In [385]:

```
import requests
from fuzzywuzzy import process

def getAssetsByType(Resource,Type,Preview):
    TABLE_CLASSTYPE = "com.infa.ldm.relational.Table";
    COL_CLASSTYPE = "com.infa.ldm.relational.Column";
    DOMAIN_CLASSTYPE = "com.infa.ldm.profiling.DataDomain";
    CORE_NAME = "core.name";
    CORE_RESOURCE_NAME = "core.resourceName";
    BGTERM = "com.infa.ldm.bg.BGTerm";
    DATASET_FLOW = "core.DataSetDataFlow";

    catalogServer = #
    uid = #
    pwd = #

    total = 1000
    offset = 0
    pageSize = 300
    attrCount=0
    custAttrCount=0

    query = catalogServer + '/access/2/catalog/data/objects?q=core.resourceName%3A%22' + \
    Resource + '%22%20AND%20core.allclassTypes%3A%22' + Type + '%22&offset=0&pageSize='+ str(pageSi
ze) + \
    '&related=false'

    while (offset < total):

        #make an api call
        resp = requests.get(query,auth=(uid,pwd))
        resp_dict = resp.json()
        status = resp.status_code
        if status != 200:
            print("error! " + str(status) + str(resp.json()))
            break
        total = resp_dict["metadata"]["totalCount"]
        offset += pageSize

        # for each attribute found...
        retMap = {}
        for attrDef in resp_dict["items"]:
            attrCount+=1
            attrId = attrDef["id"]
            for fact in attrDef["facts"]:
                if(fact["attributeId"] == CORE_NAME):
                    attrVal = fact["value"]
                    retMap[attrId] = attrVal

    return retMap
```

In [386]:

```
class FuzzyBGAssociater():
    """Sample REST API Program that associates data assets with business glossary terms
        based on fuzzy name matches"""

    def __init__(self):
        # Thresholds go from 1-100 where 100 stands for exact match.
        self.threshold = 80
        self.bg_resource="BG_DEFAULT_RESOURCE"
        self.resource="OrderEntry"
```

In [387]:

```
def main():
    fbg = FuzzyBGAssociater()
    #try:
        #login()
    #except:
```

```
        #print("Not Able to Login")

    try:
        termMap = getAssetsByType("BG_DEFAULT_RESOURCE","com.infa.ldm.bg.BGTerm", False)

        columnMap = getAssetsByType("OrderEntry","com.infa.ldm.relational.Column", False)
        print(str(len(termMap)) + ":" + str(len(columnMap)))
    except Exception as e:
        print(e)

    i = 1
    for columnId in columnMap.keys():
        colName = columnMap[columnId]
        results = process.extractBests(colName,termMap.values(),score_cutoff=80)
        if results:
            i+=1
            print(str(i) +":"+colName+":"+results[0][0])
```

In [388]:

```
main()
```

```
62:67
2:GENDER:Gender
3:GENDER:Gender
4:CATEGORY_DESCRIPTION:CAMEO Category Description
5:PRODUCT_ID:ID
6:PRODUCT_STATUS:Product
7:PRODUCT_ID:ID
8:PRODUCT_ID:ID
9:PRODUCT_ID:ID
10:PRODUCT_NAME:Product
11:CUST_FIRST_NAME:Name
12:CUST_FIRST_NAME:Name
13:CUST_LAST_NAME:Name
14:CUST_LAST_NAME:Name
15:CATEGORY_NAME:Name
16:WAREHOUSE_NAME:Name
17:TRANSLATED_NAME:Name
18:PROMO_NAME:Name
19:CUSTOMER_ID:ID
20:CUSTOMER_ID:ID
21:CUSTOMER_ID:ID
22:SUPPLIER_ID:ID
23:ACCOUNT_MGR_ID:ID
24:LOCATION_ID:ID
25:ACCOUNT_MGR_ID:ID
26:CATEGORY_ID:ID
27:PROMOTION_ID:ID
28:CATEGORY_ID:ID
29:CUST_EMAIL:Email
30:CUST_EMAIL:Email
31:PROMO_ID:ID
32:SALES_REP_ID:ID
33:WAREHOUSE_ID:ID
34:ORDER_ID:ID
35:ORDER_ID:ID
36:LINE_ITEM_ID:ID
37:WAREHOUSE_ID:ID
38:LANGUAGE_ID:ID
39:NLS_LANGUAGE:Language
40:NLS_LANGUAGE:Language
41:PRODUCT_DESCRIPTION:Product Descriptions
```