Data Modelling and Normalization

Mullins chapter 3

Bjarte Wang-Kileng

HVL

February 9, 2026

Western Norway University of Applied Sciences

Outline

1. Data modelling – A short repetition

2. Normalisation

Outline

1. Data modelling – A short repetition

2. Normalisation

Entities, occurences and attributes

- Data modelling and UML (or E/R diagrams) have been studied in earlier courses (e.g. DAT107).
- Data model – Abstraction of real world things.
- The *entities* are the «objects» that are stored in the database, e.g. *student*.
- Occurence is an instance of an entity, e.g. the student *Anne Annesen* is an occurence of *student*.
- Attributes are the characteristics of an entity. *Name*, *birth date* and *phone number* are three attributes of a occurence of *student*.

Attributes

An attribute does one of three things:

1. Identifies an entity:
   - *Candidate key*.
   - Immutable.
2. Relates entities:
   - *Foreign key*.
   - Refers to the primary key of an occurrence of another entity.
3. Describes an occurrence of an entity.

Functional relations between attributes of an enity

- A functional relation between X and Y is written as:

$$X \rightarrow Y$$

- A functional relation between two attributes X and Y means that for a given X there is precisely one value of Y.
- A functional relation must therefore exist between a candidate key and any other attribute of the entity.

$$\text{Student number} \rightarrow \text{Student first name}$$

Or, using data

$$111 \rightarrow \text{Ole}$$

$$222 \rightarrow \text{Per}$$

$$333 \rightarrow \text{Ole}$$

More on functional relations

- Assume a compound key

$$X = (\text{Student number}, \text{Student first name})$$

and an attribute

$$Y = \text{Student first name}$$

- Since Y is a part of X, we say that $Y \subseteq X$ (Y is a subset of X).
- Obviousely, if $Y \subseteq X$ then $X \to Y$.
- A functional relation $X \to Y$ is said to be trivial if $Y \subseteq X$.

Superkeys of an enity

- A superkey is an attribute, or combination of attributes that are unique within the entity.
- Any combination of attributes that include a candidate key will always be a superkey.

$$(\text{Student number}, \text{First name})$$

$$(\text{Student number}, \text{Last name})$$

$$(\text{Student number}, \text{First name}, \text{Last name})$$

$$(\text{Student number}, \text{Norwegian national security number})$$

$$(\text{Norwegian national security number}, \text{First name})$$

Superkeys, candidate keys and foreign keys

- Both candidate keys and foreign keys can be compound keys.
  - Can consist of several attributes.
- A candidate key is a minimal superkey.
  - No subset of the attributes is a superkey.
  - Below is a superkey that is **not** a candidate key, why?

$$(\text{Student number}, \text{First name})$$

  - The first attribute, *Student number* is a superkey on its own.
- A foreign key must be a candidate key of the referenced entity.

Primary key

- One key chosen from the candidate keys.
- Used to identify an entity occurrence.

Conceptual data model

- High level, business oriented view.
- Focus on the most important entities, attributes and relationships.
- Can contain many-to-many relationships.
- Cardinality, optionality and data types can be skipped.

Logical data model

- Fully normalised entities.
- All attributes are defined.
- All candidate keys, primary keys and foreign keys are defined.
- No many-to-many relationships.

Physical data model

- The logical model must be transformed into a physical implementation in a DBMS.
- Details in chapter 4.

Outline

1. Data modelling – A short repetition

2. Normalisation

Normalisation

- There are many correct models of the world.
- Not all equivalent models are equally good when it comes to:
  - reading data,
  - manipulating data.

Normalisation

Identify the one best place where each fact belongs.

Normalisation

Design approach that minimises data redundancy and optimises data structures.

First normal form

Domain

Domain of an attribute is the universe of values of the attribute.

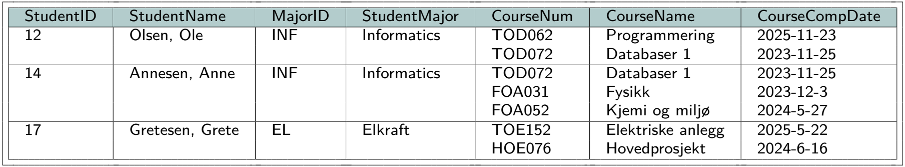
1NF

A row is in first normal form if and only if all underlying domains contain atomic values only.

Atomic value

Wheter or not a value is atomic depends on the use of the value!

An unnormalised entity



StudentID	StudentName	MajorID	StudentMajor	CourseNum	CourseName	CourseCompDate
12	Olsen, Ole	INF	Informatics	TOD062 TOD072	Programmering Databaser 1	2025-11-23 2023-11-25
14	Annesen, Anne	INF	Informatics	TOD072 FOA031 FOA052	Databaser 1 Fysikk Kjemi og miljø	2023-11-25 2023-12-3 2024-5-27
17	Gretesen, Grete	EL	Elkraft	TOE152 HOE076	Elektriske anlegg Hovedprosjekt	2025-5-22 2024-6-16

Table: Unnormalised Student data

- How does this entity break with the first normal form?
  - Repeating groups (the courses).
  - Attribute *StudentName* is not atomic.
    - But this depends on our use of the data.

Entities in 1NF

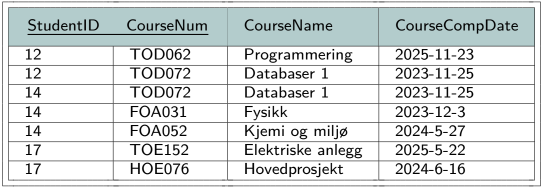
StudentID	LastName	FirstName	MajorID	StudentMajor
12	Olsen	Ole	INF	Informatics
14	Annesen	Anne	INF	Informatics
17	Gretesen	Grete	EL	Elkraft

Table: Entity *Student* in 1NF

StudentID	CourseNum	CourseName	CourseCompDate
12	TOD062	Programmering	2025-11-23
12	TOD072	Databaser 1	2023-11-25
14	TOD072	Databaser 1	2023-11-25
14	FOA031	Fysikk	2023-12-3
14	FOA052	Kjemi og miljø	2024-5-27
17	TOE152	Elektriske anlegg	2025-5-22
17	HOE076	Hovedprosjekt	2024-6-16

Table: Entity *Course* in 1NF

Major and minor components of data



StudentID	CourseNum	CourseName	CourseCompDate
12	TOD062	Programmering	2025-11-23
12	TOD072	Databaser 1	2023-11-25
14	TOD072	Databaser 1	2023-11-25
14	FOA031	Fysikk	2023-12-3
14	FOA052	Kjemi og miljø	2024-5-27
17	TOE152	Elektriske anlegg	2025-5-22
17	HOE076	Hovedprosjekt	2024-6-16

Table: Entity *Course* in 1NF

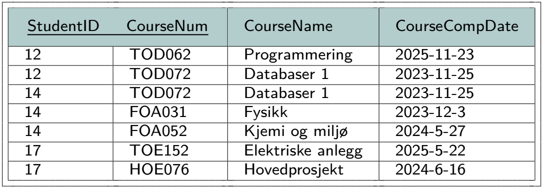
- What about the *CourseComp* column?
  - What if queries ask about courses that completed in e.g. 2024.
  - Should we use separate *Year*, *Month* and *Date* columns?
- Always best to combine a major and minor part into one column.
- DBMS will have functions to get the year from a date colum.

Second Normal Form

2NF

A row is in second normal form if and only if it is in first normal form and every non-key attribute (i.e. not part of any candidate key) is fully dependent on a candidate key (or on another non-key attribute).

- Can you see any problems with entity *Course* ?



StudentID	CourseNum	CourseName	CourseCompDate
12	TOD062	Programmering	2025-11-23
12	TOD072	Databaser 1	2023-11-25
14	TOD072	Databaser 1	2023-11-25
14	FOA031	Fysikk	2023-12-3
14	FOA052	Kjemi og miljø	2024-5-27
17	TOE152	Elektriske anlegg	2025-5-22
17	HOE076	Hovedprosjekt	2024-6-16

  - *CourseName* depends on *CourseNum* but not on *StudentID*.
- Solution?
  - Move the attribute with the part of the primary key on which it depends to a new table.

Entities in 2NF

StudentID	LastName	FirstName	MajorID	StudentMajor
12	Olsen	Ole	INF	Informatics
14	Annesen	Anne	INF	Informatics
17	Gretesen	Grete	EL	Elkraft

Table: Entity *Student* (unchanged from the 1NF form)

StudentID	CourseNum	CourseCompDate
12	TOD062	2025-11-23
12	TOD072	2023-11-25
14	TOD072	2023-11-25
14	FOA031	2023-12-3
14	FOA052	2024-5-27
17	TOE152	2025-5-22
17	HOE076	2024-6-16

Table: Entity *Enrolment* in 2NF

CourseNum	CourseName	Credits
TOD062	Programmering	10
TOD072	Databaser 1	5
FOA031	Fysikk	10
FOA052	Kjemi og miljø	10
TOE152	Elektriske anlegg	10
HOE076	Hovedprosjekt	15

Table: Entity *Course* in 2NF

Null values and normalisation

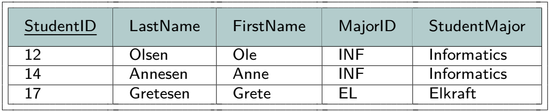
- Attribute value **null** may indicate either that a value is unknown or that the attribute is “not applicable” for this occurence of the entity.
- Value **null**, meaning “not applicable” may indicate a normalisation problem.
- How can an attribute that is “not applicable” depend fully on a candidate key?

Third normal form

3NF

A row is in third normal form if and only if it is in 2NF and every non-key attribute is nontransitively dependent (i.e. directly dependent) on the primary key (PK).

- Do you see any problems with the 2NF Student entity?



StudentID	LastName	FirstName	MajorID	StudentMajor
12	Olsen	Ole	INF	Informatics
14	Annesen	Anne	INF	Informatics
17	Gretesen	Grete	EL	Elkraft

  - *StudentMajor* depends on *StudentID* transitively through *MajorID*.
  - *StudentMajor* is not a key.
- Solution?
  - Move attributes that do not depend on the PK to a new table, together with the non-PK attribute on which they depend.

Entities in 3NF

StudentID	LastName	FirstName	MajorID
12	Olsen	Ole	INF
14	Annesen	Anne	INF
17	Gretesen	Grete	EL

Table: Entity *Student* in 3NF

MajorID	StudentMajor
INF	Informatics
EL	Elkraft

Table: Entity *Major* in 3NF

StudentID	CourseNum	CourseCompDate
12	TOD062	2025-11-23
12	TOD072	2023-11-25
14	TOD072	2023-11-25
14	FOA031	2023-12-3
14	FOA052	2024-5-27
17	TOE152	2025-5-22
17	HOE076	2024-6-16

Table: Entity *Enrolment* (unchanged)

CourseNum	CourseName	Credits
TOD062	Programmering	10
TOD072	Databaser 1	5
FOA031	Fysikk	10
FOA052	Kjemi og miljø	10
TOE152	Elektriske anlegg	10
HOE076	Hovedprosjekt	15

Table: Entity *Course* (unchanged)

Boyce–Codd normal form (BCNF or 3.5NF)

- Must be 3NF.
- For every dependency $X \to Y$, X is a superkey, or $Y \subseteq X$ (trivial).
- BCNF can only be broken if multiple overlapping candidate keys.
- Not always possible to fulfill BCNF.
- Elementary key normal form (EKNF) is a weaker form of BCNF, always possible.

Boyce-Codd

Similar to 2NF, but for keys.

Boyce-Codd demonstration

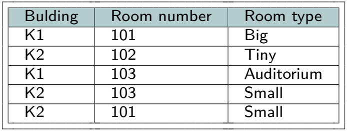
Bulding	Room number	Room type
K1	101	Big
K2	102	Tiny
K1	103	Auditorium
K2	103	Small
K2	101	Small

Table: Rooms at Kronstad

Assumptions:

- Room *number* does not identify the building.
  - E.g. room 101 exists in both K1 and K2.
- Room *type* does not identify the room number.
  - E.g. room 101 and 103 in K2 are both *small*
- Room *type* does identify the building:
  - Auditoriums and big rooms are in K1.
  - Small and tiny rooms are in K2.

Boyce-Codd demonstration contd.



Bulding	Room number	Room type
K1	101	Big
K2	102	Tiny
K1	103	Auditorium
K2	103	Small
K2	101	Small

Table: Rooms at Kronstad

- Candidate keys:
  - 1: {Bulding, Room number},
  - 2: {Room number, Room type}.
- Dependencies:
  - 1: {Building, Room number} → Room type
  - 2: Room type → Building
- Do you see any problems?
  - *Building* is not fully dependend on a candidate key (2NF requirement).
- Looks like 2NF is broken? Or?
  - No, since *Building* is a key attribute!

Boyce-Codd demonstration contd.

Bulding	Room number	Room type
K1	101	Big
K2	102	Tiny
K1	103	Auditorium
K2	103	Small
K2	101	Small

Table: Rooms at Kronstad

- Candidate keys:
  - 1: {Bulding, Room number},
  - 2: {Room number, Room type}.
- Dependencies:
  - 1: {Building, Room number} → Room type
  - 2: Room type → Building
- Dependency 2 breaks BCNF since *Room type* is not a superkey.

Fourth normal form (4NF)

- Must be BCNF and have no multivalued dependencies.
- A multivalued dependency require at least three attributes X, Y, Z.
- Y and Z have a multivalued dependency on X if:
  - There are many possible values of Y and Z for each X, and
  - there are no functional dependency between Y and Z
- If BCNF, 4NF can only be broken with a compound candidate key.

Assume entity where all attributes form a compound candidate key:

$$(\text{Student}, \text{Course}, \text{Food likes})$$

4NF is broken since:

$$\text{Student} \twoheadrightarrow \text{Course} \quad \text{and} \quad \text{Student} \twoheadrightarrow \text{Food likes}$$

with no dependecy between *Course* and *Food likes*.

More details

- Additional normal forms exist.
  - For 5NF, real world constraints for valid combinations of attributes must be implicite in the structure of the table.
  - Also a ETNF (Essential tuple normal form), DKNF (Domain-key normal form) and 6NF.
- Normalisation is done when moving from the conceptual level to the logical level.
- Only 1NF is required for a relational database.
- 3NF makes it easier to manage and maintain data integrity.
- Physical model may deviate from 3NF due to performance issues.