Project(Grading scale)

选需求分实现

需求分不是最终得分

根据需求分确定得分上限比例

	60%	70%	80%	90%	100%
2-3人组	20分	22分	24分	28分	32分
4人组	28分	30分	32分	36分	40分

Project(Requirements)

Total scores: 54

ID	Score	ID	Score
1 线路基本信息	1	11 统计站点数量	4(2+2)
2 线路站点信息	2	12 统计路线类型	2
3 站点停靠线路	2	13 查询重复站点	2
4 起止沿线站点	3(2+1)	14 查询线路换乘	4(2/3/4)
5 最短路径	6(2/4/6)	15 统计站台连接	2
6 直达路线判断	2	16 统计路线站点	2
7 线路班次信息	2	17 统计运行时间	2
8 站点某时线路	2	18 计算重复系数	4
9 站点某时某线	2	19 线路创建	4(2/4)
10 统计停靠路线	2	20 线路删除更新	4(2+2)

Project(Report)

项目报告在考试周前3周提交
项目展示为考试周前2周，共2周时间展示，顺序随机，也可以主动申请
项目报告内容

- 数据库设计
- 需求选择与实现
  - 写清楚选做的需求编号和总共完成的需求分，其中11和20分为A/B需求。
  - 写清楚需求的实现方法，可以以Cypher语句或代码的形式。

项目展示

- 每组10分钟
- 项目演示5分钟：不需要讲需求具体实现
- 项目现场测试5分钟：助教使用新的测试用例进行测试

其他说明

- 前端界面美观程度不作为主要的评分标准，不需要部署，平台不限。

Project(Test Cases)

1.查询某条线路的基本信息。 (1分)

说明：数据来源于lines.json。
查询参数距离：30(路)
查询结果举例：{

```
"route": "燎原-北路湾公交站",
"onewayTime": "约49分",
"directional": true,
"kilometer": "12.0",
"name": "30",
"runtime": "6:30-22:30",
"interval": 8,
"type": "干线”
```

}

Project(Test Cases)

2.查询某条线路方向的全部站点信息。 (2分)

说明：站点为链状结构，需要有顺序性。
查询参数举例：2路上行
查询结果举例： [{"id":"7542","name":"兴义镇(始发站)","english":"XingYiZhen"},
{"id":"7527","name":"永盛(始发站)","english":"YongSheng"},
…
{"id":"14495","name":"火车西站公交站(终点站)","english":"Huo Che Zhan"}]
全程共28个站点。

Project(Test Cases)

3.查询锦城广场站停靠的所有线路。 (同名站点按ID分组，2分)

说明：同名站点按照ID分组查询。
查询参数举例：锦城广场
查询结果举例： {
"58290":["K5路上行"],
"64355":["N26路下行","G33路上行","17路下行"],
"58289":["K5路下行"],
"64356":["N26路上行","G33路下行","17路上行"]
}
注意：课堂上讲解了在插入数据时预先设立一个字段存储停靠线路的方法，请不要采用该方法，而是在查询时计算。

Project(Test Cases)

4.查询某条线路从某站到某站，线路的运行方向、沿路站点和运行时长。(3分)

说明：运行时长由班次数据计算得出，3个需求每个1分。
查询参数举例： 10路从大悦城到小吃街(3个参数)
查询结果举例： {"name":"10路上行",
"runtime":13,
"alongStation":[{"id":"62753","name":"大悦城","english":"DaYueCheng"},
{"id":"62765","name":"肖家沟","english":"XiaoJiaGou"},
{"id":"62737","name":"华通商场","english":"HuaTang SC"},
{"id":"62729","name":"东林南路","english":"DongLinNanLu"},
{"id":"6354","name":"东林小区(始发站)","english":"DongLinXiaoQu"},
{"id":"6363","name":"东林路","english":"DongLin Road"},
{"id":"6377","name":"小吃街","english":"Snack Street"}]}

Project(Test Cases)

5.查询某两个站台之间的最短路径。 (2-6分)

说明：项目演示时提供的测试用例会比较干净，不用过多考虑细节。
使用ID进行最短路径查询得2分，使用名字进行最短路径查询得4分，
如果有更多细节考虑(最少换乘、最短运行时间)，在报告中体现，得6分。
查询参数举例：

- 基于ID查询：从ID=16115到ID=14768
- 基于名字查询：从红瓦寺到动物园(因为重名站台的原因，结果可能有多条，选最短的路径输出)

查询结果举例：[{"id":"16115","name":"红瓦寺","english":"Hongwasi"},
{"id":"59548","name":"天九街","english":"TianJiuJie"},
{"id":"5181","name":"万安路东","english":"Wan An Lu E"},
{"id":"5197","name":"万安路","english":"Wan An Lu"},
{"id":"5168","name":"万安路西","english":"Wan An Lu W"},
{“id”:“14768”,“name”:“动物园”,“english”:“Zoo”}] (基于ID和基于名字的查询应该是一致的)

Project(Test Cases)

6.查询某两个站台间是否存在直达线路。 (2分)

说明：存在返回线路方向，不存在给出提示。因为设计的原因，考虑方向和不考虑方向的难度会有差异，请在报告中给出你的思考。
查询参数举例：环球中心(始发站)、荷花池
查询结果举例： (两种答案均可)

- 考虑方向：N12路下行
- 不考虑方向(双向)： N12路下行、N12路上行

Project(Test Cases)

7.查询某条线路某个方向的全部班次信息。(2分)

说明：

- 班次信息展示：二维表格(每一列是沿线站点，每一行是某个班次)
- 班次信息的特点：
  - 每一个班次的间隔时间是固定的(下例为6分钟一个班次)，
  - 但两个站台间用时不是固定的(不是固定的2分钟一个站) 。
- 班次间隔时间的信息在lines.json文件里。 (interval字段是班次间隔)

查询参数举例：239路上行

查询结果举例：

	曾家坡	高脚碾	团结桥	羊马	羊安路东	太阳湾南	太阳湾北	文家场
班次1	07:00	07:02	07:04	07:06	07:08	07:10	07:12	07:14
班次2	07:06	07:08	07:10	07:12	07:14	07:16	07:18	07:20

示例只给出两个，
但全部班次都需要显示。
数据可能很多，
前端页面注意美观。

Project(Test Cases)

8.查询某个时刻某个站台某个时段内即将停靠的线路。 (2分)

说明：每条线路只显示最近的一个班次。站台由ID唯一确定。
若刚好在该时刻，显示即将到站。
查询参数举例：08:37分、ID=16147 (新华书店)、5分钟内即将停靠的线路。
查询结果举例：

```
{
        "53路下行":"3分钟后到站",
        "70路上行":"1分钟后到站",
        "101路":"即将到站”
}
```

Project(Test Cases)

9*.查询某个时刻某个站台线路最近的3趟班次信息。(2分)
*表示有改动
说明：可能当前时刻往后的班次信息不足3趟。
查询参数举例： 10:32分、ID=59760(地铁万盛)
查询结果举例：
{"106路上行班次1":"7分钟后到站",
"106路上行班次2":"15分钟后到站",
"106路上行班次3":"23分钟后到站",
"82路上行班次1":"1分钟后到站",
"82路上行班次2":"7分钟后到站",
"82路上行班次3":"13分钟后到站",
"99路上行班次1":"4分钟后到站",
"99路上行班次2":"10分钟后到站",
"99路上行班次3":"16分钟后到站"}

Project(Test Cases)

10.统计停靠路线最多的站点并排序 。(2分)

说明：按照ID统计，并根据数量降序排序，显示前15个。
查询结果举例：
四列，分别为ID、站台名、线路条数、停靠线路，以下为前3个

1	"818"	"金河客运站(下客点)"	13	["N12路下行", "N11路下行", "G38路上行", "G37路下行", "G28路下行", "G22路下行", "759路上行
2	"24645"	"凤溪大道和桥"	11	["N31路上行", "G90路上行", "G41路", "736路下行", "735路下行", "727A路下行", "727路下行", "3
3	"24646"	"凤溪大道和桥"	11	["N31路下行", "G90路下行", "G41路", "736路上行", "735路上行", "727A路上行", "727路上行", "3

Project(Test Cases)

11A.统计特殊站台。(2分)

说明：统计地铁站、起点站、终点站数量，并返回站点名。
地铁站：开头含地铁两字
起点站：末尾标识(始发站)/没有入射边的站
终点站：末尾标识(终点站)/没有出射边的站

查询结果举例：NA

Project(Test Cases)

11B*.统计某条线路单行站。(2分)

说明：
- 单行站是指路线的上下行名字不相同的站。
- 不考虑环线的单行站，环线的lines.json文件中directional字段为false，

最终的测试用例中不会包含环线。
- 可以根据实际情况进行合理的优化，例如去掉(始发站)和(终点站)标识，

将其认为是同一个站。
查询参数举例：208(路)
查询结果举例：
["金河旅游区(终点站)","金河旅游区","金河旅游区(始发站)",
"东源路(终点站)","莲花一区","东源路(始发站)"]

Project(Test Cases)

12.分组统计常规公交(包括干线、支线、城乡线、驳接线、社区线)、
快速公交(K字开头)、高峰公交(G字开头)、夜班公交(N字开头)的数量。(2分)

查询结果举例：NA

Project(Test Cases)

13.查询两条线路重复的站点名 。(2分)

说明：重复站点名由ID区分。
查询参数举例： 15路上行、30路下行
查询结果举例：
[{"id":"17848","name":"金河市政府(始发站)","english":"Government"},
{"id":"17824","name":"孵化园","english":"FuHuaYuan"},
{"id":"17809","name":"北门立交东","english":"BeiMenLiJiao E"}]

Project(Test Cases)

14.查询换乘线路。换乘线路数即线路停靠的所有站台停靠其他线路的数量的总和。(2/3/4分)
说明：

- 统计可以换乘的线路数量得2分。
- 统计可以换乘的线路名称得3分。
- 统计沿线每个站点可以换乘的线路得4分。
- 换乘线路注意去掉该线路本身。

查询参数举例：261路上行
查询结果举例：(2分：输出15条；3分：输出所有可换乘的线路名； 4分输出如下)

{"凤溪大道中":["70路下行","74路下行","322路上行","727路上行","735路上行","736路上行","G41路","N31路下行"],
"培风路北":["30路下行","208路上行"],
"双水村":["15路下行","16路下行","N20路上行"],
"医学院":["30路下行","208路上行"],
"花博路":["70路下行","74路下行","736路上行","G41路"],
"凤溪大道和桥":["70路下行","74路下行","322路上行","727路上行","727A路上行","735路上行","736路上行","G41路","G9
"星空大道武科路口":["15路下行","16路下行","G41路","N20路上行"],
"天府家园":["15路下行","16路下行","N20路上行"],
"花博路口":["70路下行","74路下行","736路上行","G41路"],
"星空大道南":["16路下行","30路下行","735路上行","N20路上行"]}

Project(Test Cases)

15.根据连接两个相邻站台之间线路数量排序两个相邻站台。(2分)

说明：降序排列，显示前15个，按照名字进行查询并去重，考虑方向性。
提示：match (a:Station)-->(b:Station) return distinct a.name, b.name order by ???
查询结果举例：部分如下

```
[
        "凤溪大道中-->凤溪大道和桥:9条线路",
        "凤溪大道和桥-->凤溪大道中:9条线路",
        "科北路口-->北门立交南:8条线路",
        "凤溪大道和桥-->河野:8条线路”
…
]
```

Project(Test Cases)

16.根据站点数量对线路进行排序。(2分)

说明：降序排列，显示前15条，线路含方向。
查询结果举例：

```
[
        "736路上行共46个站点",
        "736路下行共45个站点",
        "735路下行共43个站点",
        "735路上行共43个站点",
…
]
```

Project(Test Cases)

17.根据运行时间对线路进行排序。(2分)

说明：运行时间由班次数据计算而得，不要使用lines.json里的数据。
降序排列，显示前15条。
查询结果举例：

```
[
        "736路上行单程运行95分钟",
        "G90路下行单程运行94分钟",
        "736路下行单程运行94分钟",
        "G90路上行单程运行92分钟",
…
]
```

Project(Test Cases)

18.计算某条线路之间的非重复系数。 (4分)

说明：对于一条线路，有若干个站台A、B、C...。
假如A与B之间有4条线路，则站台A与B 的非重复系数为 $\frac{1}{4}$。
一条线路的非重复系数则是线路沿线所有站台间的非重复系数的平均。
提示：使用Reduce函数(自学)
查询参数与结果举例：(保留两位小数)
N12路上行：0.47
G95路上行：1.0 (G95路全程两个站，且只有一条线路直接相连，所以非重复系数为1)
208路上行：0.88
30路上行：0.63

Project(Test Cases)

19.添加一条站点数不少于10的线路。 (2-4分)

说明：
添加线路，需要设置站点和路线信息，得2分。
添加线路，设置站点和路线信息，并且加入班次信息，得4分。
该部分不进行测试，在报告中写清楚实现过程即可。

Project(Test Cases)

20A.线路的删除 (4分)

说明：
删除某条线路，若沿途站点为该线路独有，也需要删除该站点，得2分。
该部分不进行测试，在报告中写清楚实现过程即可。

Project(Test Cases)

20B.线路的更新(2分)

说明：
将某条线路沿线的某个站点替换为另一个站点，不需要修改班次信息。
返回新的路线沿途站点，得2分。
该部分不进行测试，在报告中写清楚实现过程即可。