

Python för Hantering av SQL Queries

Matthew H. Motallebipour

January 28, 2024

1 Introduktion

1.1 Relationsdatabas

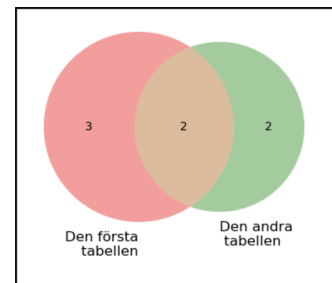
En relationsdatabas är en samling av tabeller som är relaterade till varandra med hjälp av ”nyckelegenskaper” som särskiljer raderna i varje tabell. Tanken med att ha relationsdatabaser är att undvika överflödiga rader som bara har en viss egenskap gemensamt.

1.2 CRUD

CRUD står för Create, Read, Update, och Delete, dvs skapa, läs, uppdatera och ta bort, vilka är de fundamentala funktionerna som kommer till användning just för att skapa och hantera en databas. Ett CRUD-flöde består därför av en blandning av dessa funktioner som kommer i en sekvens och möjliggör hanteringen.

1.3 LEFT JOIN och INNER JOIN

Man får se två tabeller i en databas som mängder, där join innebär att dessa kombineras för att få en ny tabell. LEFT JOIN innebär att alla rader från den första databasen tas med och motsvarande rader i den andra tabellen tillför information på samma rader som den första. Där den andra tabellen inte har någon ytterligare information kommer det att skrivas NULL. INNER JOIN, å andra sidan, fyller den nya tabellen med de rader där både den första och den andra tabellen har information att tillföra. För de rader där den ena eller den andra inte innehåller någon information kommer ingen ny rad att skapas i den nya tabellen heller.



1.4 Indexering

Indexering är att skapa en metod att söka och hitta en viss rad i en tabell utan att nödvändigtvis behöva iterera igenom varenda rad i den tabellen.

1.5 Vy

En vy är samma sak som en tabell och kan användas och hanteras på samma sätt. Det finns dock en väsentlig skillnaden; att en vy skapas i ramminnet och är tillgängligt att använda så länge databasen är aktiv och servern/ datorn är påslagen.

1.6 Lagrad procedur

Lagrad procedur, som namnet föreslår, är en samling av SQL-kommandon som redan har kompilerats och är redo för körning. Detta för att spara tid och minne, undvika redundans i koden och att ge möjligheten att tillhandahålla information till ”extern” användare utan att denne ska ha direkt tillgång till databasen och källkoden.

2 I praktiken

2.1 Bakgrund

Databasen AdventureWorks2022 är en fiktiv databas för att uppvisa Microsoft's möjligheter att hantera databaser i allmänhet. Den består av 5 scheman personal, kunder, tillverkning, försäljning, och återförsäljare, vilka i sig innehåller flera tabeller var. Originalen innehåller också redan flera vyer, vilket gör hopslagning (join) av tabellerna ännu enklare.

Det finns data om 296 anställda varav 290 jobbar kvar, 13 jobbkandidater, 181 försäljningsregioner, 1764 produkter, 6 cykelmodeller, 104 försäljningsställe med 156 kontaktpersoner, 19972 kunder och 17 säljare.

2.1.1 Installation av programvara och import av bibliotek och moduler

Förutom installation av `sqlalchemy` behövde vi importera `matplotlib.pyplot`, `venn2` från `matplotlib.venn`, `pandas`, `LabelEncoder` from `sklearn.preprocessing`, `datetime` från `datetime`, `seaborn`, `numpy`, `stats` från `scipy`, samt `sm` från `statsmodels.api`.

2.1.2 Databas, uppkoppling och åtkomst

Detta har redan diskuterats och behöver inte redovisas. Dock är det bra att påpeka att till vår *engine* har vi använt parametervärdena 'mssql', 'DESKTOP-PJ0B80O', 'AdventureWorks2022', och `integrated_security=True`.

2.1.3 Förberedelse av scheman och tabeller

Detta är också redan diskuterat och behöver inte tas upp här.

2.1.4 Databas förfrågan (Queries)

Vi tar först en titt på tabellerna i databasen ([SQL Python Exam Zomnipotential.ipynb](#)). Resultatet blir en beskrivning av alla tabeller i databasen, vad vi har använt för *Query*, de första fem raderna i tabellen och lite statistik över just den tabellen (In [10]).

Vi noterar att schemas *Person* och *Sales* innehåller mycket information som är relaterade till företagets kunder. Detta väljer vi som ämne för vår analys och framställer samma tabeller ännu en gång för att lättare undersöka sambanden mellan tabeller som innehåller all information om kunderna. Koden skiljer sig inte annat än att vi bara väljer de tabeller som tillhör schemas *Person* och *Sales* (In [11]). Sedan att vi får en hel del tabeller som egentligen inte kommer till någon användning är något vi kan försumma när det kommer till automatisk utvinning of information, trots att det minskar överskådligheten. Detta eftersom vi ritar grafer för alla kolumner i alla tabeller, istället för att välja ut "rätt" kolumn, vilket skulle kräva mer analytisk kodskrivning genom t ex regex-uttryck.

Det var först här som vi noterade att det finns många egenskaper hos kunderna som skulle kunna undersökas närmare för att se ett mönster. Vi tittar också på views som redan finns presenterade i den originella databasen och speciellt där kunderna är uppradade. Den största skillnaden mot tidigare koder är att vi istället använder `inspector.get_view_names` (In [12]).

Vi bestämmer oss således för att jämföra köpvanan hos kvinnor och män och undersöka om vi kan hitta olika mönster i hur män och kvinnor investerar i sin hälsa/ ekonomi, alternativt för en miljövänligare transportmedel. Från de senaste utskrifterna kan vi konstatera att för en rättvis och noggrann jämförelse mellan könen behöver vi tabeller med många kunder i. Därför väljer vi `vIndividualCustomer` och `vPersonDemographics` från schema *Person*. Vi gör en *JOIN* mellan de två tabellerna och får (In [15]).

Vi gör en klassificering av data i kolumnen för årsinkomst och får

```
YearlyIncome
25001-50000      5713
50001-75000      5483
0-25000          2922
75001-100000     2762
greater than 100000 1628
Name: count, dtype: int64
```

För att kunna använda denna kolumn i våra statistiska beräkningar ersätter vi varje intervall med dess max-gräns, dvs:

0-25000 with **25,000**
25001-50000 with **50,000**
50001-75000 with **75,000**
75001-100000 with **100,000**
greater than 100000 with **500,000**

för att göra talen i kolumnerna konvertibla till heltal inför nästa steg som är att skapa en korrelationsmatris. Vi kommer inte att använda själva talen i våra beräkningar så det finns ingen risk att de genererar felaktiga resultat. Med andra ord använder vi årsinkomsten endast som en ordinal, kategorisk variabel (In [17]).

2.1.5 Övergång från kategorisk data till ordinal data

Bortsett från födelsedatum för kunderna är varken omvandlingen eller kategoriseringen lika komplicerade att utföra för de andra kolumnerna. Här använder vi *LabelEncoder* (In [19]) till att beräkna företagskunders respektive ålder. För de andra kolumnerna (In [23] - [28]) är det som sagt inte lika dramatiskt utan alla följer samma mönster.

3 Analys och slutsatser

3.1 Ett första försök: hitta samband mellan Total Purchase YTD och andra faktorer

Innan vi går vidare med att jämföra könen, gör vi en analys för att se om det eventuellt finns något samband mellan det belopp man spenderat under det nuvarande året och något av de relevanta egenskaperna hos företagets kunder. Vi kan använda de nya kolumnerna till att beräkna korrelationen mellan alla dessa variabler, med hopp om att hitta variabler som är mer korrelerade med andra, speciellt de som är kopplade till kundernas kön.

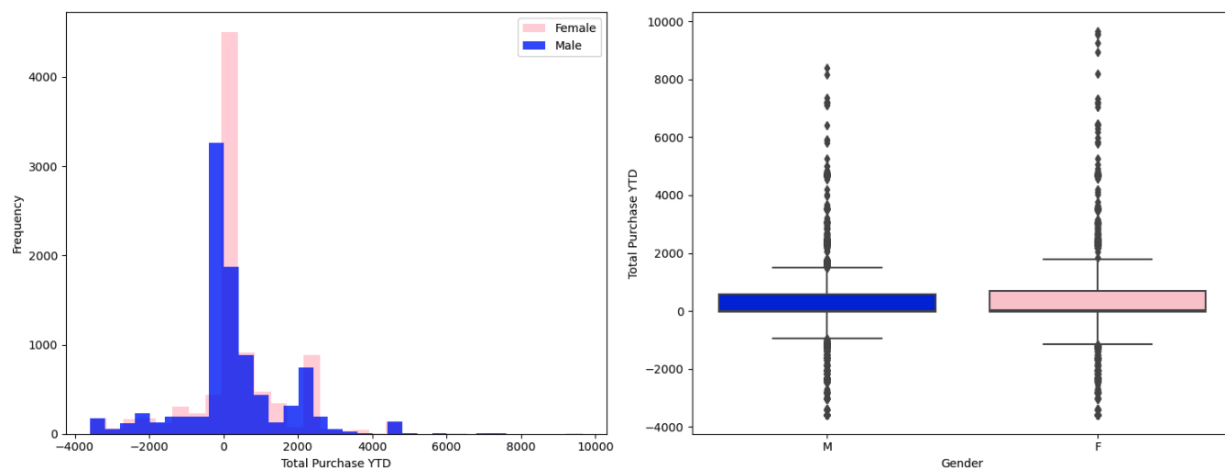
Från Heatmap:en (In [30]) kan vi endast utläsa vilka par av variabler som är mer korrelerade än andra, nämligen: Agex-Loyaltyx (0.9; detta är framför allt pga vår egen definition of Loyalty), Agex-TotalChildren (0.5), NumberChildren-TotalChildren (0.5), YearlyIncome-NumberCarsOwned (0.5), TotalChildren-Loyaltyx (-0.5), Inga av dessa är relevanta för vår studie och därför överger vi den undersökningen och söker lycka någon annanstans.

Den enda finansiellt relevanta slutsats vi kan dra är att lojalitet är "mot-korrelerad" mot antal barn som i sin tur är korrelerad med ålder. Detta betyder att de äldre tenderar att visa mindre lojalitet, vilket kan förstås i och med att de äldre har fler barn och både har mindre tid att handla och mindre pengar att spendera på egna intressen. I nästa steg tittar vi på inkomster och försöker hitta ett samband mellan könen och hur mycket de tjänar och spenderar.

3.2 Första 0-hypotensen: det finns skillnad i inkomst och köpvana hos de "mest entusiastiska" männen och kvinnorna

In [31] visar kod som tar reda på exakt hur många olika inkomster som finns. Eftersom dessa uppgår till 3994 stycken olika värden väljer vi att dela dem i inkomstnivåer; i 30 nivåklasser (bins=30); som det framgår från In [32]. I histogrammen på nästa sida ser vi en viss skiftning i kvinnornas totala köp i det föreliggande året jämfört med männens. Även i vår Boxplot ser vi denna skiftning, med extremvärden som ligger längre bort från boxen jämfört med schemat för männen. Därför kan det finnas en chans att vi åtminstone kan hitta en signifikant skillnad. Men vår beräkning (In [33]) visar att det inte finns någon signifikant skillnad i generella drag mellan hur mycket män och kvinnor spenderar på köp av cykel och tillbehör.

T-statistic: 0.7885667920652966
P-value: 0.43037536312561386
Confidence Interval: (-23.324626429997387, 54.72475825253415)



3.3 Andra 0-hypotensen: det finns en signifikant skillnad mellan könen i den högre ”spenderarklassen”

Om vi tittar på spridningsdiagrammet för hela populationen, märker vi att de högsta beloppen spenderas av de som är mellan 40 och 55 år gamla. Så vi separerar den gruppen som spenderar mer än 5000 dollar per

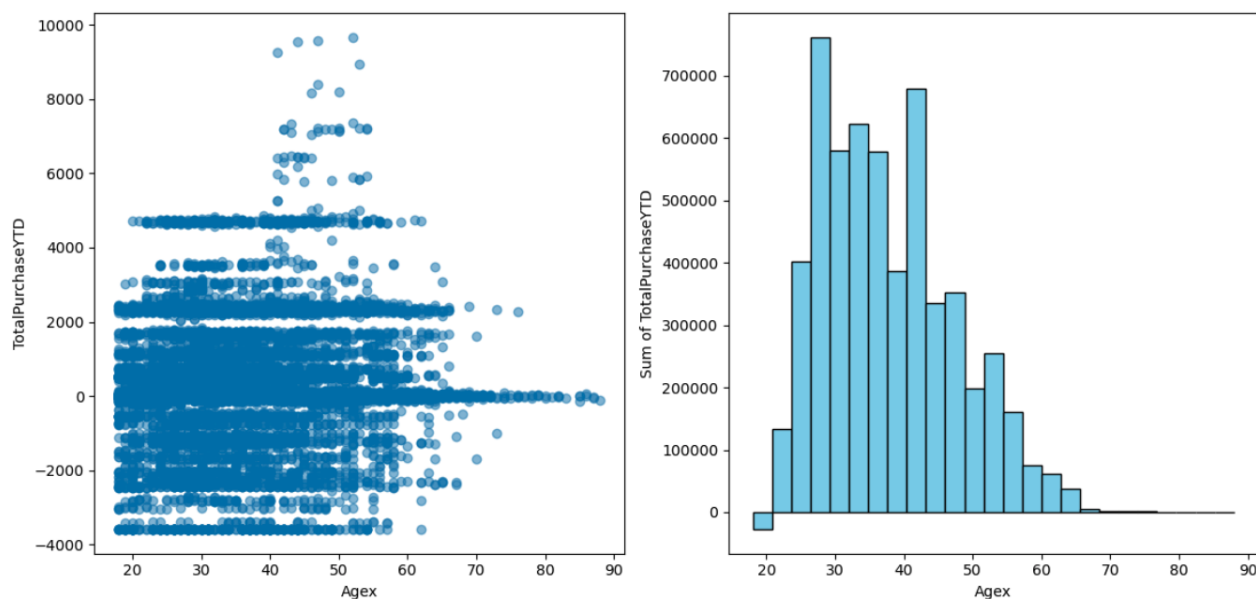


Figure 1: TotalPurchaseYTD age

år och tittar på skillnaden mellan män och kvinnor inom den gruppen av kunder.

T-statistic: 0.6506527831141421
P-value: 0.5185818626557512
Fail to reject the null hypothesis: There is no significant difference between the groups.

Vi finner återigen ingen signifikant skillnad mellan dessa värden.

3.4 Tredje 0-hypotensen: det finns en signifikant skillnad i köpvana mellan kvinnor och män i de olika inkomstklasserna

Efter att inte ha lyckats med att hitta någon skillnad mellan de, från båda könen, som spenderar mest på köp av cyklar och tillbehör gör vi ett försök med att se om det förekommer någon signifikant skillnad i hur

mycket män och kvinnor i varje inkomstklass

	YearlyIncome	DiffMeans	ConfidenceInterval
0	0-25000	36.945527	(-35.62309491373986, 109.5141486056196)
1	25001-50000	-3.755627	(-68.14993872067124, 60.63868448816346)
2	50001-75000	-1.190903	(-74.36801905155842, 71.98621260098254)
3	75001-100000	-0.272717	(-115.50074949730732, 114.9553150625617)
4	greater than 100000	138.540454	(-20.90936173573448, 297.99027048409255)

Figure 2: Income Classes.

spenderar på sin hobby iår.

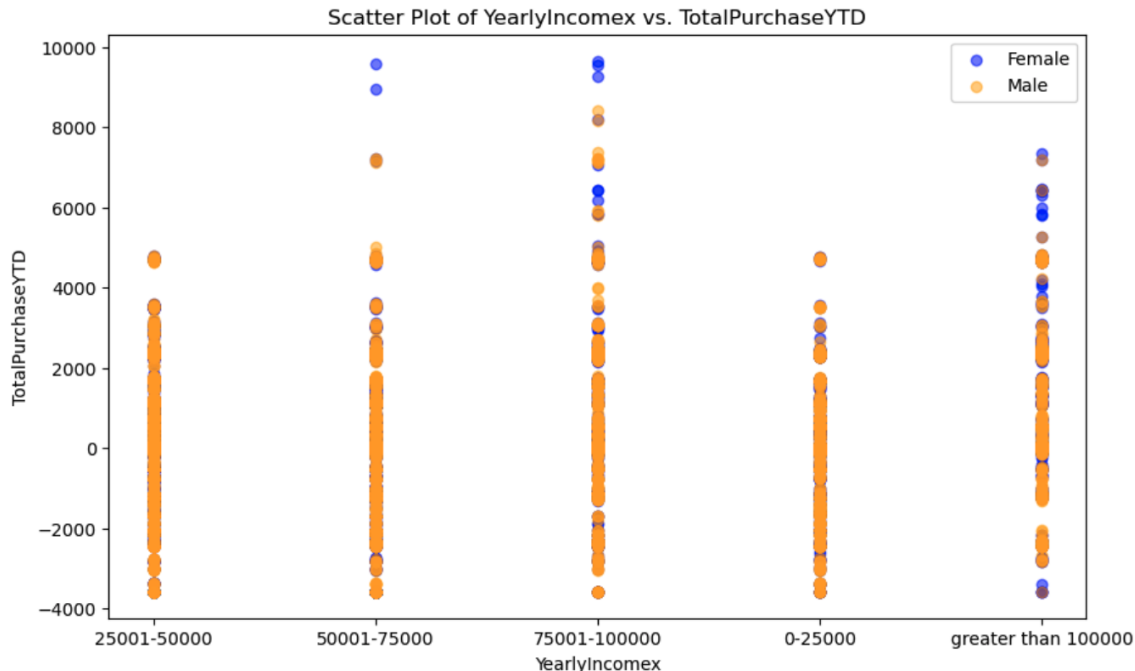


Figure 3: Shopping habit for Income Classes.

Även här hittar vi alltså ingen signifikant skillnad mellan män och kvinnors köpvana, när det kommer till cykel och tillbehör. Med andra ord innefattar konfidensintervallerna för samtliga inkomstklasser nollpunkten med mycket goda marginaler.

4 Conclusion

Genom att titta på tre olika hypoteser har vi konstaterat att köpvanan inte skiljer sig signifikant mellan män och kvinnor vare sig i de högre inkomstklasserna eller någon annan uppdelning av kunderna i könsgupper. Om man är mån att hitta skillnader får man pröva fler hypoteser och eventuellt använda sig av andra metoder, som intervjuer.

5 Framtida undersökningar

Det finns fortfarande saker att undersöka, exempelvis genom att dela beloppet man spenderat under året med ens årliga inkomst kan man också hitta ny information som kan vara en skiljepunkt mellan män och kvinnor, även om informationen vi hittills bearbetat fram inte tyder på det.

Generellt sett, skulle man också kunna använda multipel regression för att eventuellt kunna hitta samband mellan flera variabler samtidigt. Detta ingår dock varken i statistik- eller den föreliggande kursen och har därför inte beaktats.