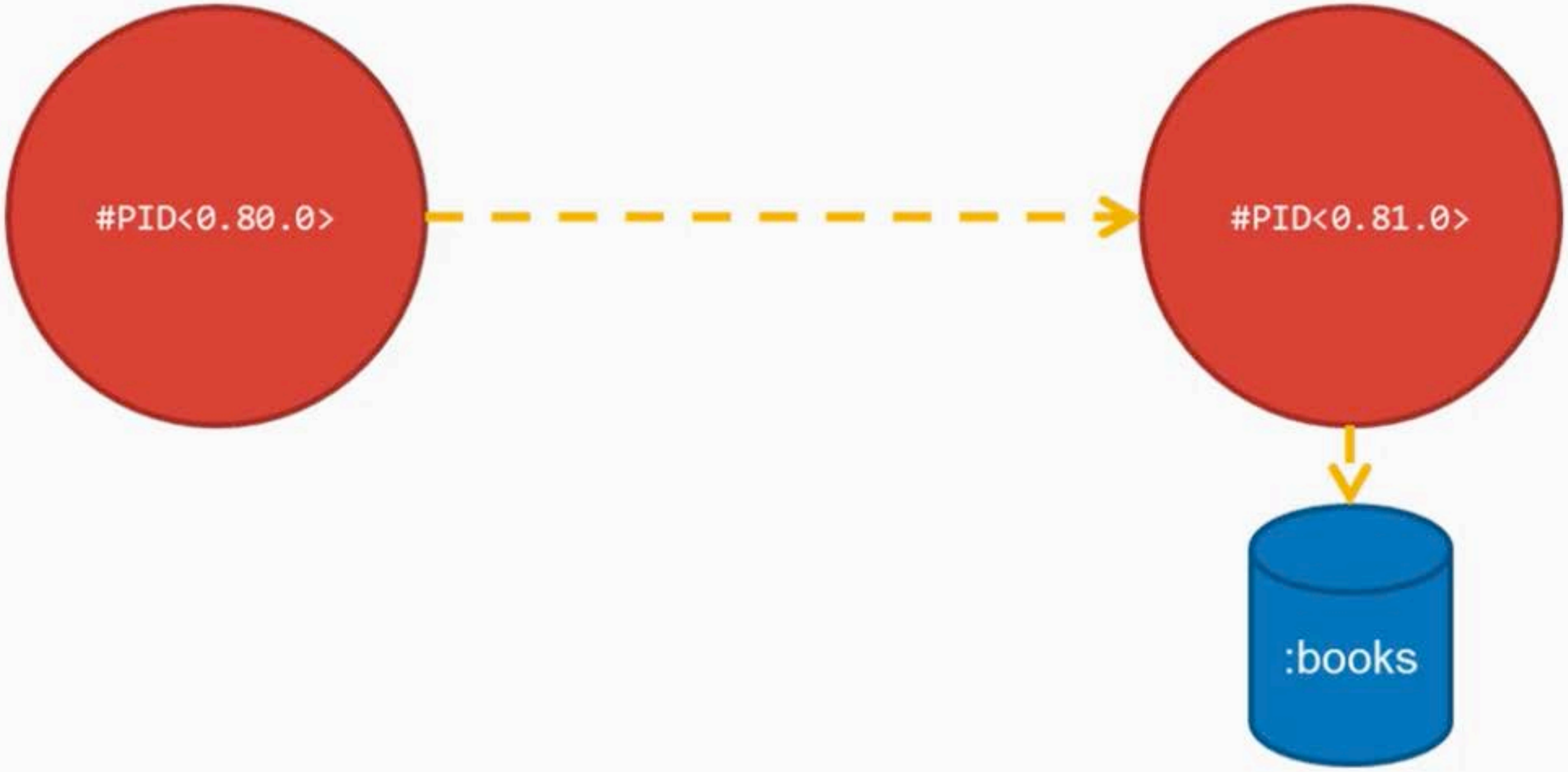
DETS and Mnesia

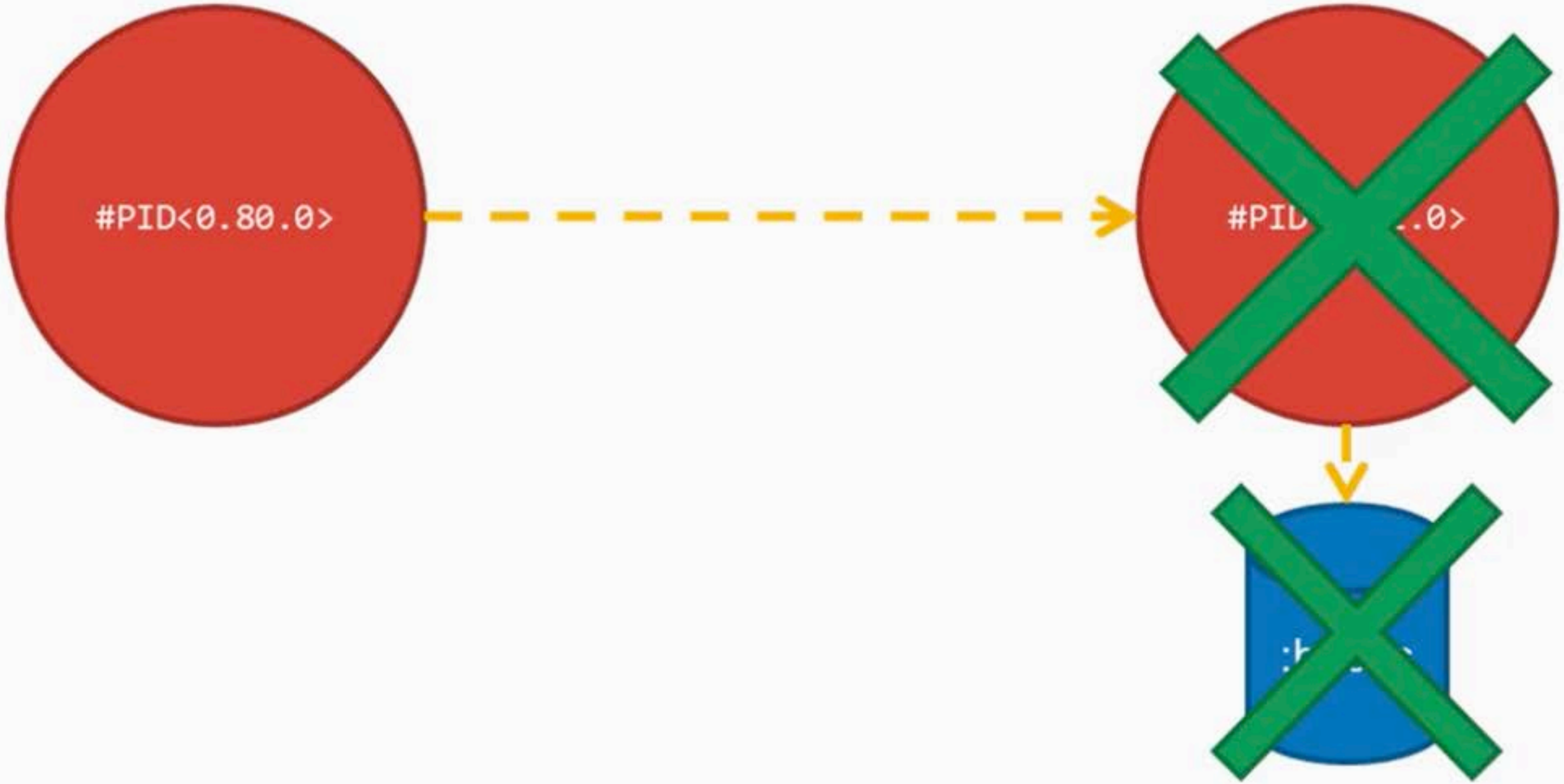
In this Video, we are going to take a look at...

- DETS
- Mnesia
- Further options

ETS – Persistence



ETS – Persistence



DETS

Disk-based Erlang Term Storage

- Persistent file-based data store
- Tables accessible by name
- API compatible with ETS
- Needs to be closed properly
- MUCH slower than ETS

DETS

```
➜ book_catalog iex
Erlang/OTP 20 [erts-9.0] [source] [64-bit] [smp:8:8] [ds:8:8:10] [async-threads:
10] [hipe] [kernel-poll:false] [dtrace]

Interactive Elixir (1.4.5) - press Ctrl+C to exit (type h() ENTER for help)
iex(1)> {:ok, table} = :dets.open_file(:books, [type: :set])
{:ok, :books}
iex(2)> table |> :dets.insert({:fable, 5, 120})
:ok
iex(3)> table |> :dets.insert({:novel, 7, 210})
:ok
iex(4)> table |> :dets.lookup(:fable)
[{:fable, 5, 120}]
iex(5)> table |> :dets.close()
:ok
iex(6)> :erlang.halt
➜ book_catalog
```

DETS

```
➜ book_catalog iex
Erlang/OTP 20 [erts-9.0] [source] [64-bit] [smp:8:8] [ds:8:8:10] [async-threads:
10] [hipe] [kernel-poll:false] [dtrace]

Interactive Elixir (1.4.5) - press Ctrl+C to exit (type h() ENTER for help)
iex(1)> {:ok, table} = :dets.open_file(:books, [type: :set])
{:ok, :books}
iex(2)> table |> :dets.lookup(:novel)
[{:novel, 7, 210}]
iex(3)>
```

What If We Need More Robustness Still?

Mnesia

The distributed data store

- Disk + memory consistent persistency model
- Location transparency
- Transactions
- Fast queries

Mnesia - Showcase

```
➜ book_catalog iex
Erlang/OTP 20 [erts-9.0] [source] [64-bit] [smp:8:8] [ds:8:8:10] [async-threads:
10] [hipe] [kernel-poll:false] [dtrace]

Interactive Elixir (1.4.5) - press Ctrl+C to exit (type h() ENTER for help)
iex(1)> :mnesia.create_schema([node()])
:ok
iex(2)> :mnesia.start()
:ok
iex(3)> :mnesia.create_table(Book, [attributes: [:title, :rating, :pages]])
{:atomic, :ok}
iex(4)> :mnesia.transaction(fn ->
...(4)> :mnesia.write({Book, :fairy_tale, 9, 329})
...(4)> :mnesia.write({Book, :horror_novel, 10, 128})
...(4)> :mnesia.write({Book, :fable, 5, 120})
...(4)> end)
{:atomic, :ok}
iex(5)> :mnesia.transaction(fn -> :mnesia.read(Book, :fairy_tale) end)
{:atomic, [{Book, :fairy_tale, 9, 329}]}
iex(6)>
```

More Options?

- Try Ecto!
  - https://github.com/elixir-ecto/ecto

Summary

- Explored ETS
- Discussed DETS and Mnesia