Gowtham SB
www.linkedin.com/in/sbgowtham/ Instagram - @dataengineeringtamil

🧠 Python Guide: Closures

✅ What is a Closure?

> A **closure** is a function that **remembers** the variables from its **enclosing scope** — even after the outer function has finished executing.

In simple words:

> The **inner function** "remembers" the values passed to the **outer function".

🔍 Closure Example:

```
def outer(msg):
    def inner():
        return f"Message is: {msg}"
    return inner

say_hi = outer("Vanakkam da mapla")
print(say_hi())  # Output: Message is: Vanakkam da mapla
```

🔎 What's Happening Here?

Line	Meaning
`outer("Vanakkam da mapla")`	`msg = "Vanakkam da mapla"` is set
`inner()`	Uses `msg` from `outer()`
`return inner`	Returns the `inner()` function itself (not the result)

`say_hi()`	Executes the returned function, which still "remembers" `msg`

🎯 Even though `outer()` has finished, the `inner()` function **still has access** to `msg`.

✅ That's a **Closure**.

🎯 Real-Life Analogy:

Imagine a **custom voice recorder** app where:

```
recorder = create_recorder("Gowtham says:")
recorder("I am here")  # Output: Gowtham says: I am here
recorder("Let's go")   # Output: Gowtham says: Let's go
```

That **"Gowtham says:"** is remembered forever — just like how `msg` is remembered in a closure.

✅ Why Use Closures?

Benefit	Explanation
Remembers outer variables	Even when outer function is gone
Used in HOF Type 2	Return function with context saved
Build reusable logic	Like dynamic SQL builders, custom filters, etc.
Foundation for decorators	All Python decorators rely on closures

---

🛠 Closure Use Cases:

1. **SQL Query Builders**
2. **Custom Greeting Generators**
3. **Event Handlers**
4. **Decorators (`@login_required`)**
5. **Validation/Plugin Systems**

🧪 Debug Tip: See Closure with `.__closure__`

```
say_hi = outer("Vanakkam da mapla")
print(say_hi.__closure__[0].cell_contents)  # Output: Vanakkam da mapla
```

It shows what the closure **remembers**.

✅ Closure Pattern Template

```
def outer(value):
    def inner():
        # uses `value` here
        return ...
    return inner
```

💡 TL;DR:

A **closure** is when a function **returns another function**,
and that inner function **remembers the outer function's variables**, even after the outer is done.

About the Author

Gowtham SB is a **Data Engineering expert, educator, and content creator** with a passion for **big data technologies, as well as cloud and Gen AI** . With years of experience in the field, he has worked extensively with **cloud platforms, distributed systems, and data pipelines**, helping professionals and aspiring engineers master the art of data engineering.

Beyond his technical expertise, Gowtham is a **renowned mentor and speaker**, sharing his insights through engaging content on **YouTube and LinkedIn**. He has built one of the **largest Tamil Data Engineering communities**, guiding thousands of learners to excel in their careers.

Through his deep industry knowledge and hands-on approach, Gowtham continues to **bridge the gap between learning and real-world implementation**, empowering individuals to build **scalable, high-performance data solutions**.

Socials

YouTube - https://www.youtube.com/@dataengineeringvideos

Instagram - https://instagram.com/dataengineeringtamil

Instagram - https://instagram.com/thedatatech.in

Connect for 1:1 - https://topmate.io/dataengineering/

LinkedIn - https://www.linkedin.com/in/sbgowtham/

Website - https://codewithgowtham.blogspot.com

GitHub - http://github.com/Gowthamdataengineer

Whats App - https://lnkd.in/g5JrHw8q

Gowtham SB
www.linkedin.com/in/sbgowtham/ Instagram - @dataengineeringtamil

📧 **Email** - atozknowledge.com@gmail.com

📱 **All My Socials** - https://lnkd.in/gf8k3aCH