Oppgaver til repetisjonsforelesning H23 (devlis basert på Prøveeksamen H20)

1 Modellering

1.1 Fileshare-model [Redusert fra prøveeksamen H20]

Lag en ER-modell for databasen til et nettsted (kalt FileShare) som har som formål å gjøre det mulig for brukere å lagre, konvertere og dele filer.

- Du må gjerne dele modellen opp i deler (f.eks. over flere sider), men sørg for at det fremkommer tydelig hvilke deler av modellen som besvarer hvilke oppgaver.
- Du må gjerne inkludere kommentarer i modellene. Dersom det er uklarheter eller tvetydigheter i oppgavebeskrivelsen, bruk sunn fornuft og skriv en kommentar til modellene dine om hvilke antagelser og tolkninger du eventuelt gjør.
- Dersom du ser at det er mulig å modellere deler av oppgaven på flere ulike måter, diskuter kort hvordan din fremgangsmåte er i forhold til de ulike alternativene.
- Tegn og skriv tydelig.

Modellen skal inneholde følgende:

- FileShare identifiserer sine brukere internt med unike IDer (f.eks. `324`). FileShare krever at brukeren oppgir en e-postadresse. FileShare tillater ikke brukere å registrere mer enn én e-postadresse, og den samme eposten kan ikke benyttes av flere brukere. I tillegg kan brukere legge inn navn (bestående av for- og etternavn), samt brukerens telefonnumre (potensielt fler nummere per bruker).
- En av kjernefunksjonene til FileShare er å lagre informasjon om filer lastet opp av brukere på FileShare. Hver bruker kan opprette én eller fler mapper, men en mappe er opprettet av nøyaktig én bruker. Hver mappe har en unik sti (f.eks. `/home/jon/xls/`). En mappe kan så inneholde potensielt mange filer.
- Filer blir lagret i filsystemet til FileShare. Filene identifiseres internt av en unik kombinasjon av mappen der filen er lagret i filsystemet og filens navn (f.eks. `lønn.xls`).

- Brukere kan utføre operasjoner på filer, og en operasjon identifiseres med dens navn. F.eks. "Brukeren `324` utførte `KjørFil`-operasjonen på filen `run.exe` (i mappen `/home/jon/xls`)". En bruker kan utføre mange operasjoner på samme fil; en operasjon kan utføres på mange filer av en bruker; og en operasjon kan utføres av mange brukere på samme fil. Ettersom opprettelsen av en fil også blir lagret som en operasjon vil alle filer være del av minst én operasjonsutførelse.

1.2 Realisering [Redusert fra prøveeksamen H20]

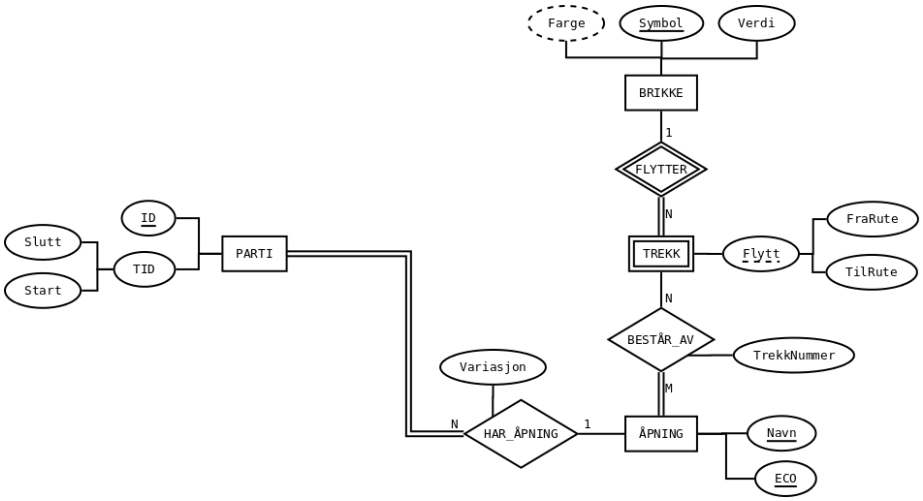

Figure 1: ER-modell

Realiser ER-modellen i figur 1 til et relasjonsdatabaseskjema. Relasjonsdatabaseskjemaet skal være korrekt (tilsvare ER-modellen), effektivt (unngå redundans og begrens antallet tabeller), og tydelig (lett å forstå). Bruk realiseringsalgoritmen for å danne et slikt relasjonsdatabaseskjema. For hver relasjon, spesifiser relasjonens navn og navnet til hvert attributt. Du skal ikke spesifisere datatyper/domener for attributtene, og ikke benytte SQL i denne oppgaven. Marker primærnøkler og kandidatnøkler tydelig. Skriv fremmednøkler enten med ord ($T(A)$ referer til $S(B)$) eller på formen $T(A) \to S(B)$ for å indikere at relasjon T sin A-attributt er en fremmednøkkel som peker på relasjon S sin B-attributt.

2 SQL

(Skjema og enkelte oppgaver basert på Prøveeksamen H20)

Du er ansatt som ny databaseadministrator hos Julenissens Verksted, hvor det er mye som skal gjøres før jul. Databasen til Julenissen er laget av følgende

SQL-script:

```
CREATE TABLE barn(
    bid int PRIMARY KEY,
    navn text NOT NULL,
    snill boolean NOT NULL
);

CREATE TABLE gave(
    gid int PRIMARY KEY,
    navn text NOT NULL,
    nyttig boolean NOT NULL
);

CREATE TABLE ønskeliste(
    barn int REFERENCES barn(bid),
    gave int REFERENCES gave(gid)
);
```

Tabellen `barn` inneholder informasjon om alle barn, hvor hvert barn har en unik ID (`bid`), et navn og en boolsk verdi som er sann om barnet har vært snill det siste året og usann hvis ikke.

Tabellen `gave` inneholder informasjon om gavene som Julenissens Verksted kan lage. Hver gave har en unik ID (`gid`), et navn samt en boolsk verdi som sier om gaven er en nyttig gave eller ikke.

Den siste tabellen, `ønskeliste`, inneholder alle barns ønsker, hvor hver rad inneholder et barns ID sammen med IDen til en gave det barnet ønsker seg. Merk: Et barn kan ønske seg mange gaver og en gave kan naturligvis bli ønsket av mange barn.

Under er et eksempel på hvordan databasen *kan* se ut.

barn

bid	navn	snill
0	Ola	t
1	Kari	t
2	Per	f
3	Nils	t
4	Mari	f
5	Kine	f
6	Jasmin	t

gave

gid	navn	nyttig
0	Lekebil	f
1	Sokker	t
2	Sykkel	t
3	Lekepistol	f
4	Dataspill	f
5	Fuglebrett	t
6	Dukke	f
7	Bok	t

ønskeliste

barn	gave
1	0
1	6
0	6
0	5
3	2
5	1
5	3
5	4
4	3
4	0
6	7

2.1 Antall gaver [Endret fra prøveeksamen H20]

Skriv en spørring som finner antall nyttige gaver hvert barn ønsker seg. Merk at det kan finnes barn som ikke ønsker seg noen nyttige gaver, og disse skal også med i svaret. Resultatet skal inneholde navn på barnet og antall gaver barnet ønsker seg.

2.2 Like ønkser [Endret fra prøveeksamen H20]

Skriv en SQL-spørring som finner alle par av *ulike* barn som har ønsket seg det samme. Svaret skal kun inneholde unike rader.

2.3 Populære gaver [Fra prøveeksamen H20]

Skriv en spørring som finner de tre mest populære nyttige gavene, og de tre mest populære unyttige gavene. Resultatet skal inneholde navn på gaven, antall barn som ønsker gaven, samt hvorvidt den er nyttig eller ikke.

2.4 Gaveliste [Fra prøveeksamen H20]

Skriv en spørring som tilordner gaver til barn i henhold til følgende regler:

- Snille barn får alt de ønsker seg
- Usnille får kun nyttige ting de ønsker seg. Om de ikke ønsker seg noen nyttige ting får de gaven med navn `'Genser'`.

Spørringen skal skrive ut navnet på barnet og navnet på gaven.

2.5 Programmering med SQL

Implementer en funksjon/metode som setter inn et nytt barn i `barn`-tabellen. Du velger selv om du vil bruke Python eller Java. Metoden har følge signatur i Python:

```python
def sett_inn_barn(conn, navn):
    # TODO
```

og følgende signatur i Java:

```java
void settInnBarn(Connection conn, String navn) throws SQLException {
    // TODO
}
```

Funksjonen/metoden tar som første argument et `Connection`-object med tilkobling til databasen. Det tredje argument er en streng som inneholder navnet til barnet som skal settes og er oppgitt av en bruker av programmet. Barnet skal få en `bid`-verdi som er `1` høyere enn høyeste `bid`-verdi til barn allerede i databasen. Du kan anta at alle barn er snille til det motsatte er bevist, og skal derfor ha `snill=true` når de settes inn i databasen.

I oppgaven trenger du ikke håndtere feil (`Exceptions`), men implementasjonen skal ikke være sårbar for SQL-injection-angrep via `navn`-argumentet.

2.6 Sikkerhet

Gitt følgende funksjon i Python:

```python
def skriv_ut_gave(conn, gave):
    cur = conn.cursor()
    q = "SELECT gid, navn, nyttig FROM gave WHERE navn = '" + gave + "';"
    cur.execute(q)
    for row in cur:
        print("Gave: " + str(row[0]) + ", " + row[1] + ", " + str(row[2]))
```

eller følgende metode i Java:

```java
void skrivUtGave(Connection conn, String gave) throws SQLException {

    Statement stmt = conn.createStatement();
    String q = "SELECT gid, navn, nyttig FROM gave WHERE navn = '" + gave + "';"
    ResultSet res = stmt.executeQuery(q);
    while (res.next()) {
        System.out.println("Gave: " + res.getInt(1) + ", "
            + res.getString(2) + ", " + res.getBoolean(3));
    }
}
```

Oppgi en verdi for argumentet `gave` som fører til at funksjonen/metoden heller printer ut informasjon om alle barn (og ingenting annet).

3 Relasjonsmodellen og dekomponering

3.1 Nøkler [Fra prøveeksamen H20]

Gitt følgende relasjon

$R(A, B, C, D, E)$

med FDene

1. $A \rightarrow B$
2. $BC \rightarrow D$
3. $DE \rightarrow A$

Hvilke kandidatnøkler har R? Vis hvordan du kommer frem til svaret.

3.2 Tapsfri dekomponering [Fra prøveeksamen H20]

Gitt følgende relasjon

$R(A, B, C, D, E, F)$

med kandidatnøkler AB og CD, og FDer:

1. $AB \rightarrow C$
2. $AB \rightarrow D$
3. $CD \rightarrow A$

4. $CD \rightarrow B$
5. $A \rightarrow E$
6. $C \rightarrow F$

Dekomponer relasjonen tapsfritt til BCNF. Vis hvordan du kommer frem til svaret. For hver relasjon du får underveis i dekomponeringen, list opp hvilke FDer som holder, og hvilke kandidatnøkler relasjonen har.