

Analiza el siguiente código en JavaScript y determina qué patrón de diseño se está utilizando y describirlos

```
1 class Database {
2   constructor() {
3     if (!Database.instance) {
4       Database.instance = this;
5     }
6     return Database.instance;
7   }
8
9   query(sql) {
10    console.log("Ejecutando consulta:", sql);
11  }
12 }
13
14 const db1 = new Database();
15 const db2 = new Database();
16
17 console.log(db1 === db2); // Output: true
18 db1.query("SELECT * FROM users");
```

Singleton

Singleton: Debido a la clase Logger

```
1 class Logger {
2   constructor() {
3     this.logs = [];
4   }
5
6   log(message) {
7     this.logs.push(message);
8     console.log("Log registrado:", message);
9   }
10
11   static getInstance() {
12     if (!Logger.instance) {
13       Logger.instance = new Logger();
14     }
15     return Logger.instance;
16   }
17 }
18
19 const logger1 = Logger.getInstance();
20 const logger2 = Logger.getInstance();
21
22 console.log(logger1 === logger2); // Output: true
23 logger1.log("Error: No se puede conectar al servidor");
```

```
1 class User {
2   constructor(name) {
3     this.name = name;
4   }
5
6   greet() {
7     console.log("Hola, soy", this.name);
8   }
9 }
10
11 class UserFactory {
12   createUser(name) {
13     return new User(name);
14   }
15 }
16
17 const factory = new UserFactory();
18 const user1 = factory.createUser("Juan");
19 const user2 = factory.createUser("Maria");
20
21 user1.greet(); // Output: Hola, soy Juan
22 user2.greet(); // Output: Hola, soy Maria
```