

El endpoint que está obteniendo mejores resultados es el uso de pools para conectarse con la base de datos desde el servidor de express. Esto se debe a que, al tener varias conexiones (en la configuración está de 5 a 10), el servidor es capaz de atender varios requests a la vez mediante el uso de promesas. También, lo que diferencia este endpoint con el ORM es que este se comunica con la base de datos mediante un stored procedure, que ya tiene un execution plan optimizado. El ORM traduce los métodos del objeto de la tabla a queries de SQL, pero el DBMS tiene que crear un execution plan y optimizar el query por lo que va a tardar más. El más lento en general es utilizar una única conexión, ya que, a pesar de que utiliza un stored procedure, sólo puede atender un request a la vez.

## First Test (Ejecutado después del app start)

The screenshot shows the Apache JMeter 5.5 interface. The left sidebar displays a tree view with 'Test Plan' expanded, containing 'Single', 'Pool', and 'ORM', each with an 'HTTP Request' child. The main panel is titled 'HTTP Request' and shows the configuration for the selected request. The 'Basic' tab is active, displaying the following settings:

- Name: HTTP Request
- Comments: (empty)
- Web Server: (empty)
- Protocol [http]: (empty)
- Server Name or IP: localhost
- Port Number: 3000
- HTTP Request: GET
- Path: /api/single/Cargar%20vacios%20a%20camion
- Content encoding: (empty)
- ☐ Redirect Automatically
- ☒ Follow Redirects
- ☒ Use KeepAlive
- ☐ Use multipart/form-data
- ☐ Browser-compatible headers

The 'Parameters' tab is also visible, showing a table for parameters to be sent with the request:

Name	Value	URL Encode?	Content-Type	Include Equals?
------	-------	-------------	--------------	-----------------

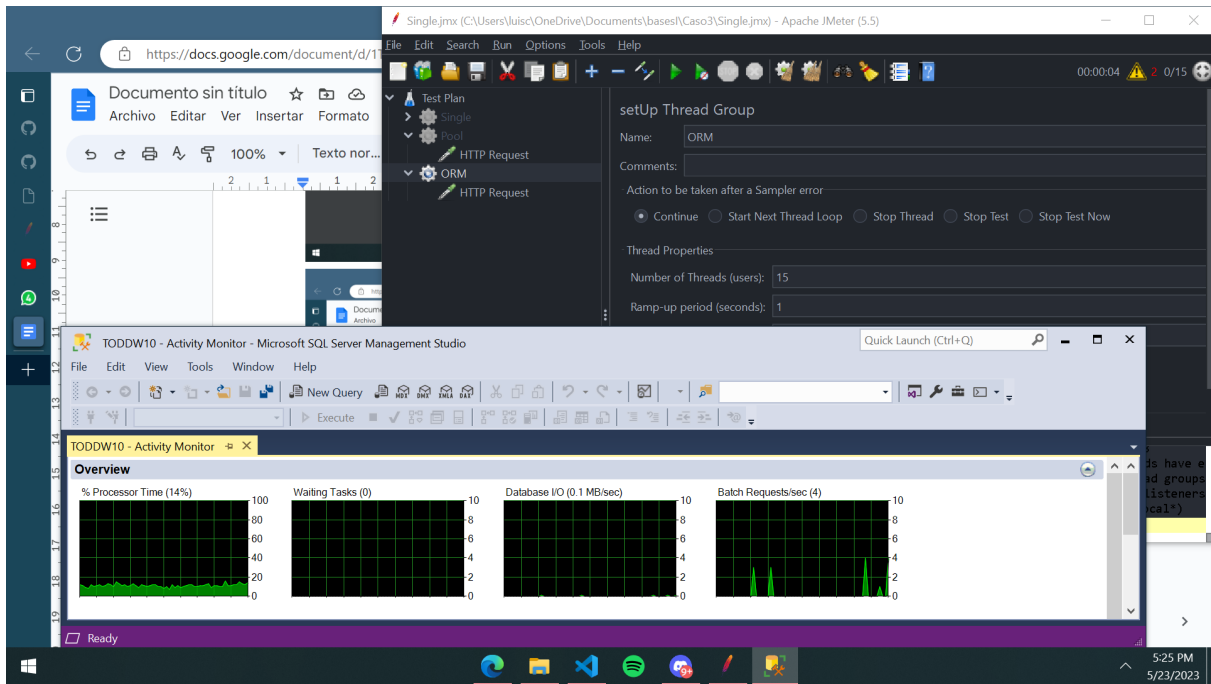
At the bottom, a log window displays the following messages:

```
74 2023-05-23 17:10:44,115 INFO o.a.j.t.JMeterThread: Inread is done: Single 1-15
75 2023-05-23 17:10:44,116 INFO o.a.j.t.JMeterThread: Thread finished: Single 1-15
76 2023-05-23 17:10:44,116 INFO o.a.j.e.StandardJMeterEngine: All Setup Threads have ended
77 2023-05-23 17:10:44,315 INFO o.a.j.e.StandardJMeterEngine: No enabled thread groups found
78 2023-05-23 17:10:44,315 INFO o.a.j.e.StandardJMeterEngine: Notifying test listeners of end of test
79 2023-05-23 17:10:44,315 INFO o.a.j.g.u.JMeterMenuBar: setRunning(false, *local*)
80
```

This screenshot shows the Apache JMeter 5.5 interface with two additional windows open. The top-left window is a Google Docs document titled 'Documento sin título'. The bottom window is the 'TODDW10 - Activity Monitor - Microsoft SQL Server Management Studio', which displays an 'Overview' section with four performance graphs:

- % Processor Time (16%): A line graph showing processor usage over time, with values generally below 20%.
- Waiting Tasks (0): A line graph showing the number of waiting tasks, which remains at 0.
- Database I/O (0.1 MB/sec): A line graph showing database input/output activity, with values around 0.1 MB/sec.
- Batch Requests/sec (4): A line graph showing the number of batch requests per second, with values around 4.

The JMeter interface in the background shows the 'HTTP Request' configuration for the 'Pool' test plan, with the path set to /api/pool/Cargar%20vacios%20a%20camion.



## Second Test (Ejecutado después de un restart del app)

