Why does passing tokio_postgres::Transaction as a reference ask to indicate the anonymous lifetime?

Asked 1 year, 6 months ago

Active 1 year, 6 months ago

Viewed 247 times

0

I'm using tokio_postgres to connect to a database and trying to start a transaction

```
let trans = client.transaction().await.unwrap();
trans.query("select * from abc", &[]).await.unwrap();
// ...
trans.commit().await.unwrap();
```

Everything works as expected. Now I want to put the code between `trans.query` and `trans.commit` into a separate function

```
async fn tx_work(trans: &tokio_postgres::Transaction) {
    trans.query("select * from abc", &[]).await.unwrap();
    // ...
    trans.commit().await.unwrap();
}
```

and call it in `main`:

```
let trans = client.transaction().await.unwrap();
tx_work(&trans).await.unwrap();
```

The code doesn't compile:

```
error[E0726]: implicit elided lifetime not allowed here
   --> src/abc.rs:209:28
    |
209 | async fn tx_work(trans: &tokio_postgres::Transaction) {
    |                          ^^^^^^^^^^^^^^^^^^^^^^^^^^^- help: indicate the anonymous lifetime: `<'_>`
```

This didn't help:

```
async fn tx_work<'a>(trans: &'a tokio_postgres::Transaction)
```

What do I do?

rust lifetime rust-tokio

Share
Improve this question
Follow

edited Jun 18 '20 at 1:59

asked Jun 18 '20 at 1:37

user3839198
75 ● 8

It's hard to answer your question because it doesn't include a minimal reproducible example. We can't tell what crates (and their versions), types, traits, fields, etc. are present in the code. It would make it easier for us to help you if you try to reproduce your error on the Rust Playground if possible, otherwise in a brand new Cargo project, then edit your question to include the additional info. There are Rust-specific MRE tips you can use to reduce your original code for posting here. Thanks!
– Shepmaster
Jun 18 '20 at 1:50

Please edit your question and paste the exact and entire error that you're getting — that will help us to understand what the problem is so we can help best. Sometimes trying to interpret an error message is tricky and it's actually a different part of the error message that's important. Please use the message from running the compiler directly, not the message produced by an IDE, which might be trying to interpret the error for you.
– Shepmaster
Jun 18 '20 at 1:52

I edited the question and included the entire error message.
– user3839198
Jun 18 '20 at 2:00

1 Answer

Active Oldest Votes

2

Based on the definition of Transaction it seems you want the lifetime specifier/parameter on the struct itself, not on the function parameter reference:

```
async fn tx_work(trans: &tokio_postgres::Transaction<'_>)
```

That's using the anonymous lifetime, but you can also explicitly specify the lifetime parameter:

```
async fn tx_work<'a>(trans: &tokio_postgres::Transaction<'a>)
```

It is common to relate these lifetimes to other existing lifetimes, for example, if you already had an explicit lifetime elsewhere and it makes sense to do so, you might pass it as the parameter directly.

This is because `tokio_postgres::Transaction` does not fully specify the type anymore than `Vec` would (compared to `Vec<u8>`), i.e. the lifetime specifiers are part of the type name, so you need `tokio_postgres::Transaction<'some_lifetime>`, but apparently you can use the anonymous lifetime `'_` too.

More specifically, here, the lifetime parameter on `Transaction` pertains to the lifetime of the references contained within the `Transaction` struct, whereas a lifetime on the *reference* to the `Transaction` struct (like you initially attempted) pertains to … well, the lifetime of that very reference.

If this is all still confusing, I encourage you to read the excellent chapter on lifetimes from the book. It is an integral part of the Rust programming paradigm.

Share
Improve this answer
Follow
edited Jun 18 '20 at 6:13

answered Jun 18 '20 at 2:03

Jorge Israel Peña
33.9k ● 15 ● 86 ● 118

1

This answer would be improved by discussing why the compiler gives this error.
– Shepmaster
Jun 18 '20 at 2:03

Jorge, thanks, it works, but honestly I don't know why. It will be really helpful if you can explain the answer a bit more, why `&tokio_postgres::Transaction<'_>` is a ref on struct `Transaction` while `&tokio_postgres::Transaction` is a ref on the ref to the struct?
– user3839198
Jun 18 '20 at 2:22

That's my fault, I just realized I used confusing language. What I meant was, where you needed the lifetime specifier `<'lifetime>` is on the type `Transaction`, i.e. `Transaction<'lifetime>`, you can liken it to how `Vec` on its own isn't a type (yet), but something like `Vec<String>` is, in other words, `Transaction` on its own is incomplete. I meant to say that `Transaction` is where you needed to add the lifetime parameter, not on the reference function parameter like you initially attempted.
– Jorge Israel Peña
Jun 18 '20 at 6:05

To make things more concrete, you use either `<'_>` (the anonymous lifetime) or you explicitly specify a lifetime parameter, I'll add that to my answer.
– Jorge Israel Peña
Jun 18 '20 at 6:07

Add a comment

Your Answer

Post Your Answer

Not the answer you're looking for? Browse other questions tagged rust lifetime rust-tokio or ask your own question.

The Overflow Blog

- Sequencing your DNA with a USB dongle and open source code
- Don't push that button: Exploring the software that flies SpaceX rockets and...

Featured on Meta

- Providing a JavaScript API for userscripts
- Congratulations to the 59 sites that just left Beta

Related

Why does the lifetime name appear as part of the function type?

Why is a lifetime needed when implementing a trait on a reference type if the lifetime is otherwise unused, in Rust < 1.31?

Does <'a, 'b: 'a> mean that the lifetime 'b must outlive the lifetime 'a?

Why can't I store a value and a reference to that value in the same struct?

Why does my trait need a lifetime parameter?

Why does the Rust compiler not optimize code assuming that two mutable references cannot alias?

Does a generic lifetime materialize as the reference's lifetime or the referenced value's lifetime?

How to resolve "indicate anonymous lifetime <'_>" error?

Hot Network Questions

How to plot molecules with angles and bond lengths

Why is the light source not showing but light is being cast on the object

Is Elon Musk really exploiting a loophole to avoid taxes?

Why does the prophecy imply Macbeth has to murder the king?

I've got a material setup that blends two shaders decently, but how would I apply it to a circular target?

more hot questions

Question feed

STACK OVERFLOW

Questions
Jobs
Developer Jobs Directory
Salary Calculator
Help
Mobile

PRODUCTS

Teams
Talent
Advertising
Enterprise

COMPANY

About
Press
Work Here
Legal
Privacy Policy
Terms of Service
Contact Us
Cookie Settings
Cookie Policy

STACK EXCHANGE NETWORK

Technology
Culture & recreation
Life & arts
Science
Professional
Business
API
Data

Blog
Facebook
Twitter
LinkedIn
Instagram