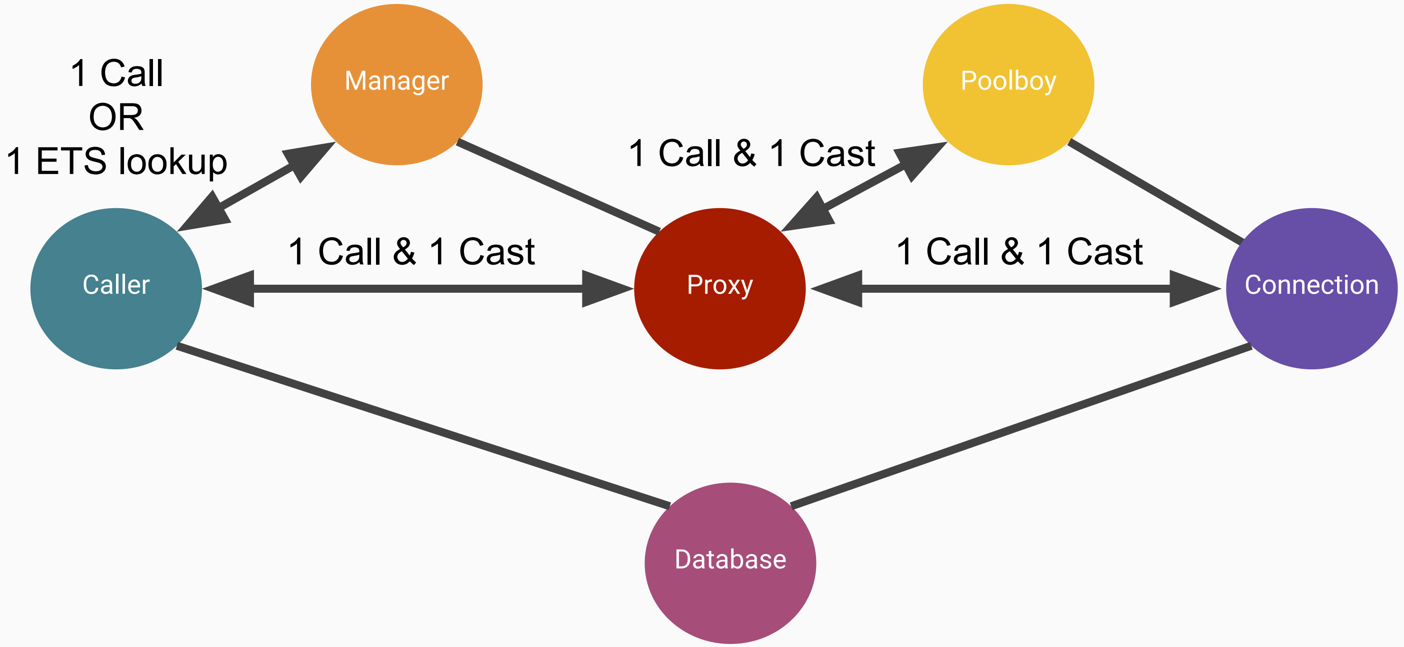
Extended Queries in Ecto v2.0

- First unique (cachable) query per repo generates SQL statement
- First (cachable) statement per connection does 2 round trips
  - Prepare
  - Execute
- Next (cachable) statement per connection does 1 round trip
  - Execute
- Transaction and savepoint queries prepared on connect
- Each adapter has different and complex logic
- User can not tell if cached query used

Sandbox

- Uses a single transaction for duration of sandbox
- BEGIN TRANSACTION at start of sandbox
- ROLLBACK TRANSACTION at end of sandbox
- Rewrites transaction queries to savepoints
  - BEGIN TRANSACTION to SAVEPOINT
  - COMMIT TRANSACTION to RELEASE SAVEPOINT
  - ROLLBACK TRANSACTION to ROLLBACK SAVEPOINT; RELEASE SAVEPOINT

Sandbox


1 Call
OR
1 ETS lookup
Manager
1 Call & 1 Cast
Poolboy
Caller
1 Call & 1 Cast
Proxy
1 Call & 1 Cast
Connection
Database

Sandbox Modes

- :auto
  - Single caller and multiple sandboxes
- :manual
  - Multiple callers and multiple sandboxes
- {:shared, pid}
  - Multiple callers and single sandbox

Sandbox Errors

- DBConnection.OwnershipError
    - Ecto.Adapters.SQL.checkout/2
    - Ecto.Adapters.SQL.allow/4
    - Ecto.Adapters.SQL.mode/2
    - caller: pid
- DBConnection.ConnectionError
    - :sys.get_state/1
    - ownership_timeout: timeout

Sandbox Lessons

- PostgreSQL failed transactions require savepoints to rollback error
  - mode: :savepoint
- InnoDB does not release some locks until transaction rolled back
  - pool_size: 1
- Isolation level must be set before savepoints
  - Isolation: level
- Can hide race conditions

New Features

- Ecto v2.1
  - MyRepo.stream/2
- Ecto v2.2 (maybe)
  - Ecto.Adapters.SQL.Stage.stream/3
  - Ecto.Adapters.SQL.Stage.producer/5
  - Ecto.Adapters.SQL.Stage.producer_consumer/5
  - Ecto.Adapters.SQL.Stage.consumer/5