Tutorials

Neo4J and Cypher

This tutorial was prepared by Yang Wu

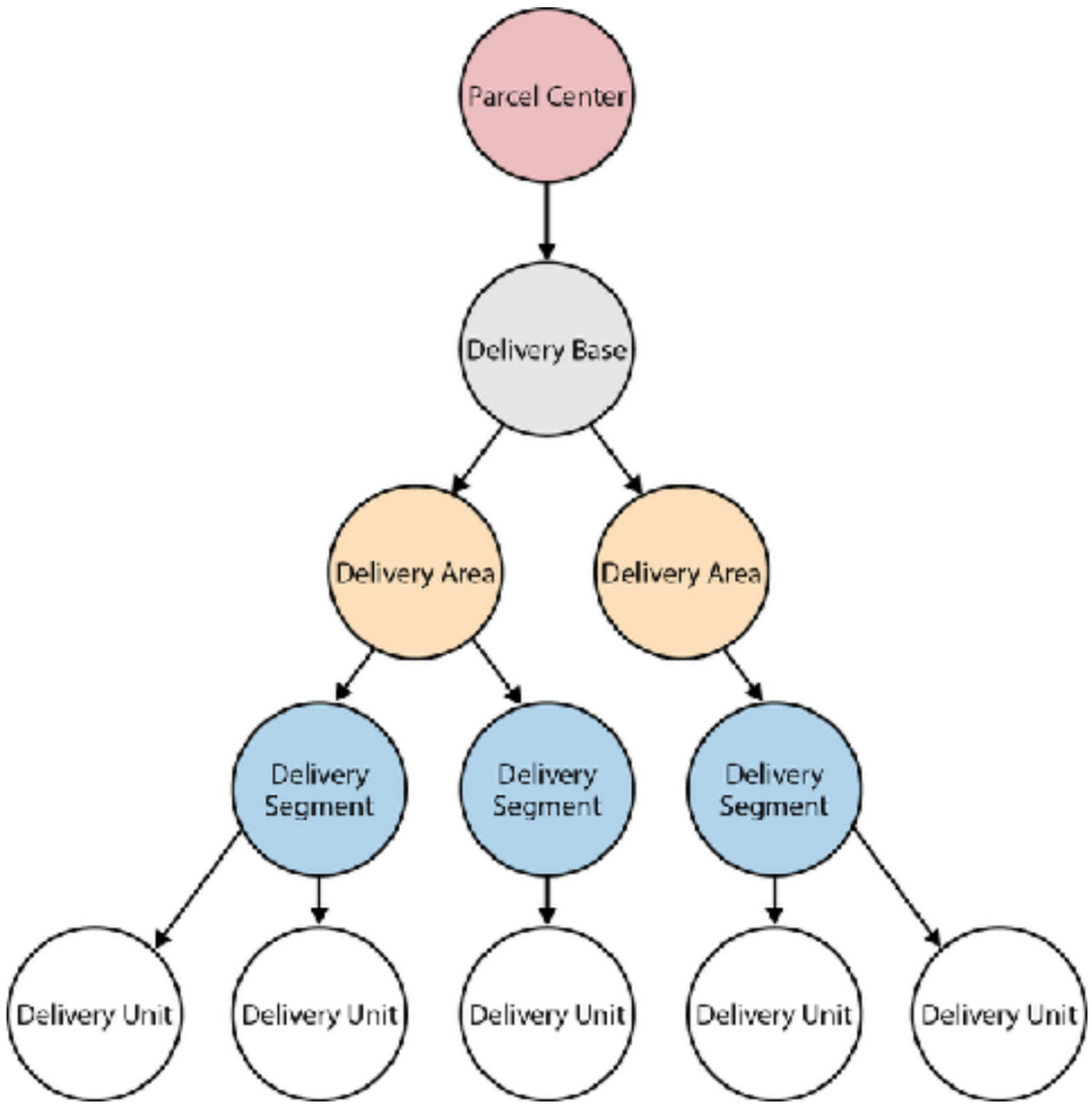
学习目标

- 了解原生图数据库的使用方法
- 掌握属性图的具体实现方法及特点
- 学习使用Cypher编写图查询语句

- 参考书籍：《图数据库》

示例： Geospatial and Logistics———Global Post

- 电商网购触发大量的物流需求，物流路径的计算要求在秒级别完成。
- 快递站结构日益复杂:
  - 快递站的层次结构：从上到下依次是集散中心(ParcelCenter)，分发基地(DeliveryBase)，配送区(DeliveryArea)以及快递点(Segment).
- 更为复杂物流计算还需要考虑运输手段、节假日限制、货物属性、交通状态等，完整的物流建模是一个复杂的动态知识图谱。



物流数据模型：图的具体结构举例

图的结构为:

a. 从分发基地到快递点之间每个下层结点至多只有一个父结点

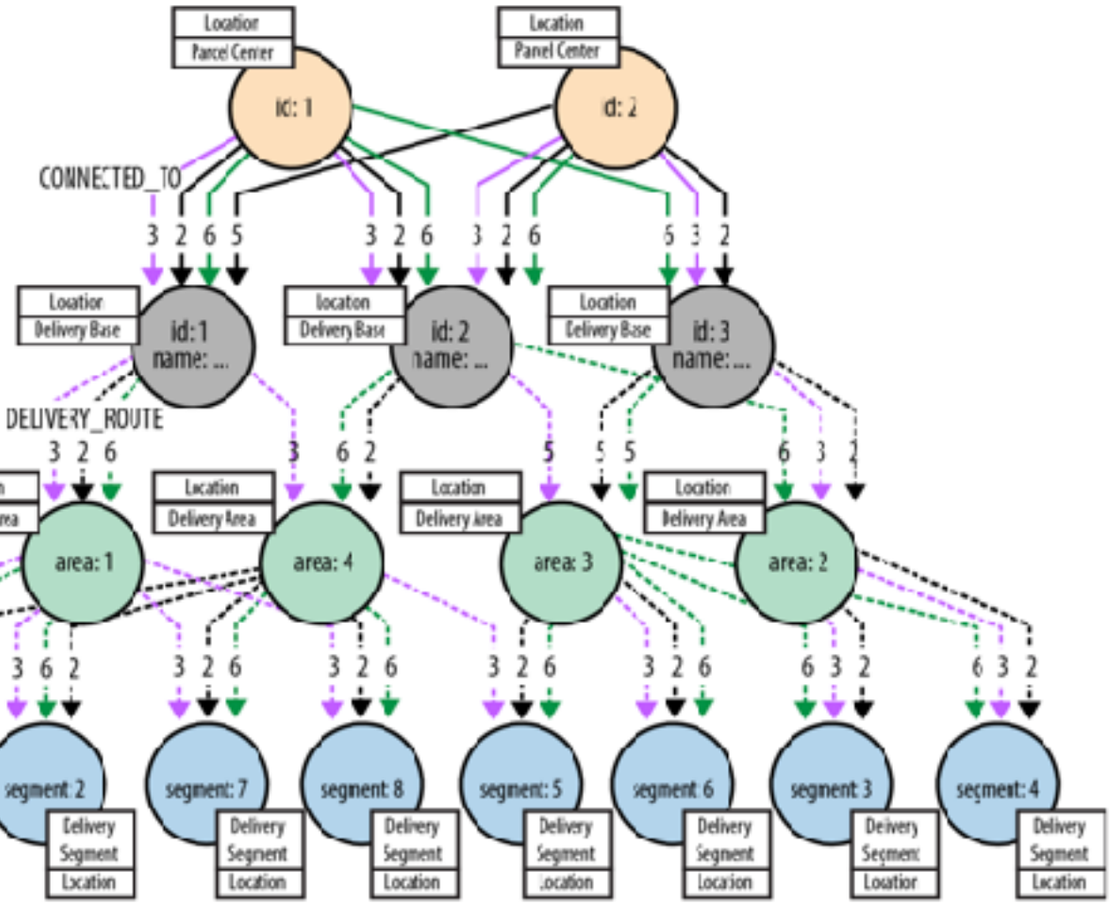
(用*DELIVERY_ROUTE*关系标识)

b. 分发基地(*DeliveryBase*)可以存在到多个集散中心的连接

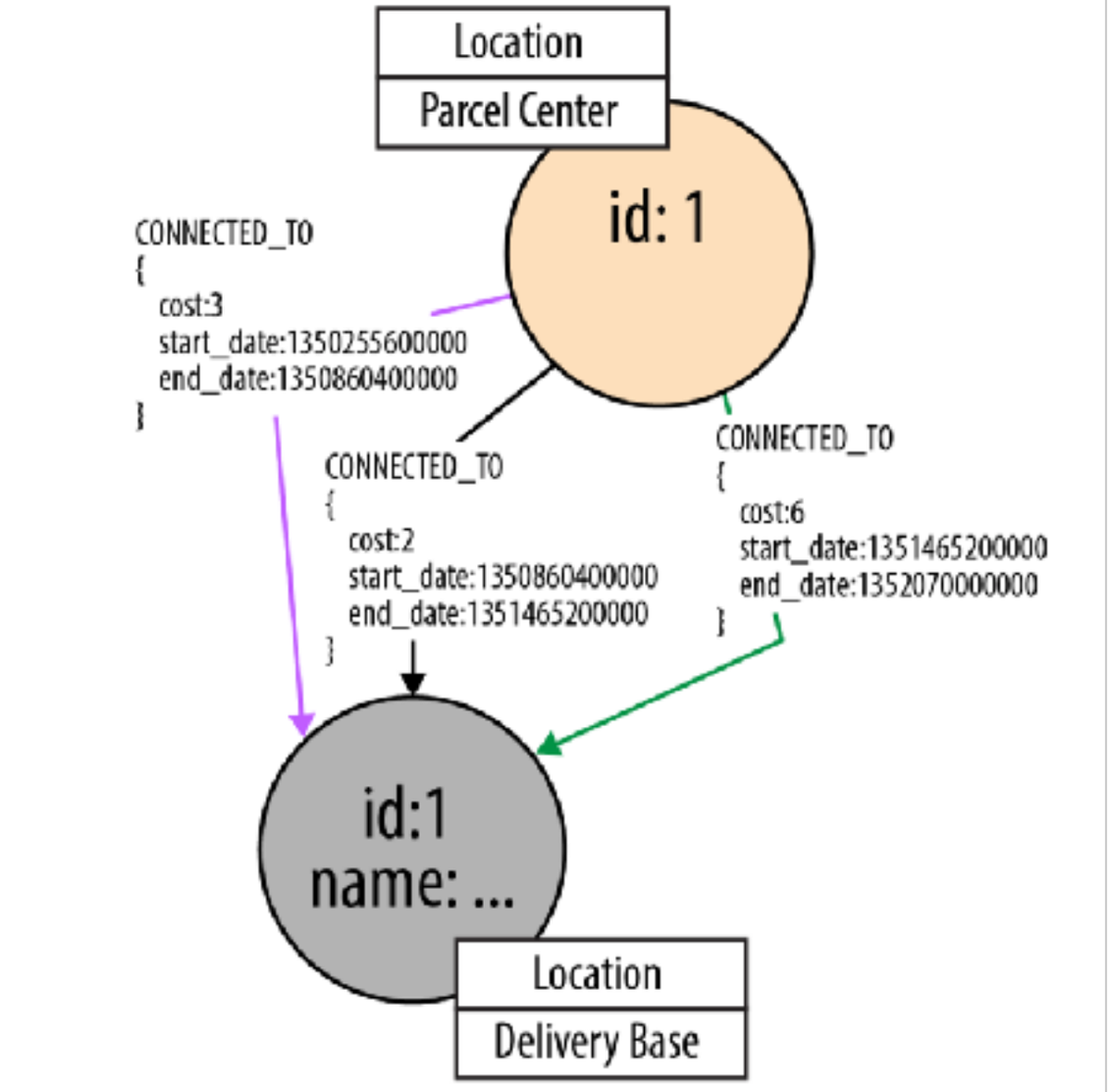
(用*CONNECTED_TO*关系标识)

c. 在满足*a,b*的条件下，整个快递运送体系有可能随时间发生改变

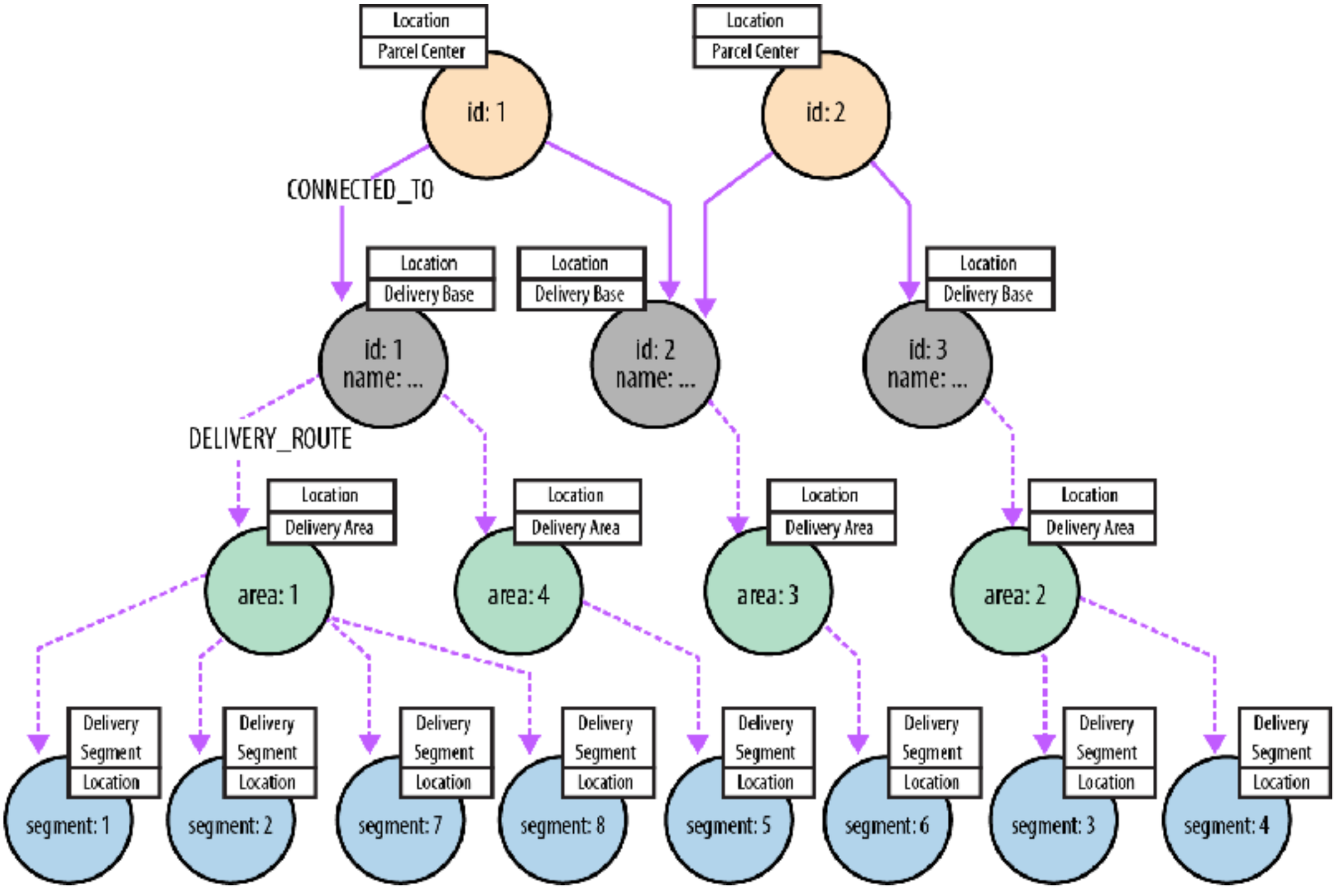
(比如*A1*快递点星期六星期天不上班)



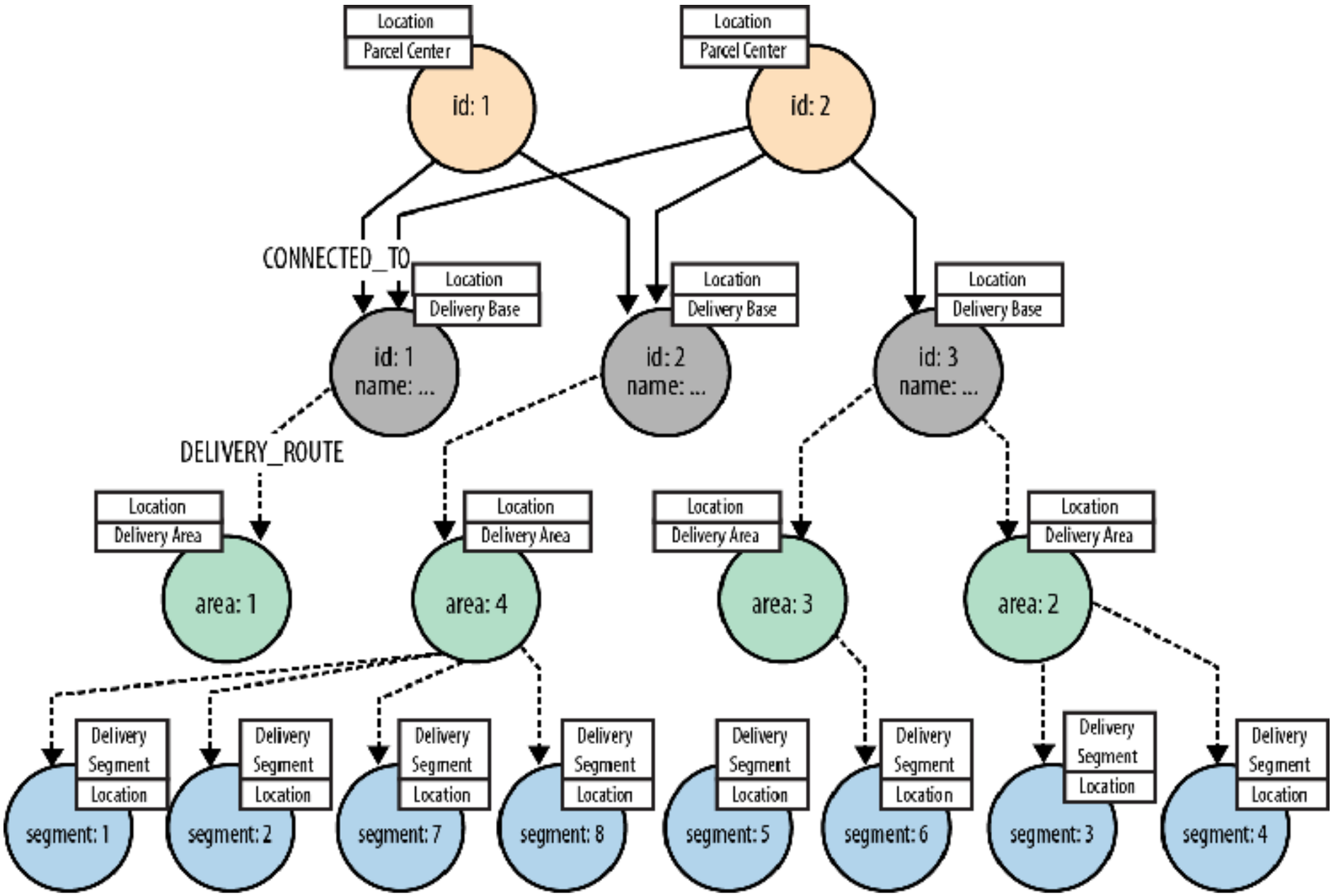
物流数据模型：图的具体结构举例



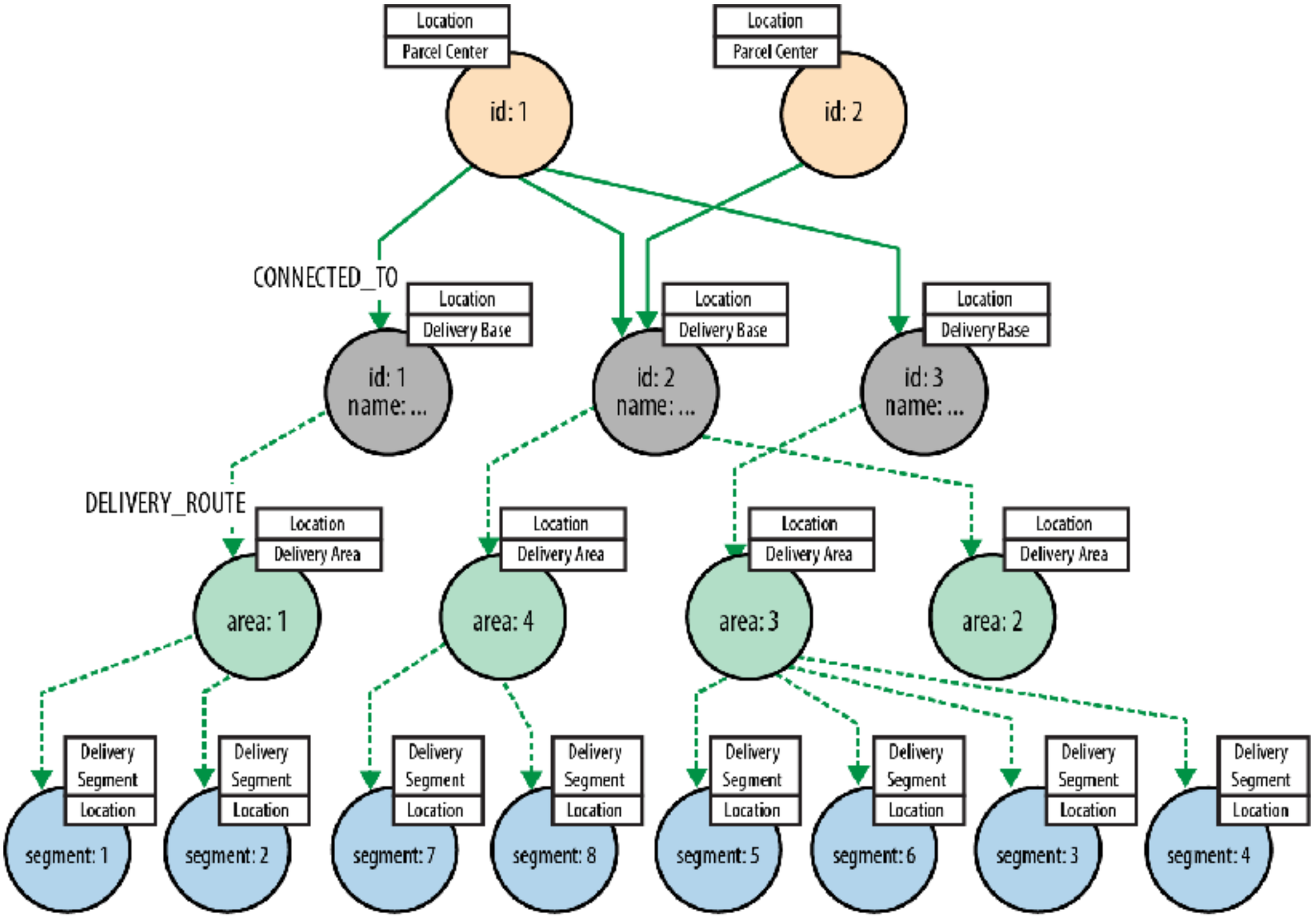
物流数据模型：动态图谱



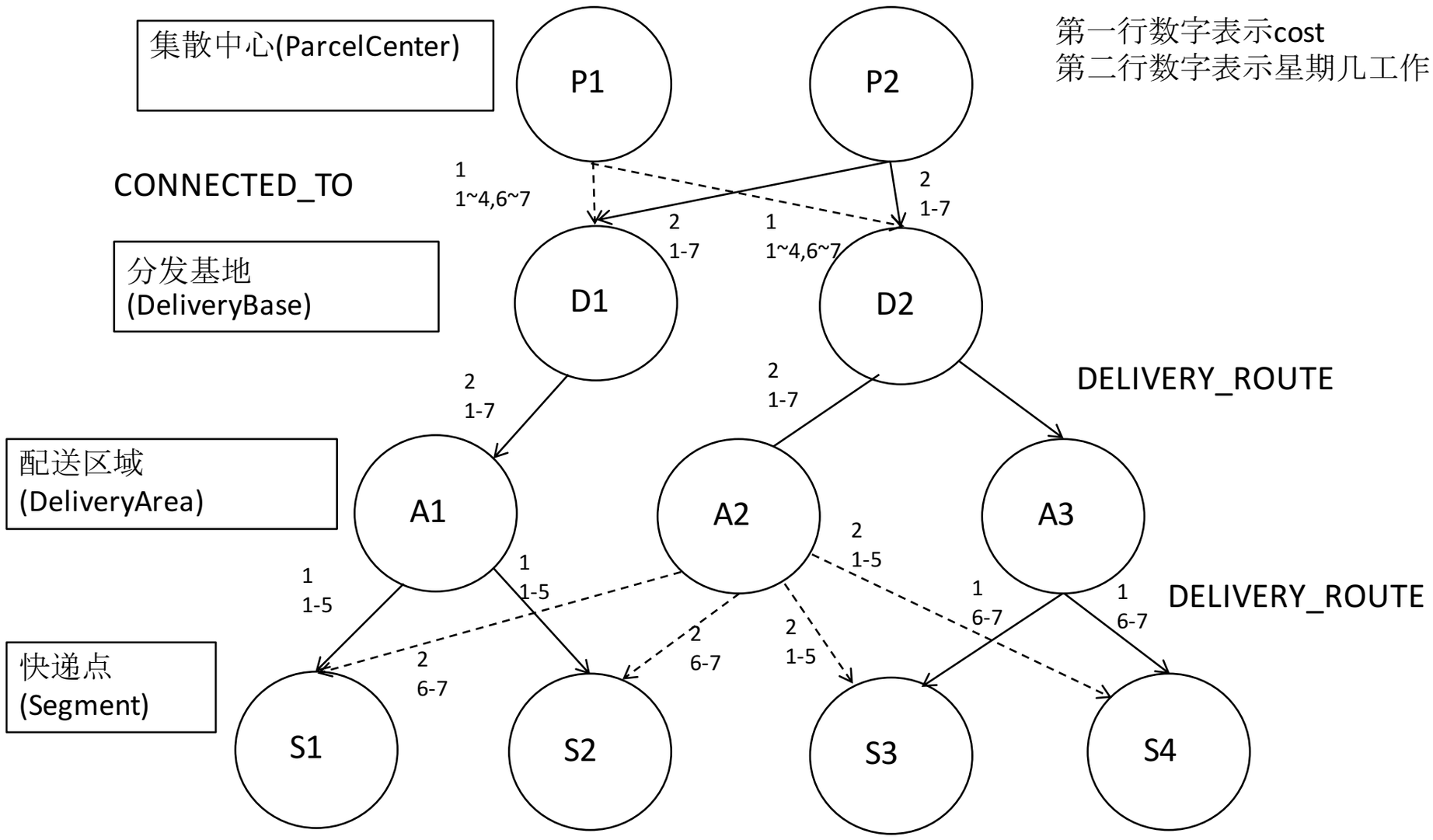
物流数据模型：动态图谱



物流数据模型：动态图谱



物流数据模型：路径计算



第一行数字表示cost
第二行数字表示星期几工作

问题 a. 给定某快递发货地点和收货地点(都是Segment) 和在哪个时间段内流转(比如星期4到星期5之间)要求必须经过至少一个集散中心(ParcelCenter)，求该快递的最短cost.

物流数据模型：路径计算——测试数据生成代码

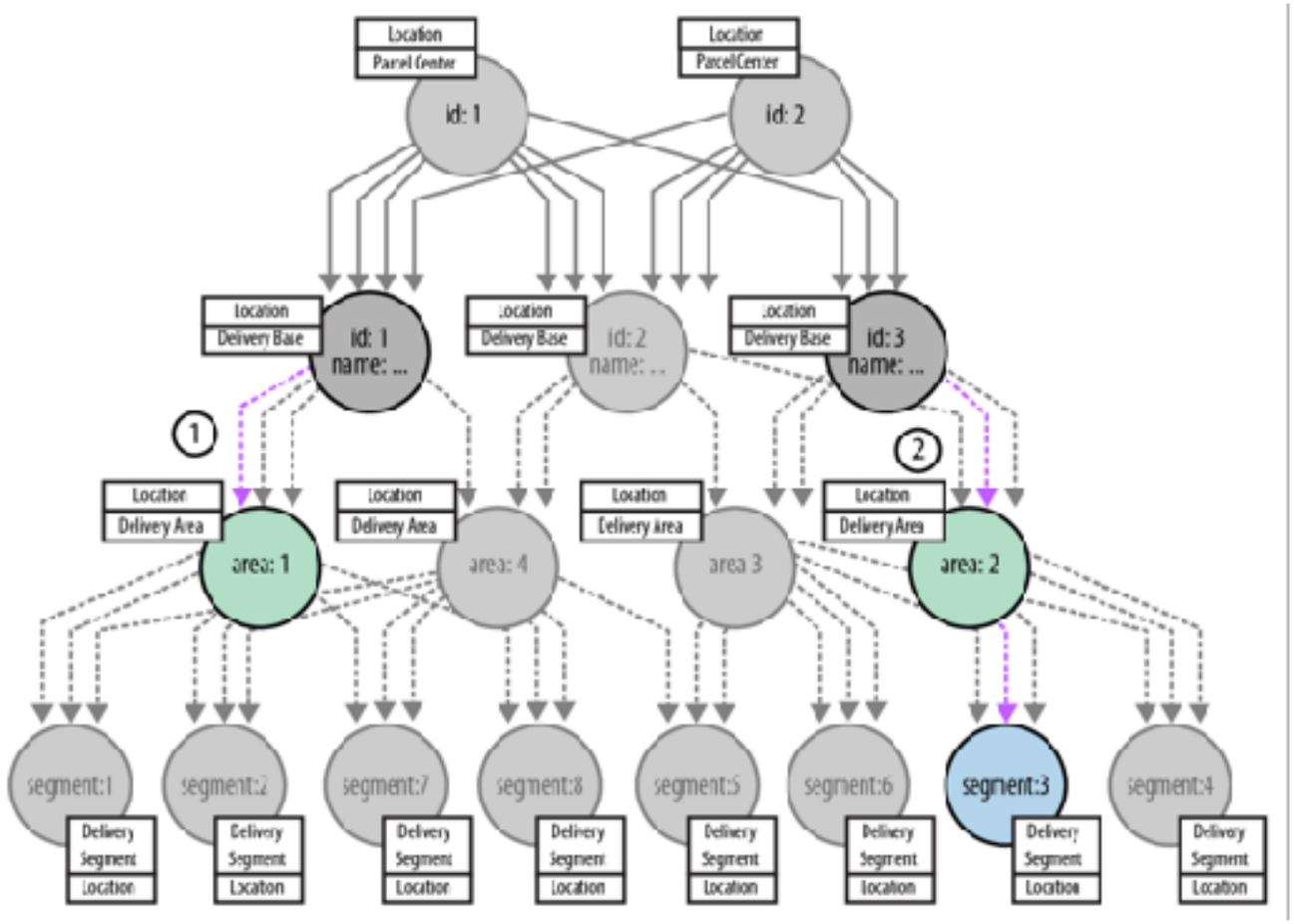
数据图的生成代码:

```
//------------------------------------------------------
CREATE (p1:ParcelCenter{name:"P1"})
CREATE (p2:ParcelCenter{name:"P2"})
CREATE (d1:DeliveryBase{name:"D1"})
CREATE (d2:DeliveryBase{name:"D2"})
CREATE (a1:DeliveryArea{name:"A1"})
CREATE (a2:DeliveryArea{name:"A2"})
CREATE (a3:DeliveryArea{name:"A3"})
CREATE (s1:Segment{name:"S1"})
CREATE (s2:Segment{name:"S2"})
CREATE (s3:Segment{name:"S3"})
CREATE (s4:Segment{name:"S4"})
CREATE (p1)-[:CONNECTED_TO{start_date:1,end_date:4,cost:1}]->(d1)
CREATE (p1)-[:CONNECTED_TO{start_date:6,end_date:7,cost:1}]->(d1)
CREATE (p1)-[:CONNECTED_TO{start_date:1,end_date:4,cost:1}]->(d2)
CREATE (p1)-[:CONNECTED_TO{start_date:6,end_date:7,cost:1}]->(d2)
CREATE (p2)-[:CONNECTED_TO{start_date:1,end_date:7,cost:2}]->(d1)
CREATE (p2)-[:CONNECTED_TO{start_date:1,end_date:7,cost:2}]->(d2)
CREATE (d1)-[:DELIVERY_ROUTE{start_date:1,end_date:7,cost:2}]->(a1)
CREATE (d2)-[:DELIVERY_ROUTE{start_date:1,end_date:7,cost:3}]->(a2)
CREATE (d2)-[:DELIVERY_ROUTE{start_date:1,end_date:7,cost:2}]->(a3)
CREATE (a1)-[:DELIVERY_ROUTE{start_date:1,end_date:5,cost:1}]->(s1)
CREATE (a1)-[:DELIVERY_ROUTE{start_date:1,end_date:5,cost:1}]->(s2)
CREATE (a2)-[:DELIVERY_ROUTE{start_date:6,end_date:7,cost:2}]->(s1)
CREATE (a2)-[:DELIVERY_ROUTE{start_date:6,end_date:7,cost:2}]->(s2)
CREATE (a2)-[:DELIVERY_ROUTE{start_date:1,end_date:5,cost:2}]->(s3)
CREATE (a2)-[:DELIVERY_ROUTE{start_date:1,end_date:5,cost:2}]->(s4)
CREATE (a3)-[:DELIVERY_ROUTE{start_date:6,end_date:7,cost:1}]->(s3)
CREATE (a3)-[:DELIVERY_ROUTE{start_date:6,end_date:7,cost:1}]->(s4)
//------------------------------------------------------------
```

物流数据模型：路径计算——查询举例

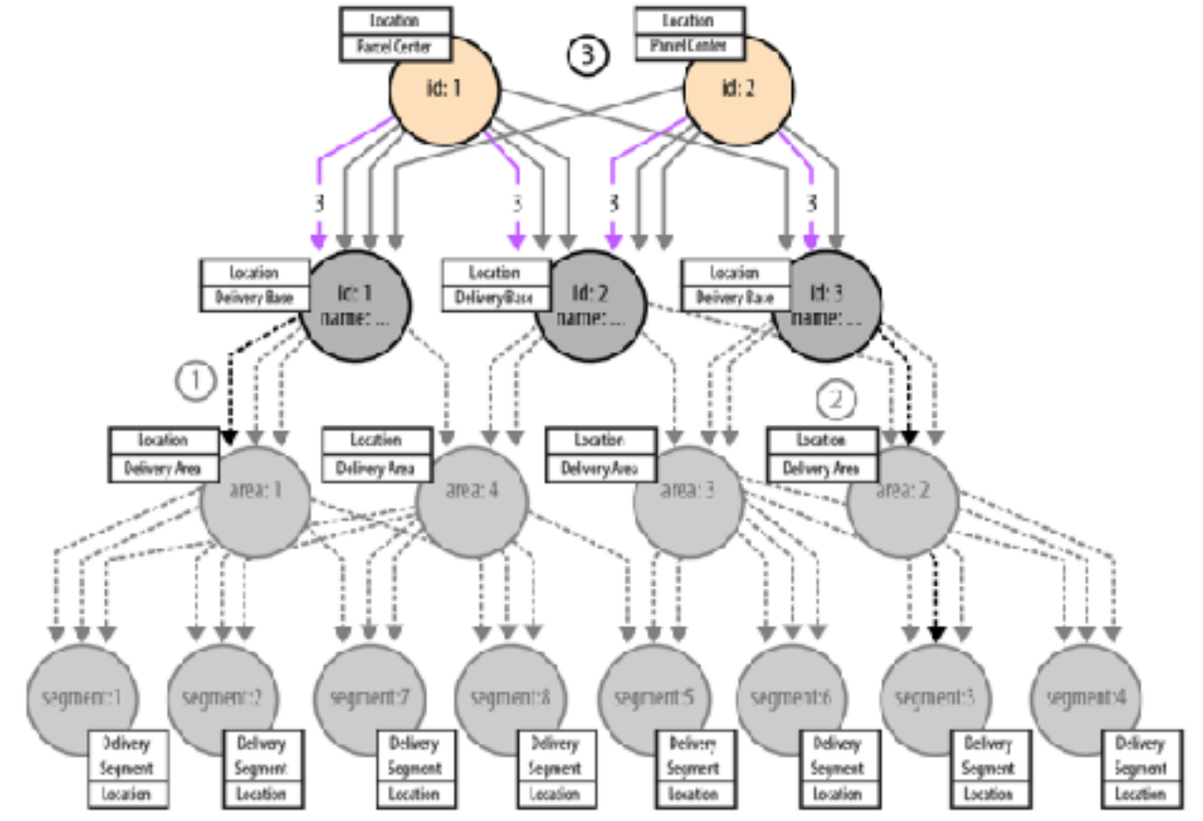
查询从S1到S4在星期4到星期5的最快路径

```
// 选定S1和S4结点
MATCH (s:Segment {name:"S1"}),
      (e:Segment {name:"S4"})

// 从S1向上走1~2个DELIVERY_ROUTE关系
MATCH upLeg = (s)<-[:DELIVERY_ROUTE*1..2]-(db1)

//走的DELIVERY_ROUTE必须在包裹运送期间内一直开启
WHERE all(r in relationships(upLeg)
        WHERE r.start_date <= 4
        AND r.end_date >= 5)

// S4类似处理
WITH e, upLeg, db1
MATCH downLeg = (db2)-[:DELIVERY_ROUTE*1..2]->(e)
WHERE all(r in relationships(downLeg)
        WHERE r.start_date <= 4
        AND r.end_date >= 5)
```

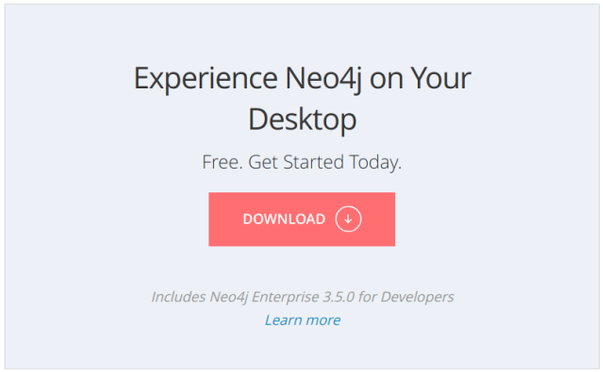


物流数据模型：路径计算——查询举例

```
WITH db1, db2, upLeg, downLeg

// 选择从upLeg到downLeg上层的路径
MATCH topRoute = (db1)<-[:CONNECTED_TO]-()-[:CONNECTED_TO*1..3]-(db2)
WHERE all(r in relationships(topRoute)
      WHERE r.start_date <= 4
      AND r.end_date >= 5)

WITH upLeg, downLeg, topRoute,
// 上层路径按代价(cost)排序，选择最小的.
reduce(weight=0, r in relationships(topRoute) | weight+r.cost) AS score
   ORDER BY score ASC
   LIMIT 1

// 打印结点
RETURN (nodes(upLeg) + tail(nodes(topRoute)) + tail(nodes(downLeg))) AS n
//-------------------------------------------------------------------------
```



物流数据模型：实例文件运行

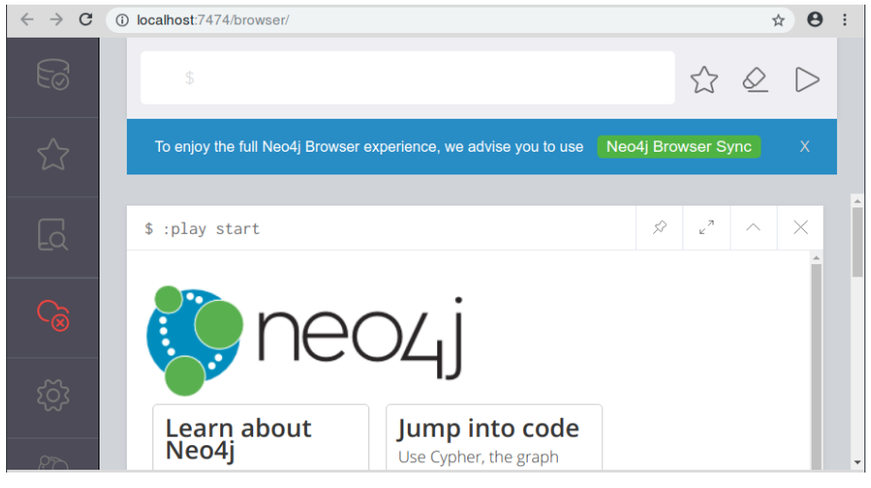


1. 下载Neo4j的server community版本。

2. 切换到下载文件的bin目录运行neo4j

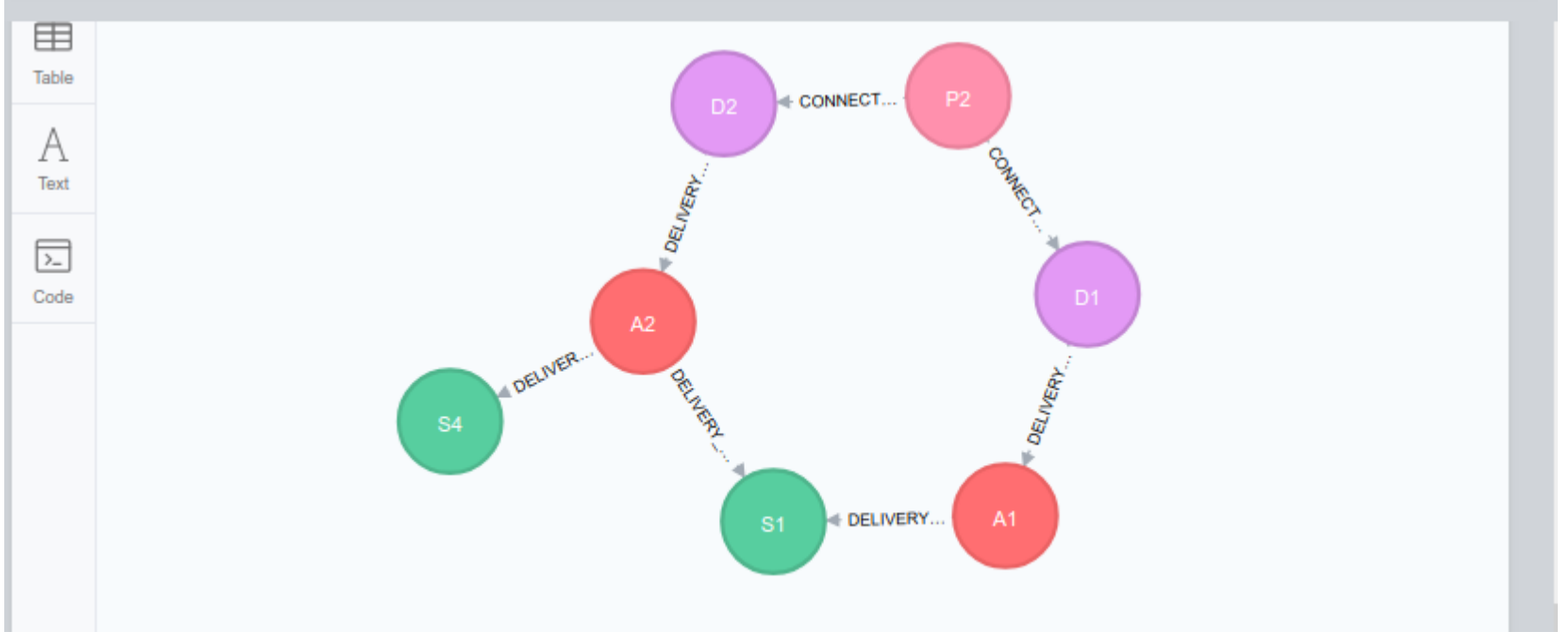
```
yancy@yancy-OptiPlex-390: ~/Downloads/neo4j-community-3.5.0-unix/neo4j-community-3.5.0/bin
yancy@yancy-OptiPlex-390:~$ cd ~/Downloads/
agraph-6.4.4-ubuntu-16.04-20180910/ neo4j-community-3.5.0-unix/
graph-databases-use-cases/          onedrive-d/
ideaIC-2018.3/
yancy@yancy-OptiPlex-390:~$ cd ~/Downloads/neo4j-community-3.5.0-unix/neo4j-comm
unity-3.5.0/bin/
yancy@yancy-OptiPlex-390:~/Downloads/neo4j-community-3.5.0-unix/neo4j-community-
3.5.0/bin$ ./neo4j start
Active database: graph.db
Directories in use:
  home:         /home/yancy/Downloads/neo4j-community-3.5.0-unix/neo4j-community
-3.5.0
  config:       /home/yancy/Downloads/neo4j-community-3.5.0-unix/neo4j-community
-3.5.0/conf
  logs:         /home/yancy/Downloads/neo4j-community-3.5.0-unix/neo4j-community
-3.5.0/logs
  plugins:      /home/yancy/Downloads/neo4j-community-3.5.0-unix/neo4j-community
-3.5.0/plugins
  import:       /home/yancy/Downloads/neo4j-community-3.5.0-unix/neo4j-community
-3.5.0/import
  data:         /home/yancy/Downloads/neo4j-community-3.5.0-unix/neo4j-community
-3.5.0/data
  certificates: /home/yancy/Downloads/neo4j-community-3.5.0-unix/neo4j-community
-3.5.0/certificates
```

3. 打开localhost:7474即可进入web界面



物流数据模型：示例文件运行

```
3 MATCH (s:Segment {name:"S1"}),
4       (e:Segment {name:"S4"})
5 // 从S1向上走1~2个DELIVERY_ROUTE关系
6 MATCH upLeg = (s)<-[:DELIVERY_ROUTE*1..2]-(db1)
7 // 走的DELIVERY_ROUTE必须在包裹运送期间内一直开启
8 WHERE all(r in relationships(upLeg)
```



可以看出，Neo4j已经将相关的结点和关系打印出来了(从A2到S1的关系是不符合要求的，时间不对，这里显示的是子图)，从S1到S4的最短路径为S1->A1->D1->P2->D2->A2->S4