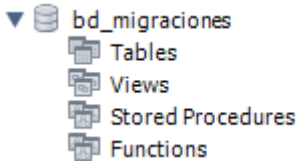
# Migraciones

Pasos para crear migraciones, **un ejemplo en el cual se crean dos tablas (usuario y publicación)**

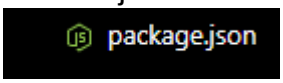
1. Primero se debe crear un base de datos vacía en mysql, ya que más adelante se necesita el nombre de la base de datos, para este ejemplo se crea la base de datos llamada: bd_migraciones en la aplicación “MySQL Workbench”




2. Se ejecutan los siguientes comandos en la terminal siguiendo el mismo orden
   Primero se ejecuta:

   **npm init -y**

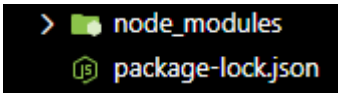
   Es el comando para iniciar un proyecto Node.js de forma rápida y automática y esto crea:




3. Al ejecutar el siguiente comando

   **npm install prisma@5.20 --save-dev**

   Se genera los siguientes archivos, esto hace que se instale Prisma como herramienta de desarrollo, es decir, solo para configurarlo, generar el cliente y manejar migraciones, pero **no** para usarse en producción.




4. Al ejecutar el siguiente comando

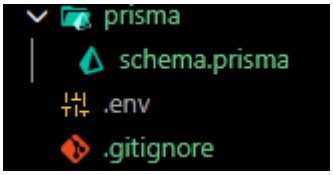
   **npm install @prisma/client@5.20**

   Se usa para instalar una versión específica y más estable de Prisma (la 5.20), evitando problemas o cambios recientes de la versión 7 que todavía no es totalmente estable.

5. Al ejecutar el siguiente comando

   **npx prisma init**

   Genera los siguientes archivos, crea la estructura básica de Prisma en tu proyecto, generando la carpeta *prisma/* con el archivo *schema.prisma* y un archivo *.env* donde irá la conexión a la base de datos.



6. Se modifica del archivo ".env" el "DATABASE_URL" que trae por defecto y se cambia por la siguiente estructura

**DATABASE_URL="mysql://ADMINSTRADOR:CONTRASEÑA@localhost:PUERTO/BASE_DE_DATOS"**

Por lo tanto se debe tener la siguiente información:

- Administrador (por defecto es *root*, si no ha sido personalizado)
- Contraseña (si tiene o no)
- Puerto (puede ser 3306 o 3307 u otra si ha sido personalizado)
- Nombre de la base de datos

En mi caso utilizo "MySQL Workbench", y para crear alguna base de datos en la aplicación solicita un usuario que es "root" y una contraseña la cual es "soft123", su puerto de conexión configurado es "3306" y el nombre de la base de datos es "bd_migraciones". Recuerde que esta información puede cambiar en su computadora

Según los datos así queda esa línea de código del DATABASE_URL en el archivo ".env"

DATABASE_URL="mysql://root:soft123@localhost:3306/bd_migraciones"

En caso de que no pida contraseña no se escribe en el espacio que le corresponde, ejemplo:

DATABASE_URL="mysql://root:@localhost:3306/bd_migraciones"

7. Se modifica el archivo "schema.prisma", aquí es donde se define la base de datos para Prisma: aquí escribes los modelos (tablas), sus campos, relaciones y el proveedor que usarás en este caso (MySQL)

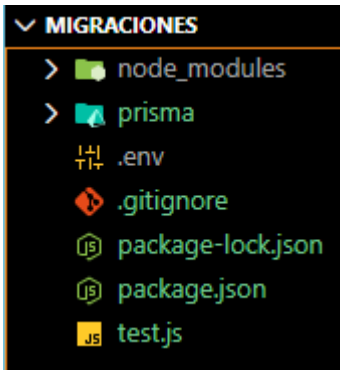
```
generator client {
  provider = "prisma-client-js"
}
datasource db {
  provider = "mysql"
  url      = env("DATABASE_URL")
}
model Usuario {
  id     Int    @id @default(autoincrement())
  nombre String
  email  String @unique
  publicaciones  Publicacion[]
}
model Publicacion {
  id          Int @id @default(autoincrement())
  titulo      String
  contenido   String
  usuario     Usuario    @relation(fields: [usuarioId], references: [id])
  usuarioId   Int
}
```

**8.** Se ejecuta el comando

**npx prisma generate**

Sirve para crear automáticamente el código del Prisma Client, que es la librería que te permite usar tu base de datos desde JavaScript de forma fácil (consultar, insertar, actualizar, etc.). si se ejecuta y no existen errores en la terminal entonces todo estaría correcto hasta el momento

**9.** Se realiza una prueba de conexión a la base de datos, creando un archivo llamado **test.js** en el proyecto




El cual contiene el siguiente código

```
const { PrismaClient } = require('@prisma/client');
const prisma = new PrismaClient();

async function main() {
    try {
        await prisma.$connect();
        console.log("Prisma se conectó correctamente");
    } catch (error) {
        console.error("Error al conectar Prisma:", error);
    } finally{
        await prisma.$disconnect();
    }
}
main();
```

**10.** Se ejecuta el comando

**node test.js**

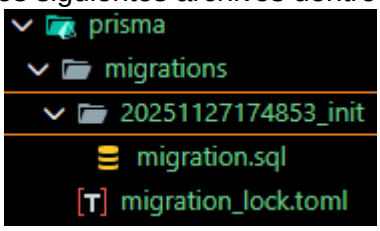
Y debe mostrar lo siguiente en la terminal

```
Prisma se conectó correctamente
```

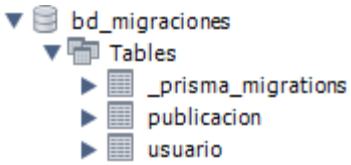
**11.** Una vez comprobado que existe conexión se procede a realizar las migraciones creando las tablas en la base de datos, por lo tanto se ejecuta el siguiente comando

**npx prisma migrate dev --name init**

Esto genera la carpeta migrations y los siguientes archivos dentro de ella




Y automáticamente se crean en la base de datos las tablas, finalizando así la migración




---

## Código schema.prisma

```
generator client {
  provider = "prisma-client-js"
}
datasource db {
  provider = "mysql"
  url      = env("DATABASE_URL")
}
model Usuario {
  id           Int           @id @default(autoincrement())
  nombre       String
  email        String    @unique
  publicaciones  Publicacion[]
}
model Publicacion {
  id             Int          @id @default(autoincrement())
  titulo         String
  contenido   String
  usuario      Usuario    @relation(fields: [usuarioId], references: [id])
  usuarioId    Int
}
```

## Código test.js

```
const { PrismaClient } = require('@prisma/client');
const prisma = new PrismaClient();

async function main() {
	try {
		await prisma.$connect();
		console.log("Prisma se conectó correctamente");
	} catch (error) {
		console.error("Error al conectar Prisma:", error);
	} finally {
		await prisma.$disconnect();
	}
}
main();
```