# Decorator

## Decorator

- Higher Order function

## Decorator

- Higher Order function
    - A wrapper over a pre existing function

# Decorator

- Higher Order function
    - A wrapper over a pre existing function
    - Extends the basic function’s ability

# Decorator

- Higher Order function
  - A wrapper over a pre existing function
  - Extends the basic function’s ability
  - Used to generalise over large number of functions

## Decorator Syntax : Extending a Function

## Decorator Syntax : Extending a Function

```
existing_function(params)
```

## Decorator Syntax : Extending a Function

```
#do something before a function
existing_function(params)
#do something after a function
```

## Decorator Syntax : Extending a Function

```
wrapper_function(params):
    #do something before a function
    existing_function(params)
    #do something after a function
```

# Decorator Syntax : Extending a Function

```
def decorator_function(existing_function):
    wrapper_function(params):
        #do something before a function
        existing_function(params)
        #do something after a function
    return wrapper_function
```

## Decorator Use Case

- Used with scalable frameworks : Dask, PySpark

## Decorator Use Case

- Used with scalable frameworks : Dask, PySpark
- Extensively used with model deployment

Notebook