NTNU

Institutt for datateknologi og informatikk

TDT4145 - Datamodellering og databasesystemer

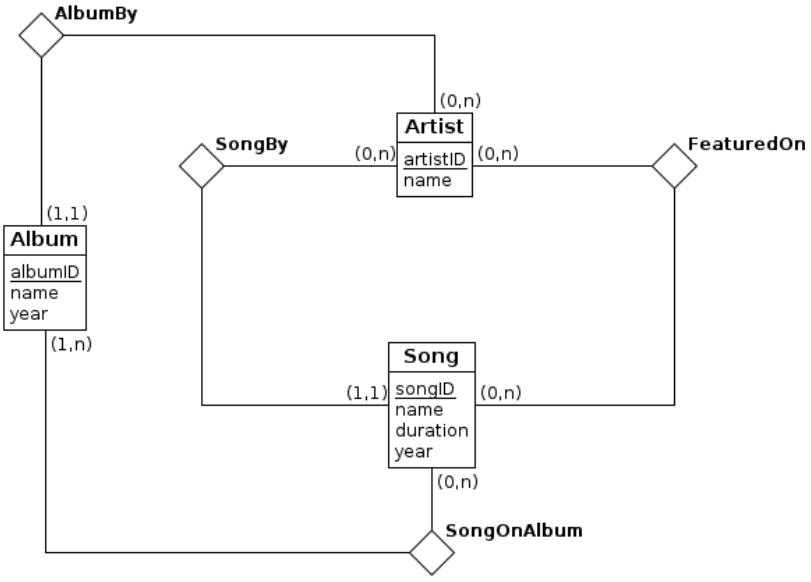
Øving 3

Læringsmål for denne øvingen er å:

- Lære seg å lage tabeller, sette inn, oppdatere og slette data i SQL.
- Lære seg SQL og relasjonsalgebra som spørrespråk.
- Kunne vurdere og forbedre relasjonsdatabaseskjema med utgangspunkt i prinsipper fra normaliseringsteori.
- Kunne vurdere og finne forekomster av funksjonelle avhengigheter i en tabell med data.

Oppgave 1: Lage SQL-tabeller og legge inn data

Følgende ER-diagram er gitt:



Fra diagrammet over kan vi se at modellen har noen begrensninger, for enkelhets skyld:

- En sang kan og må ha kun én hovedartist
- Et album kan og må høre til kun én artist

Databasen består dermed av følgende tabeller (understreket tekst signaliserer tabellens primærnøkkel):

- **artist(**artistID, name)
- **album**(albumID, name, year, artistID)
- **song**(songID, name, duration, year, artistID)
    - artistID er en fremmednøkkel som refererer til artist.artistID
- **featuredOn**(artistID, songID)
    - artistID and songID er fremmednøkler som refererer til henholdsvis artist.artistID og song.songID
- **songOnAlbum**(songID, albumID)
    - songID and albumID er fremmednøkler som refererer til henholdsvis song.songID og album.albumID

a) Vi ønsker at dersom man sletter eller oppdaterer en rad i artist-tabellen, skal tilsvarende referanser til denne raden også slettes eller oppdateres. Dette betyr at hvis man for eksempel sletter en artist, skal song og featuredOn også slette de radene der referanser til denne artisten inngår. Hvordan kan man spesifisere en slik restriksjon i SQL?

b) Lag en ny database i SQLite og skriv kode som konstruerer tabellene over i SQL. Velg selv passende datatyper for attributtene og husk primær- og fremmednøkler. Ta også med kravet presentert i a).

c) Skriv SQL-setninger som legger inn følgende data i databasen:

artist(1, “Dua Lipa”)
artist(2, “DaBaby”)
album(1, “Future Nostalgia”, 2022, 1)
song(1 “Levitating”, 203, 2022, 1)
songOnAlbum(1, 1)
featuredOn(2, 1)

d) Skriv en SQL-setning som oppdaterer navnet til DaBaby til Jonathan Lyndale Kirk.

e) Skriv en SQL-setning som sletter Jonathan Lyndale Kirk fra databasen.

Oppgave 2: SELECT-spørringer i SQL

I denne oppgaven skal dere gjøre spørringer mot en eksisterende SQLite-database. Databasen er en enkel musikkdatabase bestående av artister, album og låter, ispedd litt grunnleggende informasjon om disse entitetene, deriblant deres navn, utgivelsesår og sanglengde i sekunder.
Videre inkluderer databasen også informasjon om hvilke artister som **gjester** (på engelsk kjent som "featuring artist") på de ulike låtene, altså artister som er med på en låt uten at de er hovedartisten. Et eksempel på dette er låta "Baby" av Justin Bieber med Ludacris; Bieber er hovedartist, mens Ludacris kun har et gjestevers.

Databasen finner du i fila **musikk.db** og den ble laget ved hjelp av ER-diagrammet over. Åpne fila på den måten du selv liker best (via sqlite3-kommandoen, DB Browser, e.l.) og spør i vei! Svarene dine skal inneholde både spørringen og resultatet du får.

a) Skriv en spørring som returnerer sang-ID, tittel, lengde, år og artist-ID for alle sanger i databasen.
b) Skriv en spørring som returnerer navn og utgivelsesår for alle album utgitt før 2017.
c) Skriv en spørring som returnerer navn og utgivelsesår for alle sanger utgitt mellom 2018 og 2020 (inklusiv), sortert etter utgivelsesår.
d) Skriv en spørring som returnerer artistnavnet og sangnavnet for alle artister som har gjestet på en låt (hvor vedkommende altså ikke er hovedartist), sortert etter artistnavn og sangnavn.
e) Skriv en spørring som returnerer sangnavn, albumnavn og sangens utgivelsesår for alle låter av Ariana Grande, sortert etter år, albumnavn og sangnavn.
f) Skriv en spørring som, for alle låter hvor Ty Dolla Sign er enten hovedartist eller gjest, returnerer hovedartistens navn og sangnavnet, sortert etter artistnavn og sangnavn.
g) Skriv en spørring som returnerer artistnavn og sangnavn for alle låttitler som inkluderer tekststrengen "the".
h) Skriv en spørring som returnerer artistnavnet og antallet gjesteopptredener for den artisten i databasen med flest gjesteopptredener (låter hvor vedkommende er hovedartist teller ikke).
   (HINT: her kan en HAVING-betingelse komme godt med.)

Oppgave 3: Flere spørringer i relasjonsalgebra

Bruk samme database som i forrige oppgave og konstruér relasjonsalgebraspørringer som…

a) Finner artistnavn og sangnavn for alle låter i databasen som ikke er med på et album. Resultatsrelasjonen skal døpes songsWithoutAlbums(artistName, songName).
b) Finner alle artister som har gjestet på en låt fra dette tiåret hvis artistnavn begynner med B, samt alle artister som har gitt ut en låt på 2000-tallet (altså 2000-2009). Spørringen skal returnere artist- og låtnavn.
c) For alle artister, returnerer artistnavnet og antallet låter av dem i databasen, sortert synkende.

Task 4: Introduksjon til normaliseringsteori

Betrakt følgende tabell:

filmID	name	year	directorID	directorName	directorBirthYear
1	PlayTime	1967	1	Jacques Táti	1908
2	Mon Oncle	1953	1	Jacques Táti	1908
3	Spring Breakers	2012	2	Harmony Korine	1973
4	Monsieur Hulot's Holiday	1953	1	Jacques Táti	1908
5	Trafic	1971	1	Jacques Táti	1908
6	The Watermelon Woman	1996	3	Cheryl Dunye	1966

a) Dette er et eksempel på elendig databasedesign. Det er fordi vi lagrer mye av den samme informasjonen flere ganger (kjent som redundans); vi kan sannsynligvis strukturere tabellen på en smartere måte for å unngå dette. Vi får også problemer med innsetting, oppdatering og sletting av data; om vi sletter en film og databasen ikke inneholder flere filmer av samme regissør, mister vi all informasjon om regissøren.
   Hvis vi oppdager at Jacques Tatis fødselsår faktisk er 1907 og at a-en i etternavnet hans skrives uten aksent, hvor mange celler må vi oppdatere?

b) Kan du foreslå et alternativt design som gjør at vi trenger å oppdatere færre celler når vi endrer Tatis fødselsår og navn? (Tips: Normaliseringsteorien tilbyr som regel

løsninger som innebærer å splitte opp en tabell i flere deltabeller. I tilfellet over blander vi informasjon om filmer og regissører i samme tabell.)

Du kan anta at directorName og directorBirthYear vil ha samme verdier for identiske directorID-verdier. Med andre ord har vi følgende *funksjonelle avhengighet*: directorID -> directorName, directorBirthYear.

Oppgave 5: Funksjonelle avhengigheter, nøkler og tillukning

a) Betrakt følgende tabell. Hvilke påstander kan umulig stemme? Begrunn svarene dine.

A	B	C	D
a_1	b_1	c_1	d_1
a_1	b_1	c_1	d_2
a_2	b_2	c_2	d_2
a_3	b_4	c_3	d_3
a_3	b_3	c_3	d_1
a_4	b_3	c_4	d_2

1) A -> A
2) A -> B
3) A -> C
4) AB -> C
5) C -> D
6) D -> C
7) ABCD er en supernøkkel for tabellen
8) ABC er en supernøkkel for tabellen
9) D er en kandidatnøkkel for tabellen
10) ABD er en kandidatnøkkel for tabellen

a) X+ kalles tillukningen av X og er mengden av alle attributter som er funksjonelt avhengig av X (basert på en mengde funksjonelle avhengigheter F).
Gitt tabellen R = {A, B, C, D} and F = {D -> A, B-> D, ABD -> C}.
Finn D+, BC+, AB+, BD+. Hvor mange kandidatnøkler har R?