# Proxy design pattern
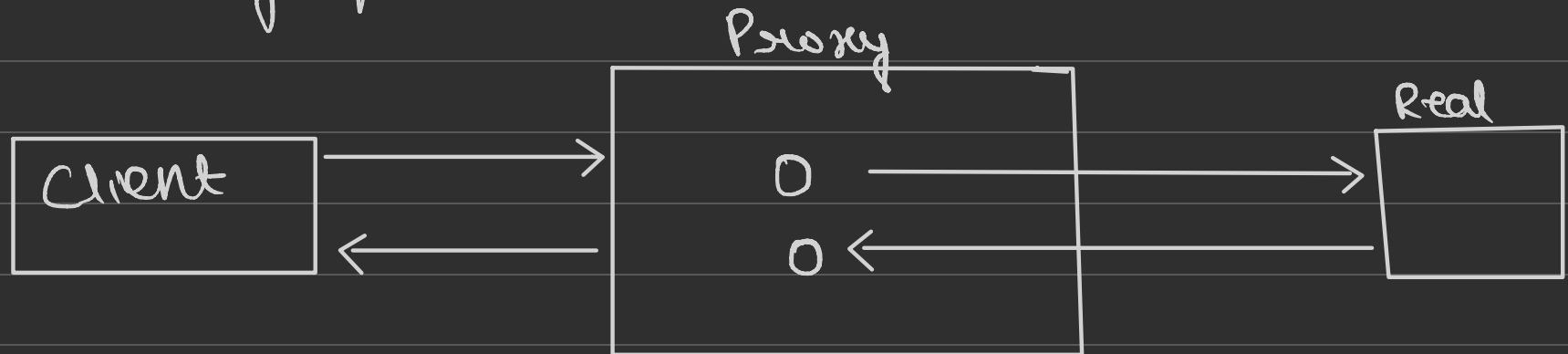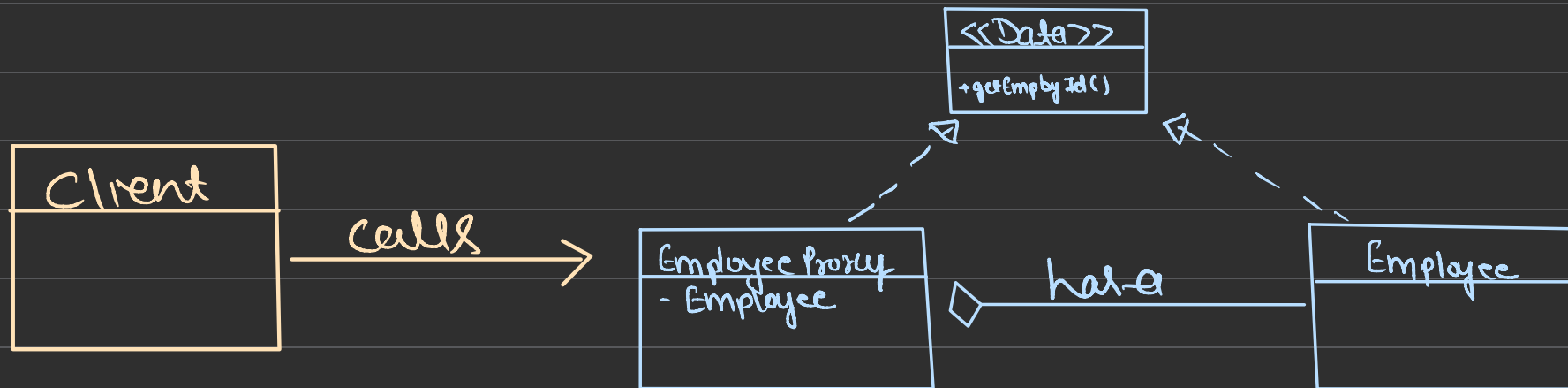
- Structural design pattern

- Placeholder for another object to control access
  to it.

- Call greeph

Proxy

Client ──────────▶ ┌─────────────┐ ──────────▶ Real
                   │      O      │            ┌──────┐
┌─────────┐        │             │            │      │
│ Client  │        │      O ◀──  │ ◀──────    │      │
└─────────┘ ◀───── │             │            └──────┘
                   └─────────────┘

# Cache -



```
Client  ──calls──▶  Employee Proxy
                    - Employee

                    <<Data>>
                    +getEmpby Id ( )

                    Employee

        has-a
```
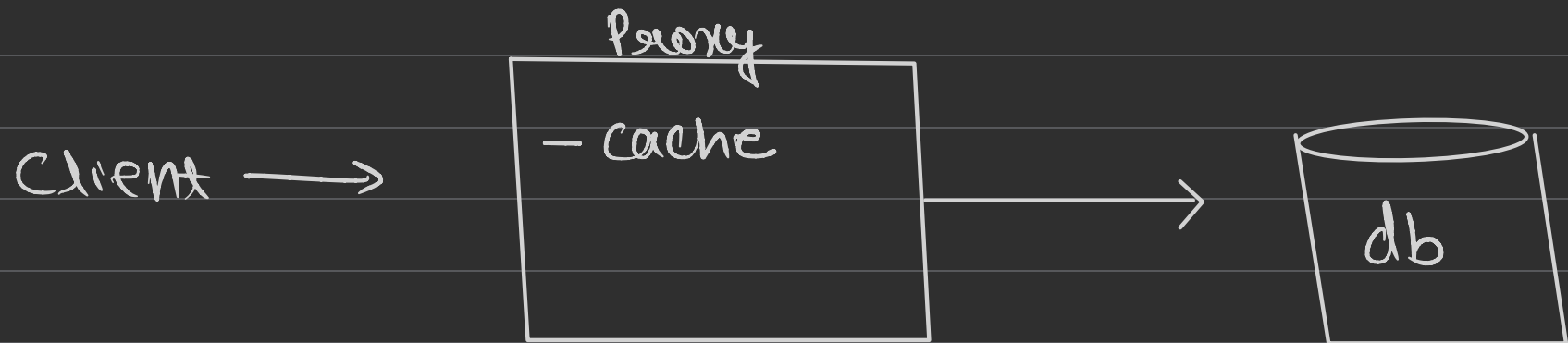
# How proxy can help—

① Cache

Proxy

Client ———→

| Proxy |
| --- |
| — Cache |

———→ db

② Access Control— proxy object can check whether request needs to be sent to real object or not? based on user's access. eg. few functions are available only for admins.

③ Lazy initialization — let's say we don't want to initialize the real object until we use it. thus can be achieve via proxy objects.