YrkesCo

PostgreSQL

Bakgrund

Ni läser i en yrkeshögskola YrkesCo och många yrkeshögskolor jobbar med många excelfiler för att tracka

olika saker såsom studenter, utbildare, utbildningsledare, anställda, kurser, program, konsulter mm. Det är

inte ovanligt att en yh-anordnare också är uppdelad till flera enheter i flera orter i Sverige. Det är inte

ovanligt att många anställda sitter med samma excelfiler och/eller egna excelfiler för att hålla koll på sina

arbetsområden/utbildningar. I kombination med excelfilerna finns också viss information i lärplattformar.

konceptuell Model

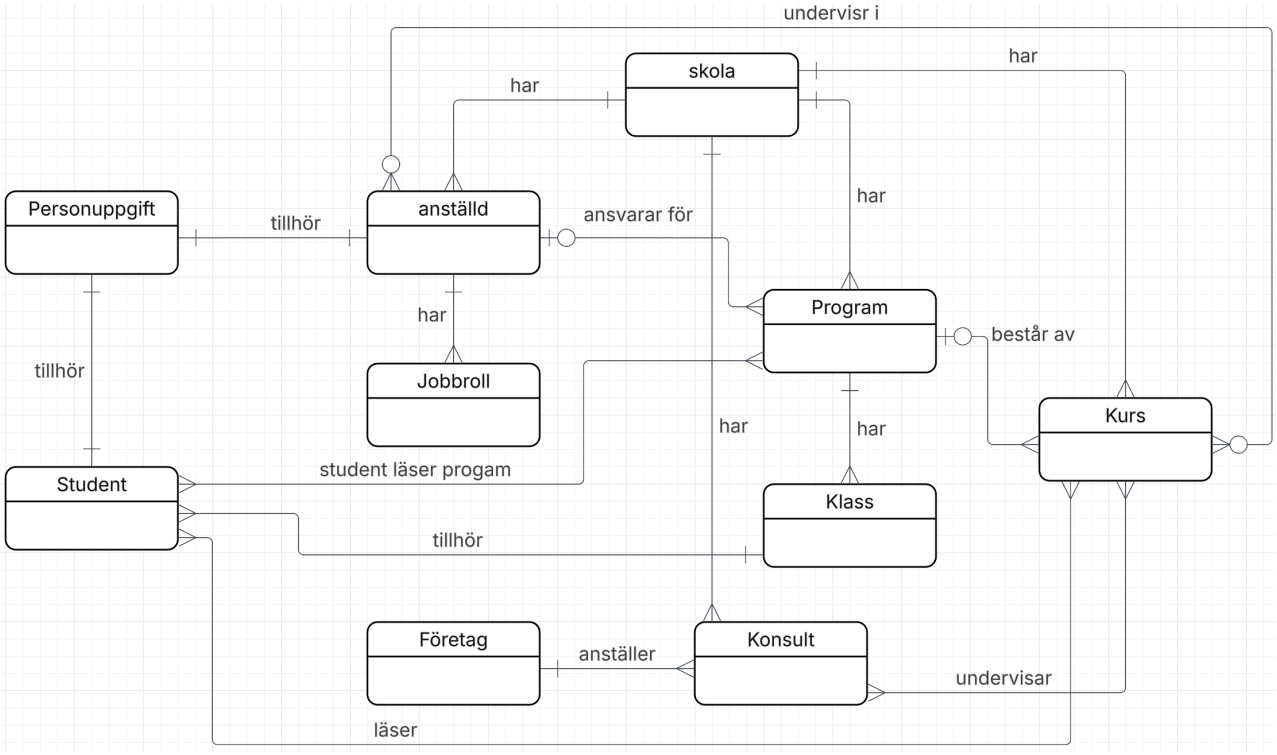
Modellen beskriver hur YrkesCos verksamhet hänger ihop på en övergripande nivå.

Studenter är primärt knutna till klasser som tillhör program.

Undervisningen genomförs av utbildare, som antingen kan vara fastanställda eller konsulter.

För att hantera integritet och regelverk kommer personuppgifter vara åtskilda från övrig verksamhetsinformation.

Sammanfattning, så visar modellen hur studenter, kurser, program, klasser, personal, konsulter och anläggningar samverkar på ett sätt som speglar hur YrkesCo faktiskt bedriver sin verksamhet.



Logisk Model

Den logiska modellen är ett mellansteg mellan den konceptuella modellen och den faktiska databasen.

I den logiska modellen har jag brutit ner verksamheten till konkreta **tabeller med** attribut, tydliga primärnycklar och relationer.

Fokus här ligger på på struktur**,** beroenden och affärsregler.

Centrala principer i modellen

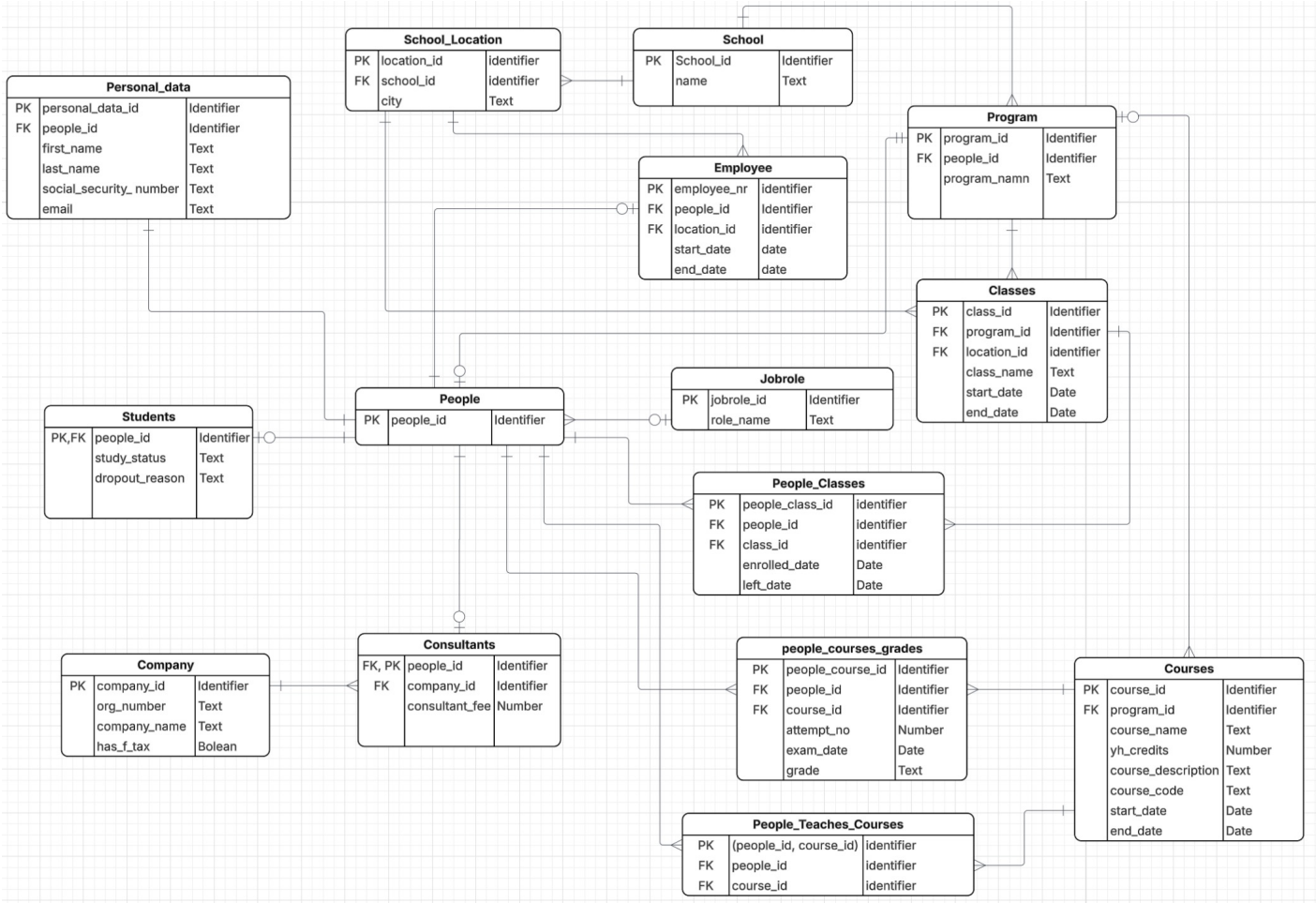
Varje tabell representerar en tydlig sak i verksamheten

(t.ex. student, anställd, kurs, klass, program).

Attribut ligger där de logiskt hör hemma, inte där de råkar behövas.

Exempel på hur modellen är uppbyggd

Tabellen People fungerar som en superklass och identiteten ärvs vidare till students som är en subklass.



Fysisk model_DDL

Tabellerna måste skapas i ordning från oberoende entiteter till beroende och avslutas med brygg tabeller.

UUID - Universally Unique Identifier.

```sql
CREATE SCHEMA IF NOT EXISTS core;

SET search_path TO core;

-- core.people

CREATE TABLE people (
    people_id UUID PRIMARY KEY DEFAULT gen_random_uuid()
);

-- core.personal_data

CREATE TABLE personal_data (
    personal_data_id UUID PRIMARY KEY DEFAULT gen_random_uuid(),
    people_id UUID NOT NULL UNIQUE,

    first_name VARCHAR(100) NOT NULL,
    last_name  VARCHAR(100) NOT NULL,

    social_security_number VARCHAR(13) NOT NULL UNIQUE,
    email VARCHAR(255) NOT NULL UNIQUE,

    CONSTRAINT chk_ssn_format
        CHECK (social_security_number ~ '^[0-9]{8}-[0-9]{4}$'),

    CONSTRAINT fk_personal_data_people
        FOREIGN KEY (people_id)
        REFERENCES people(people_id)
        ON DELETE CASCADE
);
```

Insättning av data vid tom databas

Man börjar alltid med grundtabeller utan beroenden, därefter personer och roller, sedan utbildningsstrukturen och sist bryggtabellerna som kopplar ihop allt. Det här är extra viktigt vid frontend-utveckling för att undvika brutna referenser och fel i databasen.

Verifiering av join – exempel 1

Den här joinen visar vilket program som finns och vilken utbildningsledare som är ansvarig för programmet. Kopplingen går från program till person och vidare till personuppgifter.

```
yrco_db=# SELECT
  pr.program_name,
  pd.first_name,
  pd.last_name
FROM core.program pr
JOIN core.people p          ON pr.people_id = p.people_id
JOIN core.personal_data pd ON p.people_id = pd.people_id;
  program_name  | first_name | last_name
----------------+------------+-----------
 Data Analytics | Ulf        | West
```

Verifiering av join – exempel 2

Detta visar förelningen av betyg för kursen Pandas
Kopplingen går via course_id, som finns i people_courses_grades och courses.

```
FROM core.people_courses_grades pc
JOIN core.courses c ON pc.course_id = c.course_id
WHERE c.course_code = 'PAND'
GROUP BY grade;
 grade | count
-------+-------
 G     |    36
 IG    |    14
 VG    |    22
(3 rows)
```

Verifiering av join – exempel 3

Denna är frågan visar vilka konsulter som undervisar vilka kurser och vilket företag de representerar.

```
yrco_db=# SELECT
  pd.first_name,
  pd.last_name,
  co.company_name,
  cr.course_code,
  cr.course_name
FROM core.people_teaches_courses ptc
JOIN core.people p         ON ptc.people_id = p.people_id
JOIN core.personal_data pd ON p.people_id = pd.people_id
JOIN core.consultants c    ON p.people_id = c.people_id
JOIN core.company co       ON c.company_id = co.company_id
JOIN core.courses cr       ON ptc.course_id = cr.course_id
ORDER BY co.company_name, cr.course_code;
 first_name | last_name |     company_name      | course_code |  course_name
------------+-----------+-----------------------+-------------+---------------
 Helena     | Mark      | Analytics Partners AB | DCLN        | Data Cleaning
 Helena     | Mark      | Analytics Partners AB | PYTH        | Python
 Lars       | Ek        | DataConsult AB        | PAND        | Pandas
 Lars       | Ek        | DataConsult AB        | SQL         | SQL
(4 rows)
```

Kontroll av ”constraints” - exempel 1

Yh-credits måste vara > 0

- Misslyckad uppdatering av yh_credits

```
yrco_db=# UPDATE core.courses
SET yh_credits = -10
WHERE course_code = 'PAND';
ERROR:  new row for relation "courses" violates check constraint "chk_yh_credits"
DETAIL:  Failing row contains (cbc94aa7-54cf-42be-b782-2cb060a19781, 6eac724b-a449-4188-809e-18a2d16bc80c
, PAND, Pandas, -10, Data analysis with Pandas, null, null).
```

- Lyckad uppdatering av yh_credits

```
yrco_db=# UPDATE core.courses
SET yh_credits = 60
WHERE course_code = 'PAND';
UPDATE 1
yrco_db=#
```

Kontroll av ”constraints” - exempel 2

”Org-number” måste vara i formatet: ######-####

- Misslyckad insättning av ett nytt konsult bolag

```
yrco_db=# INSERT INTO core.company (
  company_name,
  org_nr,
  has_f_tax
)
VALUES (
  'Fake Company AB',
  'ABC-123',
  true
);
ERROR:  new row for relation "company" violates check constraint "chk_org_nr_format"
DETAIL:  Failing row contains (9a827470-5acc-42f1-93b8-ea043b491125, Fake Company AB, ABC-123, t).
```

- Lyckad insättning av ett nytt konsult bolag

```
yrco_db=# INSERT INTO core.company (
  company_name,
  org_nr,
  has_f_tax
)
VALUES (
  'Fake Company AB',
  '556789-1234',
  true
);
INSERT 0 1
```

Kontroll av ”constraints” - exempel 3

Konsultarvode måste vara > 0

- Misslyckad uppdatering av consultant_fee (konsultarvode)

```
yrco_db=# UPDATE core.consultants
SET consultant_fee = -500
WHERE company_id = '712eb850-491e-4045-8ec4-e68d52563913';
ERROR:  new row for relation "consultants" violates check constraint "chk_consultant_fee"
DETAIL:  Failing row contains (e737ba0a-aa84-4ad8-86f9-6f6554b77d78, 712eb850-491e-4045-8ec4-e68d52563913
, -500.00).
```

- Lyckad uppdatering av consultant_fee (konsultarvode)

```
yrco_db=# UPDATE core.consultants
SET consultant_fee = 1250
WHERE company_id = '712eb850-491e-4045-8ec4-e68d52563913';
UPDATE 2
```

Relationship statement

Skola

- En skola har en eller flera anställda, och varje anställd tillhör exakt en skola.
- En skola har ett eller flera program, och varje program tillhör exakt en skola.
- En skola erbjuder fristående kurser, och varje fristående kurs tillhör exakt en skola.

Anställd

- En anställd tillhör exakt en skola, och en skola kan ha flera anställda.
- En anställd har en eller flera jobbroller, och en jobbroll kan innehas av flera anställda.
- En anställd ansvarar för ett eller flera program, och varje program har exakt en ansvarig anställd.
- En anställd undervisar en eller flera kurser, och en kurs kan undervisas av flera anställda.
- En anställd har exakt en uppsättning personuppgifter.

Jobroll

- En jobbroll kan tilldelas en eller flera anställda, och varje anställd har minst en jobbroll.

Personuppgift

- Personuppgifter tillhör exakt en individ i skolan.
- Varje individ i skolan (student, anställd eller konsult) har exakt en uppsättning personuppgifter.

Relationship statement (forts)

Student

- En student är en individ i skolan.
- En student läser exakt ett program åt gången, men kan läsa flera program över tid.
- En student tillhör exakt en klass åt gången, men kan tillhöra flera klasser över tid.
- En student tar en eller flera kurser, och en kurs kan tas av flera studenter.

Program

- Ett program tillhör exakt en skola, och en skola kan ha flera program.
- Ett program består av en eller flera kurser, och varje kurs kan tillhöra noll eller ett program.
- Ett program har en eller flera klasser, och varje klass tillhör exakt ett program.
- Ett program har exakt en ansvarig anställd, och en anställd kan ansvara för flera program.

Klass

- En klass tillhör exakt ett program, och ett program kan ha flera klasser.
- En klass har flera studenter, och en student går i en klass åt gången.

Relationship statement (forts)

Kurs

- En kurs kan ingå i ett program eller vara fristående.
- En kurs kan tas av flera studenter, och en student kan ta flera kurser.
- En kurs kan undervisas av en eller flera konsulter och/eller anställda

Konsult

- En konsult är en individ i skolan men är inte anställd av skolan.
- En konsult tillhör exakt ett företag, och ett företag kan ha flera konsulter.
- En konsult undervisar en eller flera kurser, och en kurs kan undervisas av flera konsulter.

Företag

- Ett företag har en eller flera konsulter, och varje konsult tillhör exakt ett företag.

Krav	Förtydligande av krav	Implementerat (Ja/Nej)
om studenter, förnamn, efternamn, personnummer, email	Information om studenter ska omfatta personuppgifter såsom förnamn, efternamn, personnummer och e-postadress. Personuppgifter ska lagras separat från övrig information för att underlätta åtkomstkontroll och skydd av känsliga uppgifter.	ja
utbildare kan vara konsulter	Utbildare kan vara antingen fast anställda av skolan eller externa konsulter. Konsulter är knutna till externa företag och är inte anställda av YrkesCo.	ja
de planerar att anställa fasta utbildare (BONUS)	YrkesCo planerar att anställa fasta utbildare. Fasta utbildare är anställda av skolan och kan ha olika utbildningsrelaterade roller, såsom lärare eller utbildningsledare.	ja
utbildningsledare och deras personuppgifter	Utbildningsledare är anställda med en särskild roll. Deras personuppgifter ska lagras separat från roll- och ansvarsinformation för att underlätta dataskydd och åtkomstkontroll.	ja
utbildningsledare har hand om 3 klasser	Varje utbildningsledare ansvarar för ett program. Ett program ges i tre omgångar och har därmed tre klasser, vilket innebär att utbildningsledaren ansvarar för tre klasser via programmet.	ja
kurser med namn, kurskod, antal poäng, kort beskrivning av kursen	Kurser ska lagras med kurskod, namn, antal YH-poäng och en kort beskrivning. Kurser är fristående objekt som kan ingå i program, klasser eller ges som fristående kurser.	ja
program har ett antal kurser knutna till sig	Ett program består av ett antal kurser som tillsammans utgör utbildningens innehåll. Kurser är normalt kopplade till ett program, men systemet ska även kunna hantera fristående kurser som inte ingår i något program.	ja

Krav	Förtydligande av krav	Implementerat (Ja/Nej)
ett program blir beviljat i tre omgångar, dvs att det finns 3 klasser	Ett utbildningsprogram beviljas i tre omgångar, vilket innebär att samma program startar tre gånger och därmed har tre tillhörande klasser. Varje klass representerar en specifik startomgång med egna datum.	ja
det finns även fristående kurser (BONUS)	Systemet ska stödja fristående kurser som inte är kopplade till något utbildningsprogram. Studenter ska kunna läsa och examineras på fristående kurser utan att tillhöra en klass.	ja
konsulter, deras företag, företagsinfo som organisationsnummer, har F-skatt, address, hur mycket de tar i arvode per timma	Konsulter representerar externa utbildare och är kopplade till ett företag. För varje företag lagras organisationsnummer, information om F-skatt samt adress. För varje konsult lagras timarvode. Konsulter är inte anställda av YrkesCo utan anlitas via sitt företag.	ja
YrkesCo har två anläggningar, en i göteborg och en i stockholm, i framtiden kanske de kommer expandera till flera orter (BONUS)	YrkesCo ska kunna ha flera anläggningar kopplade till samma skola. Varje anläggning representar en fysisk plats (stad), och systemet ska kunna hantera framtida expansion till nya orter utan strukturella ändringar.	ja
känsliga personuppgifter bör läggas i egna entiteter, eftersom det blir lättare att kontrollera vem som får access till dem tabellerna.	Ppersonuppgifter ska läggas i egna entiteter, eftersom det blir lättare att kontrollera vem som får access till dem tabellerna.	ja
<u>Nya krav</u>		
Klasser skapas per program och anläggning		ja
Studenter läser flera kurser och kan ha flera tenta försök		ja
Det skall inte gå att fylla i att en student har hoppat av utan orsak		ja