

成都市主城区生鲜冷链服务系统 运行环境及部署说明

本系统运行环境及部署说明主要包括三部分：**数据入库、服务发布和项目启动**。本文档将对其每一步配置进行详尽描述，辅助系统顺利完成本地运行。**需要注意的是，由于代码中引用了较多的配置命名，配置项目时建议全部采取与本文档相同的命名，避免命名问题导致的项目运行失败。**

1 数据入库

本系统需要使用到的数据如图 1 即 Data 文件夹所示，包括：零售商店 shp 文件夹中的生鲜零售商店 POI 数据 Shapefile 文件 Store.shp、各商店生鲜简单数据 freshprice.csv 文件、各商店售卖种类数 number.json 文件、各商店生鲜统计数据 price.json 文件和成都市行政区划成都市.json 文件。






 零售商店shp	2023/1/2 19:31	文件夹	
 freshprice.csv	2022/12/21 21:49	Microsoft Excel ...	36 KB
 number.json	2022/12/30 14:38	JSON File	2 KB
 price.json	2023/1/1 19:57	JSON File	359 KB
 成都市.json	2022/12/17 19:18	JSON File	7 KB

图 1 系统使用数据

其中，需要将 Store.shp 和 freshprice.csv 导入 PostGIS 数据库中，在创建好数据库后（本系统数据库名为 **webgis**），shp 文件可以通过 PostGIS 自带工具 PostGIS Shapefile Import/Export Manager 进行直接导入，由于操作较为简单具体过程可见相关网络教程如 https://blog.csdn.net/qq_35732147/article/details/85228444。

值得一提的是 csv 文件导入可以直接在 pgAdmin 中操作，首先需要在 webgis 数据库架构中表右键新建 **freshprice** 表，然后在数据库查询工具输入如表 1 所示代码创建表字段名，之后右键 freshprice 表选择导入数据，在出现的图 2 导入 freshprice.csv 配置中选择相关配置（文件路径）。

同时**可能会出现二进制路径报错无法导入**，可以参考图 3 二进制路径配置方法配置好文件选项卡中配置中的二进制路径后重新导入。

表 1 查询工具创建表 SQL 语句

CREATE TABLE freshprice(storeid BIGINT, type TEXT, price FLOAT8);

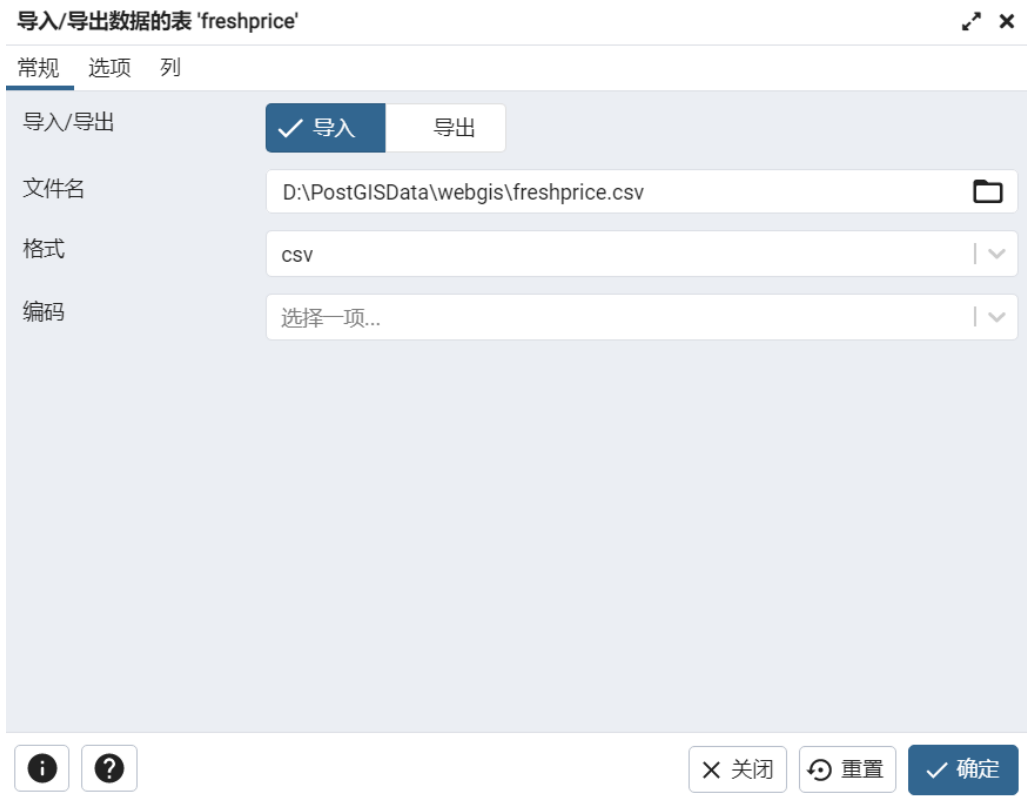


图 2 导入 freshprice.csv 配置

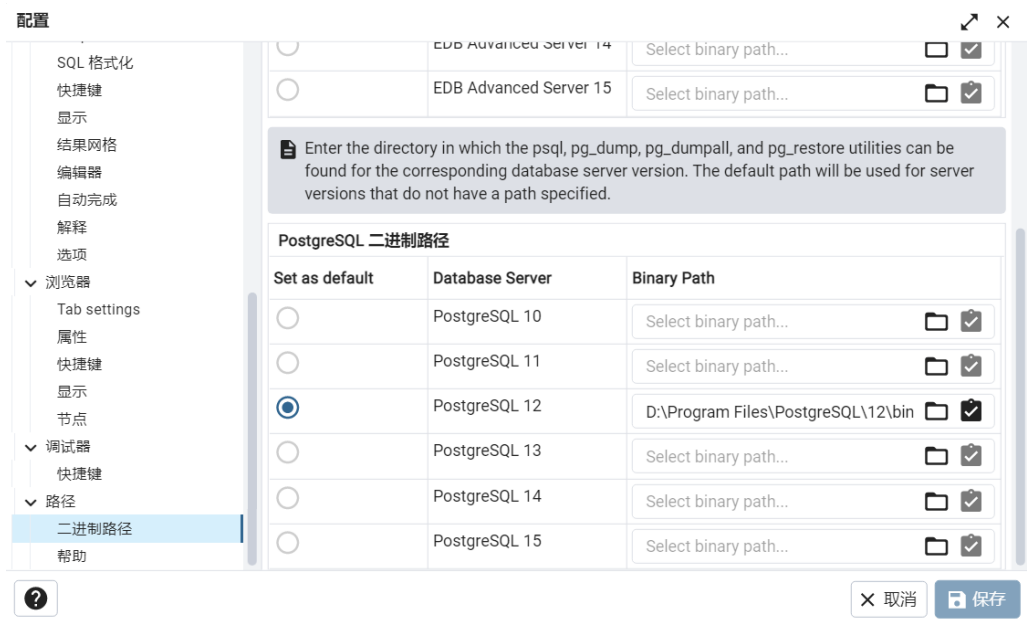


图 3 二进制路径配置

而 number.json、price.json 和成都市.json 在项目文件中已经放置好位置，可直接使用，无需进行下一步操作。

2 服务发布

服务发布主要变现在使用 GeoServer 发布 PostGIS 数据库以及配置好相应的 sqlview。

PostGIS 数据库发布体现在：首先，在左侧数据菜单依次创建好对应的工作空间（本系统命名为 **WebGIS**）、存储仓库（本系统命名为 **Data**）。在新建工作空间配置页面输入名称和 url，在新建存储仓库配置页面选择刚才新建的工作空间，填写存储仓库名称及 PostGIS 数据库连接参数信息（数据库名称、账户、密码、接口等等），如图 4 所示。



图 4 发布 PostGIS 数据库

然后，在图层里添加新的资源选择创建好的工作空间:存储仓库选项（或创建存储仓库后点击保存并关闭按钮，GeoServer 将跳转至图层发布界面），页面中会列出存储仓库中所有能够进行发布的图层。

最后，与发布 Shapefile 数据一样点击发布按钮，进行图层发布。发布的时候需要保持坐标系信息与原始数据不变，计算图层的原始边界范围和经纬度范围，

如图 5 所示。点击保存并关闭按钮完成发布后，可以在图层预览页面中对刚发布的图层进行预览即发布成功。

坐标参考系统

本机SRS

EPSG:4326

EPSG:WGS 84...

定义SRS

EPSG:4326

查找...

EPSG:WGS 84...

SRS处理

强制声明



边框

Native Bounding Box

最小 X

最小 Y

最大 X

最大 Y

103.975494384761 30.5923728942871 104.155883789062 30.7401771545410

从数据中计算

Compute from SRS bounds

纬度/经度边框

最小 X

最小 Y

最大 X

最大 Y

103.975494384761 30.5923728942871 104.155883789062 30.7401771545410

Compute from native bounds

图 5 设置坐标系与边框

接下来进行 sqlview 视图的配置，以保证查询功能的成功实现。配置开始如图 6 所示，最终结果应当如图 7 所示。

新建图层

添加一个新图层

添加图层 WebGIS:Data

您可以通过手动配置属性的名称和类型创建一个新的要素类型。 [创建新的要素类型...](#)

在数据库中，你也可以通过配置一个本地的SQL语句创建一个新的要素类型。 [配置新的SQL视图...](#)

Here is a list of resources contained in the store 'Data'. 点击你要配置的图层

<< < 1 > >> 从0 到 0的结果(共0项)

Search

发布的	图层名称	操作
✓	store	再次发布
	freshprice	发布

<< < 1 > >> 从0 到 0的结果(共0项)

图 6 SQL 视图配置选项

<input type="checkbox"/>	类型	标题	图层名称	存储仓库	启用?	Native SRS
<input type="checkbox"/>	点	store	WebGIS:store	Data	✓	EPSG:4326
<input type="checkbox"/>	面	store_dist	WebGIS:store_dist	Data	✓	EPSG:4326
<input type="checkbox"/>	面	store_fresh	WebGIS:store_fresh	Data	✓	EPSG:4326
<input type="checkbox"/>	面	store_name	WebGIS:store_name	Data	✓	EPSG:4326

图 7 最终图层

配置三个 SQL 视图的具体参数如下图表所示，三个 sqlview 命名为 **store_dist**、**store_fresh** 和 **store_name**。

命名视图名尽量保持与本系统一致。

①store_dist

表 2 store_dist SQL 语句

<pre>select store.id,store.latitude,store.longitude,store.name,store.adname, ST_Distance(ST_SetSRID(ST_MakePoint(%lng%,%lat%),4326)::geography, ST_SetSRID(ST_MakePoint(store.longitude,store.latitude),4326)::geography) as distance,store.geom from store where ST_Distance(ST_SetSRID(ST_MakePoint(%lng%,%lat%),4326)::geography, ST_SetSRID(ST_MakePoint(store.longitude,store.latitude),4326)::geography)<%dist%</pre>		
--	--	--

SQL视图参数

从SQL猜想的参数 添加新的参数 删除所选

<input type="checkbox"/>	名称	D默认值	验证的正则表达式
<input type="checkbox"/>	lng	104.09	
<input type="checkbox"/>	dist	3000	
<input type="checkbox"/>	lat	30.71	

☐ 特殊 SQL 字符溢出

属性

刷新 ☐ Guess geometry type and srid

名称	类型	SRID	标识符
id	Double		<input type="checkbox"/>
latitude	BigDecimal		<input type="checkbox"/>
longitude	BigDecimal		<input type="checkbox"/>
name	String		<input type="checkbox"/>
adname	String		<input type="checkbox"/>
distance	Double		<input type="checkbox"/>
geom	Geometry	-1	<input type="checkbox"/>

图 8 store_dist 具体配置

②store_fresh

表 3 store_fresh SQL 语句

<pre>select store.id,store.latitude,store.longitude , store.name,store.adname,</pre>		
--	--	--

ST_Distance(
ST_SetSRID(ST_MakePoint(%lng%,%lat%),4326)::geography,
ST_SetSRID(ST_MakePoint(store.longitude,store.latitude),4326)::geography
) as distance,store.geom, fresh.type,fresh.price
from store,fresh
where store.id=fresh.storeid
and fresh.type like '%%word%%' and fresh.price>=%min%
and fresh.price<=%max%
order by(fresh.price)

SQL视图参数

从SQL猜想的参数 添加新的参数 删除所选

<input type="checkbox"/>	名称	D默认值	验证的正则表达式
<input type="checkbox"/>	min	0	
<input type="checkbox"/>	lng	109	
<input type="checkbox"/>	max	400	
<input type="checkbox"/>	word		
<input type="checkbox"/>	lat	31	

☐ 特殊 SQL 字符溢出

属性

刷新 ☐ Guess geometry type and srid

名称	类型	SRID	标识符
id	Double		<input type="checkbox"/>
latitude	BigDecimal		<input type="checkbox"/>
longitude	BigDecimal		<input type="checkbox"/>
name	String		<input type="checkbox"/>
adname	String		<input type="checkbox"/>
distance	Double		<input type="checkbox"/>
geom	Geometry	-1	<input type="checkbox"/>
type	String		<input type="checkbox"/>
price	Double		<input type="checkbox"/>

图 9 store_fresh 具体配置

③store_name

表 4 store_name SQL 语句

select * from(
select store.id,store.latitude,store.longitude ,
store.name,store.adname,
ST_Distance(
ST_SetSRID(ST_MakePoint(%lng%,%lat%),4326)::geography,
ST_SetSRID(ST_MakePoint(store.longitude,store.latitude),4326)::geography

```

) as distance,store.geom
from store
) as tmp
where name like '%%word%%'
order by(distance)

```

从SQL猜想的参数 添加新的参数 删除所选

<input type="checkbox"/> 名称	D默认值	验证的正则表达式
<input type="checkbox"/> lng	104.19	
<input type="checkbox"/> word		
<input type="checkbox"/> lat	30.71	

☐ 特殊 SQL 字符溢出

属性
刷新 ☐ Guess geometry type and srid

名称	类型	SRID	标识符
id	Double		<input type="checkbox"/>
latitude	BigDecimal		<input type="checkbox"/>
longitude	BigDecimal		<input type="checkbox"/>
name	String		<input type="checkbox"/>
adname	String		<input type="checkbox"/>
distance	Double		<input type="checkbox"/>
geom	Geometry	-1	<input type="checkbox"/>

图 10 store_name 具体配置

正则表达式全部为空。

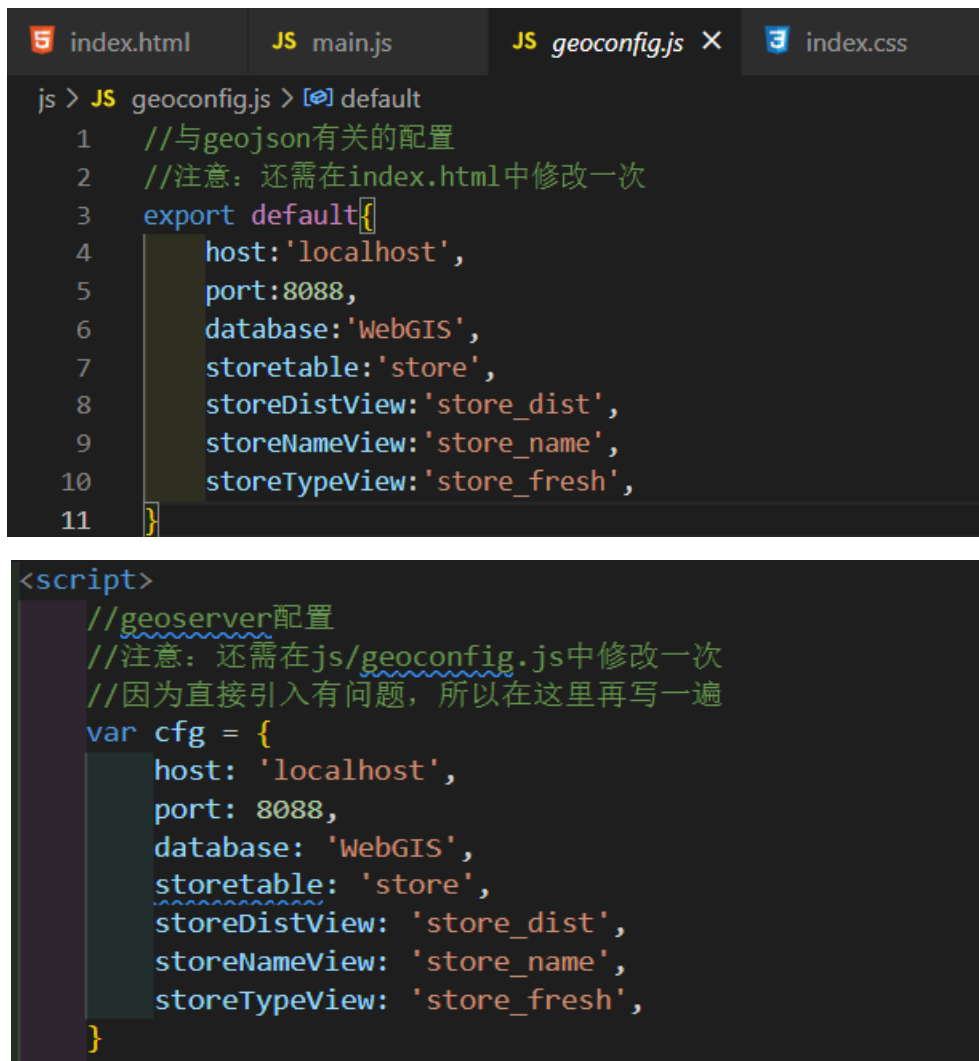
3 项目启动

首先需要对 GeoServer 配置文件进行修改，即如图 11 所示的 js 文件夹中的 **geoconfig.js** 文件修改以及 **index.html** 文件部分修改，将其中的参数修改为本机 GeoServer 参数。

然后需要电脑拥有 **Node.js 环境** 和 **npm 环境**，系统才能本地部署顺利运行。下载链接 <http://nodejs.cn/download/>，安装配置教程 <https://www.runoob.com/nodejs/nodejs-install-setup.html>。

在系统文件夹运行 node 终端，输入 **npm install** 安装项目依赖，输入 **npm start** 运行本地服务系统，如图 12 所示。因为本截图是在已经搭建出来的项目情况下输入显示的，注意 **install** 显示应当与本截图不同。

打开提示的本地端口网站即可发现系统成功运行，且在上述配置成功的前提下功能完善健全。




The image shows two code editors. The top editor, 'geoconfig.js', contains a JavaScript object 'default' with the following properties: host: 'localhost', port: 8088, database: 'WebGIS', storetable: 'store', storeDistView: 'store_dist', storeNameView: 'store_name', and storeTypeView: 'store_fresh'. The bottom editor, 'index.html', shows a script tag with a comment indicating the configuration is for Geoserver and that it needs to be modified in 'js/geoconfig.js'. It then defines a 'cfg' object with the same properties as the one in 'geoconfig.js'.

```
js > JS geoconfig.js > default
1 //与geojson有关的配置
2 //注意: 还需在index.html中修改一次
3 export default{
4   host: 'localhost',
5   port: 8088,
6   database: 'WebGIS',
7   storetable: 'store',
8   storeDistView: 'store_dist',
9   storeNameView: 'store_name',
10  storeTypeView: 'store_fresh',
11 }

<script>
//geoserver配置
//注意: 还需在js/geoconfig.js中修改一次
//因为直接引入有问题, 所以在这里再写一遍
var cfg = {
  host: 'localhost',
  port: 8088,
  database: 'WebGIS',
  storetable: 'store',
  storeDistView: 'store_dist',
  storeNameView: 'store_name',
  storeTypeView: 'store_fresh',
}
```

图 11 代码中 Geoserver 配置



The image shows a terminal window with the following commands and output: 'npm install' is run, followed by 'npm start'. The output shows the project name 'webgisproject@1.0.0' and the command 'parcel index.html'. The server is running at 'http://localhost:1234' and was built in 2.42s.

```
PS D:\学习\HZT\大三上\网络GIS课程设计\WebGISProject> npm install
up to date in 2s
PS D:\学习\HZT\大三上\网络GIS课程设计\WebGISProject> npm start

> webgisproject@1.0.0 start
> parcel index.html

Server running at http://localhost:1234
√ Built in 2.42s.
```

图 12 Nodejs 本地部署