1. Overview

## Generate the size breakup excels from ABM

## Generate the Ritemate size breakup sheet

## Allocate the TIS No.

## Copy to Order Sheet

## Copy to Finance Sheet

## Generate Order Form

## Order Trace

## Test Report recording

## Invoice and Packing List Checking

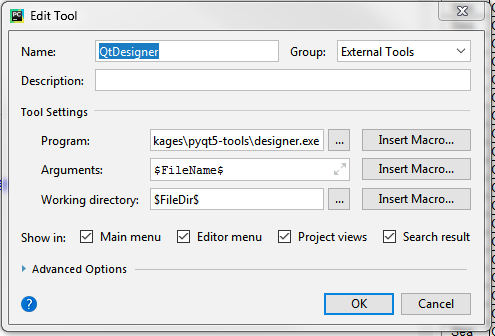
## Shipment booking check

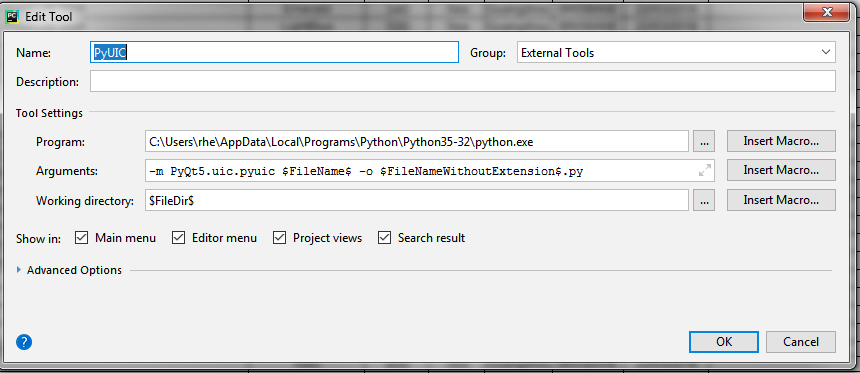
.

1. Database
2. Trouble shooting

Git can’t find the TISProduction/TISProduction 下面的文件，而是把整个目录看成一个文件。原因是这个目录下面也有个.git ，删掉以后就可以了

External Tools- UI Designer





在TISDesk应用程序里面要调用orders（django的Model），必须在调用前建立django环境，导入settings，如下：

sys.path.append(**'C:\\Users\\rhe\\PyCharm\\TISProduction'**) *# 将项目路径添加到系统搜寻路径当中*os.environ[**'DJANGO\_SETTINGS\_MODULE'**] = **'TISProduction.settings'** *# 设置项目的配置文件*django.setup() *# 加载项目配置*

ORDER\_FIELD=OrderedDict([(**'B'**,**'Customer'**),(**'C'**,**'Supplier'**),(**'D'**,**'Style'**),(**'E'**,**'TISNo'**),(**'F'**,**'InternalNo'**),(**'G'**,**'ShipCode'**),(**'H'**,**'ETD'**),(**'I'**,**'ETA'**),(**'J'**,**'InStore'**),(**'K'**,**'ABMInStore'**),(**'L'**,**'CTM'**),(**'M'**,**'Commodity'**),(**'N'**,**'Colour'**),(**'O'**,**'Quantity'**),(**'P'**,**'Freight'**),(**'Q'**,**'FOBPort'**),(**'R'**,**'ETAPort'**),(**'S'**,**'OrderDate'**),(**'T'**,**'PPSample'**),(**'U'**,**'SSSample'**),(**'V'**,**'TestReport'**),(**'W'**,**'Material'**),(**'X'**,**'Cartons'**),(**'Y'**,**'Volume'**),(**'Z'**,**'Weight'**)])  
FINANCE\_FIELD=OrderedDict([(**'A'**,**'Customer'**),(**'B'**,**'Supplier'**),(**'C'**,**'CTM'**),(**'D'**,**'OrderDate'**),(**'E'**,**'TISNo'**),(**'F'**,**'InternalNo'**),(**'G'**,**'Style'**),(**'H'**,**'Commodity'**),(**'I'**,**'Quantity'**),(**'J'**,**'FreightPrice'**),(**'K'**,**'ExchangeRate'**),(**'L'**,**'LandCost'**),(**'M'**,**'FreightCost'**),(**'N'**,**'Sales'**),(**'O'**,**'MarkUp'**),(**'P'**,**'FOBPort'**),(**'Q'**,**'Freight'**),(**'R'**,**'ETD'**),(**'S'**,**'ETA'**),(**'T'**,**'InStore'**),(**'U'**,**'FactoryPayMonth'**),(**'V'**,**'ClientPayMonth'**),(**'W'**,**'SellPrice'**),(**'X'**,**'PurchasePrice'**),(**'Y'**,**'Deposit'**),(**'Z'**,**'FTYInvoice'**),(**'AA'**,**'Fabric'**),(**'AB'**,**'Colour'**),(**'AC'**,**'Remark'**)])

**class** Order(models.Model):  
 tis\_no=models.CharField(max\_length=15,default=**'TIS18-SO'**)  
 internal\_no=models.CharField(max\_length=11,default=**'2018-PO'**)  
 ctm\_no=models.CharField(max\_length=50)  
 client=models.TextField(max\_length=50)  
 supplier=models.TextField(max\_length=50)  
 product=models.ForeignKey(Product,on\_delete=models.PROTECT,)  
 colour=models.TextField(max\_length=20)  
 quantity=models.IntegerField()  
 shipment=models.ForeignKey(Shipment,on\_delete=models.SET\_NULL,null=**True**)  
 purchase\_price=models.DecimalField(max\_digits=5,decimal\_places=2,default=0)  
 sell\_price=models.DecimalField(max\_digits=5,decimal\_places=2,default=0)  
 size1=models.IntegerField(default=0)  
 size2=models.IntegerField(default=0)  
 size3=models.IntegerField(default=0)  
 size4=models.IntegerField(default=0)  
 size5=models.IntegerField(default=0)  
 size6=models.IntegerField(default=0)  
 size7=models.IntegerField(default=0)  
 size8=models.IntegerField(default=0)  
 size9=models.IntegerField(default=0)  
 size10=models.IntegerField(default=0)  
 size11=models.IntegerField(default=0)  
 size12=models.IntegerField(default=0)  
 size13=models.IntegerField(default=0)  
 size14=models.IntegerField(default=0)  
 size15=models.IntegerField(default=0)  
 size16=models.IntegerField(default=0)  
 size17=models.IntegerField(default=0)  
 size18=models.IntegerField(default=0)  
 size19=models.IntegerField(default=0)  
 size20=models.IntegerField(default=0)  
 size21=models.IntegerField(default=0)  
 size22=models.IntegerField(default=0)  
 size23=models.IntegerField(default=0)  
 size24=models.IntegerField(default=0)  
 size25=models.IntegerField(default=0)  
 size26=models.IntegerField(default=0)  
 size27=models.IntegerField(default=0)  
 size28=models.IntegerField(default=0)  
 size29=models.IntegerField(default=0)  
 size30=models.IntegerField(default=0)

**class** Product(models.Model):  
  
 style\_no=models.TextField(max\_length=50)  
 commodity=models.TextField(max\_length=100)  
 fabric=models.TextField(max\_length=100)  
 fabric\_usage=models.DecimalField(max\_digits=4,decimal\_places=2)  
 quantity\_per\_carton=models.IntegerField()  
 volume\_per\_carton=models.DecimalField(max\_digits=4,decimal\_places=3)  
 weight\_per\_carton=models.DecimalField(max\_digits=3,decimal\_places=1)

**class** Shipment(models.Model):  
 SEAFREIGHT=**'S'** AIRFREIGHT=**'F'** COURIER=**'C'** MODE=(  
 (SEAFREIGHT,**'Seafreight'**),  
 (AIRFREIGHT,**'Airfreight'**),  
 (COURIER,**'Courier'**),  
 )  
 ETD\_PORT=(  
 (**'GUANGZHOU'**,**'Guangzhou'**),  
 (**'NINGBO'**,**'Ningbo'**),  
 (**'SHANGHAI'**,**'Shanghai'**),  
 (**'QINGDAO'**,**'Qingdao'**),  
 (**'XINGANG'**,**'Xingang'**),  
 (**'BEIJING'**,**'Beijing'**),  
 (**'OTHER'**,**'other'**),  
 )  
 ETA\_PORT = (  
 (**'BRISBANE'**, **'Brisbane'**),  
 (**'SYDNEY'**, **'Sydney'**),  
 (**'MELBOURNE'**, **'Melbourne'**),  
 (**'OTHER'**, **'Other'**),  
 )  
 code=models.TextField(max\_length=20)  
 etd=models.DateField(default=datetime.datetime.now().date())  
 eta=models.DateField()  
 instore=models.DateField()  
 instore\_abm=models.DateField()  
 total\_quantity=models.IntegerField(default=0)  
 volume=models.DecimalField(max\_digits=4,decimal\_places=1,default=0)  
 cartons=models.IntegerField(default=0)  
 mode=models.TextField(max\_length=1,choices=MODE,default=SEAFREIGHT)  
 etd\_port=models.TextField(choices=ETD\_PORT, default=**'GUANGZHOU'**)  
 eta\_port=models.TextField(choices=ETA\_PORT, default=**'BRISBANE'**)  
 container=models.TextField(max\_length=50,null=**True**,blank=**True**)

**class** FabricTrim(models.Model):  
 colour\_solid=models.TextField(max\_length=20)  
 order=models.ForeignKey(Order,on\_delete=models.CASCADE,)  
  
  
**class** SampleCheck(models.Model):  
 PPSAMPLE=**'P'** SSSAMPLE=**'S'** FABRIC=**'F'** TESTREPORT=**'T'** DOCUMENT=**'D'** INSPECTION=**'I'** TYPE=(  
 (PPSAMPLE,**'PP Sample'**),  
 (SSSAMPLE, **'Shipping Sample'**),  
 (FABRIC, **'Fabric Swatch'**),  
 (TESTREPORT, **'Test Report'**),  
 (INSPECTION, **'Inspection'**),  
 (DOCUMENT, **'Shipping Doc'**),  
 )  
 NONEED=**'N'** NEED=**'Y'** SENT=**'S'** RECEIVED=**'R'** APPROVED=**'A'** REJECTED=**'R'** STATUS=(  
 (NONEED,**'No Need'**),  
 (NEED, **'Need'**),  
 (SENT, **'Sent'**),  
 (RECEIVED, **'Received'**),  
 (APPROVED, **'Approved'**),  
 (REJECTED, **'Rejected'**),  
 )  
 type=models.CharField(max\_length=1,choices=TYPE)  
 status=models.CharField(max\_length=1,choices=STATUS,default=NONEED)  
 check\_date=models.DateTimeField(auto\_now\_add=**True**)  
 comment=models.TextField(max\_length=500,null=**True**,blank=**True**)  
 ref=models.TextField(max\_length=100)  
 order=models.ForeignKey(Order,on\_delete=models.CASCADE,)  
 fabric=models.ForeignKey(FabricTrim,on\_delete=models.CASCADE,)

**Invoice and Packing list**

**Packing List**

**"""  
The diction of packing\_lis is as below:  
{'detail':[{'form':,'to:','carton\_qty':,'colour\_detail':,'per\_carton\_pcs':,'per\_carton\_gw':,'per\_carton\_nw':,  
 'subtotal':,'length':,'width':,'height':,'size\_qty':{-xs:, ...}}],  
 'summary':{-COBALT BLUE:{'Order Qty':{'total':, 'size\_qty':{-xs:,....}},  
 'Actual Qty':{'total':, 'size\_qty':{-xs:,....}},  
 'Balance':{'total':, 'size\_qty':{-xs:,....}},  
 'Ratio':{'total':, 'size\_qty':{-xs:,....}}  
 }  
 ...  
 }  
 'total\_quantity':,'total\_carton':,'total\_gw':,'total\_nw':,'total\_volume':,'Style:,'style\_description':,  
 'invoice\_no':,'date':,'TISNo:  
}  
"""**

**After consolidate:**

**{'AW18F206-1': {'TIS17-SO4369':{'RM108V3R', [{'total\_gw': 365.20000000000005, 'total\_volume': 1.8095, 'total\_quantity': 862.0, 'style\_description': "Men's Woven L/S Shirt 3M Tape", 'total\_carton': 44.0, 'detail': [],’summary’:{}}]**

**Invoice:**

**OrderedDict([('AW18F206-1', [{'detail': OrderedDict([('TIS17-SO4369', OrderedDict([('RM1050R', [{'price': 9.5, 'amount': 8189.0, 'colour': 'ALL', 'quantity': 862}])])),**

**('TIS17-SO4370', OrderedDict([('RM107V2', [{'price': 7.05, 'amount': 20889.15, 'colour': 'ALL', 'quantity': 2963}])])**

**Actually:**

**{'AW18F206-1': {'TIS17-SO4369':{'RM1050R': [{'price': 9.5, 'amount': 8189.0, 'colour': 'ALL', 'quantity': 862}]}},**

**{'TIS17-SO4370':{'RM107V2': [{'price': 7.05, 'amount': 20889.15, 'colour': 'ALL', 'quantity': 2963}]}},**

**'AW18F205':{}**

**}**

Full Text Search:

1. <http://www.rkblog.rk.edu.pl/w/p/fulltext-search-sqlite-and-django-app/>

2. Error : sqlite3.OperationalError: no such module: fts3

downloaded and copied the sqlite precompiled Binaries for Windows win32 from [here](http://www.sqlite.org/download.html) into my Python DLL directory (C:\Users\rhe\AppData\Local\Programs\Python\Python35-32\DLLs)

3. When I used the QListWidget.findItems(str,QMatchFlags), waste much time: (1) at first PyCharm can’t find the QMatchFlags in QCore.Qt. Actually, it is no problem, when I import Qt, then use Qt.QMatchExactly as 2nd params. (2) PyCharm will prompt the QtMatchFlag in method, don’t use it. Just use the correct format QListWidget.findItems(‘

**Q & A**

**判断数据是否存在**

用get查询的时候，查询不到内容的时候会抛出异常，同样查询结果多余1条的时候也会抛出异常。因此，不能使用get，而应该使用filter。

filer若是查询不到数据，会返回一个空的查询集，[]  type类型是：Queryset。

查询到多余一条的时候会，还是会返回一个包含多个对象的查询集。

所有用filter查询到的是否为'[ ]'来判断是否存在。

|  |  |
| --- | --- |
| 1  2  3  4  5 | userinfo = Users.objects.filter(email = request.POST['email'])  if userinfo.exists():      print("yes,we have this email")  else:     print("sorry,email is not register") |

另外还可以使用count（）这个方法，userinfo.count() =0的话表示不存在数据，大于0的话表示存在一条或多条。

**当用annotate时**，如果model有meta class Meta 有 定义ordering=(**'tis\_no'**,**'colour'**)

时，会自动生成group by tis\_no,colour, 因此必须在语句中显式加入order\_by

如：client\_count=Order.objects.order\_by(**'client'**).values(**'client'**).annotate(clients=Count(**'client'**))

PYQT- QComboBox

当connect CurrentTextChanged, 在line\_edit 输入一个字符都会触发，当connect CurrentIndexChanged, 只会在整个内容改变而鼠标离开直到index changed的时候触发

在做split shipment 的时候，判断201807email, 偶发性出现不匹配so4555 rm200cf black. 不是固定的。 当加入logger跟踪so4445 时，又判断正常。很奇怪。经过跟踪，原因是在做product\_price.get\_formal\_colourname\_from\_alias时，因为使用dict.get(key=’’)，排序在dict是不固定的，获取的product是随机的，有些颜色是black, 有些颜色是Black. 我在程序比较字符串的时候没有做upper()转换。

当遍历List而且需要pop或者del 或者Insert时需要注意因为数据结构变化，遍历时会取到非所需的item，可以使用一个单独的序号（独立于for）并且根据实际操作来设定序号，例如当pop时，要保持本次的序号作为下次，因为序号list已经改变，当前序号的item被pop后，原来的下一个item会占用当前序号