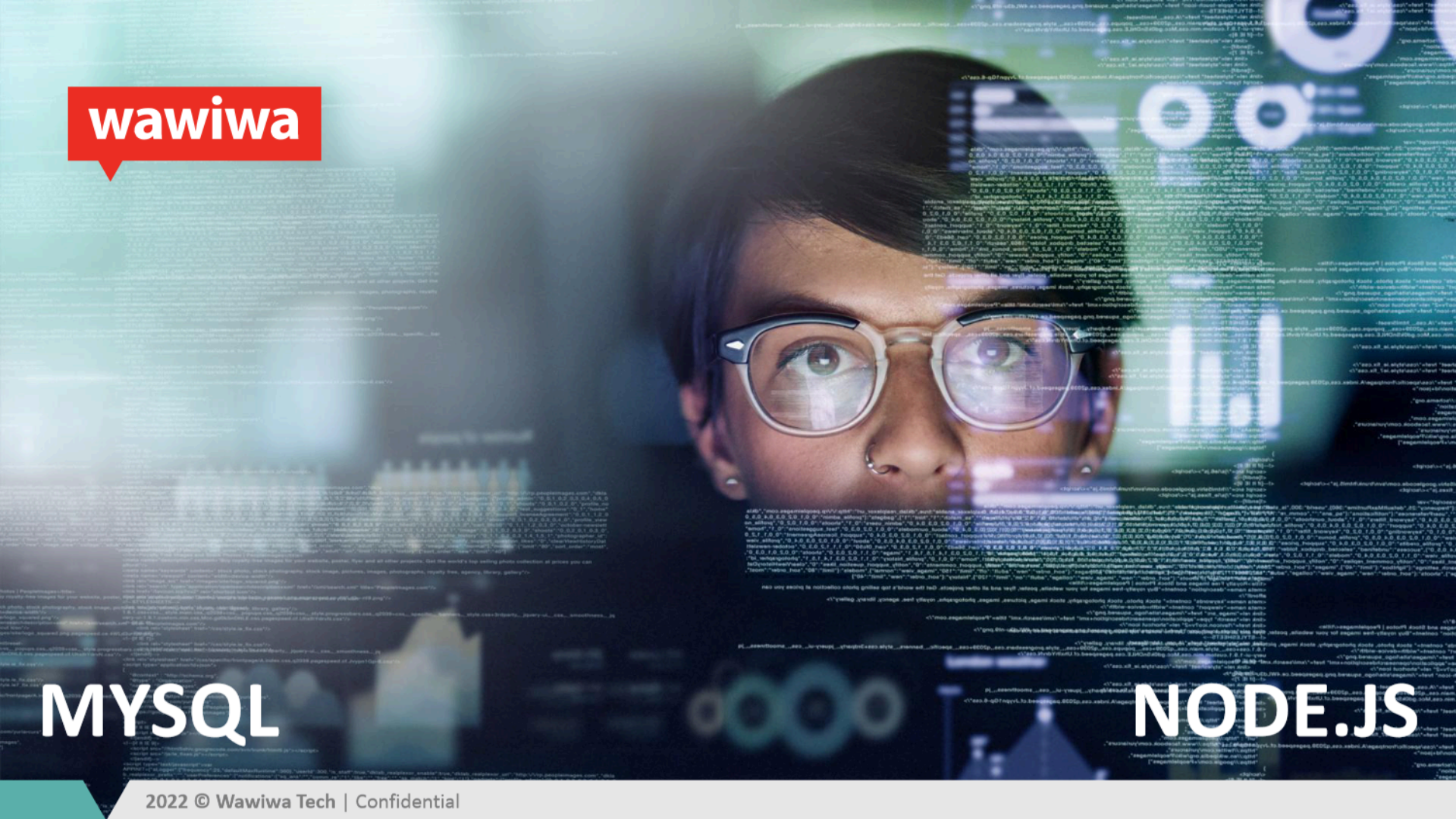


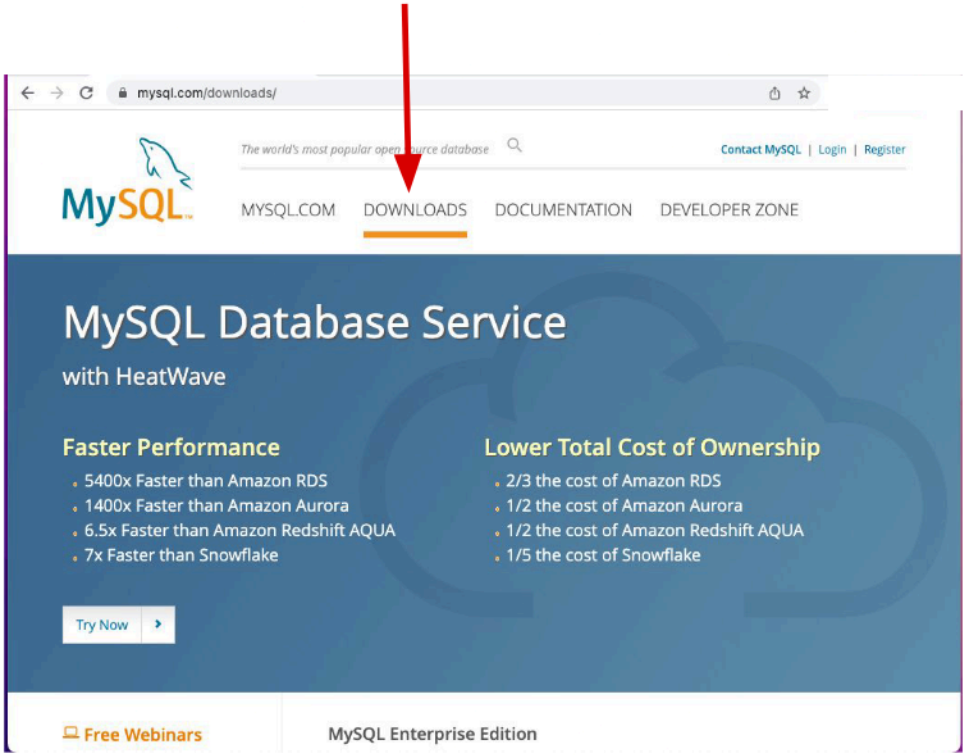
Base de datos MySQL

MySQL es un sistema de gestión de bases de datos relacionales basado en SQL (lenguaje de consulta estructurado). ... Sin embargo, el uso más común de MySQL es como base de datos web. Se puede utilizar para almacenar cualquier cosa, desde un único registro de información hasta un inventario completo de productos disponibles para una tienda en línea.

mysql.com

Descarga MySQL

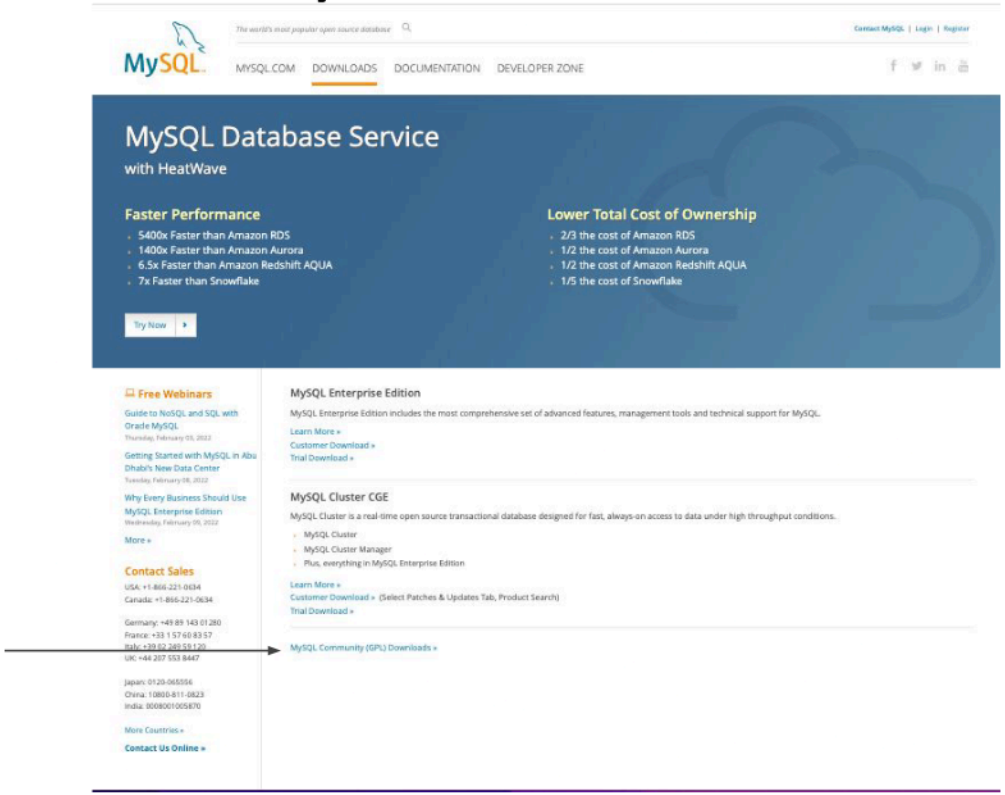
Ve a mysql.com



wawiwa

Descarga el “community server”

Ve a mysql.com

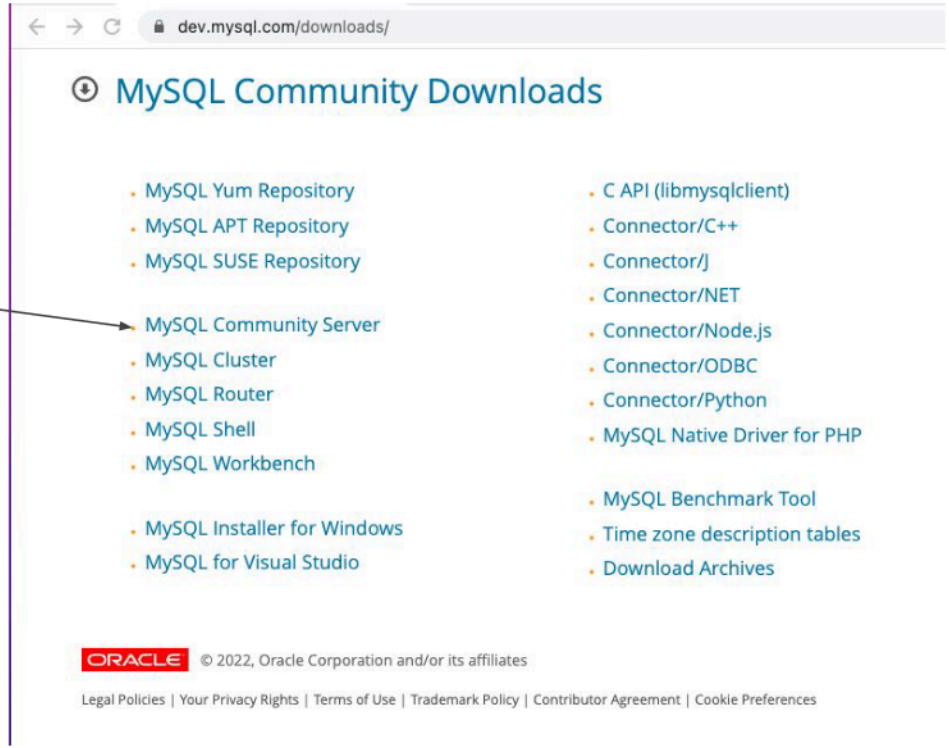


Haz click en:MySQL Community (GPL) Descargas »

Descarga el “community server”

Ve a mysql.com

Descarga: MySQL Community Server

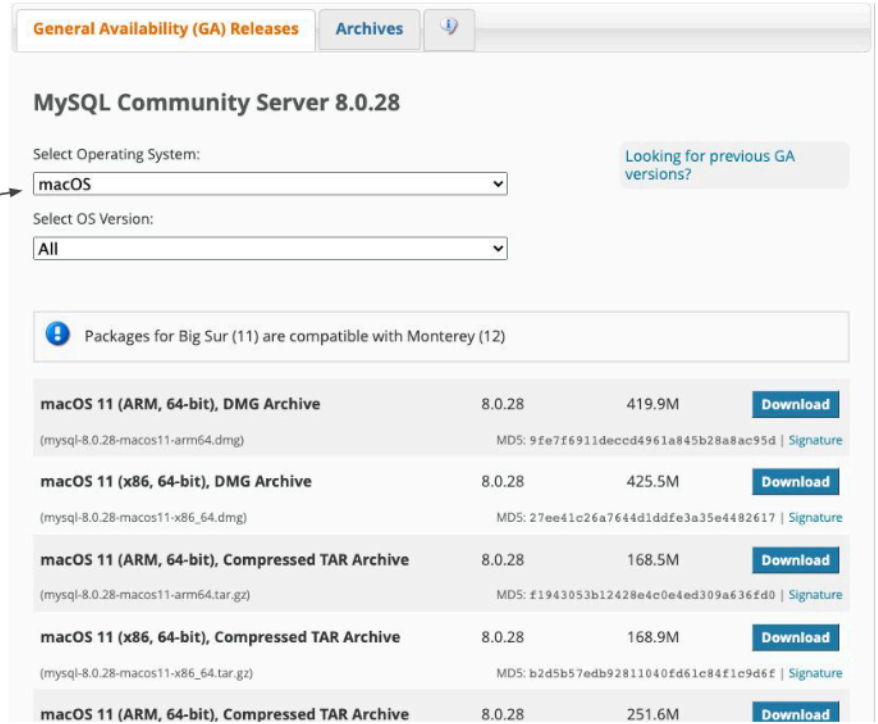


wawiwa

Descarga mysql

Ve a mysql.com

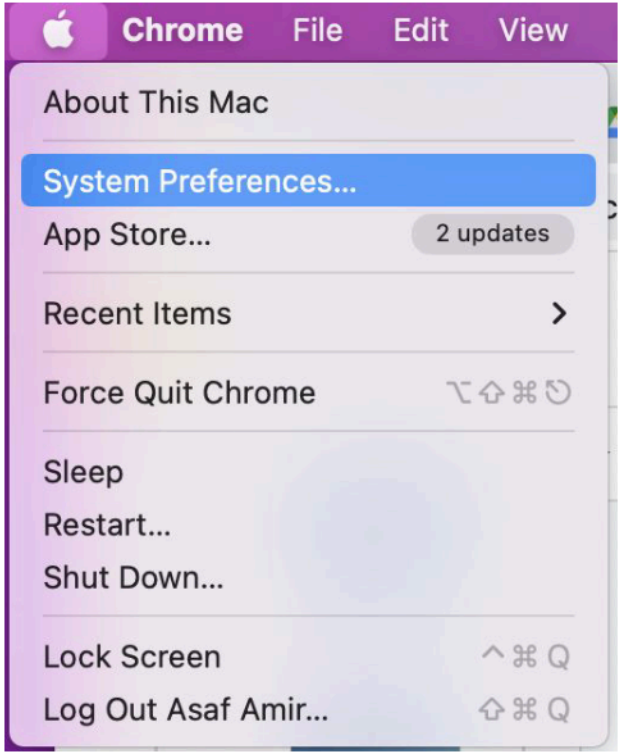
Selecciona tu sistema operativo y descarga:



Inicia el servidor

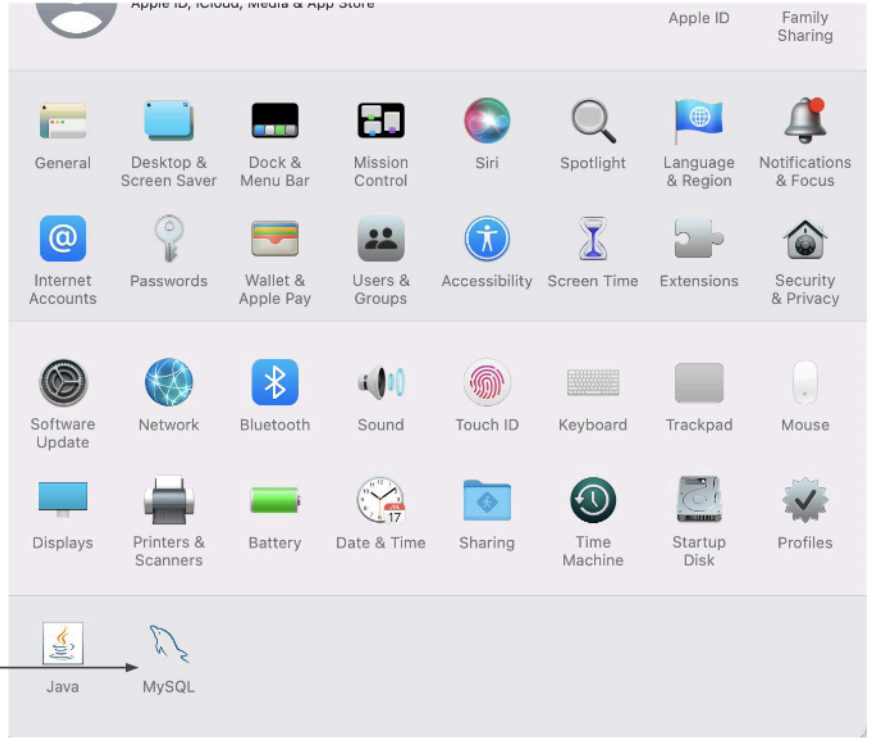
Desde mac :

Ve a



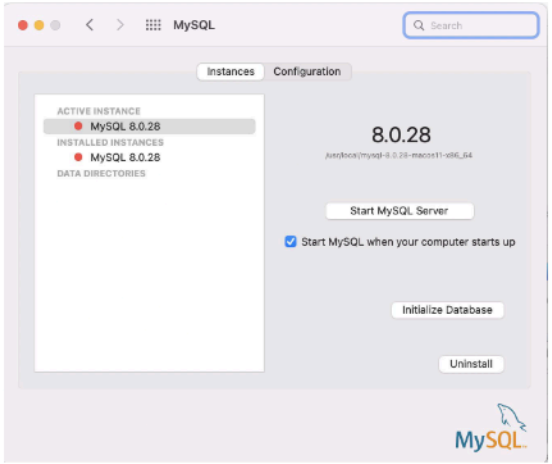
wawiwa

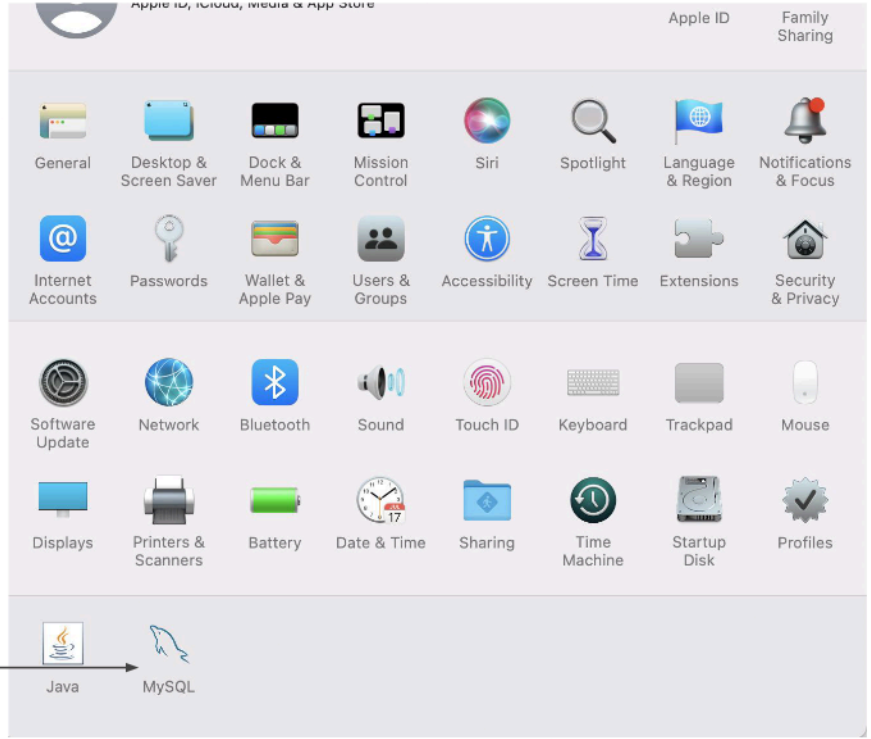
Inicia el servidor



Click on mysql

Inicia el servidor

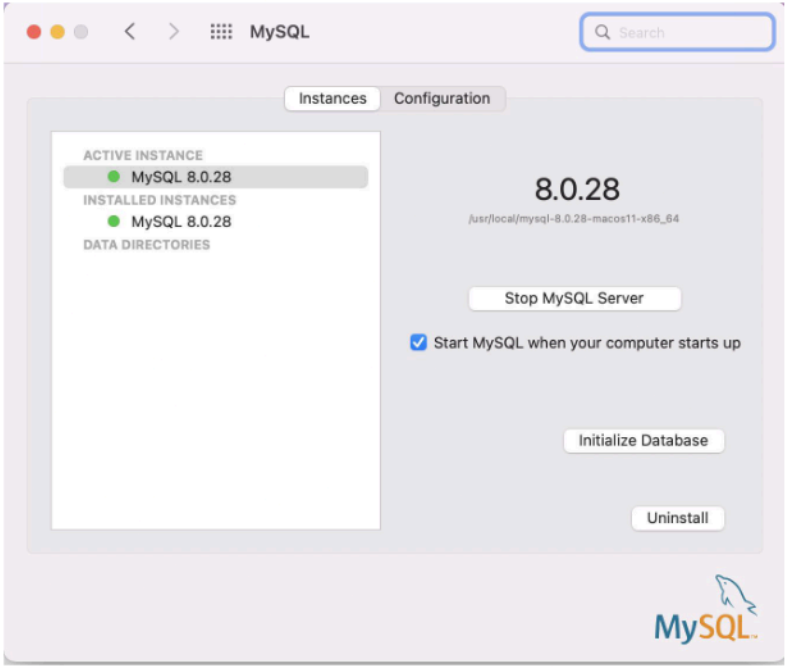




Click on mysql

Después de instalar, incial el servidor

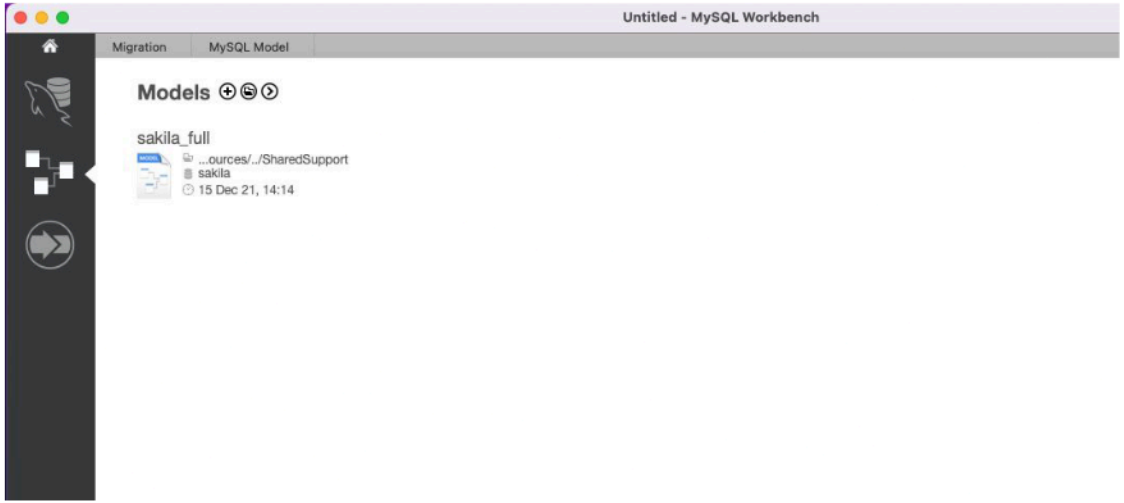
Ve a mysql.com



MySQL - descarga workbench

https://dev.mysql.com/downloads/workbench/

Después de instalarlo, ábrelo:

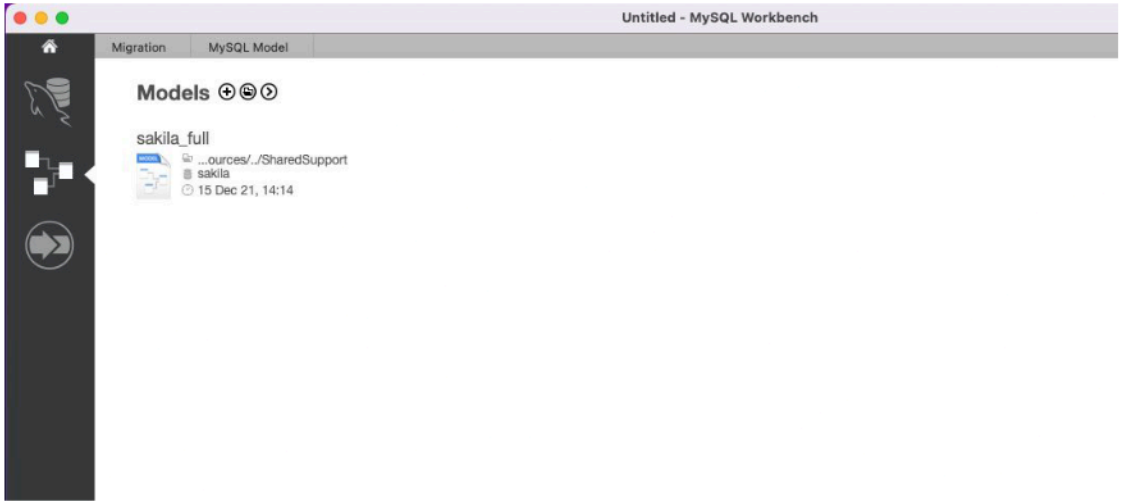


wawiwa

MySQL

https://dev.mysql.com/downloads/workbench/

Después de instalarlo, ábrelo:



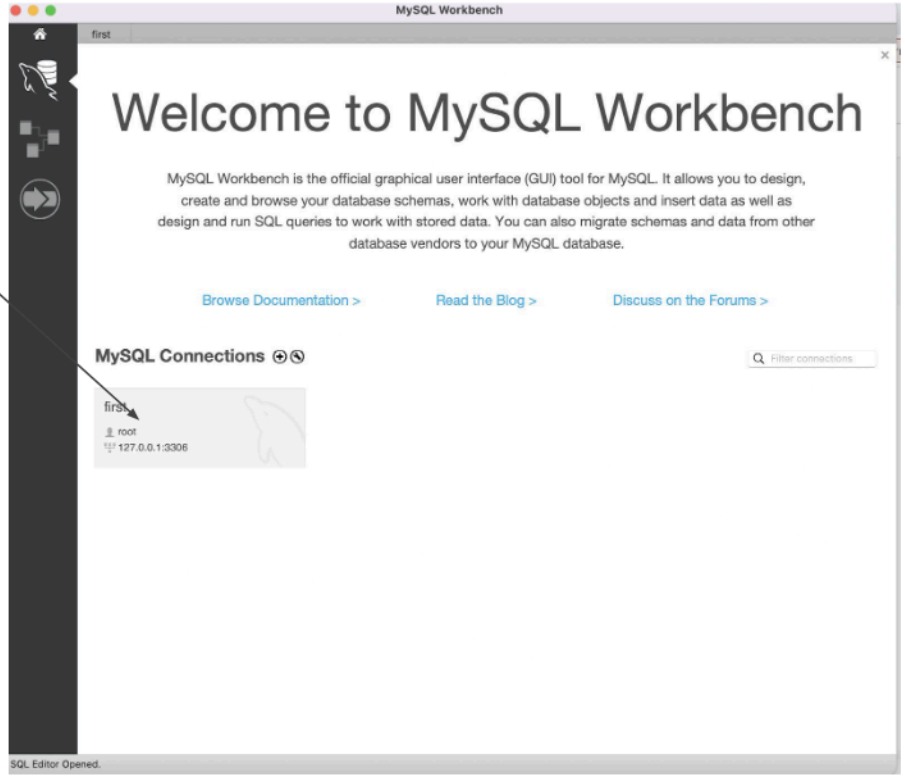
Agrega un módulo

Haz click en el ícono + y agrega un módulo

wawiwa

Agrega una conexión

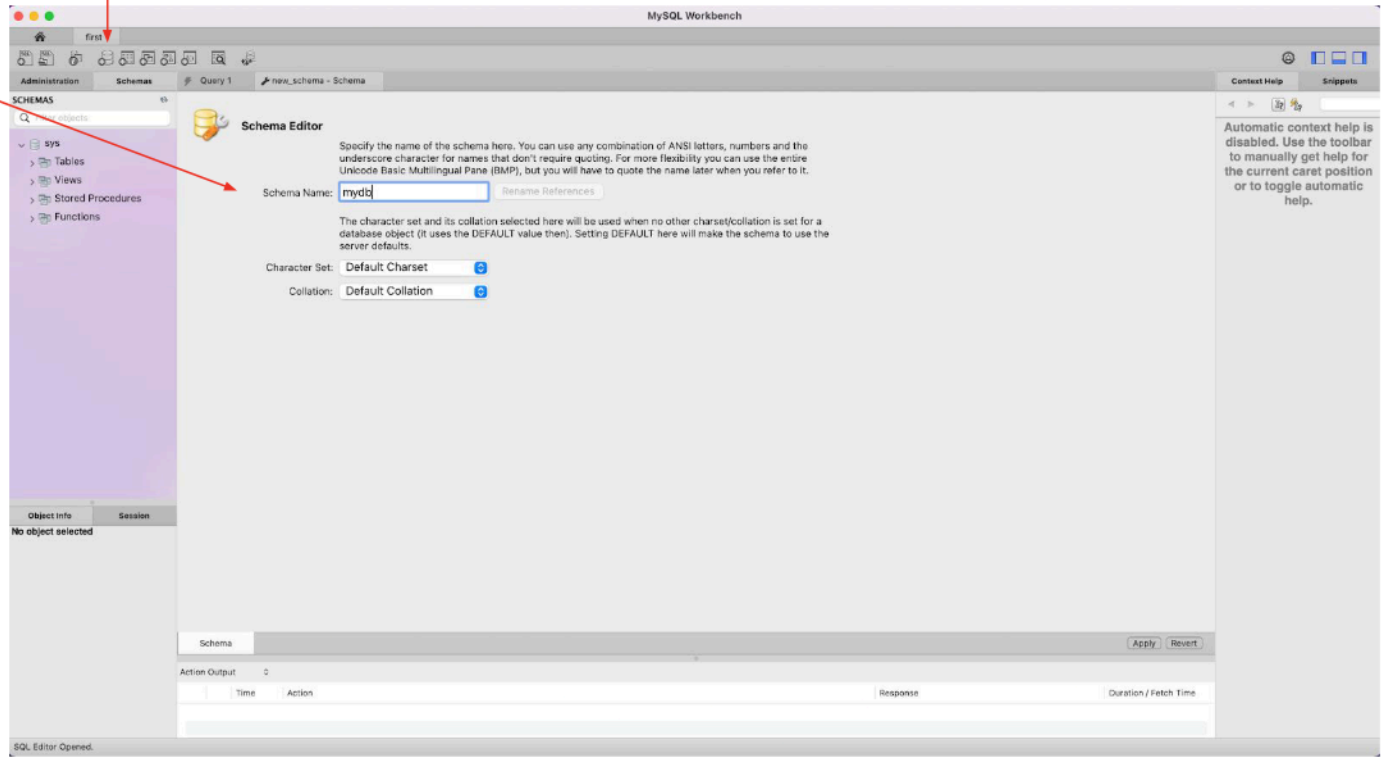
Da click en connection



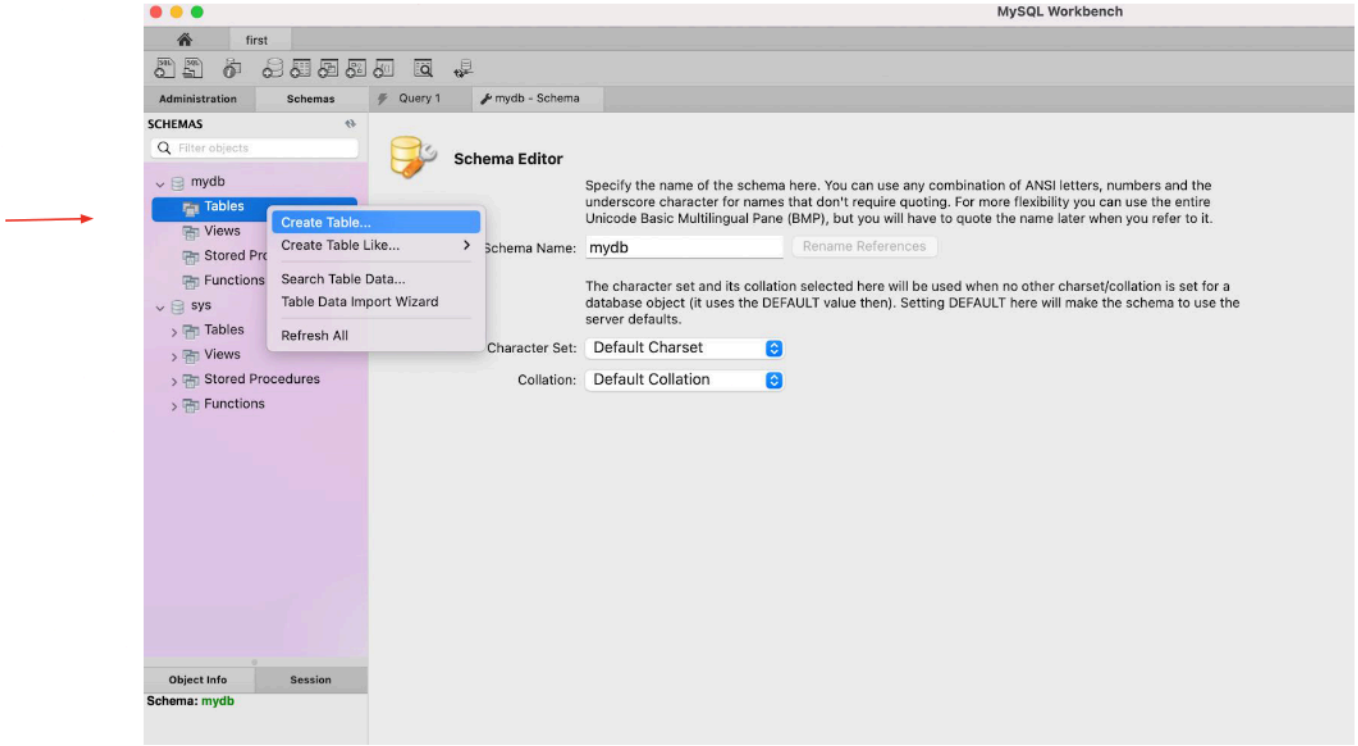
wawiwa

Crea una base de datos

Crea base de datos

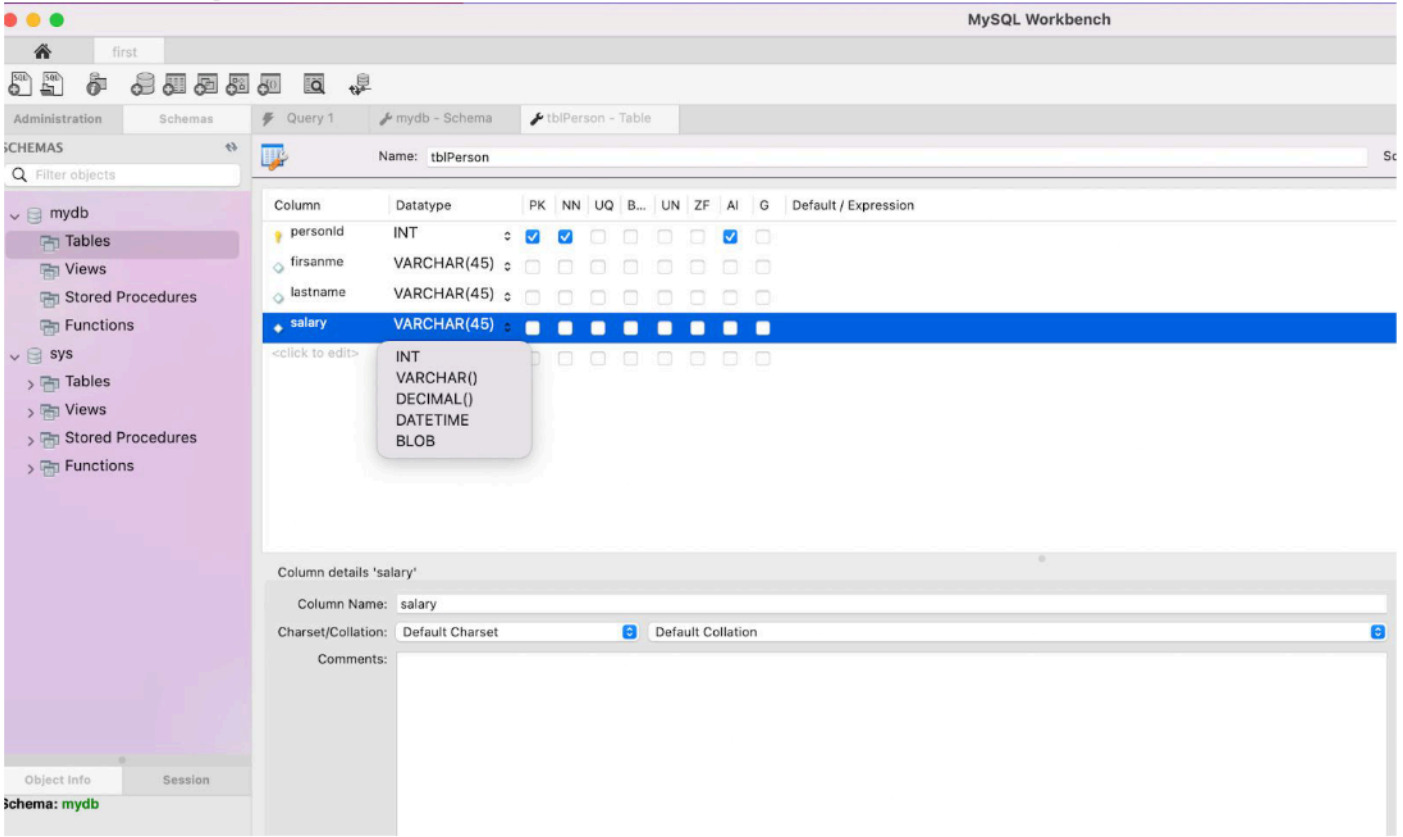


Crea una tabla



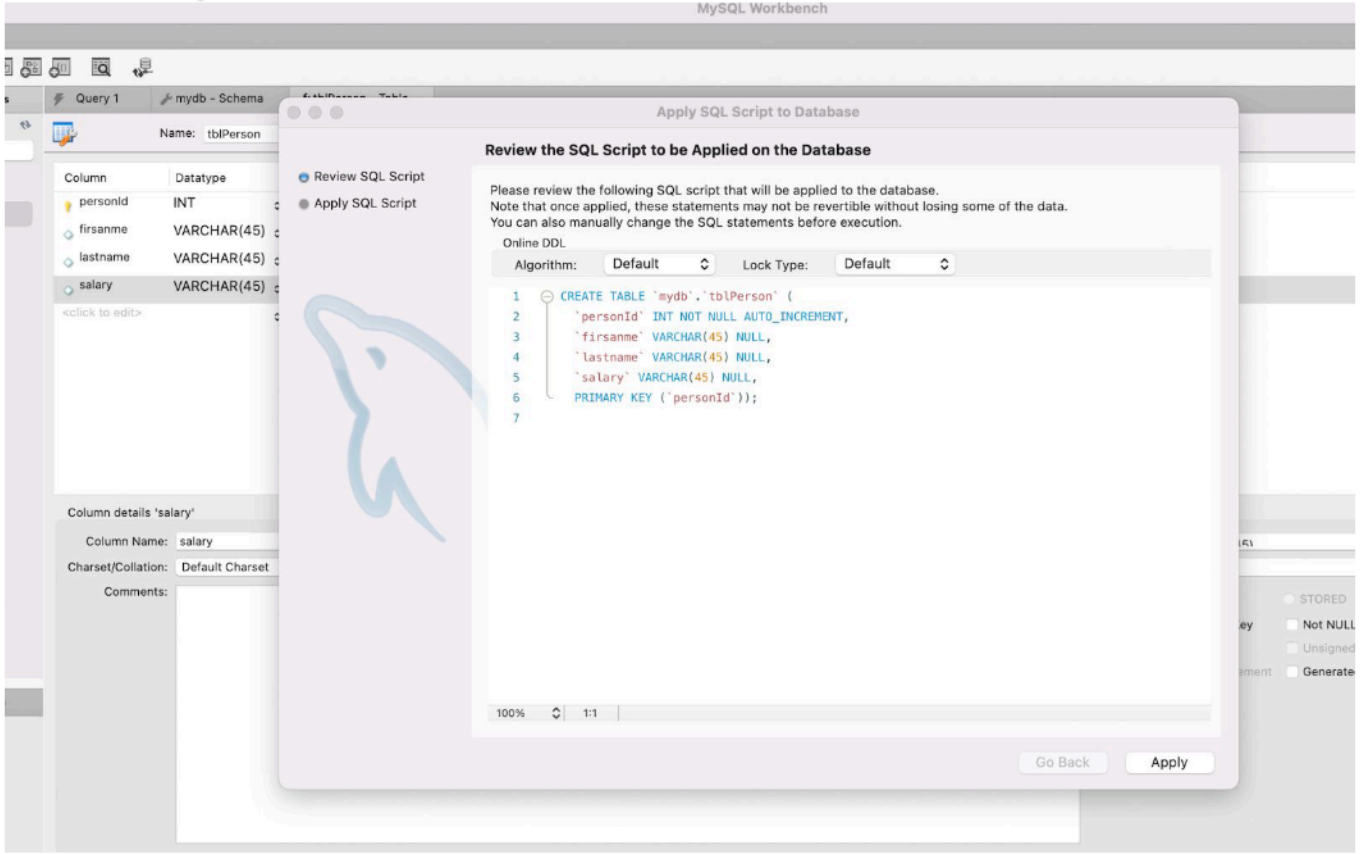
wawiwa

Agrega campos



wawiwa

Agrega campos



wawiwa

NODEJS Y MYSQL

wawiwa

Crea una conexión

```
const express = require('express')
const mysql = require('mysql');
const app = express()
const PORT = process.env.PORT || 3000
app.get('/', (req, res) => {
 res.send('Hello Heroku!')
})
const con = mysql.createConnection({
 host: "localhost",
 user: "root",
 password: "asaf7452702"
});

con.connect(function(err) {
 if (err) throw err;
 console.log("Connected!");
 });

app.listen(PORT, () => {
 console.log(`Example app listening on port ${PORT}`)
})
```

```
asafamir@Asafs-MBP chap18-heroku-app % node server
Example app listening on port 3000
Connected!
```

wawiwa

Crea una base de datos

Creamos una base de datos en MySQL, pero también lo puedes hacer con código desde node.

```
con.connect(function(err) {
 if (err) throw err;
 console.log("Connected!");
 con.query("CREATE DATABASE mydb", function (err, result) {
   if (err) throw err;
 console.log("Database created");
 });
});
```

wawiwa

Crea una tabla

Creamos una tblPerson en MySQL, pero también lo puedes hacer con código desde node.

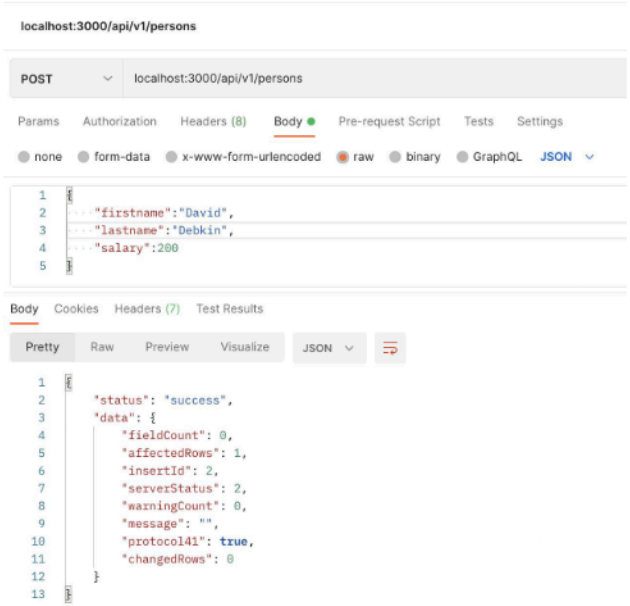
```
con.connect(function(err) {
 if (err) throw err;
 console.log("Connected!");
 createTablePerson()
});
function createTablePerson(){
 let sql = `CREATE TABLE if not exists mydb.tblPersons(personId INT
NOT NULL AUTO_INCREMENT,
   firstname VARCHAR(45) NULL, lastname VARCHAR(45) NULL,
   salary INT NULL,  PRIMARY KEY (personId))`;
   con.query(sql, function (err, result) {
   if (err) throw err;
   console.log("Table created");
 });
}
```

```
@Asafs-MBP chap18-heroku-app % node server
Example app listening on port 3000
Connected!
Table created
```

wawiwa

Inserta un registro

```
app.post('/api/v1/persons', function(req, res, next) {
 try{
   let sql = `INSERT INTO mydb.tblPersons (firstname, lastname, salary) VALUES
('${req.body.firstname}','${req.body.lastname}','${req.body.salary}')`;
    con.query(sql, function (err, result) {
        if (err) throw err;
        res.status(201).json({
          status:"success",
          data:result
        })
      });
 }
 catch(err){
      res.status(400).json({
          status:"fail",
          message:"error:😱" + err
      })
 }
});
```

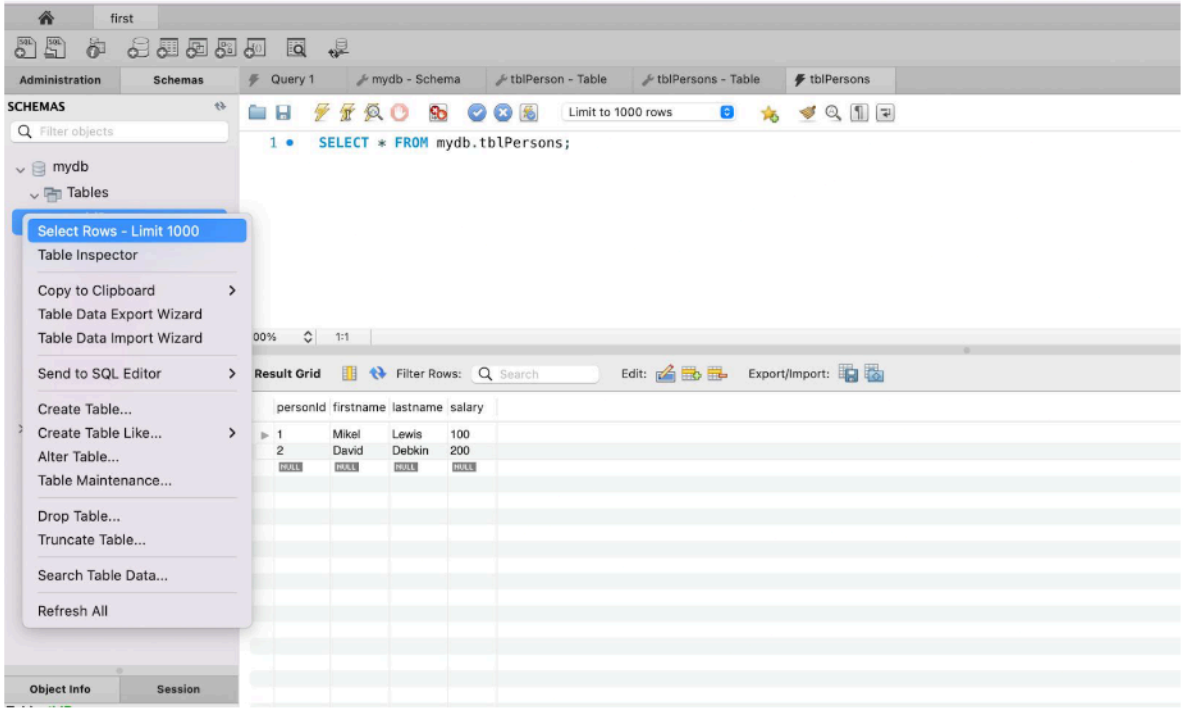


wawiwa

Actualiza un registro

```
app.patch('/api/v1/persons', function(req, res, next) {
 console.log(req.body)
 try{
   const sql = `UPDATE mydb.tblPersons SET firstname = '${req.body.firstname}' WHERE
personId = ${req.body.personId}`;
   console.log(sql)
   con.query(sql, function (err, result) {
       if (err) throw err;
       res.status(201).json({
         status:"success",
         data:result
       })
     });
 }
 catch(err){
     res.status(400).json({
         status:"fail",
         message:"error:😱" + err
     })
 }
});
```

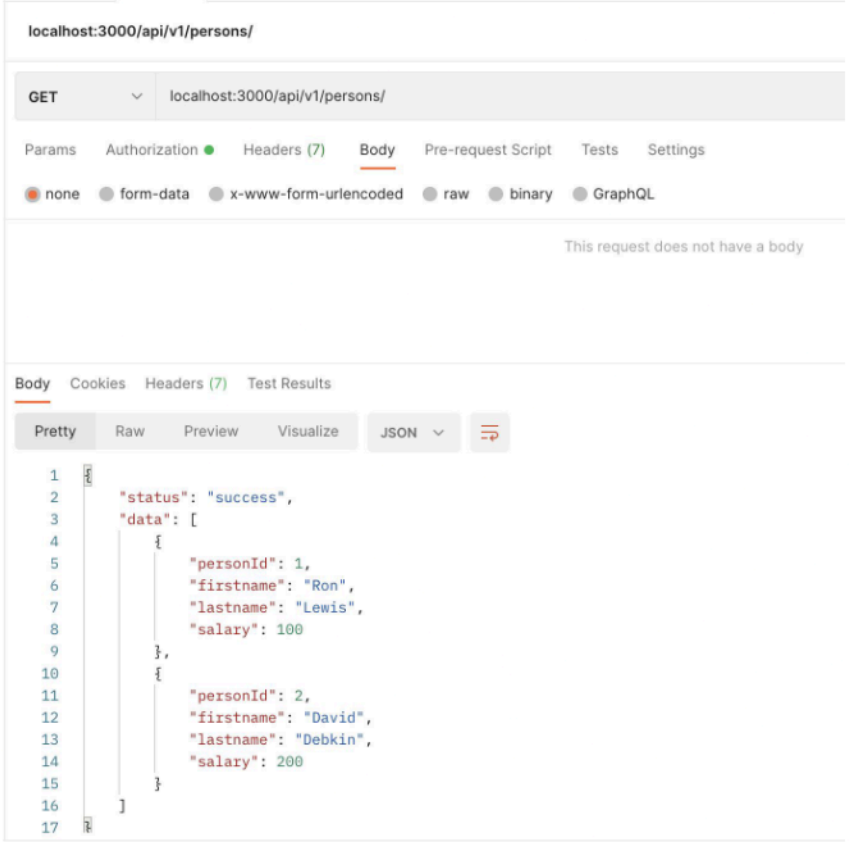
Antes de actualizar el registro



wawiwa

Selecciona todos los registros

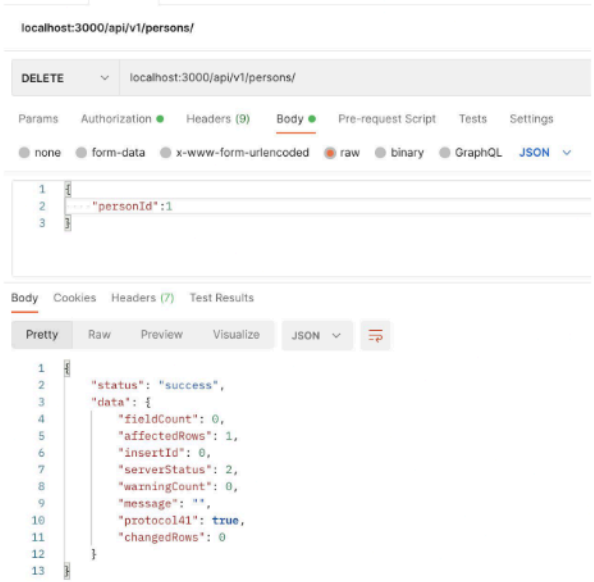
```
app.get('/api/v1/persons', (req, res) => {
 try{
   let sql = `SELECT * FROM mydb.tblPersons;`;
   con.query(sql, function (err, result) {
       if (err) throw err;
       res.status(201).json({
         status:"success",
         data:result
       })
     });
 }
 catch(err){
     res.status(400).json({
         status:"fail",
         message:"error:😱" + err
     })
 }
});
```



wawiwa

Borra

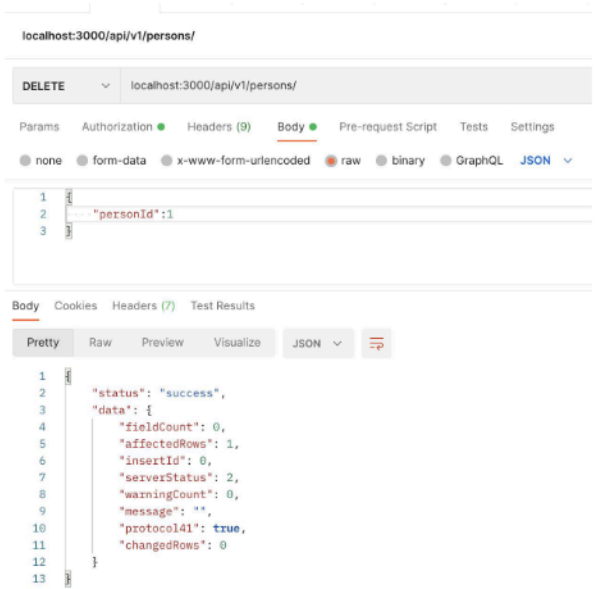
```
app.delete('/api/v1/persons', function(req, res, next) {
 console.log(req.body)
 try{
   const sql = `delete from mydb.tblPersons WHERE personId = ${req.body.personId}`;
   con.query(sql, function (err, result) {
       if (err) throw err;
       res.status(201).json({
         status:"success",
         data:result
       })
     });
 }
 catch(err){
     res.status(400).json({
         status:"fail",
         message:"error:😱" + err
     })
 }
});
```



wawiwa

Borra

```
app.delete('/api/v1/persons', function(req, res, next) {
 console.log(req.body)
 try{
   const sql = `delete from mydb.tblPersons WHERE personId = ${req.body.personId}`;
   con.query(sql, function (err, result) {
       if (err) throw err;
       res.status(201).json({
         status:"success",
         data:result
       })
     });
 }
 catch(err){
     res.status(400).json({
         status:"fail",
         message:"error:😱" + err
     })
 }
});
```



wawiwa

Más ejemplos

```
con.query("SELECT * FROM mydb.tblPersons ORDER BY firstname", function (err, result) {}

con.query("DROP TABLE IF EXISTS mydb.tblPersons", function (err, result) {}

con.query("SELECT * FROM mydb.tblPersons LIMIT 10"", function (err, result) {}
```

wawiwa

Ejemplos de “Join”

```
con.query("SELECT mydb.tblPersons.firstname AS firstname,
mydb.tblPersons.lastname AS lastname FROM mydb.tblPersons JOIN
cities ON mydb.tblPersons.cityId = mydb.cities.cityId", function
(err, result) {}
```

wawiwa

Hazlo tú mismo

Ingresa la siguiente tabla en MySQL,

tblProduct (productId, name, description, price, category, quantity)

Escribe las siguientes solicitudes en node.js:

1. Ingresa 5 “"products"” en la tabla agregando un comando
2. Muestra todos los “"products"” en la tabla.
3. Ve todas las “categories”
4. Muestra solamente” productId” y “price”.
5. Ve todos los "products" en la “category” 8
6. Ve todos los "products" que su código es 10 o 20 y su “price” entre 100 y 200
7. Ve todos los "products" ordenados por su “name”.
8. Borra todos los "products" que su “quantity” sea igual a 0


wawiwa
¿Preguntas?