# 1.新建一个项目mysql\_prisma

|  |
| --- |
|  |

# 2.安装prisma ，使用命令yarn add prisma，完成后如图

|  |
| --- |
|  |

# 3.初始化prisma，把数据库提供商设置位mysql，使用yarn dlx prisma init --datasource-provider mysql

|  |
| --- |
|  |

# 4.执行这个命令后会在项目根目录下面生成一个prisma文件夹，里面有一个schema.prisma，并且会在prisma文件夹外面生成一个.env文件。

|  |
| --- |
|  |

# 5.修改.env文件，把用户名和密码以及数据库名称改为自己的设置

|  |
| --- |
|  |

# 6.然后修改schema.prisma，添加两个模型，Student和Class

|  |
| --- |
| // This is your Prisma schema file,  // learn more about it in the docs: https://pris.ly/d/prisma-schema  // Looking for ways to speed up your queries, or scale easily with your serverless or edge functions?  // Try Prisma Accelerate: https://pris.ly/cli/accelerate-init  generator client {  provider = "prisma-client-js"  }  datasource db {  provider = "mysql"  url = env("DATABASE\_URL")  }    model Student{  id Int @id @default(autoincrement())  name String?  age Int  height Decimal @db.Decimal(5,2)  gender String?  clsId Int @default(0)  isDeleted Int @default(0)  }  model Class{  id Int @id @default(autoincrement())  name String?  } |

## 注意在schema.prisma使用枚举很烦，也没有bit类型，decimal需要写成Decimal @db.Decimal(5,2)

# 7.生成prisma客户端，使用命令yarn dlx prisma migrate dev --name init，执行成功后，会在prisma文件夹里面生成一个migrations文件夹里面有一个随机数字文件夹，文件夹里面有一个migrations.sql文件

|  |
| --- |
|  |

# 8.这里有一个小问题，他没有生成prisma客户端，我们先执行yarn dlx prisma generate使用，然后使用yarn add @prisma/client安装一下，完成后如图

|  |
| --- |
|  |

# 9.然后我们来查看一下数据库，发现生成了新街口studb，他有2个表格

|  |
| --- |
|  |

# 10.然后，我们在项目根目录新建一个index.js,插入3条数据，注意由于我们的id字段是自增长的，我们不需要给他赋值。

|  |
| --- |
| let {PrismaClient} = require('@prisma/client')  let client  = new PrismaClient()  // let newStudent = client.student.create({  //     data:{  //         name : 'xiaoming',  //         age : 14,  //         height: 172.30,  //         gender: 'Male',  //         clsId: 0,  //         isDeleted: 0  //   },  // });  // newStudent.then((data)=>console.log(data))  // let newStudent2 = client.student.create({  //     data:{  //         name : 'xiaohong',  //         age : 15,  //         height: 166.30,  //         gender: 'Female',  //         clsId: 1,  //         isDeleted: 0  //   },  // });  // newStudent2.then((data)=>console.log(data))  let newStudent2 = client.student.create({      data:{          name : 'xiaowang',          age : 15,          height: 169.00,          gender: 'Male',          clsId: 1,          isDeleted: 0    },  });  newStudent2.then((data)=>console.log(data)) |

## 注意：使用prisma客户端的create方法时，id是从1开始，如果你给id赋值为0，就会报错！

# 11.我们到数据库里面查看一下，发现数据进来了

|  |
| --- |
|  |

# 13.查询演练，在项目根目录下面新建一个querystu.js文件，添加查询所有学生的代码，

|  |
| --- |
|  |

## 效果：

|  |
| --- |
| PS F:\Projects\_f\nodejs-koudinlang\2-node-express\express+mysql\mysql\_prisma> node .\querystu.js  [  {  id: 1,  name: 'xiaoming',  age: 14,  height: 172.3,  gender: 'Male',  clsId: 0,  isDeleted: 0  },  {  id: 3,  name: 'xiaohong',  age: 15,  height: 166.3,  gender: 'Female',  clsId: 0,  isDeleted: 0  },  {  id: 4,  name: 'xiaowang',  age: 15,  height: 169,  gender: 'Male',  clsId: 1,  isDeleted: 0  }  ] |

# 13.然后我们添加代码根据id来查询一个学生

|  |
| --- |
|  |

## 效果：

|  |
| --- |
|  |

# 14.一次性创建多条记录，使用createMany方法

|  |
| --- |
|  |

## 效果：

|  |
| --- |
|  |
|  |

# 15.update方法更新数据，不知道where查找条件为什么必要要有id，没有id报错

|  |
| --- |
| let {PrismaClient} = require('@prisma/client')  let client = new PrismaClient()  let res = client.student.update({      where:{id:8,name:'Tracy',},      data:{age:14,},  })  res.then(data=>{      console.log(data);  }) |

# 16.删除数据，创建一个delstu.js