John Sandsjö

DE24 - Datamodellering

Labb - databas för yrkeshögskolan YrkesCo

Agenda

Background

Business objective

Business requirements

Modelling

- Conceptual model
- Logical model
- Physical model

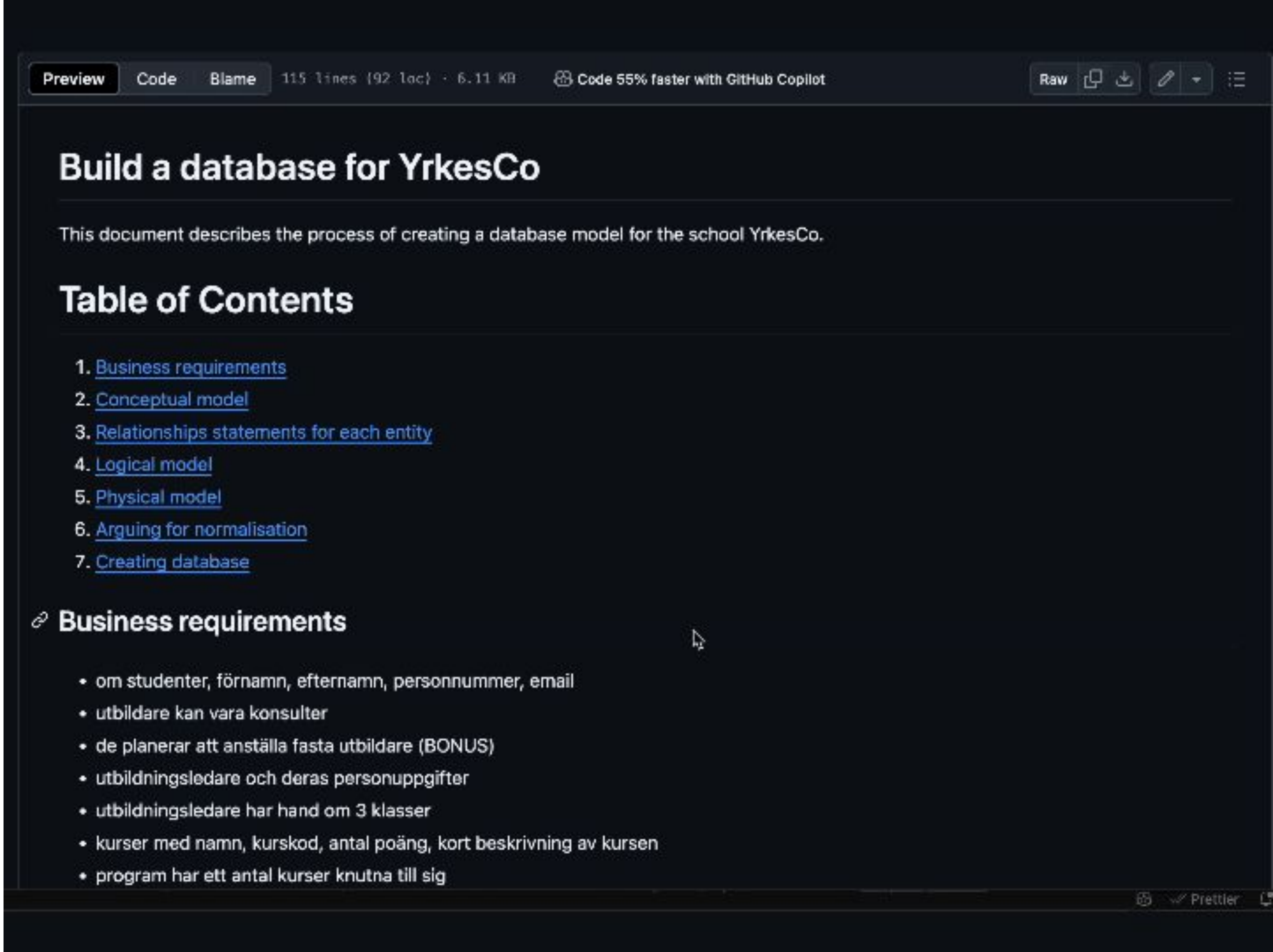
Implementation

- Creating tables
- Adding constraints

The data

Normalisation

Some query results and why they matter to the business


Preview
Code
Blame
115 lines (92 loc) · 6.11 KB
Code 55% faster with GitHub Copilot
Raw
Prettier

Build a database for YrkesCo

This document describes the process of creating a database model for the school YrkesCo.

Table of Contents

Business requirements

- om studenter, förnamn, efternamn, personnummer, email
- utbildare kan vara konsulter
- de planerar att anställa fasta utbildare (BONUS)
- utbildningsledare och deras personuppgifter
- utbildningsledare har hand om 3 klasser
- kurser med namn, kurskod, antal poäng, kort beskrivning av kursen
- program har ett antal kurser knutna till sig

Note

The information in this presentation can also be read in the yh_labb_report.md file in my GitHub repo

Background

YrkesCo is a Swedish school with focus on Data and AI. It currently has two locations, one in Gothenburg and one in Stockholm, with possibly more sites to be open in the future. The schools program are location specific but the school also offer standalone courses to the students that can be taken from both sites.

Business objectives

Build a scalable database for YrkesCo. The model should adhere to future business needs like being able to scaling the operations to more sites as well as adhering to Personal Identifiable Information (PII).

Business requirements, provided in task

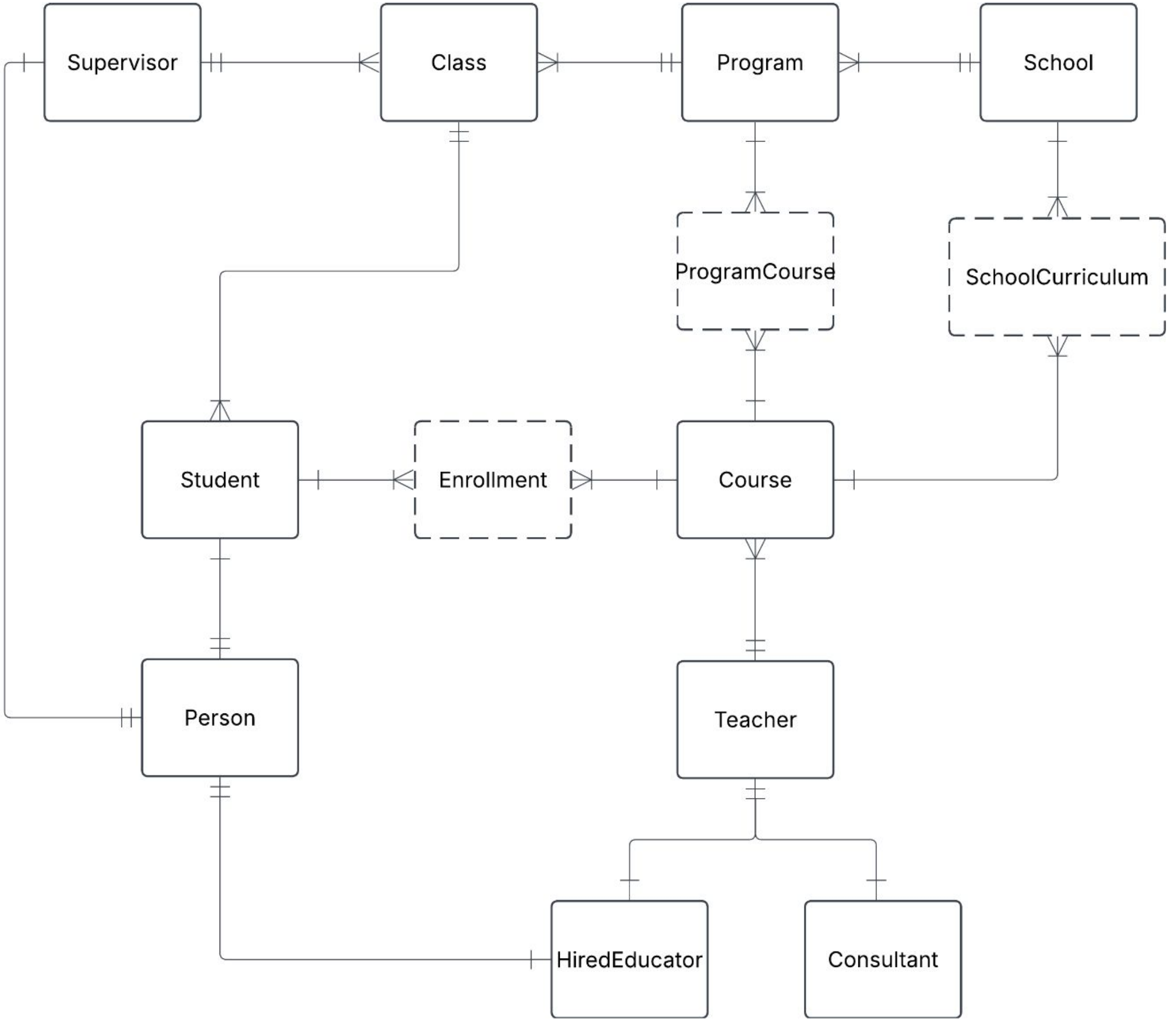
- om studenter, förnamn, efternamn, personnummer, email
- utbildare kan vara konsulter
- de planerar att anställa fasta utbildare (BONUS)
- utbildningsledare och deras personuppgifter
- utbildningsledare har hand om 3 klasser
- kurser med namn, kurskod, antal poäng, kort beskrivning av kursen
- program har ett antal kurser knutna till sig
- ett program blir beviljat i tre omgångar, dvs att det finns 3 klasser
- det finns även fristående kurser (BONUS)
- konsulter, deras företag, företagsinfo som organisationsnummer, har F-skatt address, hur mycket de tar i arvode per timma
- YrkesCo har två anläggningar, en i göteborg och en i stockholm, i framtiden kanske de kommer expandera till flera orter (BONUS)

Additional business requirements

- A course can have only one teacher
- Standalone course can be taken online by students in any of the schools. But courses are only open for students enrolled in the school.
- A program belongs to one school site
- A course can belong to many programs (e.g. Python fundamentals)

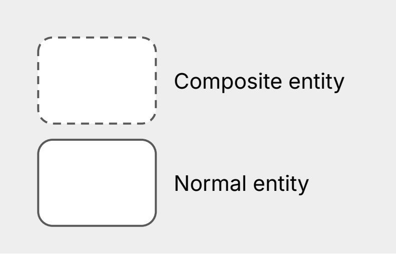
Conceptual model

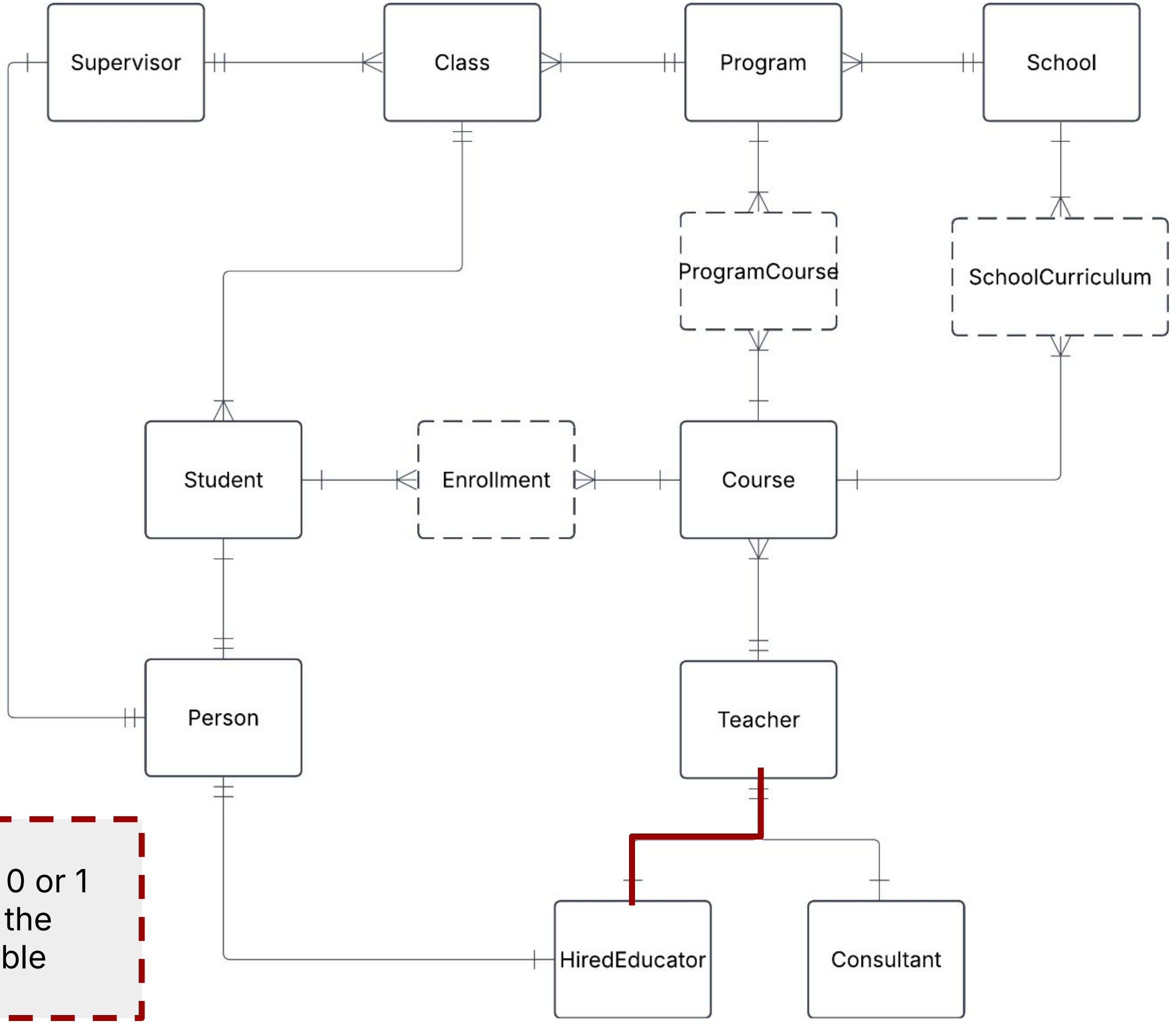
- ▯ Composite entity
- ▢ Normal entity



Supervisor
Class
Program
School
ProgramCourse
SchoolCurriculum
Student
Enrollment
Course
Person
Teacher
HiredEducator
Consultant

Conceptual model



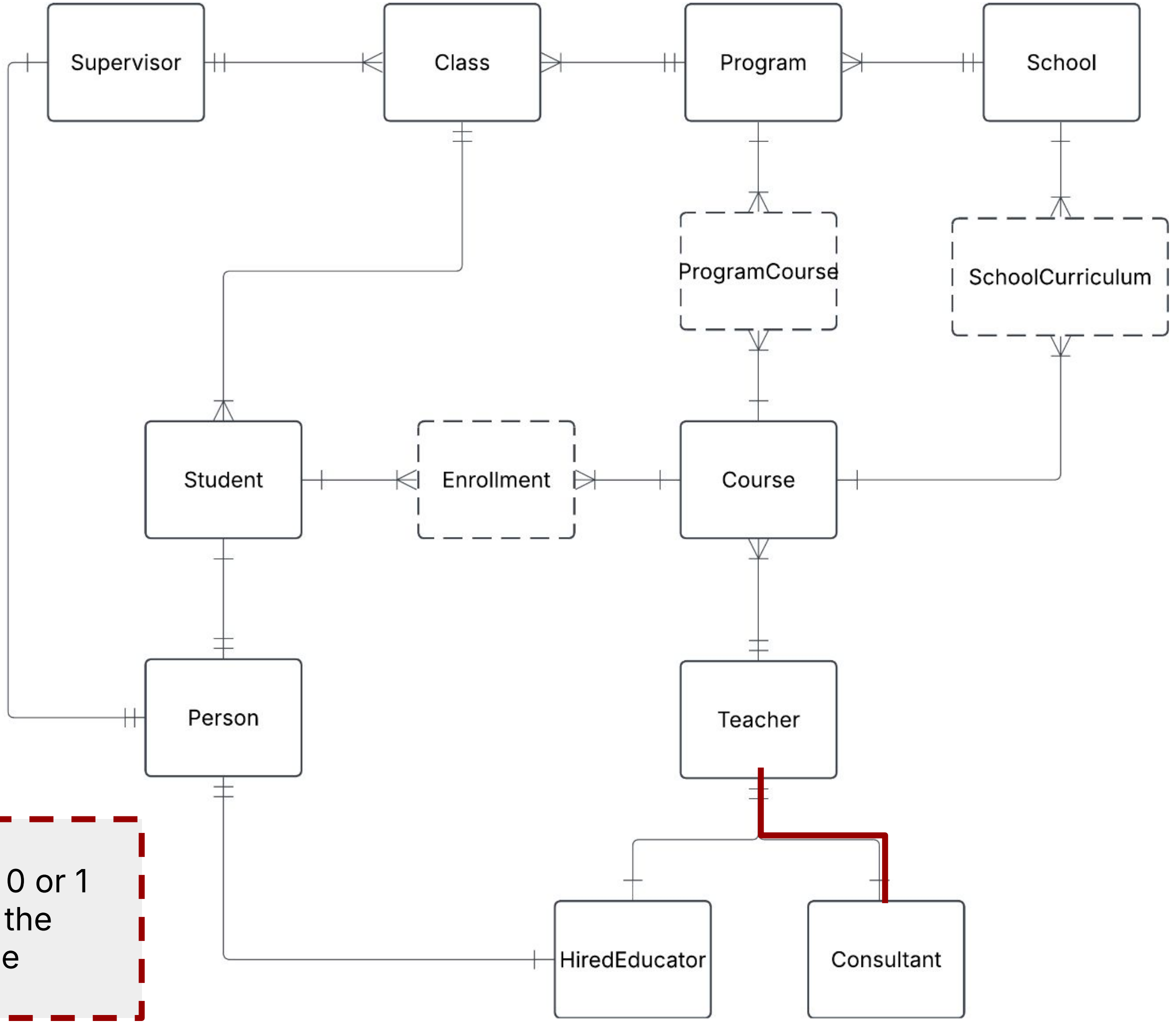


A teacher can have 0 or 1 representation in the HiredEducator table

Conceptual model

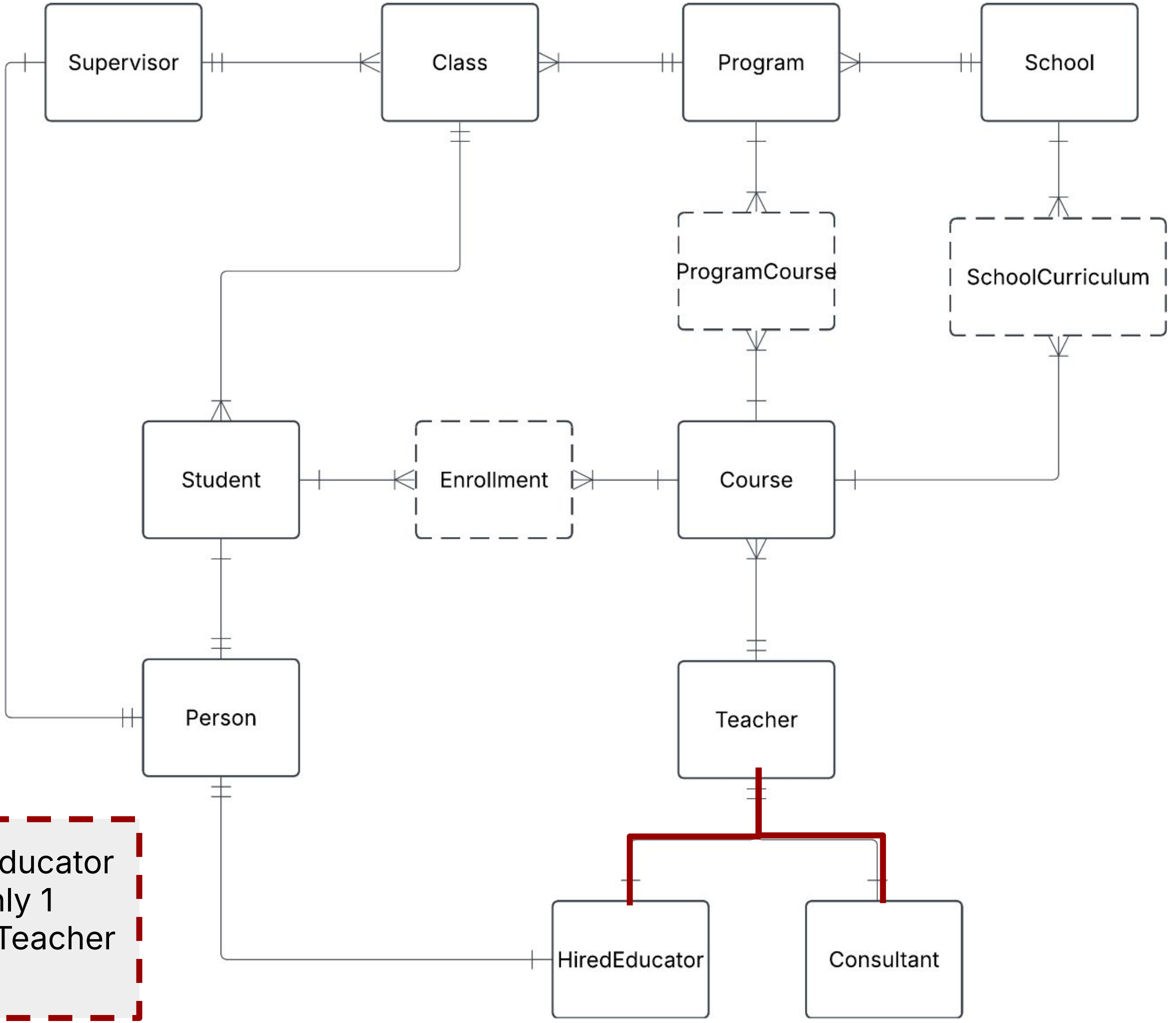
Composite entity

Normal entity



A teacher can have 0 or 1 representation in the Consultant table

Conceptual model



⬚ Composite entity

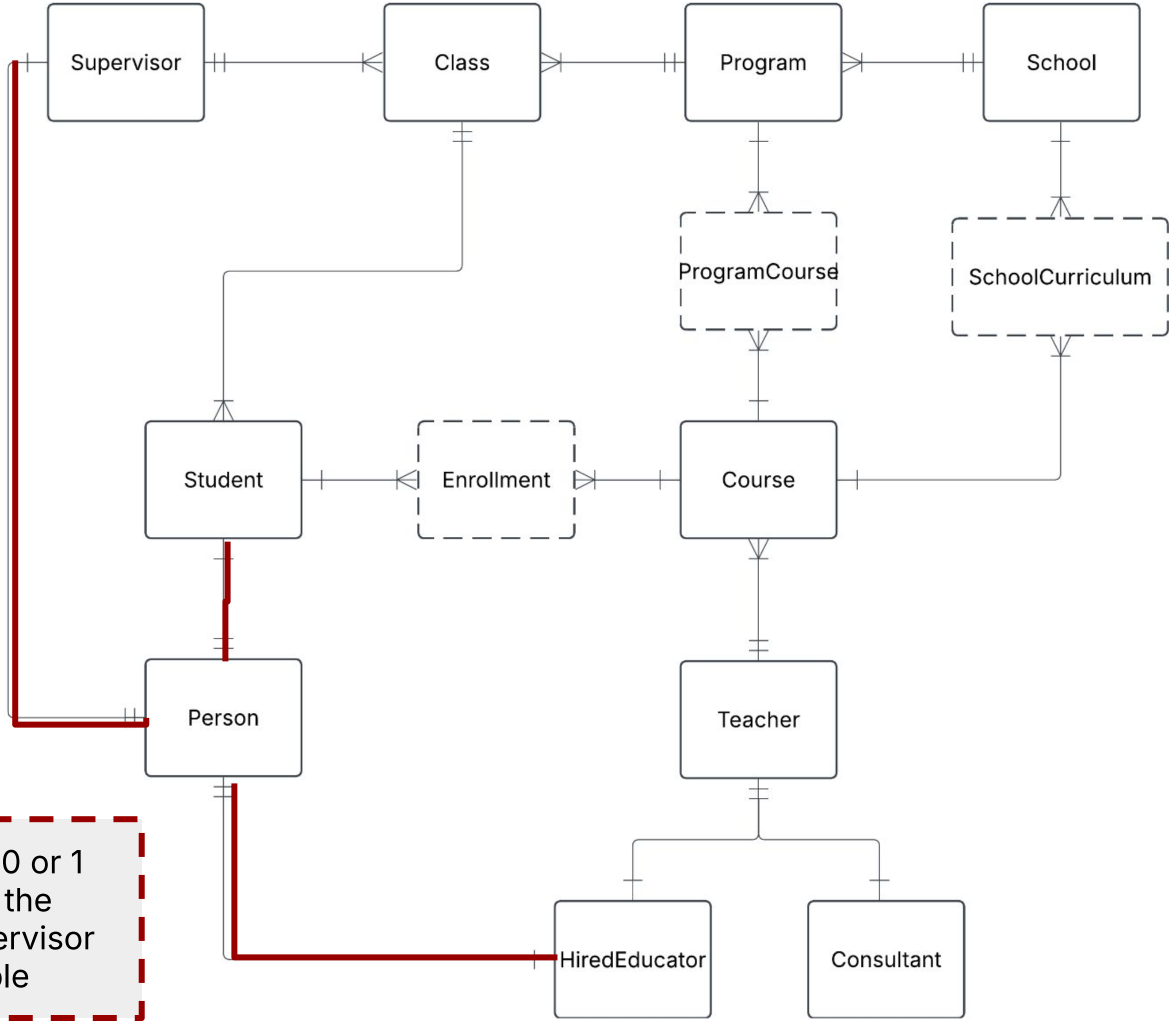
▢ Normal entity

A Consultant / HiredEducator can have 1 and only 1 representation in the Teacher table

Conceptual model

⬚ Composite entity

▭ Normal entity

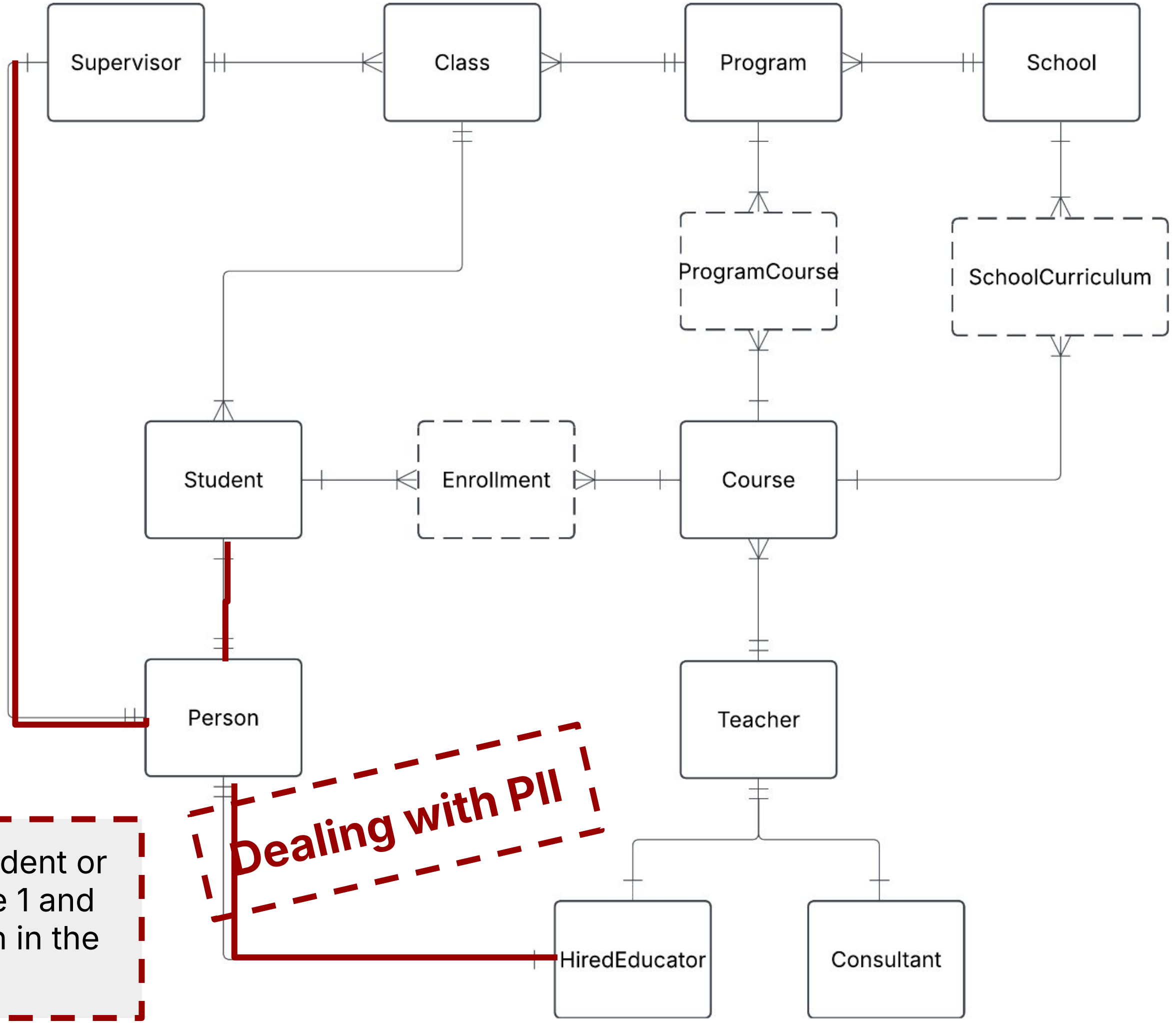


A Person can have 0 or 1 representation in the HiredEducator, Supervisor and Student table

Conceptual model

▭ Composite entity

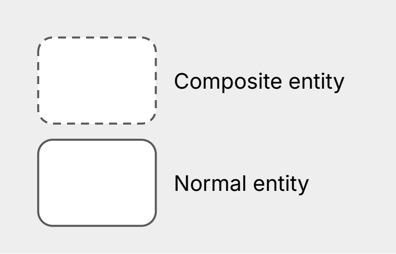
▢ Normal entity

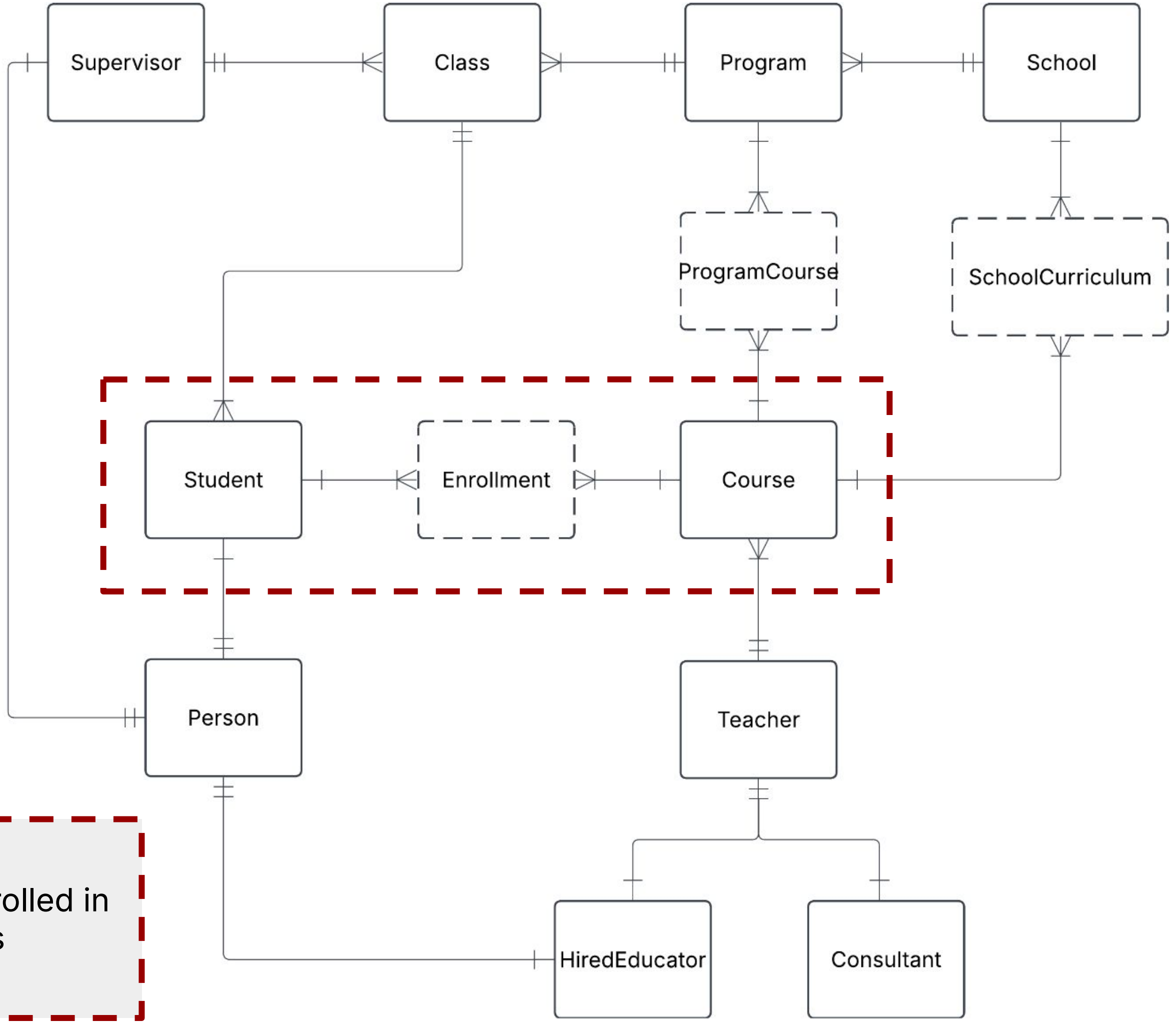


Dealing with PII

A HiredEducator, Student or Supervisor can have 1 and only 1 representation in the Person table

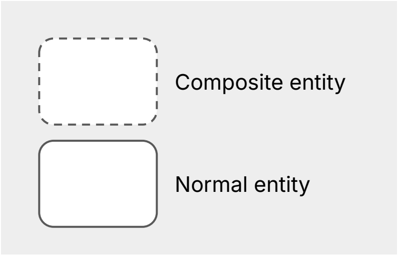
Conceptual model

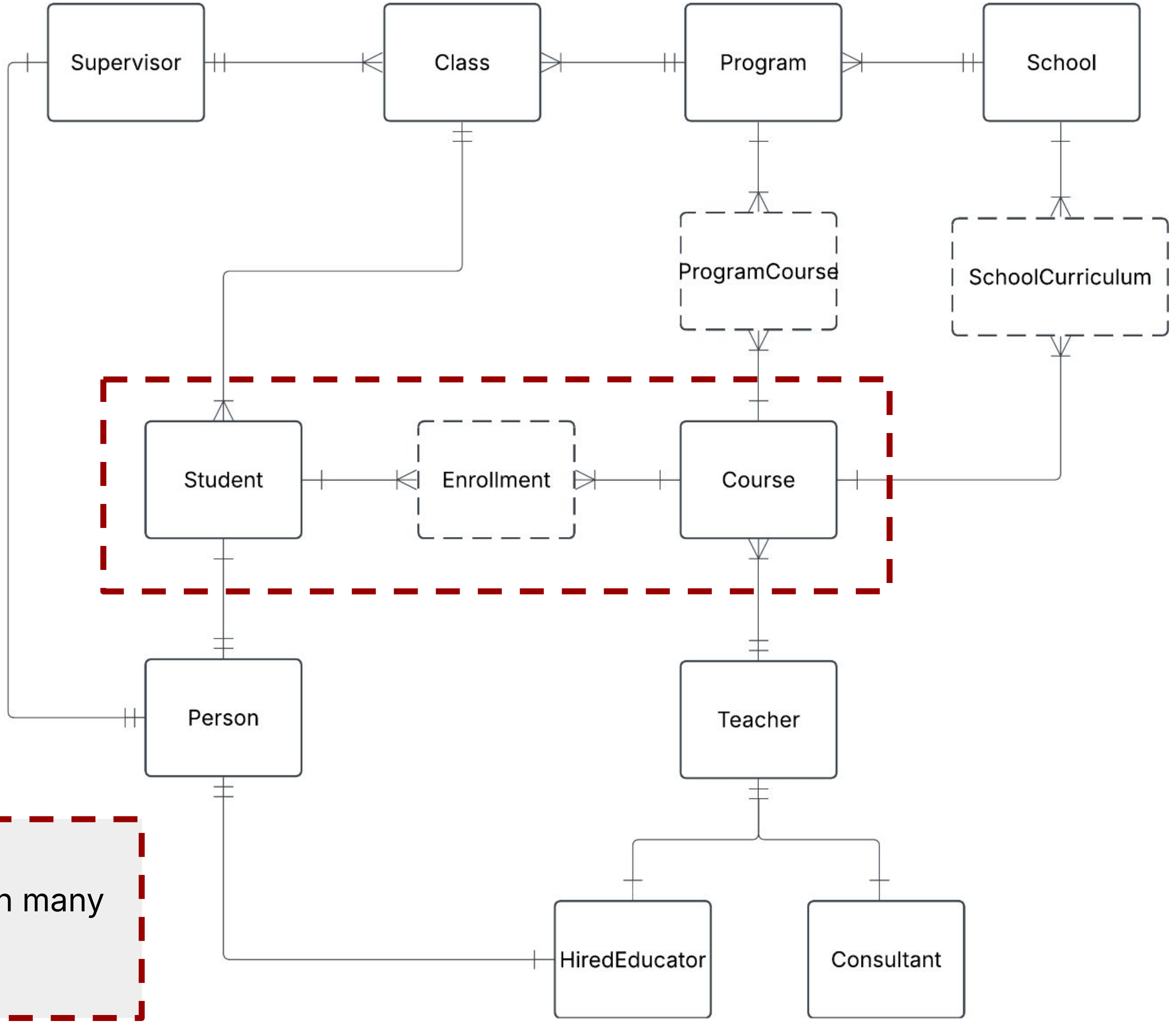




A Student can be enrolled in many Courses

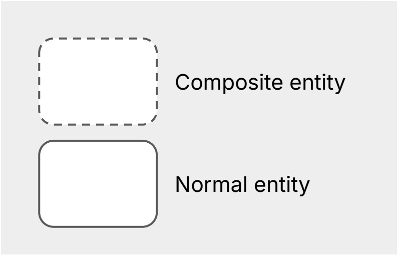
Conceptual model

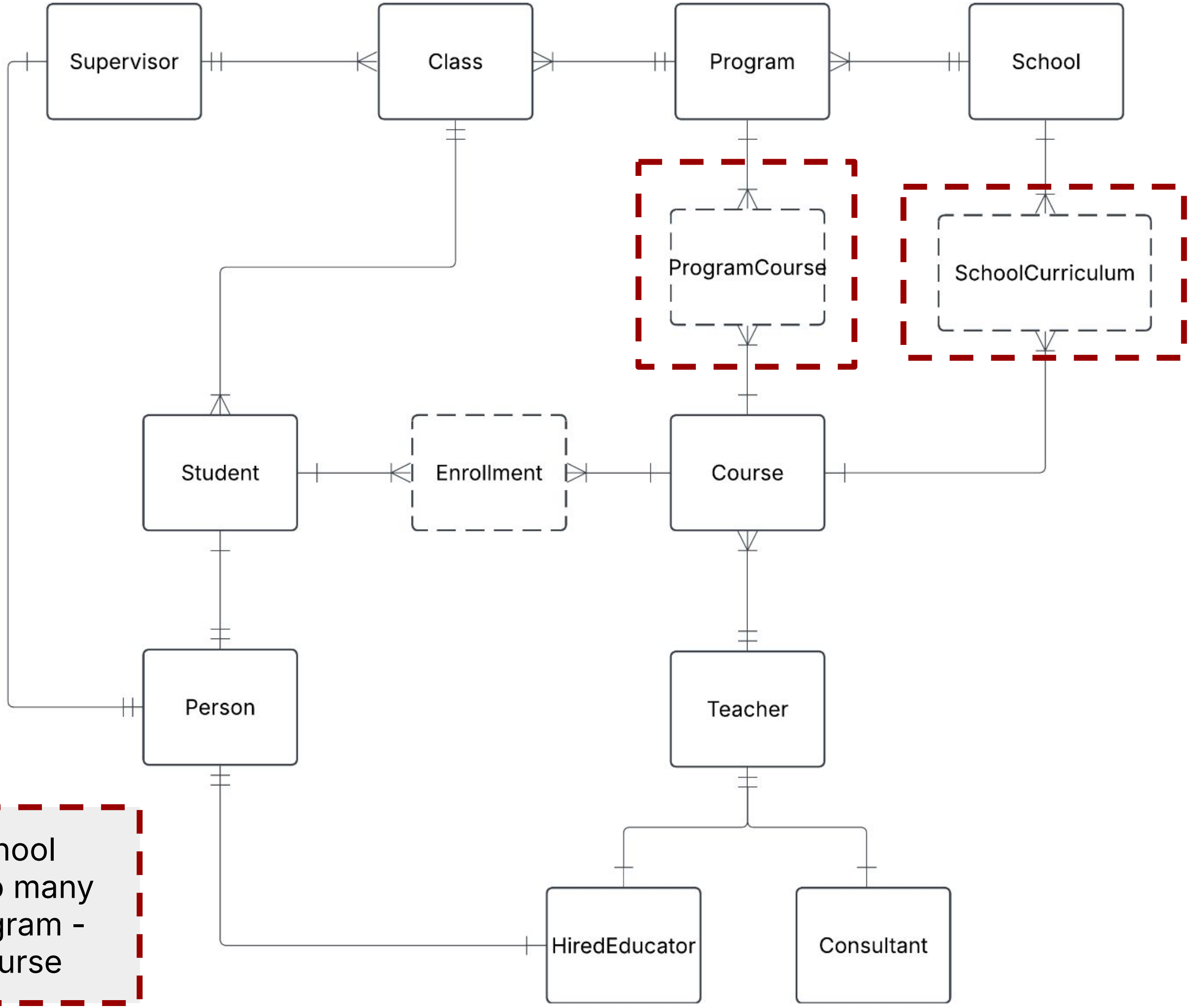




A Course can contain many Students

Conceptual model



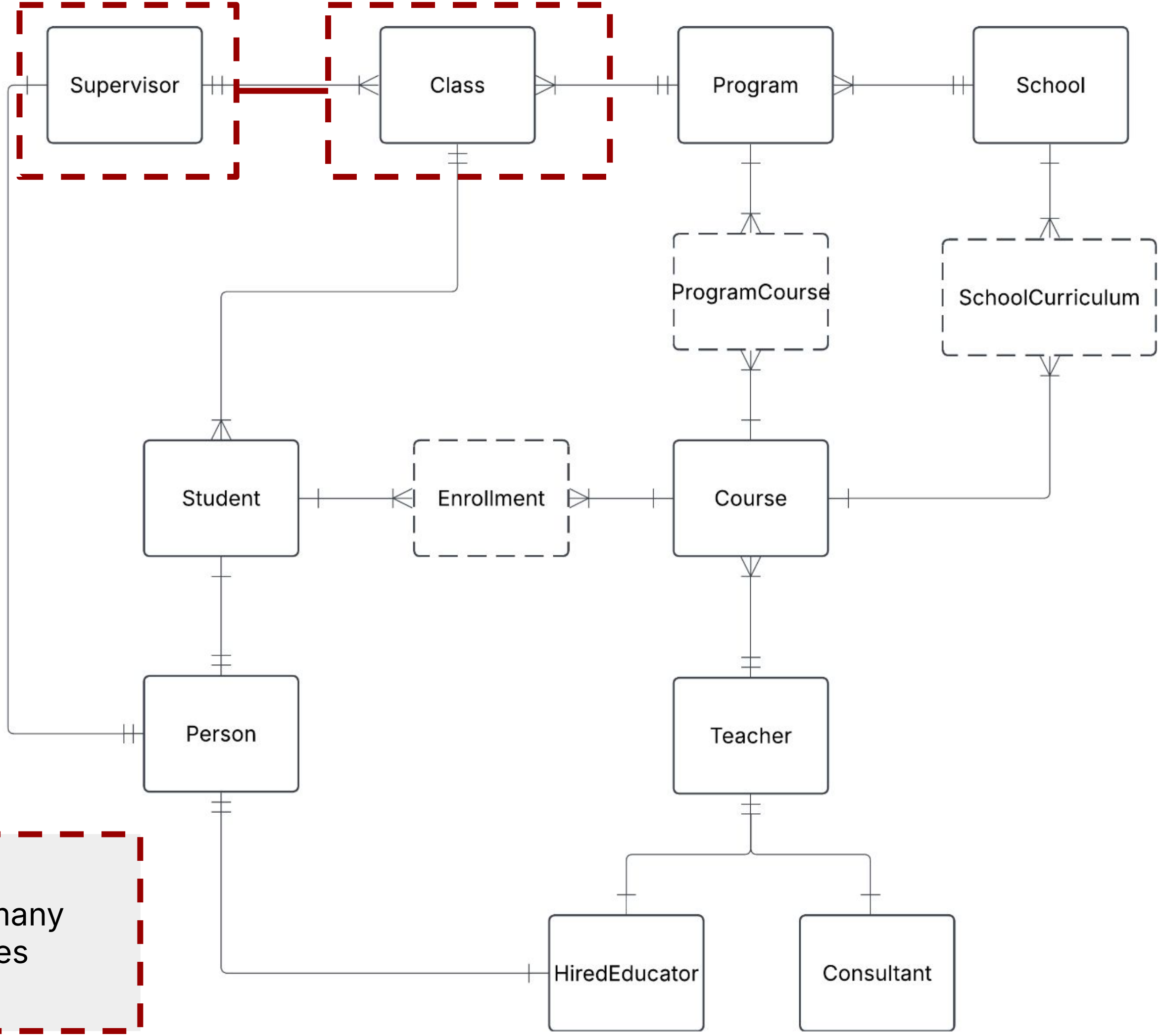


ProgramCourse and School Curriculum solves many to many relationship between Program - Course and School - Course

Conceptual model

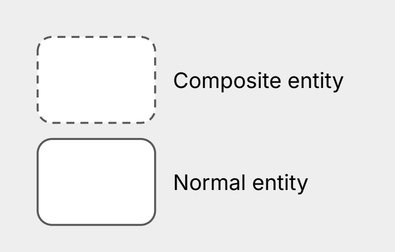
Composite entity

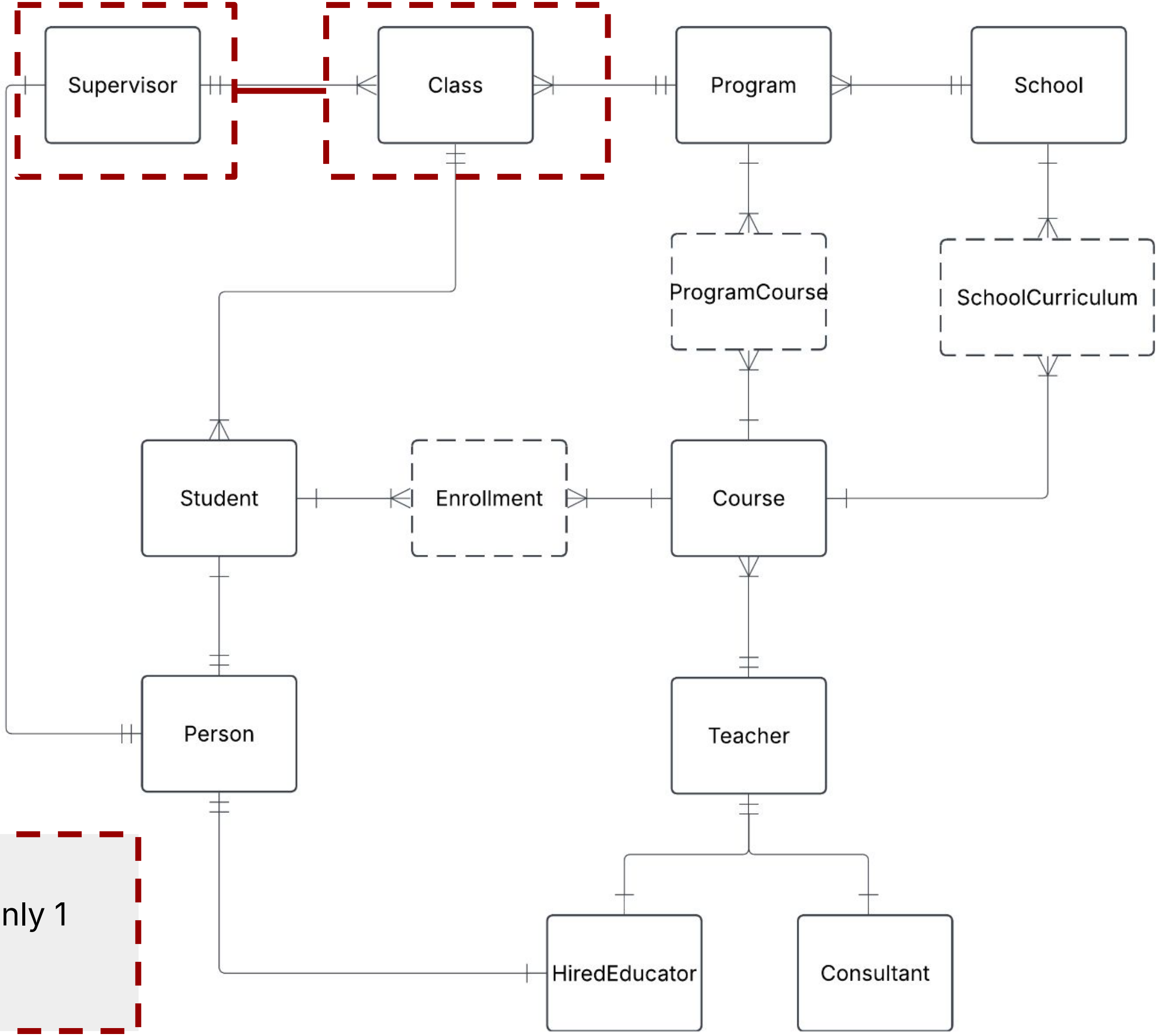
Normal entity



A Supervisor can have many (maximum of 3) classes

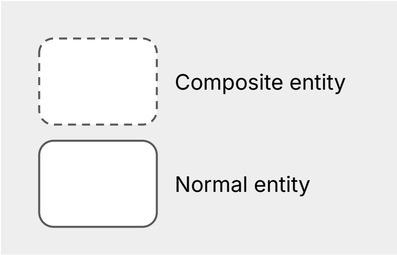
Conceptual model

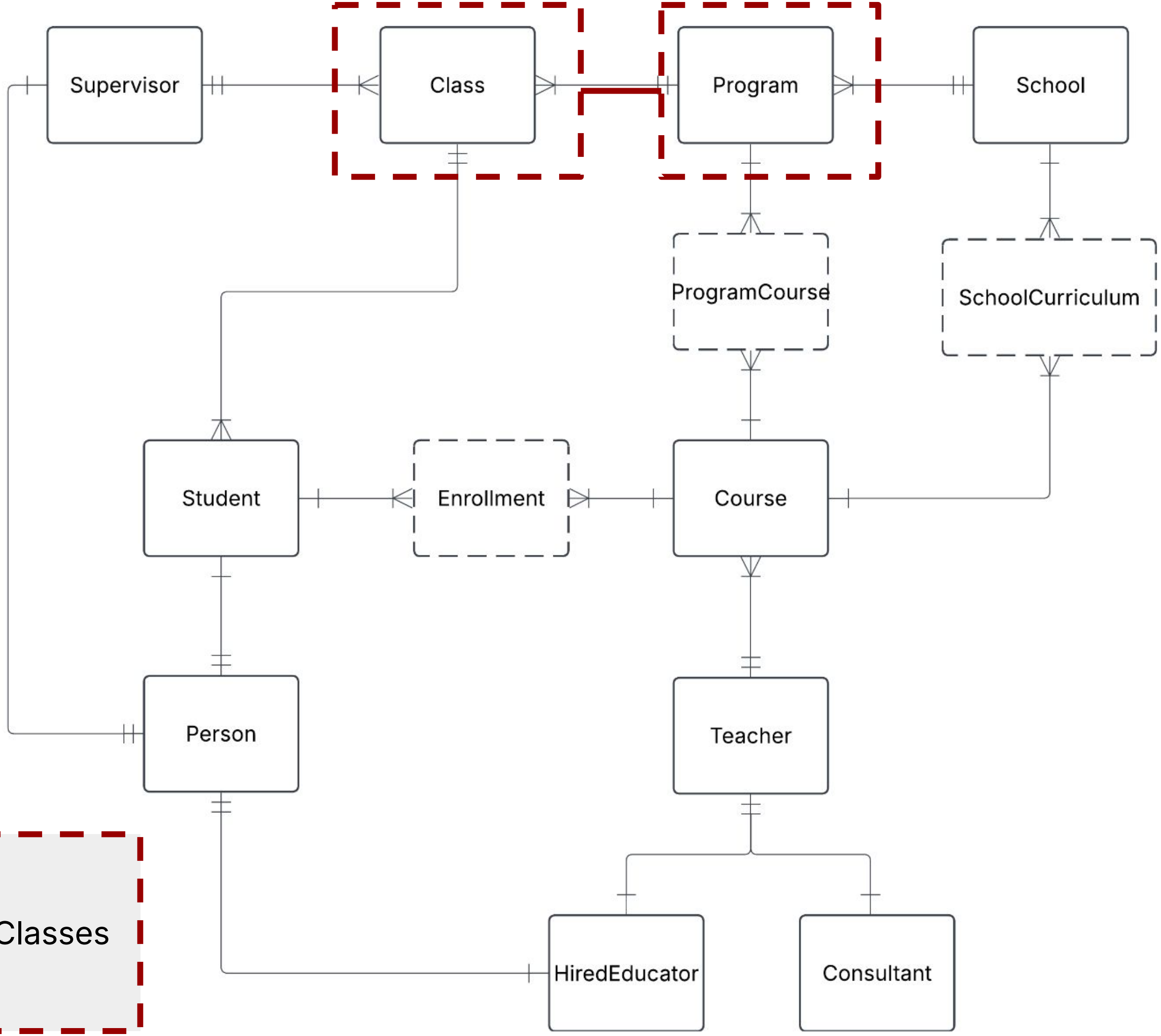




A Class can have 1 and only 1 Supervisor

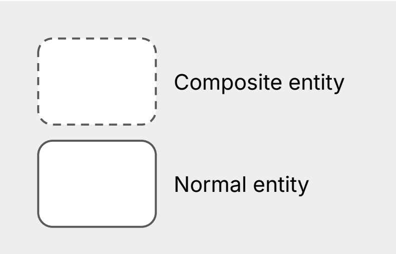
Conceptual model

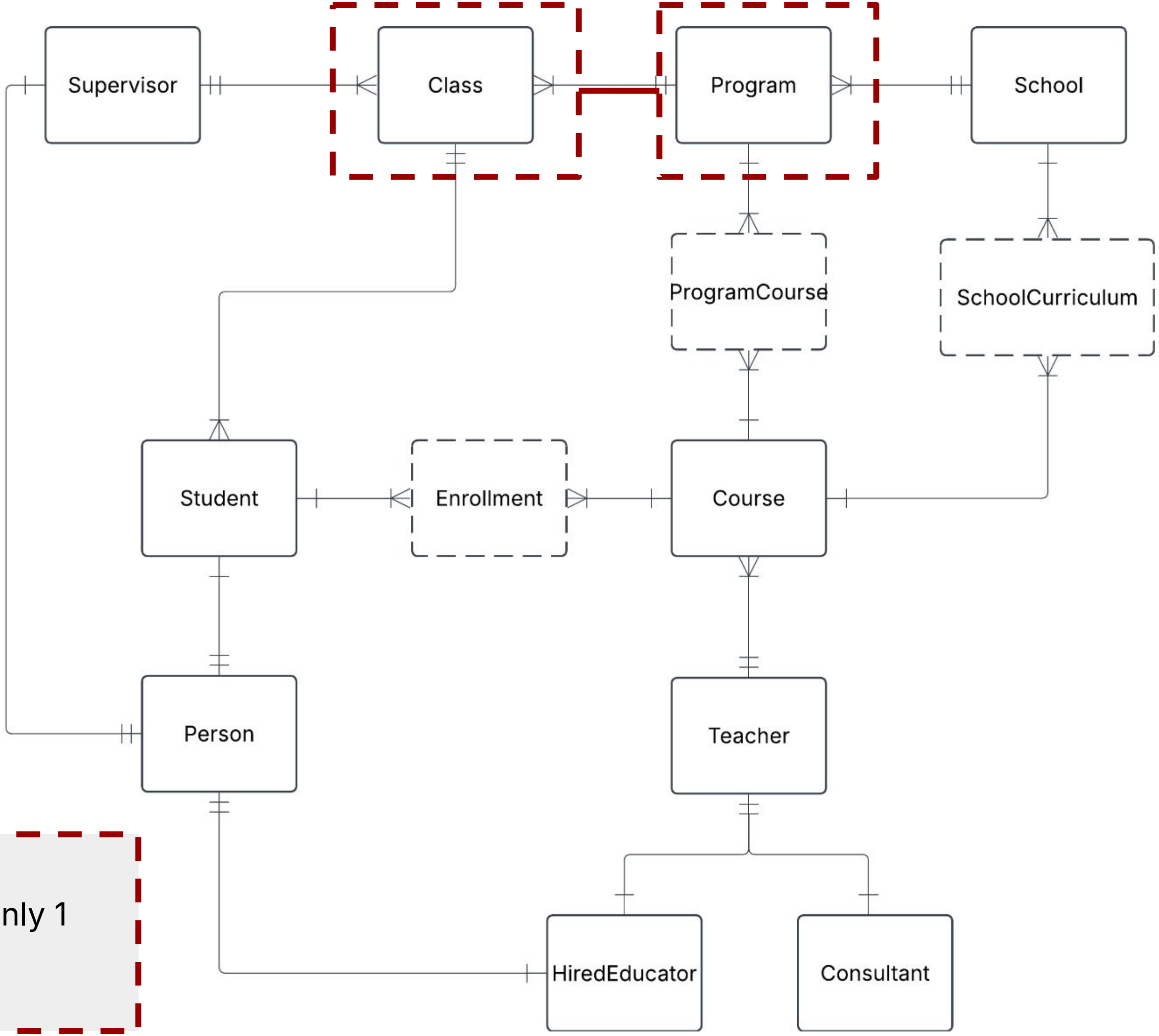




A Program can have many Classes

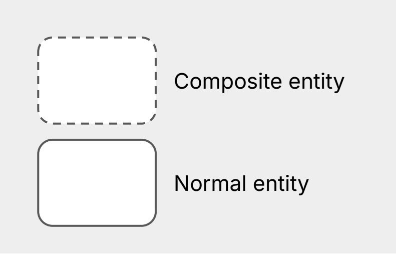
Conceptual model

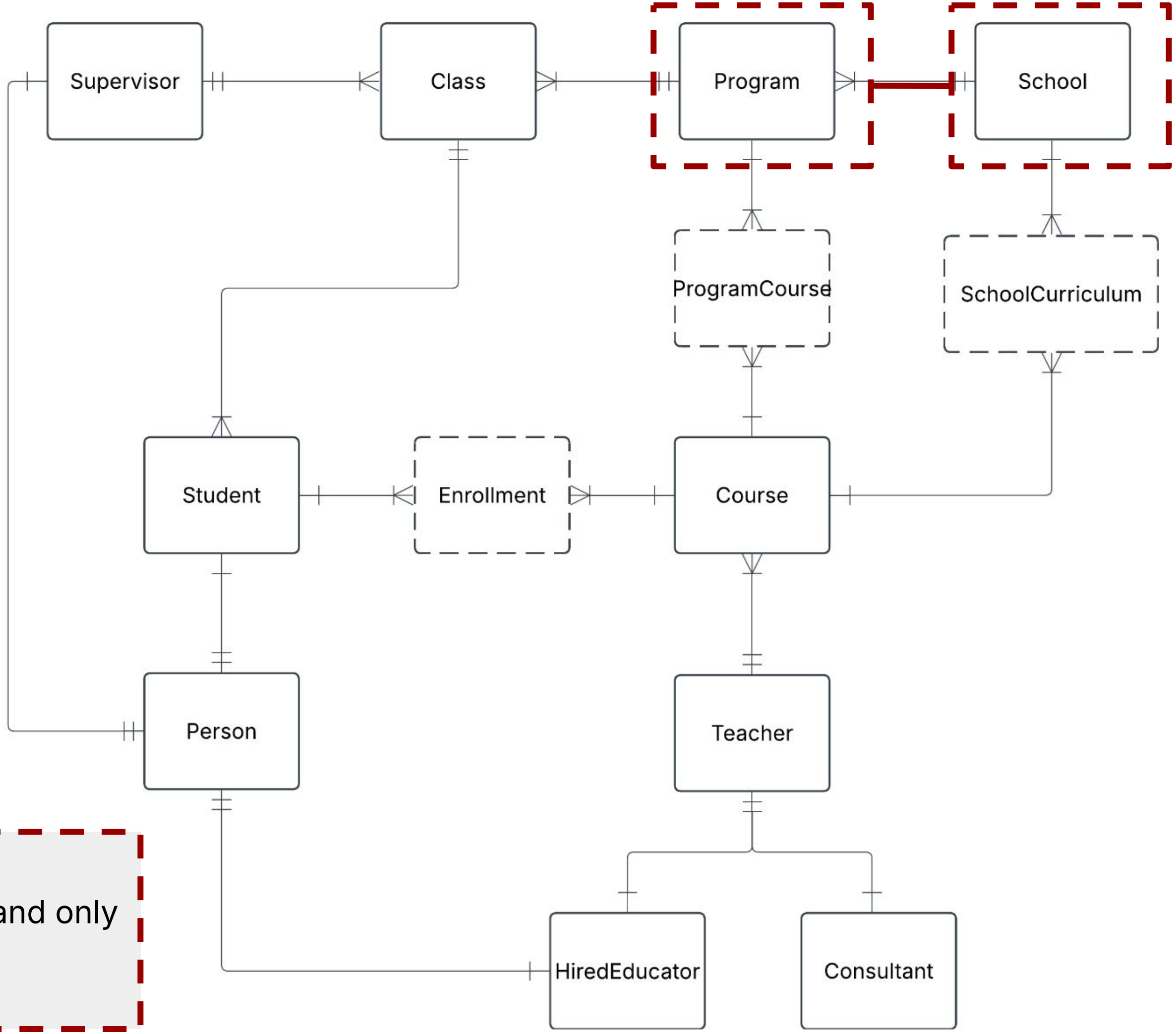




A Class can have 1 and only 1 Program

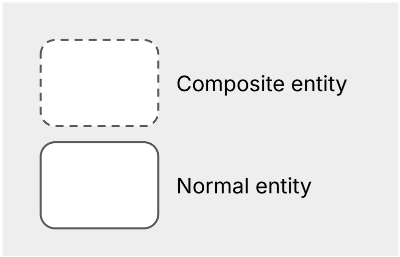
Conceptual model

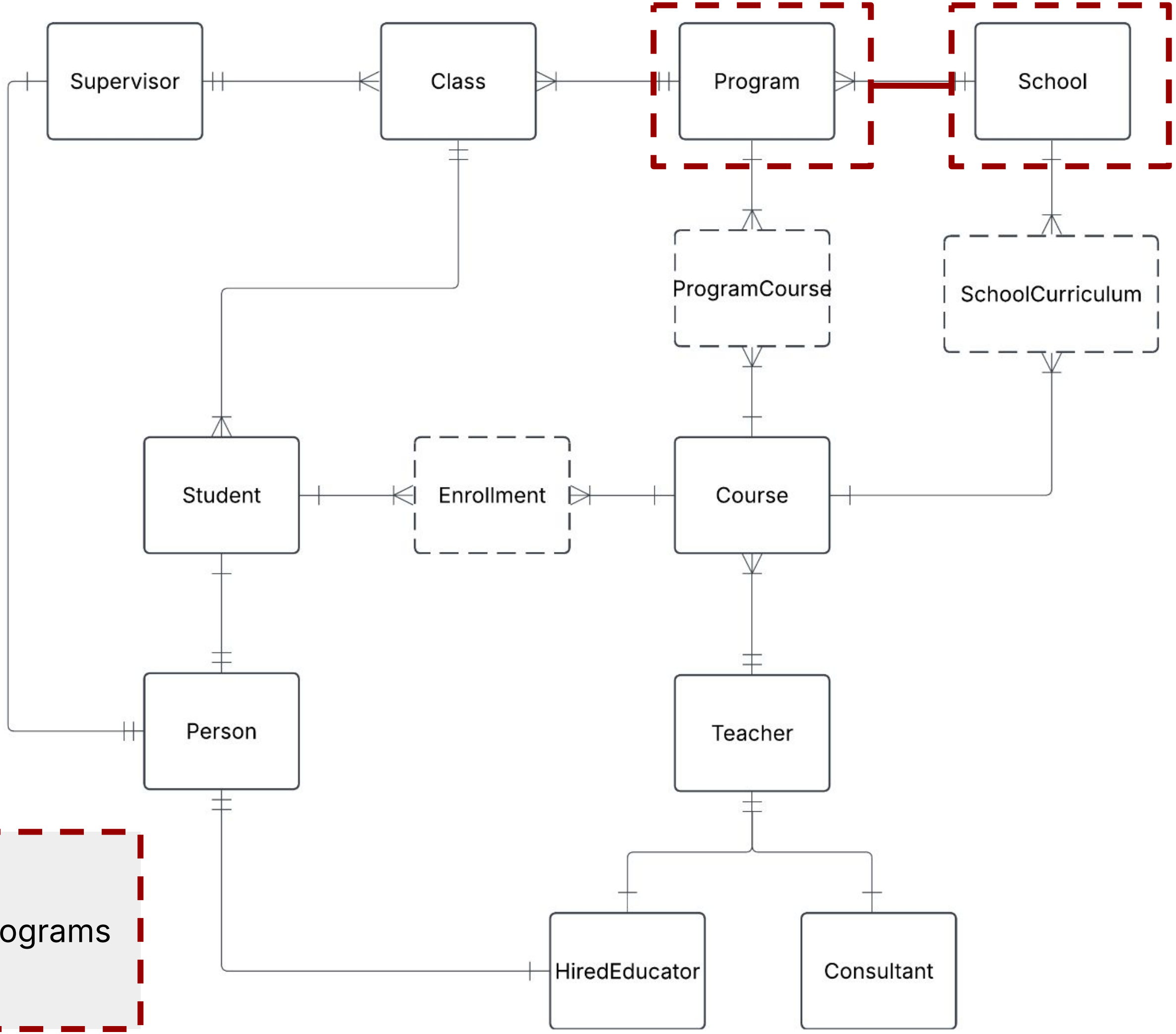




A Program can be held in 1 and only 1 School

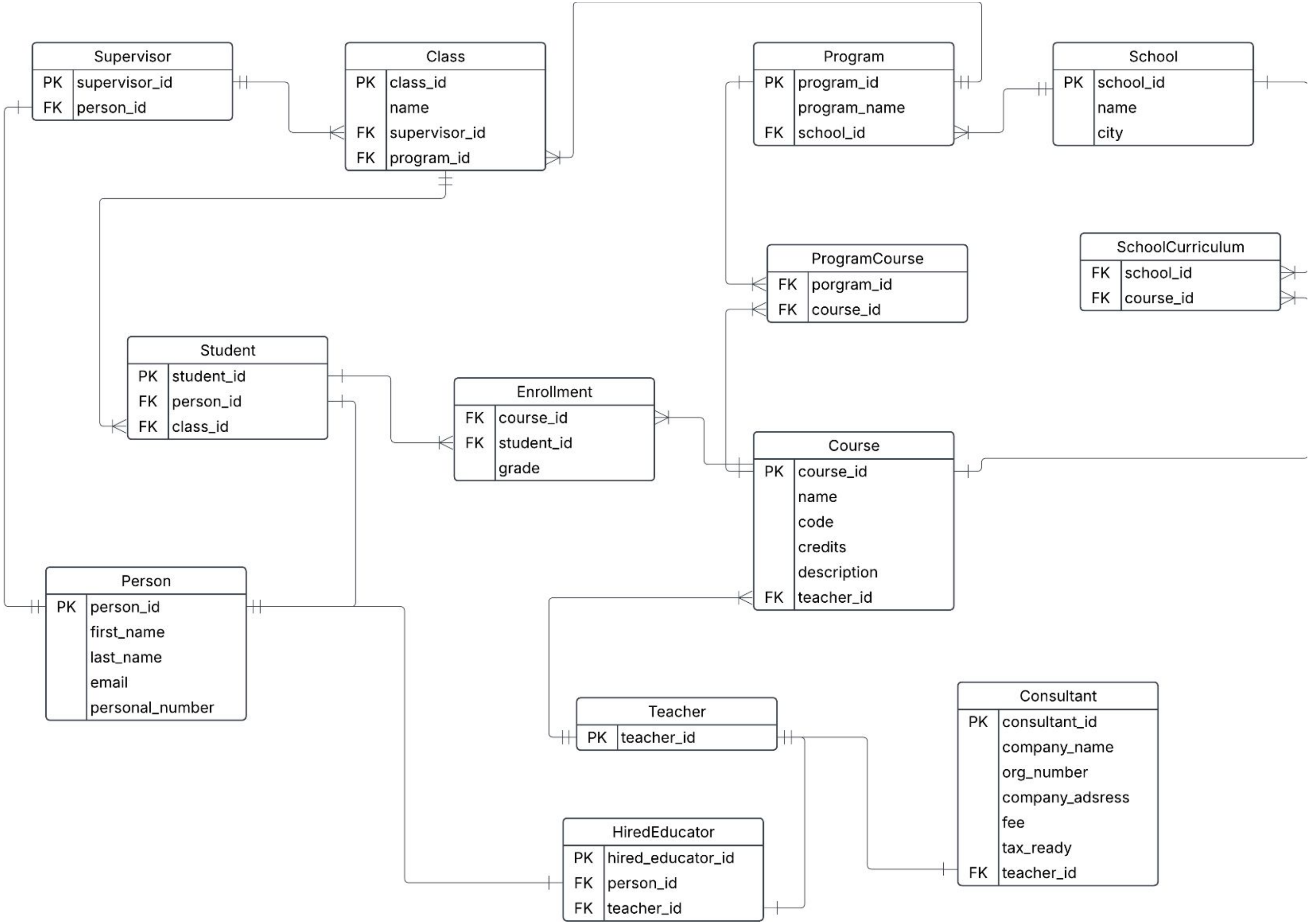
Conceptual model





A School can have many Programs

Logical model



	Supervisor
PK	supervisor_id
FK	person_id

	Class
PK	class_id
	name
FK	supervisor_id
FK	program_id

	Program
PK	program_id
	program_name
FK	school_id

	School
PK	school_id
	name
	city

	ProgramCourse
FK	porgram_id
FK	course_id

	SchoolCurriculum
FK	school_id
FK	course_id

	Student
PK	student_id
FK	person_id
FK	class_id

	Enrollment
FK	course_id
FK	student_id
	grade

	Course
PK	course_id
	name
	code
	credits
	description
FK	teacher_id

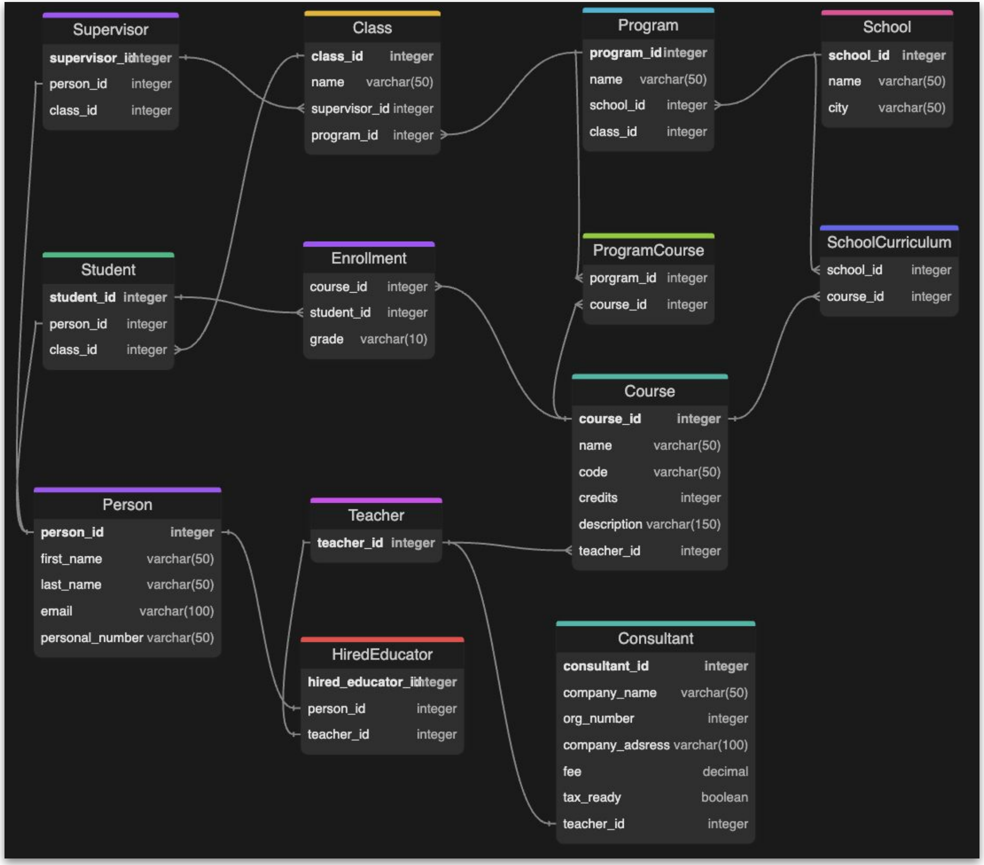
	Person
PK	person_id
	first_name
	last_name
	email
	personal_number

	Teacher
PK	teacher_id

	Consultant
PK	consultant_id
	company_name
	org_number
	company_adsress
	fee
	tax_ready
FK	teacher_id

	HiredEducator
PK	hired_educator_id
FK	person_id
FK	teacher_id

Physical model



Supervisor	
supervisor_id	integer
person_id	integer
class_id	integer

Class	
class_id	integer
name	varchar(50)
supervisor_id	integer
program_id	integer

Program	
program_id	integer
name	varchar(50)
school_id	integer
class_id	integer

School	
school_id	integer
name	varchar(50)
city	varchar(50)

Student	
student_id	integer
person_id	integer
class_id	integer

Enrollment	
course_id	integer
student_id	integer
grade	varchar(10)

ProgramCourse	
porgram_id	integer
course_id	integer

SchoolCurriculum	
school_id	integer
course_id	integer

Course	
course_id	integer
name	varchar(50)
code	varchar(50)
credits	integer
description	varchar(150)
teacher_id	integer

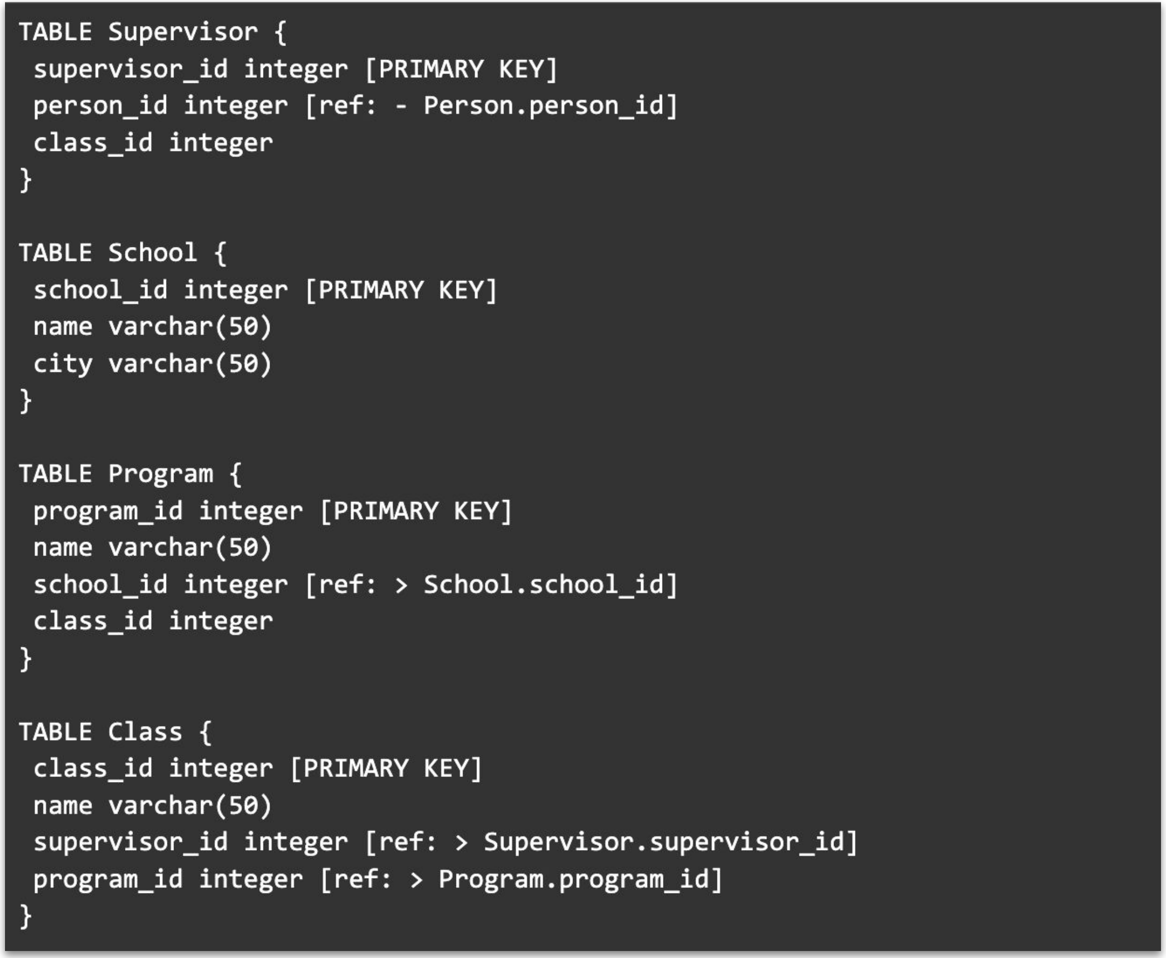
Person	
person_id	integer
first_name	varchar(50)
last_name	varchar(50)
email	varchar(100)
personal_number	varchar(50)

Teacher	
teacher_id	integer

HiredEducator	
hired_educator_id	integer
person_id	integer
teacher_id	integer

Consultant	
consultant_id	integer
company_name	varchar(50)
org_number	integer
company_adsress	varchar(100)
fee	decimal
tax_ready	boolean
teacher_id	integer

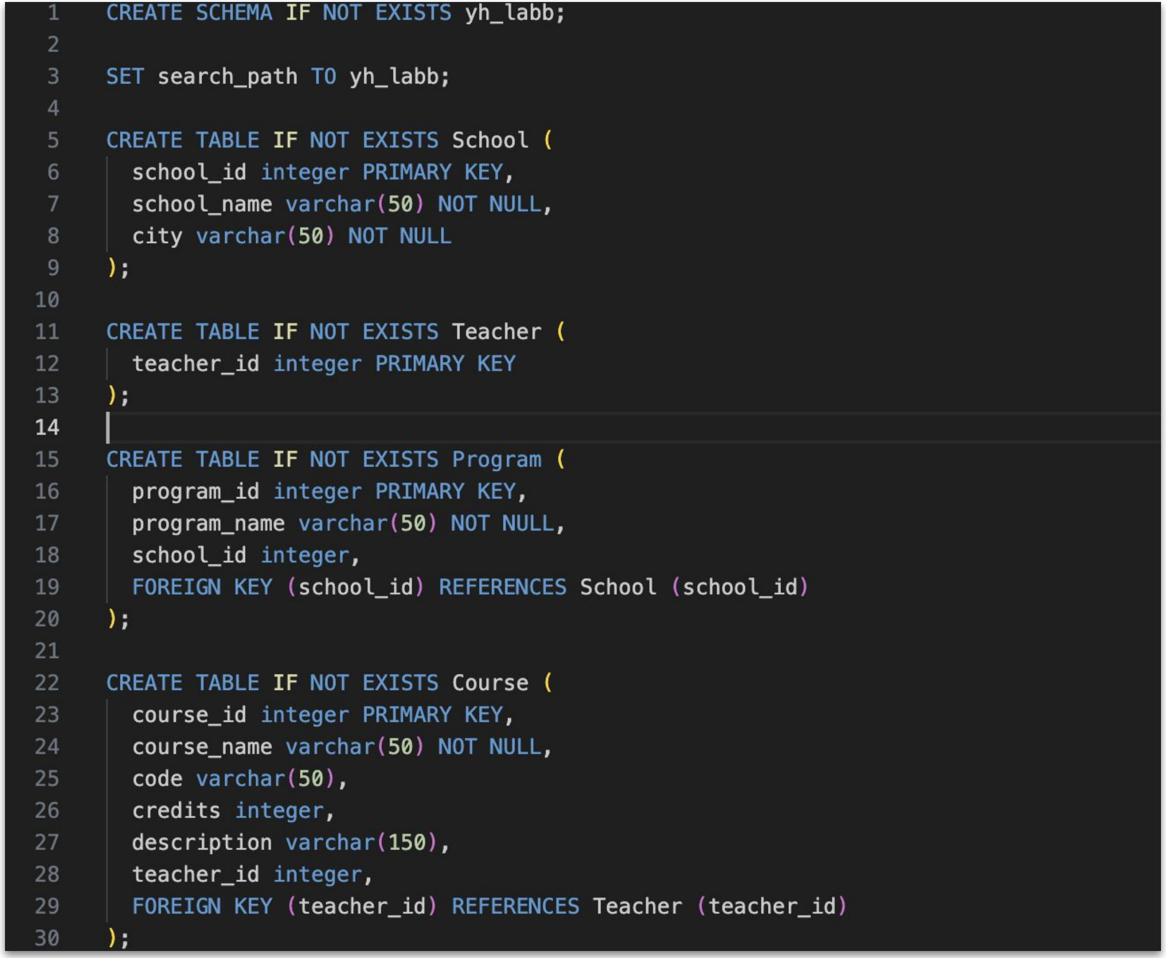
Physical model



```
TABLE Supervisor {
 supervisor_id integer [PRIMARY KEY]
 person_id integer [ref: - Person.person_id]
 class_id integer
}

TABLE School {
 school_id integer [PRIMARY KEY]
 name varchar(50)
 city varchar(50)
}

TABLE Program {
 program_id integer [PRIMARY KEY]
 name varchar(50)
 school_id integer [ref: > School.school_id]
 class_id integer
}

TABLE Class {
 class_id integer [PRIMARY KEY]
 name varchar(50)
 supervisor_id integer [ref: > Supervisor.supervisor_id]
 program_id integer [ref: > Program.program_id]
}
```

Implementation

Creating tables

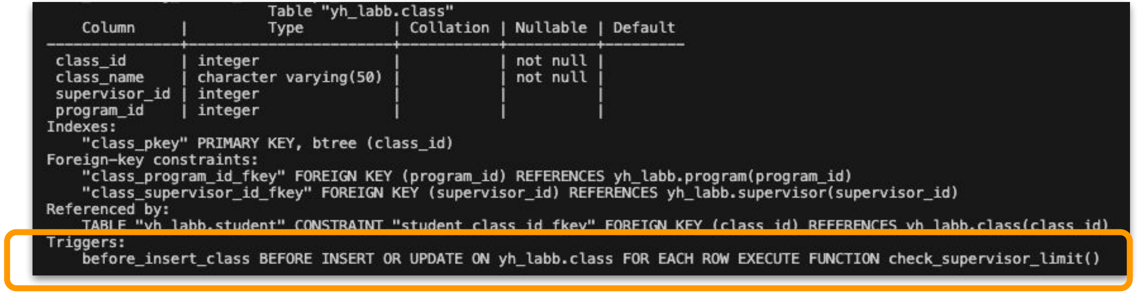


```sql
CREATE SCHEMA IF NOT EXISTS yh_labb;

SET search_path TO yh_labb;

CREATE TABLE IF NOT EXISTS School (
  school_id integer PRIMARY KEY,
  school_name varchar(50) NOT NULL,
  city varchar(50) NOT NULL
);

CREATE TABLE IF NOT EXISTS Teacher (
  teacher_id integer PRIMARY KEY
);

CREATE TABLE IF NOT EXISTS Program (
  program_id integer PRIMARY KEY,
  program_name varchar(50) NOT NULL,
  school_id integer,
  FOREIGN KEY (school_id) REFERENCES School (school_id)
);

CREATE TABLE IF NOT EXISTS Course (
  course_id integer PRIMARY KEY,
  course_name varchar(50) NOT NULL,
  code varchar(50),
  credits integer,
  description varchar(150),
  teacher_id integer,
  FOREIGN KEY (teacher_id) REFERENCES Teacher (teacher_id)
);
```

Implementation

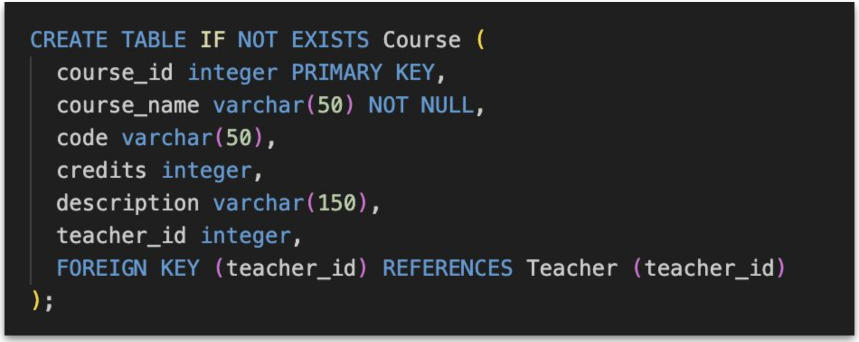
Adding constraints

Trigger



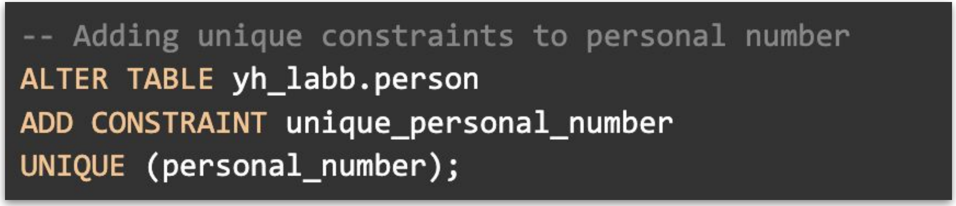
```
                         Table "yh_labb.class"
    Column     |          Type          | Collation | Nullable | Default
---------------+------------------------+-----------+----------+---------
 class_id      | integer                |           | not null |
 class_name    | character varying(50)  |           | not null |
 supervisor_id | integer                |           |          |
 program_id    | integer                |           |          |
Indexes:
    "class_pkey" PRIMARY KEY, btree (class_id)
Foreign-key constraints:
    "class_program_id_fkey" FOREIGN KEY (program_id) REFERENCES yh_labb.program(program_id)
    "class_supervisor_id_fkey" FOREIGN KEY (supervisor_id) REFERENCES yh_labb.supervisor(supervisor_id)
Referenced by:
    TABLE "yh_labb.student" CONSTRAINT "student_class_id_fkey" FOREIGN KEY (class_id) REFERENCES yh_labb.class(class_id)
Triggers:
    before_insert_class BEFORE INSERT OR UPDATE ON yh_labb.class FOR EACH ROW EXECUTE FUNCTION check_supervisor_limit()
```

NOT NULL

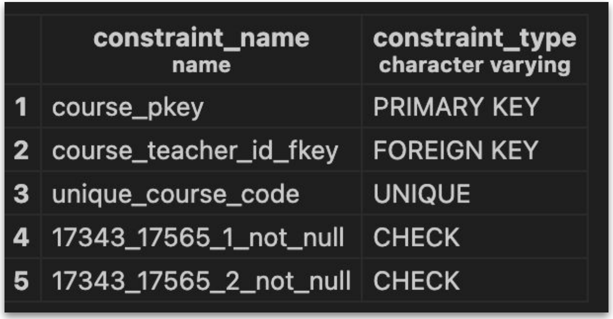


```
CREATE TABLE IF NOT EXISTS Course (
  course_id integer PRIMARY KEY,
  course_name varchar(50) NOT NULL,
  code varchar(50),
  credits integer,
  description varchar(150),
  teacher_id integer,
  FOREIGN KEY (teacher_id) REFERENCES Teacher (teacher_id)
);
```

UNIQUE



```
-- Adding unique constraints to personal number
ALTER TABLE yh_labb.person
ADD CONSTRAINT unique_personal_number
UNIQUE (personal_number);
```



	constraint_name	constraint_type
	name	character varying
1	course_pkey	PRIMARY KEY
2	course_teacher_id_fkey	FOREIGN KEY
3	unique_course_code	UNIQUE
4	17343_17565_1_not_null	CHECK
5	17343_17565_2_not_null	CHECK

The data

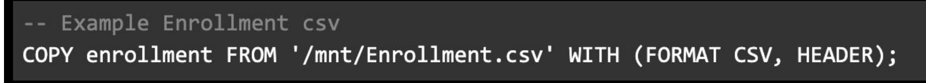
- Dummy data generated in LLM ingested with the purpose to test the relationships and joins
- Methodology for data ingestion

Prompt:
I am modelling a database for a school. See below my sql statement for creating my tables and its cardinalities. I want you to create dummy data in csv's for each of these tables? The dummy data should have the theme Data and AI and courses could be things like AI, Analytics and other development courses. The course names should have a fun twist to the course names. The School's name is YrkesCo and have one location in Stockholm, called "YrkesCo Liljeholmen" and one location in Gothenburg, called "YrkesCo Lindholmen". The student and teachers name should be Swedish sounding names, the first letter in the first_name should be the same as the first letter in the last_name.

The data

- Dummy data generated in LLM ingested with the purpose to test the relationships and joins
- Methodology for data ingestion

- Used Gemini to generate dummy data in csv's for each Table
- Added the csv's to my local machine
- Created a bind mount in docker compose with source folder from bullet above and target folder in mnt folder in container



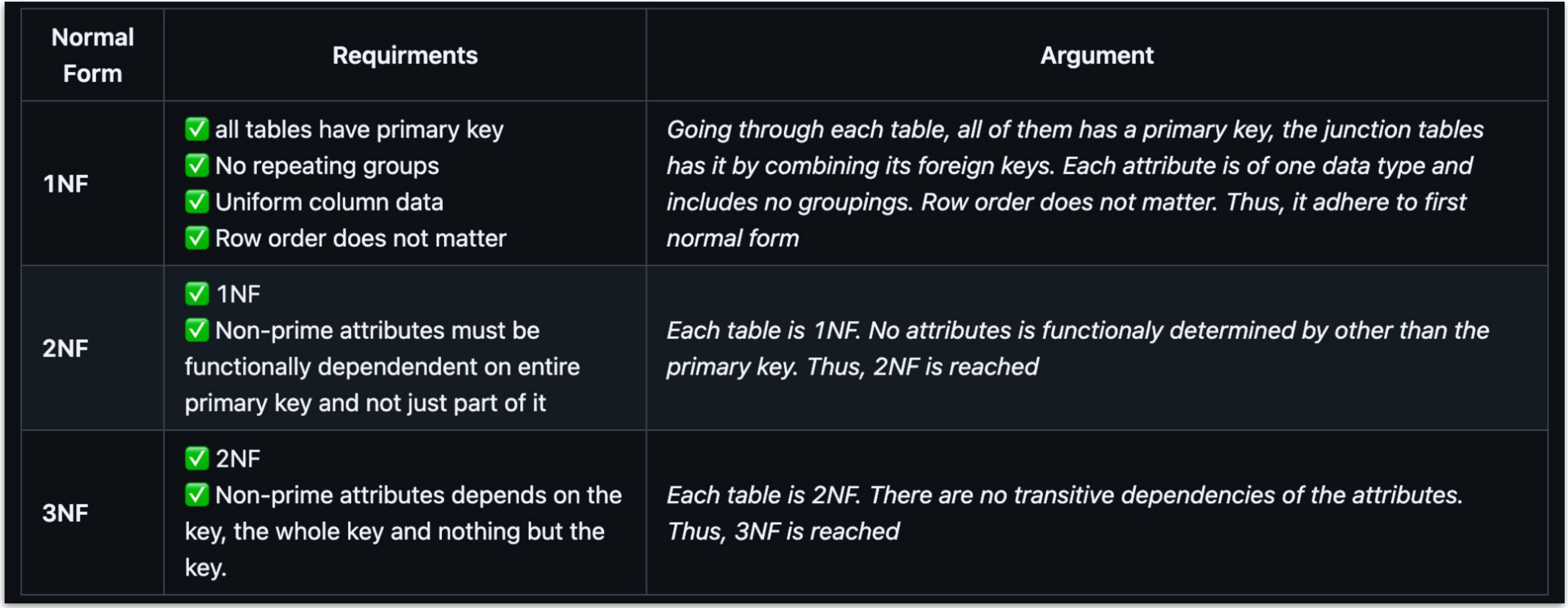
```
-- Example Enrollment csv
COPY enrollment FROM '/mnt/Enrollment.csv' WITH (FORMAT CSV, HEADER);
```

- Copied the data from the csv's to the database using the COPY command in psql

Normalisation - why do we care about it?

- Reduce data redundancy and improve data integrity
  - Changes can be done in place and integrity is ensured
- Prevent anomalies when updating, inserting
  - Ensures high data quality over the entire lifecycle
- Normalisation is a not a silver bullet for OLAP and Data Warehousing
  - For analytics purposes in OLAP systems and data warehousing, denormalization is sometimes desired to optimize read performance for analytical queries for quicker data retrieval.

Normalisation - how it is achieved?

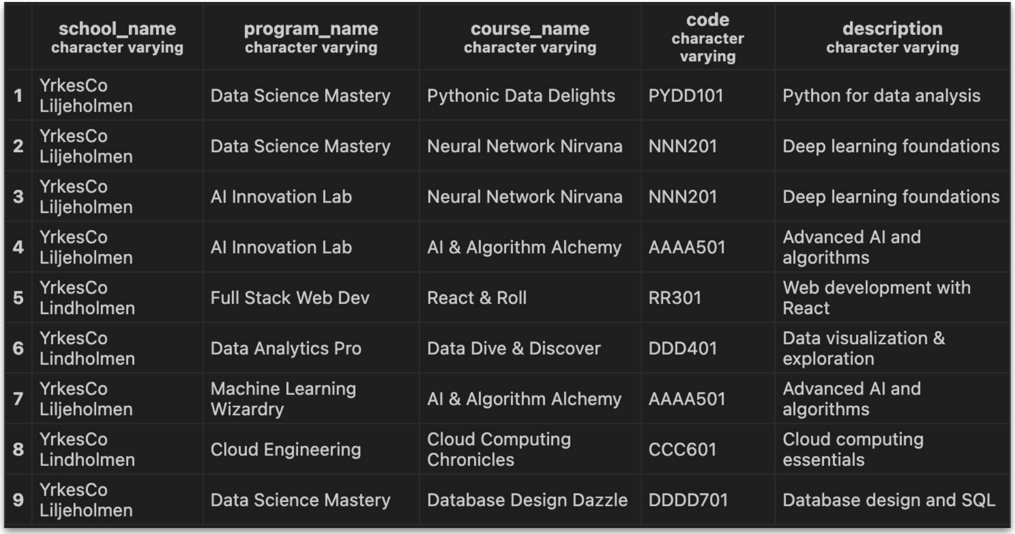


Normal Form	Requirments	Argument
1NF	✅ all tables have primary key ✅ No repeating groups ✅ Uniform column data ✅ Row order does not matter	*Going through each table, all of them has a primary key, the junction tables has it by combining its foreign keys. Each attribute is of one data type and includes no groupings. Row order does not matter. Thus, it adhere to first normal form*
2NF	✅ 1NF ✅ Non-prime attributes must be functionally dependendent on entire primary key and not just part of it	*Each table is 1NF. No attributes is functionaly determined by other than the primary key. Thus, 2NF is reached*
3NF	✅ 2NF ✅ Non-prime attributes depends on the key, the whole key and nothing but the key.	*Each table is 2NF. There are no transitive dependencies of the attributes. Thus, 3NF is reached*

Some query results and why they matter to the business

Query
for each school, list the program names and the courses offered within those programs?

Why it matters
This is showing that the school sites are having distinct programs as modelled. It also shows that more school sites can be added in the future if the business grows.



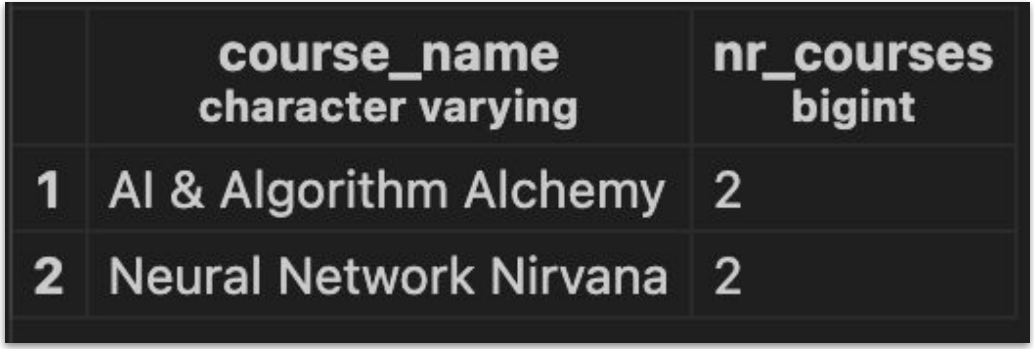
	school_name character varying	program_name character varying	course_name character varying	code character varying	description character varying
1	YrkesCo Liljeholmen	Data Science Mastery	Pythonic Data Delights	PYDD101	Python for data analysis
2	YrkesCo Liljeholmen	Data Science Mastery	Neural Network Nirvana	NNN201	Deep learning foundations
3	YrkesCo Liljeholmen	AI Innovation Lab	Neural Network Nirvana	NNN201	Deep learning foundations
4	YrkesCo Liljeholmen	AI Innovation Lab	AI & Algorithm Alchemy	AAAA501	Advanced AI and algorithms
5	YrkesCo Lindholmen	Full Stack Web Dev	React & Roll	RR301	Web development with React
6	YrkesCo Lindholmen	Data Analytics Pro	Data Dive & Discover	DDD401	Data visualization & exploration
7	YrkesCo Liljeholmen	Machine Learning Wizardry	AI & Algorithm Alchemy	AAAA501	Advanced AI and algorithms
8	YrkesCo Lindholmen	Cloud Engineering	Cloud Computing Chronicles	CCC601	Cloud computing essentials
9	YrkesCo Liljeholmen	Data Science Mastery	Database Design Dazzle	DDDD701	Database design and SQL

Query

what courses belong to more than one program?

Why it matters

This is showing that the school can have shared courses across programs as defined in the business requirements.



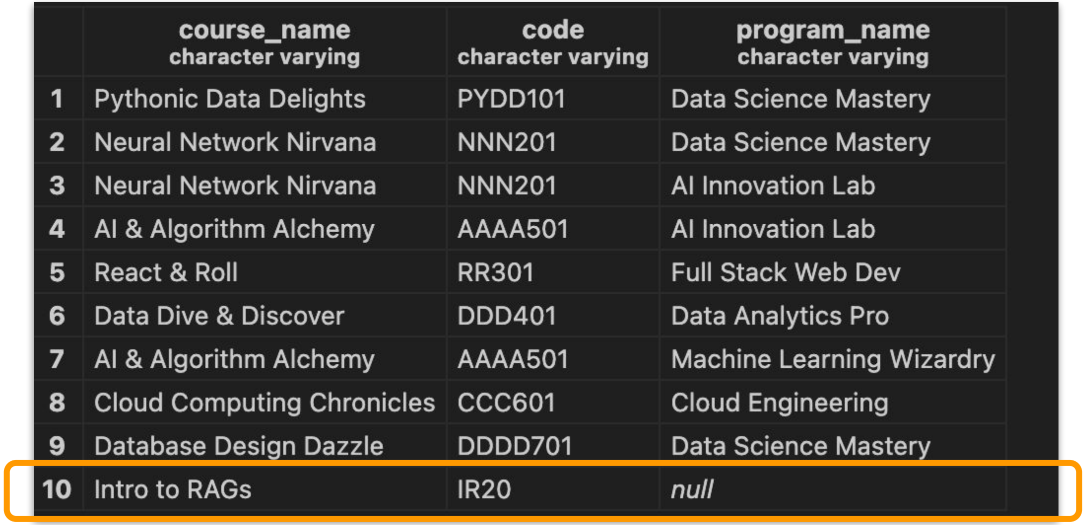
	course_name character varying	nr_courses bigint
1	AI & Algorithm Alchemy	2
2	Neural Network Nirvana	2

Query

What courses belong to which program and which courses are standalone?

Why it matters

This is showing that the school can offer standalone courses not part of any program. Open for any student to attend.



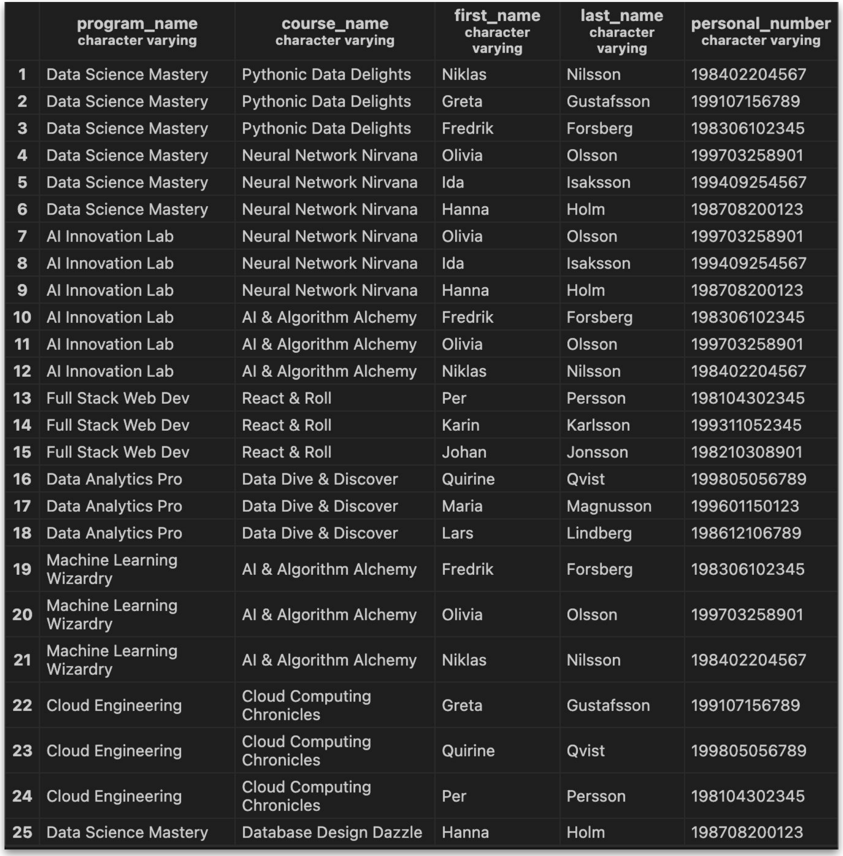
	course_name character varying	code character varying	program_name character varying
1	Pythonic Data Delights	PYDD101	Data Science Mastery
2	Neural Network Nirvana	NNN201	Data Science Mastery
3	Neural Network Nirvana	NNN201	AI Innovation Lab
4	AI & Algorithm Alchemy	AAAA501	AI Innovation Lab
5	React & Roll	RR301	Full Stack Web Dev
6	Data Dive & Discover	DDD401	Data Analytics Pro
7	AI & Algorithm Alchemy	AAAA501	Machine Learning Wizardry
8	Cloud Computing Chronicles	CCC601	Cloud Engineering
9	Database Design Dazzle	DDDD701	Data Science Mastery
10	Intro to RAGs	IR20	*null*

Query

For each program, list the names of the students and their personal number enrolled in the classes within that program.

Why it matters

This is showing that simplicity of joining tables to get lists of personal information per course



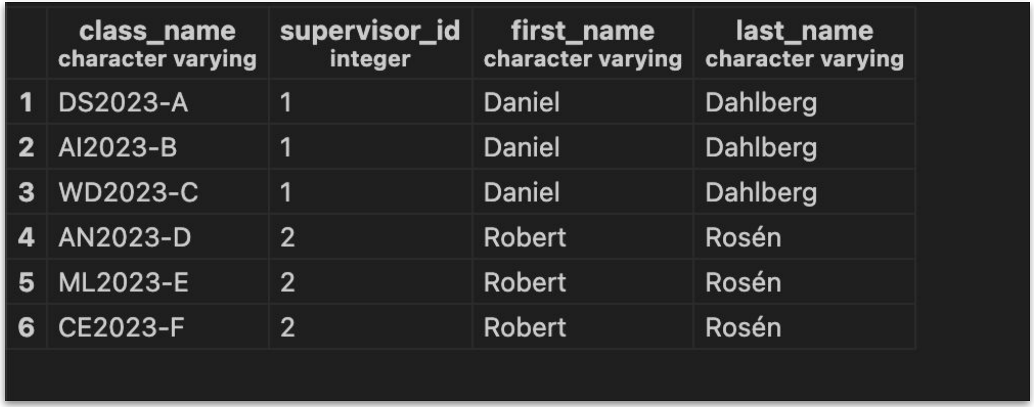
	program_name character varying	course_name character varying	first_name character varying	last_name character varying	personal_number character varying
1	Data Science Mastery	Pythonic Data Delights	Niklas	Nilsson	198402204567
2	Data Science Mastery	Pythonic Data Delights	Greta	Gustafsson	199107156789
3	Data Science Mastery	Pythonic Data Delights	Fredrik	Forsberg	198306102345
4	Data Science Mastery	Neural Network Nirvana	Olivia	Olsson	199703258901
5	Data Science Mastery	Neural Network Nirvana	Ida	Isaksson	199409254567
6	Data Science Mastery	Neural Network Nirvana	Hanna	Holm	198708200123
7	AI Innovation Lab	Neural Network Nirvana	Olivia	Olsson	199703258901
8	AI Innovation Lab	Neural Network Nirvana	Ida	Isaksson	199409254567
9	AI Innovation Lab	Neural Network Nirvana	Hanna	Holm	198708200123
10	AI Innovation Lab	AI & Algorithm Alchemy	Fredrik	Forsberg	198306102345
11	AI Innovation Lab	AI & Algorithm Alchemy	Olivia	Olsson	199703258901
12	AI Innovation Lab	AI & Algorithm Alchemy	Niklas	Nilsson	198402204567
13	Full Stack Web Dev	React & Roll	Per	Persson	198104302345
14	Full Stack Web Dev	React & Roll	Karin	Karlsson	199311052345
15	Full Stack Web Dev	React & Roll	Johan	Jonsson	198210308901
16	Data Analytics Pro	Data Dive & Discover	Quirine	Qvist	199805056789
17	Data Analytics Pro	Data Dive & Discover	Maria	Magnusson	199601150123
18	Data Analytics Pro	Data Dive & Discover	Lars	Lindberg	198612106789
19	Machine Learning Wizardry	AI & Algorithm Alchemy	Fredrik	Forsberg	198306102345
20	Machine Learning Wizardry	AI & Algorithm Alchemy	Olivia	Olsson	199703258901
21	Machine Learning Wizardry	AI & Algorithm Alchemy	Niklas	Nilsson	198402204567
22	Cloud Engineering	Cloud Computing Chronicles	Greta	Gustafsson	199107156789
23	Cloud Engineering	Cloud Computing Chronicles	Quirine	Qvist	199805056789
24	Cloud Engineering	Cloud Computing Chronicles	Per	Persson	198104302345
25	Data Science Mastery	Database Design Dazzle	Hanna	Holm	198708200123

Query

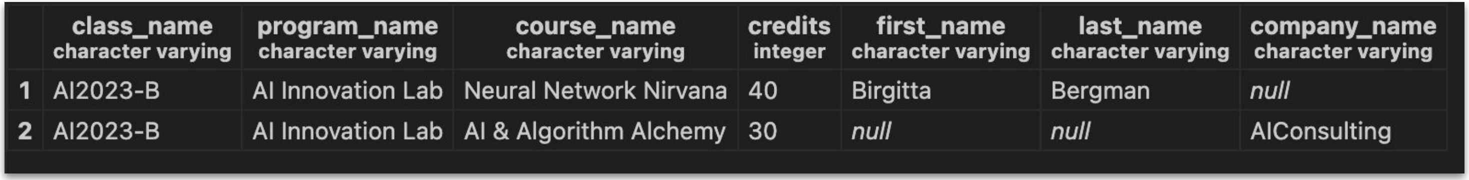
What supervisor belong to a specific class?

Why it matters

This is showing that a supervisor has 3 classes each. A trigger function is also added to enforce this business requirement.



	class_name character varying	supervisor_id integer	first_name character varying	last_name character varying
1	DS2023-A	1	Daniel	Dahlberg
2	AI2023-B	1	Daniel	Dahlberg
3	WD2023-C	1	Daniel	Dahlberg
4	AN2023-D	2	Robert	Rosén
5	ML2023-E	2	Robert	Rosén
6	CE2023-F	2	Robert	Rosén



	class_name	program_name	course_name	credits	first_name	last_name	company_name
	character varying	character varying	character varying	integer	character varying	character varying	character varying
1	AI2023-B	AI Innovation Lab	Neural Network Nirvana	40	Birgitta	Bergman	*null*
2	AI2023-B	AI Innovation Lab	AI & Algorithm Alchemy	30	*null*	*null*	AIConsulting

Query

What courses a specific class has and what educator per course

Why it matters

This is showing that for a specific program, the courses can be taught by either a hired teacher or a consultant.

Thank you for listening...