Øving 8: Egen databaseløsning

Navn: Jens Christian Aanestad  
Dato: 06.11.2023

* 1. Definere behov

**Case:** Optimalisering avstatistikkføring i fotball for fiktive FotballBet AS.FotballBet AS ønsker seg et statistikksystem for fotballklubber og fotballspillere for flere sesonger i de fem øverste fotballdivisjonene. Intensjonene er å bruke dataene for å lage en prognose for et fremtidig resultat, slik at betting vil bli mer lønnsomt.

**Dagens situasjon:** FotballBet AS har et stort utvalg av fotball statistikk, inkludert kamp resultat, målpoeng, spillerbørs, tabell etc. Statistikkinnføringen er manuell, og av den grunn hender det at feil data blir plottet inn i systemet. Noe som kan forverre prognosene.I systemet er det ingen oppdeling av divisjonene eller sesonger. Dette gjør det vanskelig å kategorisere fotballstatistikk. I tillegg er det heller ingen oppdeling av spillerstatistikk for en sesong.  
  
Det verste av alt er at det ikke er innført statistikk for kort, eller utvisning, noe som kan ha store innvirkninger på en fotballkamp.

**Fremtidig behov:** FotballBet AS ønsker å styrke dette svake leddet, ved å øke brukervennligheten for de ansatte. På denne måten vil det bli enklere å innføre statistikk, og FotballBet AS kan fortsette å spekulere i fotballbetting.

**Krav til et nytt system:** FotballBet AS ønsker å innføre divisjoner, og sesonger, der hvor en sesong har oversikten til de fem øverste divisjonene. I divisjonene skal det føres opp en ligatabell over stillingen, med poeng, målpoeng, vinn-uavgjort-tap. Det skal i tillegg være en egen inndeling for alle kamper i en divisjon. I disse kampene, skal det føres statistikk for ballbesittelse, skudd på mål, forventet mål, taklinger, gule kort, rødekort, pasninger, forseelser etc. For hver klubb skal det også føres en egen spillerstatistikk for hver spiller i klubben. Her skal statistikk som driblinger, pasninger, km løpt, skudd, antall ganger mistet ball, og spillerbørs føres opp.

**Begrensninger**

FotballBet AS er kun interessert i de siste 5 sesongene for å spare plass.

* 1. Egen relasjonsdatabase eksempler i SQL

**Datamodell (ER):**

Et bilde som inneholder skjermbilde, tekst, diagram, design

Automatisk generert beskrivelse

**Relasjonsmodell:**

Sesong(sesong\_beskrivelse)

Divisjon(divisjon\_id, divisjon\_tittel, land, sesong\_beskrivelse\*)

Ligatabell(ligatabell\_id, divisjon\_id\*)

Klubbinfo(klubbinfo\_id, ligatabell\_id\*, fotballklubb\_id\*, plassering, kamper\_spilt, vunnet, uavgjort, tapt, mål\_scoret, mål\_sluppet\_inn, gule\_kort, røde\_kort poeng)

Fotballklubb(fotballklubb\_id, divisjon\_id\*, fotballklubb\_navn, bane, etableringsår, markedsverdi, eier)

Spiller(spiller\_id, klubb\_id\*, fornavn, etternavn, nasjonalitet, alder, posisjon)

Spillerstatistikk(spillerstatistikk\_id, kamp\_id\*, skudd\_på\_mål, dribblinger, pasninger, innlegg, røde\_kort, gule\_kort, km\_tilbakelagt, antall\_mistet\_ballen)

Kamp(kamp\_id, matchday, dato)

Kampstatistikk(kampstatistikk\_id, kamp\_id\*, resultat, skudd\_på\_mål, ballbesittelse, hjørnespark, pasninger, gule\_kort, røde\_kort)

Hjemmelag(kampstatistikk\_id\*, fotballklubb\_id\*)

Bortelag(kampstatistikk\_id\*, fotballklubb\_id\*)

**Spørring 1: Finn etternavn til alle spillerne i FC Barcelona.**

SELECT s.etternavn  
FROM spiller s  
INNER JOIN fotballklubb fb ON s.fotballklubb\_id = fb.fotballklubb\_id  
WHERE fb.fotballklubb\_navn = "FC Barcelona"

**Spørring 2: Finn klubben/klubbene med flest gule kort i Premier League.**

SELECT fb.fotballklubbnavn  
FROM fotballklubb fb  
INNER JOIN klubbinfo ki ON fb.klubbinfo\_id = ki.klubbinfo\_id  
INNER JOIN ligatabell l ON l.ligatabell\_id = ki.ligatabell\_id  
INNER JOIN divisjon d ON d.divisjon\_id = l.divisjon\_id  
WHERE d.divisjon\_tittel = "Premier League"  
AND ki.gule\_kort = (SELECT MAX(ki2.gule\_kort)  
FROM klubbinfo ki2);

**Spørring 3: Finn resultatet mellom hjemmekampen til Man United mot Man City fra denne sesongen (altså 2023/2024-sesongen).**

SELECT h.resultat  
FROM hjemmelag h  
INNER JOIN bortelag b ON h.kamp\_id = b.kamp\_id  
INNER JOIN fotballklubb fk1 ON fk.fotballklubb\_id = h.fotballklubb\_id  
INNER JOIN fotballklubb fk2 ON fk.fotballklubb\_id = b.fotballklubb\_id  
INNER JOIN klubbinfo ki ON ki.fotballklubb\_id = fk1.fotballklubb\_id  
INNER JOIN ligatabell l ON l.ligatabell\_id = ki.ligatabell\_id  
INNER JOIN divisjon d ON d.divisjon\_id = l.divisjon\_id  
INNER JOIN sesong s ON s.sesong\_beskrivelse = d.sesong\_beskrivelse  
WHERE fk1.fotballklubb\_navn = "Man United"  
AND fk2.fotballklubb\_navn = "Man City"  
AND s.sesong\_beskrivelse = "2023/2024";

**Kort oppsummering om erfaringer med database og videre arbeid:**

Oppsettet av SELECT setninger var relativt enkelt å gjennomføre, så lenge det ikke var sære spørringer. I den siste spørringen ble litt for mange foreninger, men dette kunne faktisk vært unngått om jeg hadde gått veien fra kamp til divisjon og så til sesong. Er fortsatt usikker på om subtyping i denne relasjonsdatabasen er en god ide. Begrunnelsen er at sub-tabellene ikke tar inn noen unike attributter. På den andre siden ville jeg skilt mellom statistikk for hjemmelag og bortelag slik at hver og en av dem kan holde på deres statistikk i en fotball kamp.

For evt. videre arbeid ville jeg ha lagt inn flere kolonner for statistikk i tabellene kampstatistikk, og spillerstatistikk. I de tabellene jeg har nå, mangler det en god del som antall feilslåtte pasninger, fullførte pasninger, flate/korte pasninger, lobb pasninger, gjennombruddspasninger, innlegg, trivela, pasningspresisjon (%), skuddpresisjon (%), lobbskudd, chip-skudd, volley, low-driven. knockle-ball, osv. Dette orket jeg ikke å legge inn i denne oppgaven fordi det ville tatt litt for lang tid

* 1. Løsning med XML evt. JSON i MySQL

**Insert-setning med JSON:**

INSERT INTO spiller(spiller\_id, spiller\_info)  
VALUES(1, JSON\_OBJECT("klubb\_id", 1, "fornavn", "Erling", "etternavn", "Braut Haaland", "nasjonalitet", "Norge", "Alder", 23, "Posisjon", "Spiss")  
);  
  
**Select-spørring med JSON:**

SELECT JSON\_EXTRACT(spiller\_info)   
FROM spiller s   
WHERE s.spiller\_id = 1;

Noen fordeler ved å bruke lagre alle eller noen av dataene i databasen i oppgave 1.2 som JSON, er at det er en enkel og fleksibel måte å lagre data på. I tillegg er JSON menneskelig lesbart, noe som kan bidra til å forstå data. Dette kan komme til nytte i tabellene som har mange kolonner/attributter for forskjellig statistikk. Det er i dette området hvor JSON kan være nyttig for denne databasen.   
  
Siden denne databasen har kompleks sammenheng og relasjonsintegritet mellom klubb, spiller, divisjon osv. virker fullstendig bruk av JSON ugunstig siden den ikke ser relasjonene.

* 1. NoSQL-løsning

For å vurdere en NoSQL-løsning som et alternativ til vår relasjonsdatabase, må vi både se på fordeler og ulemper med en slik implementasjon.

For det første er en SQL-database velegnet når datastrukturen er kjent og relativt stabil, mens en NoSQL-database mer egnet for håndtering av semi-strukturerte eller ustrukturerte data. Tabellene i databasen til FotballBet AS kan virke ustrukturerte, og man kan av den grunn argumentere for at NoSQL er en bedre løsning. Men relasjonene mellom tabellene er fortsatt viktig siden FotballBet AS er veldig opptatt fra å skille sesong og divisjoner, for å prøve å estimere fotball resultater i divisjonene.

For det andre, SQL-databaser støtter transaksjoner, noe som er avgjørende for å opprettholde datakonsistens i statistikkinnføringer (som er en viktig faktor for FotballBet AS). I motsetning så mangler ofte NoSQL-databaser støtte for transaksjoner, som kan bli en utfordring. I tillegg kan enkelte NoSQL-databaser kreve ekstra sikkerhetstiltak når de håndterer kundedataen, ettersom de vanligvis ikke har innebygde sikkerhetsfunksjoner som SQL-databaser har.

Av disse grunnene tror jeg ikke at en NoSQL-løsning kan være egnet for relasjonsdatabasen til FotballBet AS. Men hvis FotballBet AS plutselig velger å innføre mye mer statistikk, og derfor trenger en løsning som er mer skalerbar for å takle store datamengder, kan NoSQL mulig være et bedre alternativ likevel.