Datamatiker 4.Semester

BI Preject

Daniel, Tobias og Simon.

Introduktion

På en større uddannelsesinstitution har der tidligere været drevet en form for social klub for både ansatte og studerende, hvor der blev solgt forskellige varer. Salget har været stoppet i en længere periode, men nu er der et nyt team, som vil drive klubben videre, men de mangler indsigt i forretningen, særligt i forhold til salget.

Vi skal med baggrund i historiske data hjælpe det nye team i gang. Vi får derfor udleveret alle de historiske data forretningen har og med baggrund i dette materiale, skal vi skabe et overblik til det nye team med et særligt fokus på salg. Det nye team kan desværre ikke hjælpe os med forretnings- eller data forståelsen, da de netop er tiltrådt, så det er helt op til os at fortolke det udleverede materiale.

Men det nye team ved dog at klubben har solgt forskellige produkter som mælk, øl og billetter til arrangementer. Klubbens medlemmer forudbetaler et beløb, som herefter er på medlemmets konto. Når et medlem herefter køber et produkt indtastes medlemmets ID og produktets ID i klubbens it-system, hvorefter medlemmets konto opdateres.

I denne rapport vil vi udføre en dataanalyse på den udleverede Access fil, samt category.xls filen. Access filen indeholder nedenstående tabeller:

- *Member (tabel med klubbens medlemmer)*
- *Payment (tabel med oplysninger om forudbetalinger)*
- *Product (tabel med produkter der sælges)*
- *Room (tabel med oplysninger om hvor produkter sælges)*
- *Sale (tabel med oplysninger om salg)*

Den anden fil, category.xls, indeholder følgende tabeller:

- *Ark1 (tabel med produkter som klubben sælger eller har solgt)*
- *Ark2 (tabel med produktkategorier)*
- *Ark3 (en tom tabel)*

Problemstilling

Hvordan får vi gjort den eksisterende data overskuelig med fokus på at identificere de eksisterende forretningsprocesser, samt at kunne forklare hvordan sammenhængen er i det udleverede datasæt?

Problemformulering

- Hvordan får vi hjulpet det nye team med at få et godt overblik af dataen med et særligt fokus på salg?
- Hvordan får vi fremstillet dataen i en multidimensionelt cube således dataen nemt og overskueligt kan analyseres af analytikere?
- Hvilke data skal vi analysere for at give det nye team et godt overblik?
- Hvordan kan vi samle alt relevant data i én let tilgængeligt lokation?
- Hvordan kan vi transformere dataen således den er uniform og giver mening?
- Hvornår opdateres de enkelte tabeller, rækker og hvorfor?
- Hvordan er sammenhængen i det udleverede datasæt. Identificer fejl og mangler.

Analyse og beskrivelse af datasæt

Til vores analyse har vi fået udleveret to filer; catergory.xls og fclub_safe.mdb. Herunder følger en analyse af indholdet i de to filer.

category.xls

Fungerer som en særskilt database for de forskellige produkters kategorier.

I filen har vi 2 ark hvor ark 1 er en liste over produkter samt et category id som fungerer som en foreign key til ark 2's primary key; id. Dette afgør hvilken kategori som produktet indgår i.

Access fil:

Databasen er sidst blevet brugt 2008, 7. Januar, kl. 12:19:39. Dette har vi konkluderet fordi den sidste transaktion har en timestamp i Sale tabellen.
Member tabellen indeholder både lærer og elever, da der er nogle medlemmer der har været aktive siden 1977, disse er højst sandsynlig ikke er elever.

Member tabel:

En tabel over alle nuværende og tidligere medlemmer af klubben, det år hvor de blev tilmeldt, hvorvidt de er aktive og deres balance.

id:
Surrogate key

active:
Vi antager at active attributen afgør om medlemmet stadig er aktiv. Langt størstedelen af medlemmerne er inaktive, hvor der er 1681 rækker, hvorimod det antal der er aktive er 665.
Dette kan ses vha. test:
*SELECT * FROM Member WHERE active = 0*
*SELECT * FROM Member WHERE active = 1*

Alle der er meldt ind i 2007 er aktive.
Dette kan ses vha. test:
*SELECT * FROM Member WHERE [year] = 2007*

Hvad der gør at en bruger ikke længere bliver opfattet som aktiv kan vi ikke finde frem til.

year:
Vi antager at year attributten i members tabellen indeholder data vedrørende medlemmets indmeldelsesdato i klubben.
Der er 49 rækker hvor året er 0, hvilket vi antager er en fejl.
Vi lader dette være så vi ved at alle medlemmer hvor vi har en inkorrekt dato, vil

have årstallet år 0.
Dette kan ses vha. test:
*SELECT * FROM Member WHERE [year] = 0*

balance:
Balance attributten ser ud til at beskrive det enkelte medlems rådighedsbeløb inden for klubben. Alle medlemmer der er tilmeldt i år 2007 har en positiv balance i deres rådighedsbeløb.
Dette kan ses vha. test:
*SELECT * FROM Member WHERE [year] = 2007*

De sidste to tal i balancen antager vi er øre, hvilket vil sige at 12550 vil blive oversat til beløbet 125,50 kr.
Nogle steder står der at balancen er negativ, vi antager så at det betyder at man skylder penge.

Penge bliver sat ind på balance attributen ved at vedkommende laver en "Payment". Altså, går vi ud fra at kunden laver en pengeoverførsel til klubben, hvorefter en værdi bliver sat ind i payment tabellen som gemmer en historik for overførsler til balancen.
Dette kan konkluderes ud fra denne test:
*SELECT * FROM Member where id = 2321*

Ovenstående query viser at balancen er 1900.

SELECT SUM(amount) FROM Payment WHERE member_id = 2321

Ovenstående query viser at alle indsættelser på kontoen giver 1350.00 kr.

SELECT SUM(price) FROM Sale WHERE member_id = 2321

Ovenstående query viser at alt hvad kunden har købt giver 1331.00 kr.

Når vi laver regnestykket *1350,00 kr - 1331,00 kr* giver det *19,00 kr* som er balancen hos kunden. Dermed har vi et bevis for at payments er indsættelse på kontoen og sale er salget af produkter, og at balance er rådighedsbeløbet for medlemmet.

Payment:

Payment bliver brugt, når man tilføjer penge til sin balance. Altså, når man indsætter penge ind på sin klubkonti kommer beløbet ind i payments, og amount attributtens værdi bliver lagt sammen med memberens balance attribut. Så har man styr på hvornår overførslen har fundet sted.

id:
Surrogate Key

member_id:
Foreign key til primary key i members tabellen.

timestamp:
Timestamp for hvornår at overførslen har fundet sted.

amount:
Beløbet som er blevet overført. Vi antager at de to sidste tal er ører og resten er kroner.
Amount vil svare til det samlede beløb som vedkommende sætter ind på sin konto.

Product:

Indeholder alle de produkter man kan købe, samt deres pris, om de er aktive eller ej og hvornår de er blevet deaktiveret.

id:
Surrogate key

name:
Navnet på produktet og eventuelt volumen i tilfælde af drikkevarer.

price:
Price er prisen for et produkt, hvor de sidste to tal er øre og resten er kroner.

active:
Active er om produktet stadig kan købes i butikken. 1 er aktiv, og 0 er ikke aktiv.

deactivate_date:
Den dato hvor et produkt, begivenhed eller tjeneste ikke længere sælges eller udbydes.
Det kan forekomme at en række har en deactivate_date men stadig har aktiv attributen til at være én. Her går vi ud fra at den er deaktiveret.
Der er kun en deactivate_date fra 2007, hvilket vil tyde på at det er noget nyt de er begyndt på.

Room:

Room tabellen indeholder data vedrørende de forskellige sammenhænge hvor der solgt noget i klubben. Det kan for eksempel være under en hyttetur, gennem en webshop eller i et lokale.

id:
Surrogate Key

name:
Navnet på rummet / beskrivelse af rummet

description:
Yderligere beskrivelse af rummet, begivenheden eller webshoppen.

Sale:

En tabel over de køb der er blevet foretaget af medlemmer i klubben, hvilket produkt der er tale om, hvor stort et beløb der er blevet købt for, samt hvilket room købet har fundet sted i.

id:
Surrogate Key

member_id:
Foreign key til et medlem id i members tabellen. Henviser til den member der har foretaget købet.

product_id:
Foreign key til et produkt i product tabellen. Henviser til hvilket produkt der er tale om i salget. Henviser til produktet surrogate key i products tabellen.

room_id:
Dette fortæller hvilket rum eller under hvilken begivenhed salget er blevet foretaget under.
Vi kan se at alle salg før 2007 bliver foretaget i rum 1, hvilket er websalg. De andre rum er nogle der er blevet tilføjet i år 2007. Som vi kan se på denne sql query:
*SELECT * FROM Sale WHERE room_id = 1*

Hvis vi så derimod kigger på alle salg der ikke er blevet foretaget i rum 1, kan vi se at der ikke er nogen salg før 2007. Dette kan vi se med denne sql query:
*SELECT * FROM Sale WHERE room_id != 1*

timestamp:
Timestamp er det tidspunkt hvor et salg er blevet foretaget.

price:
Det beløb som medlemmet har betalt under salget, hvor de sidste to tal er ører. Tallet bliver trukket fra det aktuelle members balance attribut i members tabellen.

Semester Groups:

Semester groups tabellen viser hvilken periode, årstal og semester et member hører til.

member_id:
Foreign key til member tabellen. Afgør hvilket medlem vi har med at gøre.

periode:

Hvis vi kigger på et par eksempler på nogle perioder, f.eks. F99 og E99, antager vi at F er det første halvår, imens E er det sidste halvår. Da vi tænker at E står for Efterår og F står for Forår. Tallet er så årstallet, 99 er f.eks. 1999.

Perioden går op til 02 (år 2002) hvor vi antager at de har skiftet system eller begyndt at undlade at notere semester gruppe for medlemmerne. Dette kan vi se med denne sql query:
*SELECT * FROM SemesterGroups*

semester:

Hvis vi nu f.eks. tager et eksempel på en semester værdi, f.eks. Dat5, så antager vi at Dat står for Datamatiker og at 5 står for hvilket semester de går på. Dette kan man se ved at kører følgende sql query:
*SELECT * FROM SemesterGroups*

En member har typisk flere semestre under sig, så for eksempel: bruger 1061 har 5 entries i semester group tabellen. Som kan se på denne sql query:
*SELECT * FROM SemesterGroups WHERE member_id = 1061*

Dette får også række antallet 2242 i semester groups til at give mening. Da der er 2356 rækker i medlemmer, og Semester groups tabellen blev inaktiv i år 2002. Her kan man se antallet af rækker i SemesterGroup tabellen:
SELECT COUNT(member_id) FROM SemesterGroups

Hvordan forventes det udleverede data sæt at være brugt i den kommende løsning

Vi forventer at samle de eksisterende datasæt i et data warehouse vha. en ETL proces som bliver skabt vha. integration services (SSIS). ETL processen vil blive kreeret i Visual Studio i et integration service projekt.
Vi forventer at transformere vores data således at vi skaber en fact table ud fra 'Sale' tabellen i datasættet, da det er dette aspekt af datasættet vi er interesseret i at analysere. Mange af attributterne fra de andre tables ender med blive flyttet over på denne nye 'Sale' table.
Vi forventer at vi kommer til at omforme tabellerne således vi får et denormaliseret datasæt i et star pattern.
Det lader ikke til at relationer bliver overført fra .mdb filen, så det lader til at vi også skal definere de forskellige relationer efter den tolkning vi har lavet af datasættet.
Vi kombinerer også .mdb filen og .xls filen.

Hvilke forretningsprocesser kan identificeres

Der er to former for forretningsprocesser, den ene er payment, som foregår i payment tabellen. Payment tabellen bliver brugt når et medlem vil tilføje kredit til deres konto, så de kan betale for deres køb i Sale tabellen. Sale er den næste forretningsprocess, hvor Sale tabellen bliver brugt, til at beskrive transaktioner af produkter der bliver købt af et medlemmerne.