

数据库课设任务

服务器配置

- 服务器需在腾讯服务器安全站上开放对应端口，Oracle默认为1521端口，远程协作api端口为：121.5.175.203:8080端口。
- Oracle安装；
- Oracle数据配置：在下载解压包，解压以后的文件夹中的network->admin文件夹下修改listener文件

```
SID_LIST_LISTENER =
(
  (SID_LIST =
    (SID_DESC =
      (SID_NAME = CLRExtProc)
      (ORACLE_HOME = C:\Database)
      (PROGRAM = extproc)
      (ENVS = "EXTPROC_DLLS=ONLY:C:\Database\bin\oraclr19.dll")
    )
    (SID_DESC =
      (GLOBAL_DBNAME = orcl)//默认名字，运行setUp时的
      (ORACLE_HOME = C:\Database)//文件地址
      (SID_NAME = orcl)//默认名字
    )
  )
)

LISTENER =
(DESCRIPTION_LIST =
  (DESCRIPTION =
    (ADDRESS = (PROTOCOL = TCP)(HOST = 172_17_0_14)(PORT = 1521))
    (ADDRESS = (PROTOCOL = IPC)(KEY = EXTPROC1521))
  )
  (DESCRIPTION =
    (ADDRESS = (PROTOCOL = TCP)(HOST = 服务器地址)(PORT = 1521))
    (ADDRESS = (PROTOCOL = IPC)(KEY = EXTPROC1521))//修改服务器地址
  )
)
```

至此可以用sql developer来进行链接并访问数据库。（已经在项目中完成）

- 在服务器配置C#环境，下载安装.NET CORE3.1 runtime Hosting Bundle即可
<https://dotnet.microsoft.com/download/dotnet-core/3.1>

Supported versions

Version	Status	Latest release	Latest release date	End of support
.NET 5.0	RC	5.0.0-rc.2	2020-10-13	
.NET Core 3.1 (recommended)	LTS	3.1.9	2020-10-13	2022-12-03
.NET Core 2.1	LTS	2.1.23	2020-10-13	2021-08-21

Release information	Build apps - SDK	Run apps - Runtime												
v3.1.9 Release notes Released 2020-10-13	<p>ⓘ This release contains multiple SDKs. If you're using Visual Studio, look for the SDK that supports the version you're using. If you're not using Visual Studio, install the first SDK listed.</p> <p>SDK 3.1.403</p> <p>Visual Studio support Visual Studio 2019 (v16.7) Visual Studio 2019 for Mac (v8.8)</p> <p>Included in Visual Studio 16.7.6</p> <p>Included runtimes .NET Core Runtime 3.1.9 ASP.NET Core Runtime 3.1.9 .NET Core Desktop Runtime 3.1.9</p>	<p>ASP.NET Core Runtime 3.1.9</p> <p>The ASP.NET Core Runtime enables you to run existing web/server applications. On Windows, we recommended installing the Hosting Bundle, which includes the .NET Core Runtime and IIS support.</p> <p>IIS runtime support (ASP.NET Core Module v2) 13.1.20268.9</p> <table border="1"> <thead> <tr> <th>OS</th> <th>Installers</th> <th>Binaries</th> </tr> </thead> <tbody> <tr> <td>Linux</td> <td>Package manager instructions</td> <td>ARM32 ARM64 ARM64 Alpine x64 x64 Alpine</td> </tr> <tr> <td>macOS</td> <td></td> <td>x64</td> </tr> <tr> <td>Windows</td> <td>Hosting Bundle x64 x86</td> <td>ARM32 x64 x86</td> </tr> </tbody> </table>	OS	Installers	Binaries	Linux	Package manager instructions	ARM32 ARM64 ARM64 Alpine x64 x64 Alpine	macOS		x64	Windows	Hosting Bundle x64 x86	ARM32 x64 x86
OS	Installers	Binaries												
Linux	Package manager instructions	ARM32 ARM64 ARM64 Alpine x64 x64 Alpine												
macOS		x64												
Windows	Hosting Bundle x64 x86	ARM32 x64 x86												

用CMD窗口dotnet + .dll文件或点击exe文件即可运行C#文件。

将任务部署到服务器上

- 在本地写好的文件中，先在本地测试好后，把StartU.cs加上为 .UseUrls("http://*:8080")来进行访问远程的8080端口。然后发布，生成publish文件后复制到服务器上即可。

后端项目

在VS中选择.NET CORE 3.1 WEB API后修改appsettings.json：

```
{
  "Logging": {
    "IncludeScopes": false,
    "Debug": {
      "LogLevel": {
        "Default": "Warning"
      }
    },
    "Console": {
      "LogLevel": {
        "Default": "Warning"
      }
    }
  },
  "ConnectionStrings": {
    "EmployeeConnection": "Data Source=(DESCRIPTION=(ADDRESS=(PROTOCOL=TCP) (HOST='121.5.175.203') (PORT=1521)) (CONNECT_DATA=(SERVICE_NAME=orcl)));Persist Security Info=True;User ID=c##test;Password=test;"
  }
}
```

```
}  
}
```

需要注意的就是Data Source部分写上自己对应的数据库地址和用户名账号密码。

然后创建一个仓库文件夹（Repositories）来进行操作，由IEmployeeRepository来管理接口，然后在EmployeeRepository来实现接口内容。

创建一个Controller来链接网络请求。

```
[Route( "api/User" )]  
[ApiController]  
[HttpGet]  
[HttpPost]  
[HttpPost( "xxx" )] //对应url/xxx  
[HttpGet( "{WantName}/{WantPwd}" )] //对应url/name/pwd详情见代码
```

API接口

Get方法

- <http://121.5.175.203:8080/api/User>

查询所有User数据

返回值：所有用户数据

- <http://121.5.175.203:8080/api/User/{name}/{pwd}>

访问对应name和pwd的人，把{name}和{pwd}换成对应的即可，

返回值：

没有则返回Not Found404

存在则返回数据

POST方法

- <http://121.5.175.203:8080/api/User>

作用：查询对应name和pwd的人

返回值：

没有则返回false

存在返回true

JSON请求为：

```
{  
  
  "name": "",  
  
  "pwd": ""  
  
}
```

- <http://121.5.175.203:8080/api/User/register>

作用：查询对应name和pwd的人

返回值：

没有则返回false

存在返回true

JSON请求为：

```
{  
  
  "name": "",  
  
  "pwd": ""  
  
}
```