Hack postgres Source Code: Vol I

Chapter 1 : C & Rust

1.24 Generics - Rust

Generics in rust allow us to reduce the duplication code.

We can create generic functions, generic structures, generic enums and generic methods in rust.

A generic function - display can be defined as -

```
fn display<T>(a:T) -> T {
    a
}
```

This function can be used to display integers or floats. If there were no generics in rust, we would have to create two functions - one for integers and other for floats .

```
fn main() {

    let a: i32 = 10;

    println!("Display integer {}", display(a));

    let x: f32 = 10.0;

    println!("Display float {}", display(x));

}
```

In the similar way, this generics concept can be extended to structures.

```
struct User<I,N> {

    emp_id: I,
    emp_name: N,

}

impl<I,N> User<I,N> {

    fn display_emp_id(&self) -> &I {
```

```
            &self.emp_id
        }
    }

    fn main() {

        let emp_1: User<i32, String> = User { emp_id:100, emp_name:String::from("John") };

        println!("emp id is {} and emp name {}", emp_1.emp_id, emp_1.emp_name);

        println!("emp id is {}", emp_1.display_emp_id());
    }
```