31/3-2025

Fra API burde tabellerne kunne sammenflettes via orders tabellen med customers tabellen via customer\_id variablen og med order\_items tabellen via order\_id variablen.

De fleste af cellerne i phone variablen i customers tabellen er tomme, så den kan måske slettes. Man kan prøver at sammenligne phone fra customers med phone fra staff og stores tabellerne for at se, hvor mange af indkøbene er af medarbejderer. Ligeledes kan E-mail også sammenlignes fra førnævnte tabeller af samme grund.

Staff og stores tabellerne kan sammenflettes via store\_name variablen fra staff tabellen med name variablen fra stores. I det tilfælde skal name, phone og email variabelerne fra stores tabellen omdøbes, samtidigt kan street variablen fra stores ses bort fra, da den er er ligeså identisk i begge tabeller og kan ”mappes” på samme måde med butiksnavnet.

Manager\_id variablen fra staffs tabellen kan droppes, da der er ingen andre af tabellerne, som har en lignende variabel, samtidigt med at der er en tom værdi i den. Den ligner også mere en index variabel end en mere informativ variabel, så, hvis der er behov for det, så kan den genlaves efterfølgende.

Products tabellen kan være hoved tabel for sammenfletning med categories tabellen via category\_id variablen i begge tabeller og med brands tabellen via brand\_di variablen i begge tabeller. Stocks tabellen kan også sammenflettes med Products tabellen via product\_id i begge tabeller. Dog har stocks product\_id 321 produkt ID’er, imens stocks kun har 313, så der vil være 8 produkter, som ikke er i stock tabellen. Efter sammenfletning af tabellerne kan vi tage højde for om de 8 produkter er værd at gemme eller hvad der skal gøres.

Det Burde kunne lade sig gøre med den python klasse, som jeg udviklede i uge 4, men revidering kan være nødvendigt efter omfanget. De fleste af funktionerne for dette er allerede tilstede i min MySQL – klasse, men jeg er nok nød til at tilføje en, som eksportere databasen som en sql-fil for at sikre adgang til denne.

2/4 -2025

Det har lige lykkedes mig at lave en backup\_db funktion, som gemmer en database fra en MySQL server som en .db-fil ved at kommunikere midlertidigt med en Sqlite server. Det er ikke den mest elegante metode, da der kort oprettes forbindelse med en anden server for bare at kunne eksportere en database. Men det lader umiddelbart til at virke.

Jeg har vist også brug for funktioner til at håndtere redigeringer af kolonne datatyper, da datasættet indeholder tidsvariabler, som kan ændre sig. Men mest python-metode at gøre det på vil være igennem pandas og datatime pakkerne, men siden data ligger på en MySQL server, tror jeg ikke at denne fremgangsmåde kan gøres på samme måde.

Det lader til at være ligetil for alle de andre datatyper, men der er et problem, som jeg skal håndtere anderledes ift. Dato. Fra orders tabellen har vi order\_date, required\_date og shipped\_date, som alle sammen er af format ”dd-mm-YYYY”, imens MySQL DATE() datatypen lader til at være i ”YYYY-mm-dd”. Jeg ved endnu ikke om det er lige til eller om jeg skal tænke på at lave en form for omstrukturering af dato-variablerne. Hvis Jeg skal omstrukturerem, vil jeg sandsynligvist ende med at lave en dag, måned og års variabel for både order, required og shipped datoerne. Det vil gøre tabellen og derved databasen markant større, så jeg skal slette disse dag, måned og års variabeller efter de er sammensluttet i den acceptable rækkefølge.

I samme sammenhæng skal jeg nok også lave ekstra funktioner ift. Datatyper. Det skal være at vise datatyper og ændre datatyper. Det vil være lettest at lave en funktion, som viser datatype, men jeg skal være forsigtigt ift. Ændre datatyper. For det første skal jeg tvinge brugeren til at definere enten én specifik kolonne ad gangen eller flere specifikke kolonner ad gangen for ikke at ændre i hele tabellen på én gang og risikere at gøre tabellen ulæselig.

En ny funktion ”describe” er udarbejdet til at vise datatype og for at danne overblik over dataindhold. Den anvender DESC for at gøre det enkelt. Change\_coltype funktion er også lavet for at ændre datatype og den tager særligt forbehold til dato og datoformater.

3/4-2025

Change\_coltype vil håndtere både andre datatyper og date. Det er ikke altid helt let at sige hvilket datoformat, som bruges i en datovariabel, så det skal gøres fleksibelt. Det vil så give mest mening at anvende regex til det. Datovariabler kan nogle gange have tidszoner, så det skal der også tages højde for. Oprindeligt havde jeg tænkt mig at lave en opdeling på + i python via split, men det vil ikke virke i en kolonne variabel i MySQL. Heldigvist fandt jeg en metode via STR\_TO\_DATE og SUBSTRING\_INDEX i MySQL, som respektivt registrere en tekststreng som dato alt efter format og gør det samme som python’s split funktion.

4/4 – 2025

Jeg overvejede at udarbejde en parser til at arbejde med MySQL klassen på, men jeg har brugt for meget til nu på at udarbejde klassen og for lidt på at arbejde på dataet.

Det sidste, jeg vil gøre for klassen (for nu) er at ændre import\_table funktionen, så den uploader tabellen i samme navn, som den uploadede fil/tabel. Det vil gøre det mere overskueligt for brugeren.

Jeg har indset at min klasse er blevet så kompleks at jeg er nød til at skrive et lille cheat sheet for at se, hvilke funktioner, som jeg har lavet og hvad de gør. -\_-

Det ser ud til at MySQL ikke kan håndtere tidlige instancer af NULL i en kolonne, så jeg tilføjede linjen:

A screen shot of a computer

AI-generated content may be incorrect.

Til try-blokken i import\_table funktionen. Og det vil nok være det sidste, som jeg gør ved MySQL-klassen for nu.

For pricerne i tabellerne, overvejede jeg at gemme dem som FLOAT eller DOUBLE altefter om der skulle prioriteres hukommelse eller præcision. Men efter at have læse lidt mere op på det, giver det mest mening at gemme det som DECIMAL(5,2), da det er mere præcist og at vi ikke kan forvente at nogle af priserne er mere end $ 99.999,99 værd. Der er ganske vist nogle høje priser, men pt er den højeste registreret pris $ 11999.99, så det burde være bedst for både præcision og lagring.

Datatype ved importering er generelt varchar(255), da der kan være meget variation i data, men generelt er de ikke helt så lange, som de kunne have været. Hvis der kunne arbejdes med længere celler, ville det give mening at ændre import\_table funktionen til at importere alt i LONGTEXT format, indtil de konverteres, men det vil så også ende med at optage meget plads, som så skal konverteres for at mindske plads. Ellers havde jeg også ændre import\_table funktionen til at importere alle tabeller som LONGTEXT format.

#####################################################################

Jeg kan se at jeg nok har misforstået lidt. Vi skal kunne arbejde med data fra både csv, database OG api. Min klasse kan oprette forbindelse med en database og den kan importere csv-filer som tabeller. Men jeg har ikke skænket api-importering en tanke i hele forløbet. Så jeg er ikke færdig med at arbejde med klassen overhovedet. Det skal jeg have set ind i.

Derudover har jeg opdaget at det også vil give mening for mig at oprette en funktion, som laver hurtig statistik over tabellerne. Det vil give et bedre overblik over indhold i større tabeller, samt at hvis jeg implementere muligheden for at se mængden af tomme celler, kan det hjælpe ift. At se om alt er gjort ordentligt.

Derved kan det også give mening at prøve at se ind i mulgheden for at lave full-join med tabeller igen for at sikre at man ikke mister data igen.

Men for nu vil prioriteringen være:

1. Forbindelse og dataimport fra api.
2. Statistik tabel
3. Full-join

Hvis jeg får tid til det, ser jeg på det i weekenden.

Jeg må videre undersøge de scripts, som de oploadede til os i starten af ugen for at se, hvordan import skal kunne gøres. Heldigt at jeg havde downloaded dem i starten af ugen og lagt dem op på GitHub. Jeg vil nok ikke kunne tilgå databasen hjemmefra, men jeg burde kunne få forbindelse til api’en.

7/4-2025

Jeg har valgt at arbejde på API data fra csv filerne, da jeg ikke kan få konverteret dem rigtig til json format, så jeg kan anvende dem. Men jeg har i det mindste udarbejdet en funktion, som henter data fra API’en under forudsætning af at URL’en henviser til en tabel. Jeg har af gode grunde ikke kunne teste den, så jeg kan ikke sige om den virker.

Jeg må også prøve at se endnu en gang om jeg kan eksportere databasen som sql fil. Det lykkedes mig og nu føler jeg mig dum. En SQL fil er bare en fil med alle kommandoerne til at opbygge tabellerne i en database, så hvis bare SQL-kommandoerne til at oprette tabellerne er i en fil med sql format, vil den kunne anvendes som en SQL-database.

Jeg har også nu oprettet en funktion til at vise en statistik tabel over en tabel. Den differentiere udelukkende efter tal-variabler og ikke tal variabler. Så data skal konverteres inden, medmindre man kun vil se antal unikke og tomme celler.

Jeg har også været nødsaget til at lave en funktion til at flytte tabeller og til at skabe relationer imellem tabeller.

9/4-2025

Nu har jeg funktioner til at importere data fra API, SQL, CSV og databaser og jeg kan eksportere database som sql fil med relationer. Dog ikke med meget andet. Det Må være nok her, da jeg ellers skal have nok tid til at bearbejde data.

I staffs tabellen er den første celle i manager\_id er NULL, hvilket giver problemer ved tabel import. Jeg har sat den celle til ... og erstattet den med en tom celle efterfølgende på serveren. Jeg vil nok erstatte det med ”NA” streng senere, så det ikke giver problemer ift. Databehandling.

Orders tabellen indeholder datoer med tid. Og det er unikke tider, så tidspunkt på dagen kan ikke ignoreres så let. Jeg har derfor skrevet change\_coltype om så den opfatter tidsvariabler ved både DATE og TIME- variabler.

Endeligt har jeg inkluderet en freestyle funktion i klassen, som lader brugeren eksekvere egne kommandoer, som ikke er dækket af alle de andre funktioner. Det burde gøre det lettere at håndtere tidszonerne. Jeg vil lave en kolonne, som hedder timezone, hvor det tages med i betragning (da det ellers vil blive slettet ved date konvertering).

Jeg så på det forkerte order.csv dokument...... den version, som vi har fået til denne opgave, har ikke tidszoner, så jeg har brugt en halv time til at bekymre mig over ingenting. Men nu har jeg en freestyle funktion, som jeg bruge kunne bruge.

Jeg har diskuteret lidt ift. Telefon numre i customers tabellen, men hvis der er videre behov for at analysere dem som tal, kan de konverteres på det tidspunkt. Men der er mange muligheder for at udføre tekst analyse (såsom med CPR-numre) på det, hvis det forbliver en streng.