Loggi SLA Forecasting

Environment setup

In [1]:

```
#importing necessery libraries for future analysis of the dataset
import pandas as pd
import datetime
import numpy as np
import os
from sqlalchemy import create_engine, text
import matplotlib.pyplot as plt
%matplotlib inline
import seaborn as sns
```

SQL Connection setup

In [2]:

```
db_host = 'postgresql://team4:team4@ds4a-lbenetton-instance.c6qxfh7ops9d.us-east-2.rds.amazonaws.com/ds4a_team4'
engine=create_engine(db_host, max_overflow=20)

def run_query(sql):
    result = engine.connect().execution_options(isolation_level="AUTOCOMMIT").execute(text(sql))
    return pd.DataFrame(result.fetchall(), columns=result.keys())
```

Import Data and Start Analisys

In [3]:

```
sql = """
    select
        *
    from package_ok
    where completed between '20191201' and '20191215'
;"""
df = run_query(sql)
df.head()
```

Out[3]:

	packid	completed	itineraryid	tasktype	ackstatus	waypointrole	agreedslo1	agreedslo2	fi
0	37c04ee8-eb16-e5e2-7021-396373cac9e1	2019-12-01 06:45:08	8f89e50a-26a8-336c-e48e-f3cb921c0703	Retirada no last-mile	Realizado com sucesso	Distribution Center	D1	D1	
1	291537c9-a0d2-1245-d071-66d722e9732d	2019-12-01 00:02:51	4dbb9eb2-1c61-7f96-454b-7d1701901ca0	Entrega	Realizado com sucesso	Recipient Address	D0	D0	
2	6160b754-e84b-ceba-44da-a3db6b061888	2019-12-01 00:13:01	4dbb9eb2-1c61-7f96-454b-7d1701901ca0	Entrega	Realizado com sucesso	Recipient Address	D0	D0	
3	edcb3f98-6804-e373-e753-8a8b658a3585	2019-12-01 00:18:04	84c8bc48-cebe-56a3-724f-dcce25c2b384	Entrega	Realizado com sucesso	Recipient Address	D1	D1	
4	8a918f38-ea3c-b000-a737-62576bd854c4	2019-12-01 00:32:34	ce268aba-9a66-3860-cc22-a6497d70c592	Entrega	Realizado com sucesso	Recipient Address	D2	D2	

5 rows × 36 columns

Adding reference date fields

In [4]:

```
# Date completed
df['month'] = df['completed'].dt.month
df['day'] = df['completed'].dt.day
df['hour'] = df['completed'].dt.time

# Package delivered
df['sla_ok'] = df['deadlinetime1'] > df['firstdeliverytime1']
```

In [5]:

```
df.head()
```

Out[5]:

	packid	completed	itineraryid	tasktype	ackstatus	waypointrole	agreedslo1	agreedslo2	fi
0	37c04ee8-eb16-e5e2-7021-396373cac9e1	2019-12-01 06:45:08	8f89e50a-26a8-336c-e48e-f3cb921c0703	Retirada no last-mile	Realizado com sucesso	Distribution Center	D1	D1	
1	291537c9-a0d2-1245-d071-66d722e9732d	2019-12-01 00:02:51	4dbb9eb2-1c61-7f96-454b-7d1701901ca0	Entrega	Realizado com sucesso	Recipient Address	D0	D0	
2	6160b754-e84b-ceba-44da-a3db6b061888	2019-12-01 00:13:01	4dbb9eb2-1c61-7f96-454b-7d1701901ca0	Entrega	Realizado com sucesso	Recipient Address	D0	D0	
3	edcb3f98-6804-e373-e753-8a8b658a3585	2019-12-01 00:18:04	84c8bc48-cebe-56a3-724f-dcce25c2b384	Entrega	Realizado com sucesso	Recipient Address	D1	D1	
4	8a918f38-ea3c-b000-a737-62576bd854c4	2019-12-01 00:32:34	ce268aba-9a66-3860-cc22-a6497d70c592	Entrega	Realizado com sucesso	Recipient Address	D2	D2	

5 rows × 40 columns

In [6]:

```
#checking amount of rows in given dataset to understand the size we are working with
len(df)
```

Out[6]:

```
2127669
```

In [7]:

```
#checking type of every column in the dataset
df.dtypes
```

Out[7]:

```
packid                                 object
completed                      datetime64[ns]
itineraryid                            object
tasktype                               object
ackstatus                              object
waypointrole                           object
agreedslo1                             object
agreedslo2                             object
finalcity                              object
mesoregion                             object
companyid1                             object
companyid2                             object
packstatus1                            object
packstatus2                            object
height                                float64
length                                float64
width                                 float64
realweight                            float64
deadlinetime1                  datetime64[ns]
deadlinetime2                  datetime64[ns]
crossdockingarrivaltime        datetime64[ns]
transferdispatchtime           datetime64[ns]
transferreceivaltime           datetime64[ns]
lastmileallocationstarttime    datetime64[ns]
lastmiledriverpickuptime       datetime64[ns]
firstdeliverytime1             datetime64[ns]
firstdeliverytime2             datetime64[ns]
city                                   object
transporttype                          object
product                                object
productversion                         object
created                        datetime64[ns]
accepted                       datetime64[ns]
checkedin                      datetime64[ns]
pickupcheckout                 datetime64[ns]
distributioncenter                     object
month                                   int64
day                                     int64
hour                                   object
sla_ok                                   bool
dtype: object
```

Understadning, Wrangling and Cleaning Data

In [8]:
```
#looking to find out first what columns have null values
#using 'sum' function will show us how many nulls are found in each column in dataset
df.isnull().sum()
```

Out[8]:
```
packid                           0
completed                        0
itineraryid                      0
tasktype                         0
ackstatus                        0
waypointrole                     0
agreedslo1                       0
agreedslo2                       0
finalcity                        6
mesoregion                     122
companyid1                       0
companyid2                       0
packstatus1                      0
packstatus2                      0
height                           0
length                           0
width                            0
realweight                       0
deadlinetime1                    0
deadlinetime2                    0
crossdockingarrivaltime          0
transferdispatchtime        106645
transferreceivaltime        916843
lastmileallocationstarttime 251354
lastmiledriverpickuptime      6931
firstdeliverytime1            4124
firstdeliverytime2            4124
city                             0
transporttype                    0
product                          0
productversion                   0
created                          0
accepted                      1189
checkedin                     2865
pickupcheckout                7728
distributioncenter            4853
month                            0
day                              0
hour                             0
sla_ok                           0
dtype: int64
```

In [9]:
```
#let's proceed with examing some interesting categorical unique values

#examining the unique values of n_group as this column will appear very handy for lat
er analysis
df.agreedslo1.unique()
```

Out[9]:
```
array(['D1', 'D0', 'D2', 'D3'], dtype=object)
```

In [10]:

```
#examining the unique values of n_group as this column will appear very handy for lat
er analysis
df.finalcity.unique()
```

Out[10]:

```
array(['São Paulo', 'Brasília', 'Hortolândia', 'Campinas', 'Sumaré',
       'Porto Alegre', 'Belo Horizonte', 'Ribeirão das Neves',
       'Guarulhos', 'Barueri', 'Santana de Parnaíba', 'Santo André',
       'Osasco', 'Embu das Artes', 'Taboão da Serra', 'Salvador',
       'Carapicuíba', 'Sorocaba', 'Uberlândia', 'Curitiba', 'São José',
       'Florianópolis', 'São Caetano do Sul', 'Palhoça',
       'São Bernardo do Campo', 'Cotia', 'Valinhos', 'Mauá', 'Joinville',
       'Piracicaba', 'Diadema', 'Goiânia', 'São José dos Pinhais',
       'Aparecida de Goiânia', 'Jacareí', 'São José dos Campos',
       'Pinhais', 'Rio de Janeiro', 'São Vicente', 'Praia Grande',
       'Votorantim', 'Santos', 'Ribeirão Preto', 'Cubatão', 'Recife',
       'Niterói', 'Belford Roxo', 'Nova Iguaçu', 'Duque de Caxias',
       'Vila Velha', 'Bauru', 'Fortaleza', 'Araçatuba', 'Araraquara',
       'Limeira', 'Suzano', 'Torres', 'Vitória', 'Manaus',
       'Santana do Paraíso', 'Itauçu', 'Itapevi', 'São Leopoldo',
       'Jaboatão dos Guararapes', 'Resende', 'Itaquaquecetuba',
       'Mogi das Cruzes', 'n/a', 'Jundiaí', 'Borda da Mata',
       'Itapecerica da Serra', 'Contagem', 'Paulista', 'Queimados',
       'Serra', 'São Gabriel', 'Nova Odessa', 'São José do Rio Preto',
       'São Gonçalo', 'Guarujá', 'Mandaguari', 'Vespasiano',
       'Bragança Paulista', 'Santa Luzia', 'Poá', 'Arapongas', 'Londrina',
       'Rondonópolis', 'Valparaíso de Goiás', 'Caucaia', '\u200b',
       'Vinhedo', None, 'Campo Limpo Paulista', 'Bocaina',
       'Novo Hamburgo', 'Maringá', 'Aracaju', 'Porto Feliz', 'Rolândia',
       'Teresópolis', 'Patrocínio', 'Cunha', 'Vargem Grande Paulista'],
      dtype=object)
```

In [11]:

```
#examining the unique values of n_group as this column will appear very handy for lat
er analysis
df.city.unique()
```

Out[11]:

```
array(['São Paulo', 'Brasília', 'Campinas', 'Porto Alegre',
       'Belo Horizonte', 'Salvador', 'Sorocaba', 'Uberlândia', 'Curitiba',
       'Florianópolis', 'Joinville', 'Piracicaba', 'Goiânia',
       'São José dos Campos', 'Rio de Janeiro', 'Santos',
       'Ribeirão Preto', 'Recife', 'Vitória', 'Fortaleza', 'Manaus',
       'Rio de Janeiro - Redespacho Local', 'Maringá'], dtype=object)
```

In [12]:

```
#examining the unique values of n_group as this column will appear very handy for lat
er analysis
df.transporttype.unique()
```

Out[12]:

```
array(['Moto', 'Carro', 'Van'], dtype=object)
```

In [14]:

```
#examining the unique values of n_group as this column will appear very handy for lat
er analysis
df['product'].unique()
```

Out[14]:

```
array(['Pro'], dtype=object)
```

In [15]:

```
#examining the unique values of n_group as this column will appear very handy for lat
er analysis
df['productversion'].unique()
```

Out[15]:

```
array(['Prime', 'Start'], dtype=object)
```

Exploring and Visualizing Data

In [30]:

```
#setting figure size for future visualizations
sns.set(rc={'figure.figsize':(10,8)})
```

In [19]:

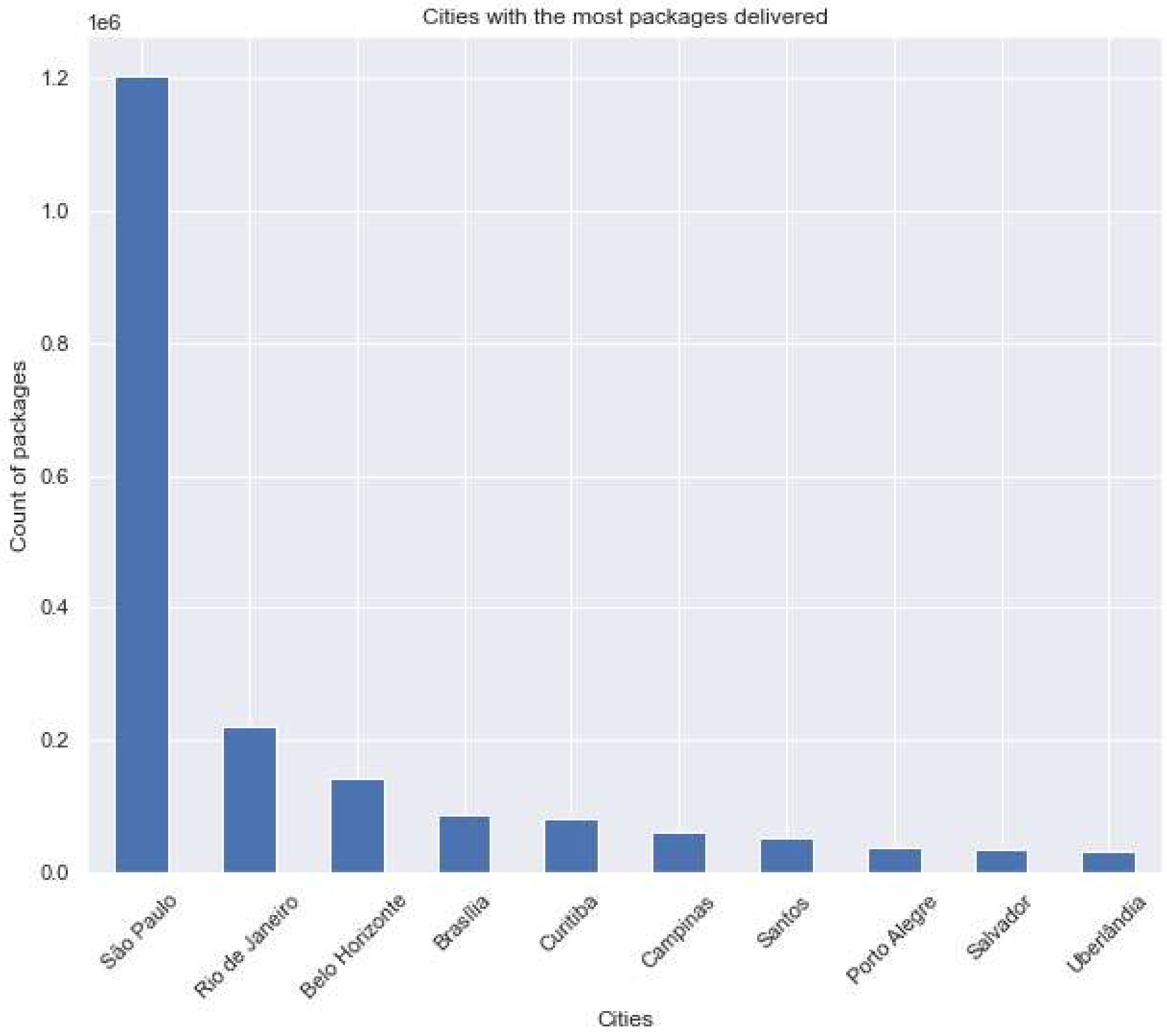
```
#let's see what city have the most qty of packages destinations
top_city=df['city'].value_counts().head(10)
top_city
```

Out[19]:

```
São Paulo          1203558
Rio de Janeiro      220634
Belo Horizonte      142685
Brasília             87915
Curitiba             82670
Campinas             60348
Santos               51297
Porto Alegre         37320
Salvador             35229
Uberlândia           32333
Name: city, dtype: int64
```

In [31]:
```
graph1=top_city.plot(kind='bar')
graph1.set_title('Cities with the most packages delivered')
graph1.set_ylabel('Count of packages')
graph1.set_xlabel('Cities')
graph1.set_xticklabels(graph1.get_xticklabels(), rotation=45)
```

Out[31]:
```
[Text(0, 0, 'São Paulo'),
 Text(0, 0, 'Rio de Janeiro'),
 Text(0, 0, 'Belo Horizonte'),
 Text(0, 0, 'Brasília'),
 Text(0, 0, 'Curitiba'),
 Text(0, 0, 'Campinas'),
 Text(0, 0, 'Santos'),
 Text(0, 0, 'Porto Alegre'),
 Text(0, 0, 'Salvador'),
 Text(0, 0, 'Uberlândia')]
```



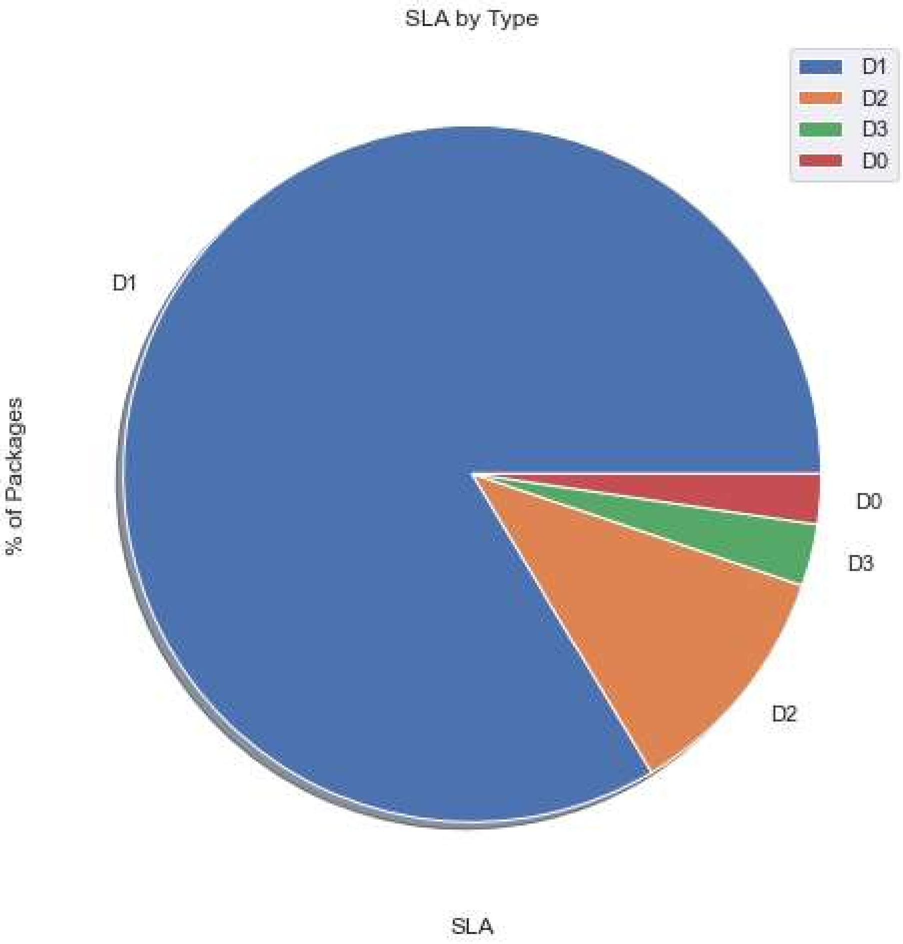
In [83]:
```
#let's see what type of SLA have the most % of packages
top_sla=df['agreedslo1'].value_counts()
top_sla
```

Out[83]:
```
D1    1778888
D2     237903
D3      61201
D0      49677
Name: agreedslo1, dtype: int64
```

```
In [91]: sns.set(rc={'figure.figsize':(10,8)})

         graph2=top_sla.plot(kind='pie', shadow=True, legend = True)
         graph2.set_title('SLA by Type')
         graph2.set_ylabel('% of Packages')
         graph2.set_xlabel('SLA')
         #graph2.set_xticklabels(graph1.get_xticklabels(), rotation=45)
```

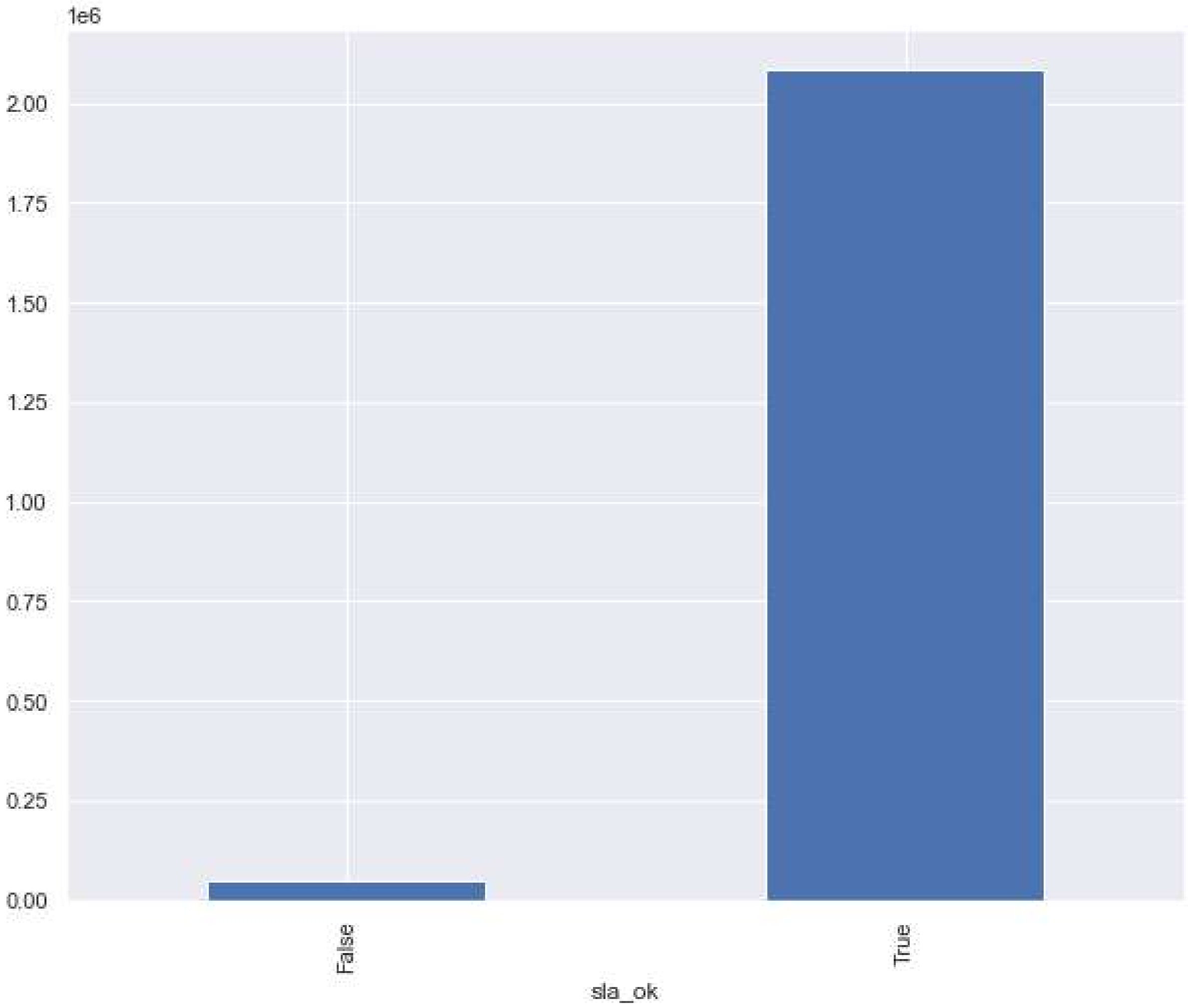
```
Out[91]: Text(0.5, 0, 'SLA')
```



```
In [90]: # Packages completed vs fault
         plt.figure(figsize=(10,8))

         df.groupby('sla_ok')['packid'].count().plot(kind='bar')
```

```
Out[90]: <matplotlib.axes._subplots.AxesSubplot at 0x21d49908ac8>
```



In [72]:

```
# Packages completed vs fault by city

city_tot = df.groupby(['city','sla_ok'])['packid'].count().reset_index()

city_ok = city_tot[city_tot['sla_ok'] == True]
city_fault = city_tot[city_tot['sla_ok'] == False]

city_sla_tot = city_tot.groupby(['city']).sum()

city_sla_ok = pd.merge(city_ok, city_sla_tot, left_on='city', right_on='city', how='left').drop(['sla_ok_x','sla_ok_y'], axis=1)

city_sla_ok['pct'] = city_sla_ok['packid_x'] / city_sla_ok['packid_y']

city_sla_ok[['city', 'pct']].sort_values('pct')
```

Out[72]:

	city	pct
1	Brasília	0.930069
8	Manaus	0.939895
0	Belo Horizonte	0.950492
13	Rio de Janeiro	0.974832
15	Salvador	0.974907
19	São Paulo	0.981987
3	Curitiba	0.984468
11	Recife	0.984779
10	Porto Alegre	0.985531
21	Vitória	0.986603
12	Ribeirão Preto	0.986887
6	Goiânia	0.986972
5	Fortaleza	0.990489
16	Santos	0.991481
20	Uberlândia	0.993103
2	Campinas	0.993123
18	São José dos Campos	0.993407
4	Florianópolis	0.993441
9	Piracicaba	0.995306
17	Sorocaba	0.996705
7	Joinville	0.996970
14	Rio de Janeiro - Redespacho Local	1.000000

In [74]:

```python
# Packages completed vs fault by transport type

tt_tot = df.groupby(['transporttype','sla_ok'])['packid'].count().reset_index()

tt_ok = tt_tot[tt_tot['sla_ok'] == True]
tt_fault = tt_tot[tt_tot['sla_ok'] == False]

tt_sla_tot = tt_tot.groupby(['transporttype']).sum()

tt_sla_ok = pd.merge(tt_ok, tt_sla_tot, left_on='transporttype', right_on='transporttype', how='left').drop(['sla_ok_x','sla_ok_y'], axis=1)

tt_sla_ok['pct'] = city_sla_ok['packid_x'] / city_sla_ok['packid_y']

tt_sla_ok[['transporttype', 'pct']].sort_values('pct')
```

Out[74]:

	transporttype	pct
1	Moto	0.930069
0	Carro	0.950492
2	Van	0.993123

In [81]:

```python
# Packages completed vs fault by SLA

slo_tot = df.groupby(['agreedslo1','sla_ok'])['packid'].count().reset_index()

slo_ok = slo_tot[slo_tot['sla_ok'] == True]
slo_fault = slo_tot[slo_tot['sla_ok'] == False]

slo_sla_tot = slo_tot.groupby(['agreedslo1']).sum()

slo_sla_ok = pd.merge(slo_ok, slo_sla_tot, left_on='agreedslo1', right_on='agreedslo1', how='left').drop(['sla_ok_x','sla_ok_y'], axis=1)

slo_sla_ok['pct'] = slo_sla_ok['packid_x'] / slo_sla_ok['packid_y']

slo_sla_ok[['agreedslo1', 'pct']].sort_values('pct')
```

Out[81]:

	agreedslo1	pct
0	D0	0.962981
2	D2	0.965368
1	D1	0.980280
3	D3	0.980523