Finding our best-performing salespeople and products

Introduction

Business Context. You work for AdventureWorks, a company that sells outdoor sporting equipment. The company has many different locations and has been recording the sales of different locations on various products. You, their new data scientist, have been tasked with the question: **"What are our best products and salespeople and how can we use this information to improve our overall performance?"**

You have been given access to the relevant data files with documentation from the IT department. Your job is to extract meaningful insights from these data files to help increase sales. First, you will look at the best products and try to see how different products perform in different categories. Second, you will analyze the best salespeople to see if the commission percentage motivates them to sell more.

Business Problem. Your task is to **write queries in SQL to carry out the requested analysis**.

Analytical Context. You are given the data as a SQLite database. The company has been pretty vague about how they expect you to extract insights, but you have come up with the following plan of attack:

1. Load the database and ensure you can run basic queries against it
2. Look at how product ratings and total sales are related
3. See how products sell in different subcategories (bikes, helmets, socks, etc.)
4. Calculate which salespeople have performed the best in the past year
5. See if total sales are correlated with commission percentage

Of course, this is only your initial plan. As you explore the database, your strategy will likely change.

Overview of the data

The data for this case is contained in the `AdventureWorks.db` SQLite database. We will be focusing on the tables that belong to the Sales and Product categories. Complete documentation for the original data (of which you have only a subset) can be found here.

Product Tables:

- **Product**: one row per product that the company sells
- **ProductReview**: one row per rating and review left by customers
- **ProductModelProductDescriptionCulture**: a link between products and their longer descriptions also indicating a "culture" - which language and region the product is for
- **ProductDescription**: a longer description of each product, for a specific region

In [46]:
```
%%sql

SELECT salesorderdetail.salesorderid,
    salesorderheader.totaldue -
    (salesorderdetail.unitpricediscount*salesorderdetail.unitprice*salesorderdetail.orderqty)
     AS ordertotal
FROM salesorderdetail

JOIN salesorderheader ON salesorderdetail.salesorderid = salesorderheader.salesorderid

GROUP BY salesorderdetail.salesorderid
ORDER BY salesorderdetail.salesorderid ASC

LIMIT 5
```

```
 * sqlite:///AdventureWorks.db
Done.
```

Out[46]:

salesorderid	ordertotal
43659	23153.2339
43660	1457.3288
43661	36865.8012
43662	32474.9324
43663	472.3108

In [47]:
```
# Name your variable order_ordertotal
# YOUR CODE HERE

order_ordertotal = """

SELECT salesorderdetail.salesorderid,
    salesorderheader.totaldue -
    (salesorderdetail.unitpricediscount*salesorderdetail.unitprice*salesorderdetail.orderqty)
     AS ordertotal
FROM salesorderdetail

JOIN salesorderheader ON salesorderdetail.salesorderid = salesorderheader.salesorderid

GROUP BY salesorderdetail.salesorderid
ORDER BY salesorderdetail.salesorderid ASC

"""
```

6.2

Using the previous query as a subquery, find the sales for each salesperson for the year 2014 and display results for the top 5 salespeople.

Hint: You can get the `salesorderid` and `salespersonid` pairs from the `salesorderheader` table.

In [48]:

```
%%sql

WITH ordertotals AS

    (SELECT salesorderdetail.salesorderid,
    salesorderheader.totaldue -
    (salesorderdetail.unitpricediscount*salesorderdetail.unitprice*salesorderdetail.orderqty)
     AS ordertotal
    FROM salesorderdetail

    JOIN salesorderheader ON salesorderdetail.salesorderid = salesorderheader.salesorderid

    GROUP BY salesorderdetail.salesorderid
    ORDER BY salesorderdetail.salesorderid ASC)

SELECT SUM(ordertotals.ordertotal) AS ordertotalsum, salesorderheader.salespersonid
FROM salesorderheader

JOIN ordertotals ON salesorderheader.salesorderid = ordertotals.salesorderid
WHERE salesorderheader.salespersonid IS NOT NULL
AND salesorderheader.salespersonid <>"" AND orderdate >= '2014-01-01'

GROUP BY salesorderheader.salespersonid
ORDER BY ordertotalsum DESC

LIMIT 5
```

```
 * sqlite:///AdventureWorks.db
Done.
```

Out[48]:

ordertotalsum	salespersonid
1558454.9372999994	289
1433849.0879759998	276
1191783.2753000003	275
1177338.4009999998	282
1171365.4701	277

salespersonid	ordertotalsum	commissionpct
...	...	...

Hint: Remember that the `businessentityid` column from the `salesperson` is compatible with the `salespersonid` column in the query

In [50]:
```
# explore salesperson
```

In [51]:
```
%%sql
SELECT * FROM salesperson LIMIT 1
```

```
 * sqlite:///AdventureWorks.db
Done.
```

Out[51]:

businessentityid	territoryid	salesquota	bonus	commissionpct	salesytd	saleslastyear	rowguid	modifieddate
274			0	0.0	559697.5639	0.0	48754992-9ee0-4c0e-8c94-9451604e3e02	2010-12-28 00:00:00

In [52]:

```
%%sql

WITH ordertotalsum AS
(WITH ordertotals AS

    (SELECT salesorderdetail.salesorderid,
    salesorderheader.totaldue -
    (salesorderdetail.unitpricediscount*salesorderdetail.unitprice*salesorderdetail.orderqty)
     AS ordertotal
    FROM salesorderdetail

    JOIN salesorderheader ON salesorderdetail.salesorderid = salesorderheader.salesorderid

    GROUP BY salesorderdetail.salesorderid
    ORDER BY salesorderdetail.salesorderid ASC)

SELECT SUM(ordertotals.ordertotal) AS ordertotalsum, salesorderheader.salespersonid
FROM salesorderheader

JOIN ordertotals ON salesorderheader.salesorderid = ordertotals.salesorderid
WHERE salesorderheader.salespersonid IS NOT NULL
AND salesorderheader.salespersonid <>"" AND orderdate >= '2014-01-01'

GROUP BY salesorderheader.salespersonid
ORDER BY ordertotalsum DESC)

SELECT ordertotalsum.salespersonid, ordertotalsum.ordertotalsum, salesperson.commissionpct
FROM ordertotalsum

JOIN salesperson ON salesperson.businessentityid = ordertotalsum.salespersonid

GROUP BY ordertotalsum.salespersonid
ORDER BY ordertotalsum.salespersonid ASC

LIMIT 5
```

```
 * sqlite:///AdventureWorks.db
Done.
```

Out[52]:

salespersonid	ordertotalsum	commissionpct
274	201288.51960000003	0.0
275	1191783.2753000003	0.012
276	1433849.0879759998	0.015
277	1171365.4701	0.015
278	490762.81000000006	0.01

In [58]:
```
# explore salesperson
```

In [59]:
```
%%sql
SELECT * FROM salesperson LIMIT 1
```

```
 * sqlite:///AdventureWorks.db
Done.
```

Out[59]:

businessentityid	territoryid	salesquota	bonus	commissionpct	salesytd	saleslastyear	rowguid	modifieddate
274			0	0.0	559697.5639	0.0	48754992-9ee0-4c0e-8c94-9451604e3e02	2010-12-28 00:00:00

In [60]:
```
# join countryregioncurrency.currencycode, countryregioncurrency.countryregioncode, and salesterritory.territoryid
```

In [61]:
```
%%sql

SELECT countryregioncurrency.countryregioncode, countryregioncurrency.currencycode, salesterritory.territoryid
FROM countryregioncurrency

JOIN salesterritory ON salesterritory.countryregioncode = countryregioncurrency.countryregioncode

GROUP BY salesterritory.territoryid
ORDER BY salesterritory.territoryid ASC

LIMIT 7
```

```
 * sqlite:///AdventureWorks.db
Done.
```

Out[61]:

countryregioncode	currencycode	territoryid
US	USD	1
US	USD	2
US	USD	3
US	USD	4
US	USD	5
CA	CAD	6
FR	EUR	7

In [62]:
```
# join currency and codes with salesperson.businessentityid
```

In [63]:

```
%%sql

WITH territorycurrency AS

    (SELECT countryregioncurrency.countryregioncode, countryregioncurrency.currencycode, salesterritory.territoryid
    FROM countryregioncurrency

    JOIN salesterritory ON salesterritory.countryregioncode = countryregioncurrency.countryregioncode

    GROUP BY salesterritory.territoryid
    ORDER BY salesterritory.territoryid ASC)

SELECT salesperson.businessentityid, territorycurrency.currencycode FROM salesperson

JOIN territorycurrency ON territorycurrency.territoryid = salesperson.territoryid
WHERE salesperson.territoryid IS NOT NULL AND salesperson.territoryid <>""

GROUP BY salesperson.businessentityid
ORDER BY salesperson.businessentityid ASC

LIMIT 5
```

```
 * sqlite:///AdventureWorks.db
Done.
```

Out[63]:

businessentityid	currencycode
275	USD
276	USD
277	USD
278	CAD
279	USD

In [65]:

```
%%sql

WITH salesordercommission AS

(WITH ordertotalsum AS
(WITH ordertotals AS

    (SELECT salesorderdetail.salesorderid,
    salesorderheader.totaldue -
    (salesorderdetail.unitpricediscount*salesorderdetail.unitprice*salesorderdetail.orderqty)
     AS ordertotal
    FROM salesorderdetail

    JOIN salesorderheader ON salesorderdetail.salesorderid = salesorderheader.salesorderid

    GROUP BY salesorderdetail.salesorderid
    ORDER BY salesorderdetail.salesorderid ASC)

SELECT SUM(ordertotals.ordertotal) AS ordertotalsum, salesorderheader.salespersonid
FROM salesorderheader

JOIN ordertotals ON salesorderheader.salesorderid = ordertotals.salesorderid
WHERE salesorderheader.salespersonid IS NOT NULL
AND salesorderheader.salespersonid <>"" AND orderdate >= '2014-01-01'

GROUP BY salesorderheader.salespersonid
ORDER BY ordertotalsum DESC)

SELECT ordertotalsum.salespersonid, ordertotalsum.ordertotalsum, salesperson.commissionpct
FROM ordertotalsum

JOIN salesperson ON salesperson.businessentityid = ordertotalsum.salespersonid

GROUP BY ordertotalsum.salespersonid
ORDER BY ordertotalsum.salespersonid ASC),

salespersoncurrency AS
(WITH territorycurrency AS

    (SELECT countryregioncurrency.countryregioncode, countryregioncurrency.currencycode, salesterritory.territoryid
    FROM countryregioncurrency

    JOIN salesterritory ON salesterritory.countryregioncode = countryregioncurrency.countryregioncode

    ORDER BY salesterritory.territoryid ASC)

SELECT salesperson.businessentityid, territorycurrency.currencycode FROM salesperson

JOIN territorycurrency ON territorycurrency.territoryid = salesperson.territoryid
```

```
ORDER BY salesperson.businessentityid ASC)

SELECT salespersoncurrency.currencycode, salesordercommission.salespersonid, salesordercommission.ordertotalsum, salesor
FROM salesordercommission

JOIN salespersoncurrency ON salesordercommission.salespersonid = salespersoncurrency.businessentityid

ORDER BY salespersoncurrency.currencycode ASC, salesordercommission.ordertotalsum DESC

LIMIT 5
```

```
 * sqlite:///AdventureWorks.db
Done.
```

Out[65]:

currencycode	salespersonid	ordertotalsum	commissionpct
AUD	286	659541.7523760003	0.018
CAD	282	1177338.4009999998	0.015
CAD	278	490762.81000000006	0.01
DEM	288	654853.9932999999	0.018
EUR	290	976707.953976	0.016

In [80]:

```
# Ex. 9
assert "salesperson_ranking_currency" in globals(), "Ex. 9 - Remember that your variable's name should be `salesperson_r
salesperson_ranking_currency_result = pd.read_sql(salesperson_ranking_currency, con=sqlite_engine)
assert len(salesperson_ranking_currency_result) == 16, "Ex. 9 - There are too many or too few rows in your result. Remem
assert set(salesperson_ranking_currency_result.columns) == {'commissionpct', 'currencycode', 'ordertotalsum', 'salespers
print("Exercise 9 looks fine for now. You will get your final grade after we've reviewed your submission.")
```

```
Exercise 9 looks fine for now. You will get your final grade after we've reviewed your submission.
```

Attribution

"AdventureWorks database", Nov 7, 2017, Microsoft Corporation, MIT License, https://github.com/microsoft/sql-server-samples/tree/master/samples/databases/adventure-works