Borrowed data escapes outside of closure.

Erroneous code example:

```
let mut list: Vec<&str> = Vec::new();

let _add = |el: &str| {
    list.push(el); // error: `el` escapes the closure body here
};
```

A type annotation of a closure parameter implies a new lifetime declaration. Consider to drop it, the compiler is reliably able to infer them.

```
let mut list: Vec<&str> = Vec::new();

let _add = |el| {
    list.push(el);
};
```

See the Closure type inference and annotation and Lifetime elision sections of the Book for more details.