Øving 8: Egen databaseløsning

## Definere behov

Jeg har valgt den akademiske verden som utgangspunkt for oppgaven. Denne problemstillingen dreier seg om modellering av en database som inneholder informasjon om forskere, akademiske institusjoner og samarbeid mellom forskere. En forsker kan enten være ansatt som professor eller laboratorieassistent. Det er tre typer professorer:

Følgende bør lagres:

• For hver forsker, hans/hennes navn, fødselsår og nåværende stilling (hvis mulig).

• For hver institusjon, navn, land og åpningsår.

• For hver institusjon, navnene på skolene (f.eks. Handels høyskole etc.)

Skolen for informatikk,. . .). En skole tilhører nøyaktig én institusjon.

• En ansettelseshistorikk, inkludert informasjon om alle ansettelser (start og slutt

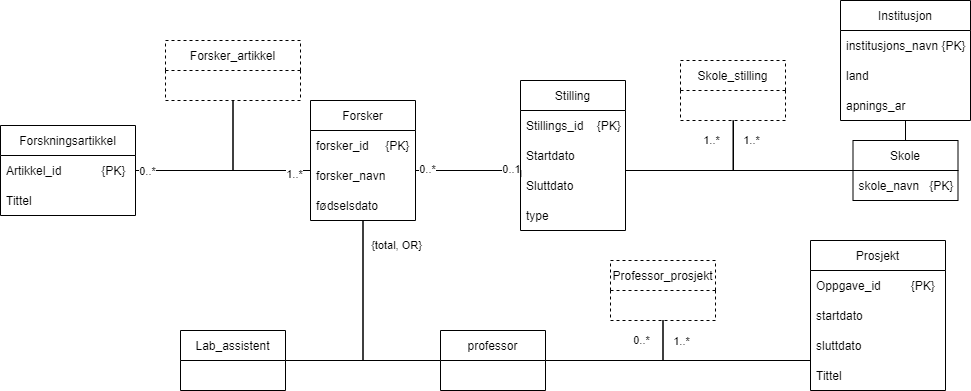
dato, stilling og hvilken skole).

• Informasjon om medforfatterskap, dvs. hvilke forskere som har vært medforfatter til en forsknings artikkel. Titlene på vanlige forskningsartikler bør også lagres.

• For hver professor, informasjon om hvilke forskningsprosjekter (tittel, startdato og slutt).

dato) han/hun er involvert i.

## Egen relasjonsdatabase med eksempler i SQL



Relasjonsmodell

***Primærnøkler er markert slik*** -> (\_primærnøkkel\_) og ***fremmednøkler slik*** -> (*fremmednøkkel\**)

Forsker( **\_forsker\_id\_** *,* forsker\_navn, fodselsdato, stilling)

Forskningsartikkel( **\_artikkel\_id\_** , *forsker\_id*\*, tittel)

Forsker\_artikkel(**(\_forsker\_id\_\*, artikkel\_id\*)**)

Stilling( **\_stillings\_id\_** , *forsker\_id\**, startdato, sluttdato og type)

Skole( **\_skole\_navn\_,** *institusjons\_navn\**) //antar hver skole og institusjon i et land har forskjellige navn

Skole\_stilling( (**\_stillings\_id\_\*, \_*skole\_navn*\_**\*))

Institusjon( **\_institusjons\_id\_ ,** land, apnings\_ar)

Lab\_assistent(**\_*forsker\_id\_\****)

Professor(**\_*forsker\_id\_\****)

Prosjekt( **\_oppgave\_id\_,** startdato, sluttdato, tittel)

Professor\_prosjekt( (**\_forsker\_id\_\*, \_oppgave\_id\_\***) )

**SQL SPØRRINGER**

**Lag en liste over alle titteler av artikler skrevet av en forsker.**

SELECT fa.tittel FROM forskningsartikkel fa JOIN forsker\_artikkel f\_a ON(fa.artikkel\_id = f\_a.forsker\_id) GROUP BY fa.tittel;

**Lag en liste over alle prosjekter en professor har vært en del av, med tittel og id av prosjektet (en professor med navn Jeff for eksempel)**

SELECT prosj.oppgave\_id, prof.navn , prosj.tittel FROM professor prof JOIN prosjekt prosj JOIN professor\_prosjekt pf ON (pf.oppgave\_id = prosj.oppgave\_id AND pf.professor\_id = prof.professor\_id) WHERE (prof.forsker\_navn = “Jeff”);

**lag en liste av alle skoler for hver institusjon**

SELECT i.institusjons\_navn, s.skole\_navn FROM institusjon i LEFT JOIN skole s ON (i.institusjons\_id = s.institusjons\_id);

**erfaringer ved bruk av databaseløsningen:** Måtte endre på ER modellen mange ganger for at spørringer henger godt sammen med ER modellen. Startet med to mange til mange relasjoner og legget til forsker\_artikkel slik at det ble 3 totalt til slutt. Jeg brukte en del tid på ER modellen, noe som gjorde de andre stegene mye lettere.

XML kan hjelpe i å lagre ustrukturert data i relasjonsdatabaser. XML er også nyttig hvis vi ønsker brukeren skal velge strukturen av informasjonen som lagres. Her har jeg valgt attributtet «type» i stilling tabellen fordi det kan være gunstig å lagre noen andre informasjon knyttet til stillingen. For eksempel kan det være nyttig å lagre type i stilling tabellen med navn og rolle beskrivelse som XML tekst.

CREATE TABLE stilling (stillings\_id INT NOT NULL,startdato DATE,sluttdato DATE, type VARCHAR(400), PRIMARY KEY (id));

INSERT INTO stilling VALUES(1,’2000-10-10’, ’2010-10-11’,'<type><navn>Assosiate professor of medicine</navn> <rolle> Initiate research, supervise doctoral degree students, participate in teaching and examination work at all levels, and undertake administrative tasks, in accordance with applicable provisions. </rolle></type>');

Select setning som kan hente informasjonen om type navn fra stillings tabellen

SELECT ExtractValue(stilling, '/type/navn') AS navn FROM stilling;

Select setning som kan hente informasjonen om type rolle beskrivelse fra stillingstabellen

SELECT ExtractValue(stilling, '/type/rolle) AS rolle\_beskrivelse FROM stilling;

Fordelene med å lagre data slik var nevnt over, men hva med ulemper? En ulempe kan være at det kan være vanskelig å holde orden på data. Siden jeg har ikke implementert noen form for kontroll for XML data-en jeg setter inn kan det fort bli kaos.

**Hvor godt egnet er NoSQL-løsning til oppgaven:** NoSQL er ikke godt egnet til denne oppgaven siden det ville ha vært mye vanskeligere å holde oversikt på relasjonene (pga. mange relasjoner, avhengigheter osv. i denne oppgaven). En NoSQL løsning kunne ha vært en bedre mulighet hvis fleksibilitet med databasen hadde hoved prioritet.