GEO 874 | HS25
Universität Zürich

5. Lecture Introduction to Databases

Logical DB Design and Physical DB Design

Esra Suel
Dept. of Geography, University of Zürich

Rolf Meile
Eidg. Forschungsanstalt für Wald, Schnee und Landschaft (WSL)

kudos to Dr. Zhiyong Zhou, Dr. Cheng Fu & Dr. Haosheng Huang for providing this document

Written exam on 30.10.2025

- 70% of the final grade

- 1 hour
- Stuff covered in lectures and practicals
- Practical-oriented
- Not allowed to take lecture notes
- Pen and paper

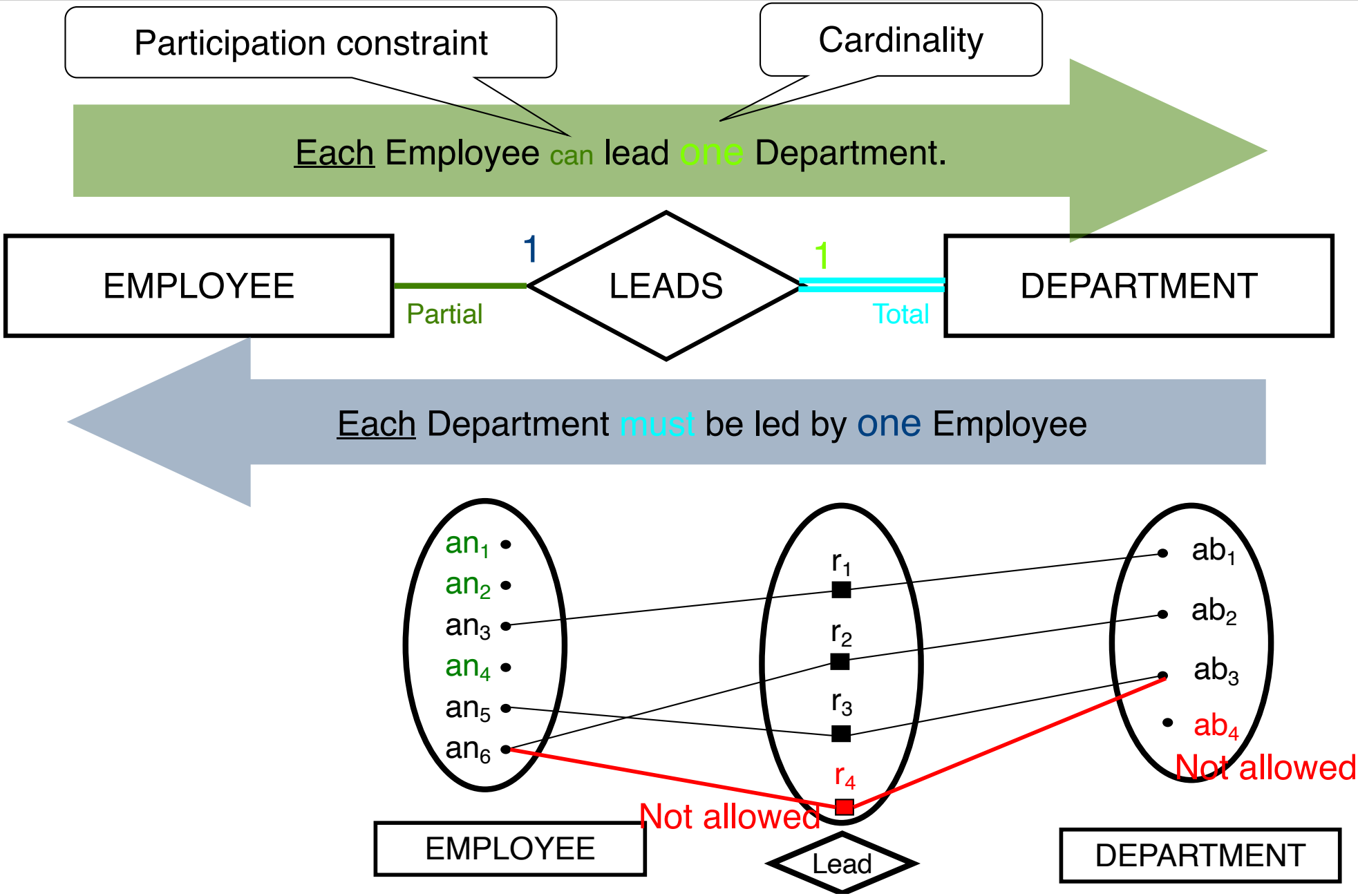
- Friday 30.10.2025 **8:40 am - 9:40 am**
- Place: **Y25-H-79** (No change unless there is further notice via OLAT)
- Be here at 8:30 am.
- No Lab on 30.10.2025

Recap: ER Modeling

Conceptual ER models are represented through ER Diagrams. ER modeling concerns the identification of

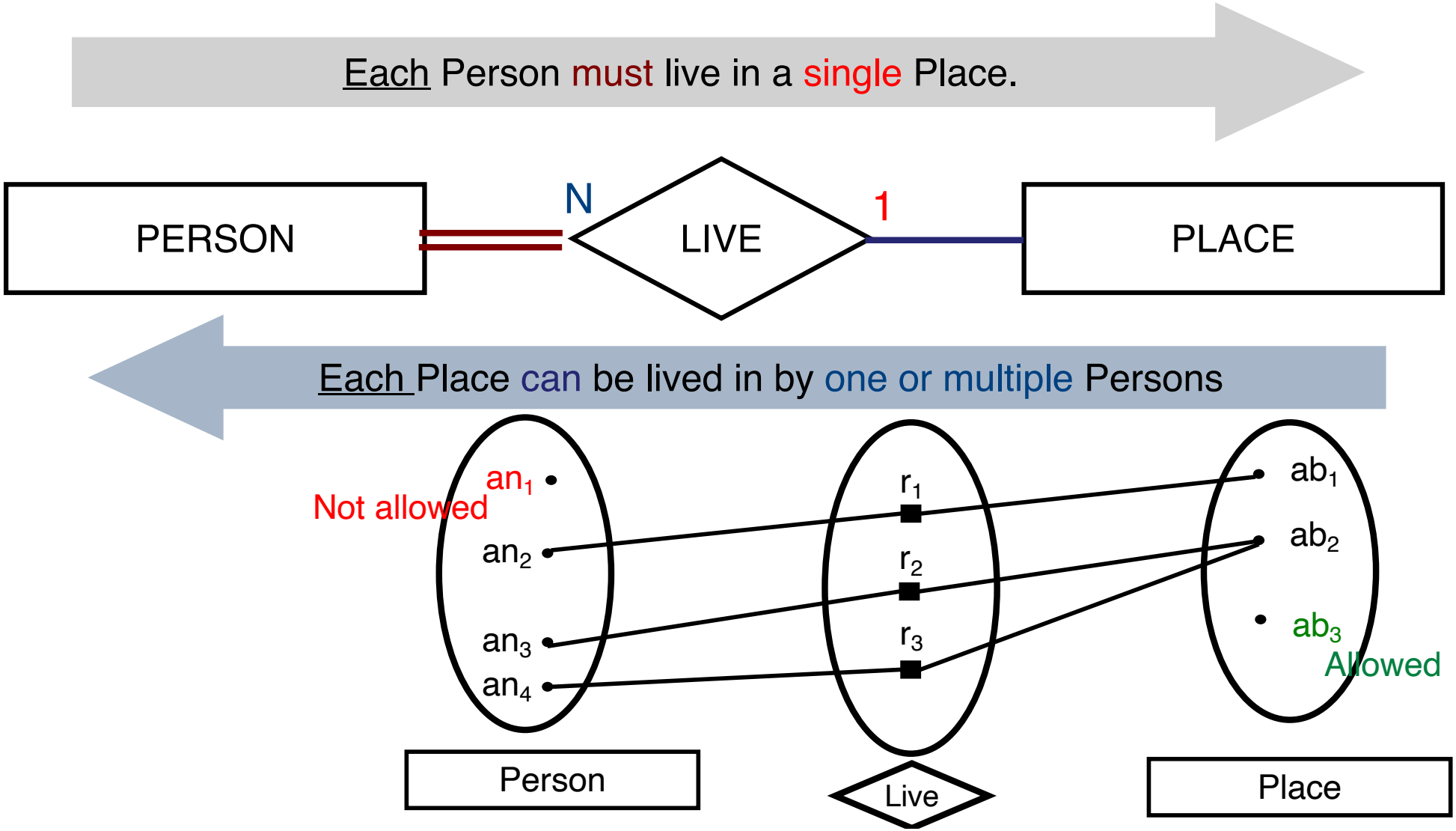
- Entities, entity types and entity sets
  - Key attributes of the entity types
- Attributes are characterized by name, data type and value domain
- Attribute types
  - Simple, composite, multi-valued, complex
- Relationship types
  - Degree (recursive, **binary**, ternary, …)
  - Constraints on relationship types
    - Cardinality constraint (1:1, 1:N, M:N for binary relationships)
    - Participation constraint (total, partial)
- Weak entity types
  - An identifying entity must be referenced to identify a weak entity (the existence dependency)
  - The primary key of an identifying entity together with a partial key together uniquely identify a weak entity.

Decoding requirements text 1



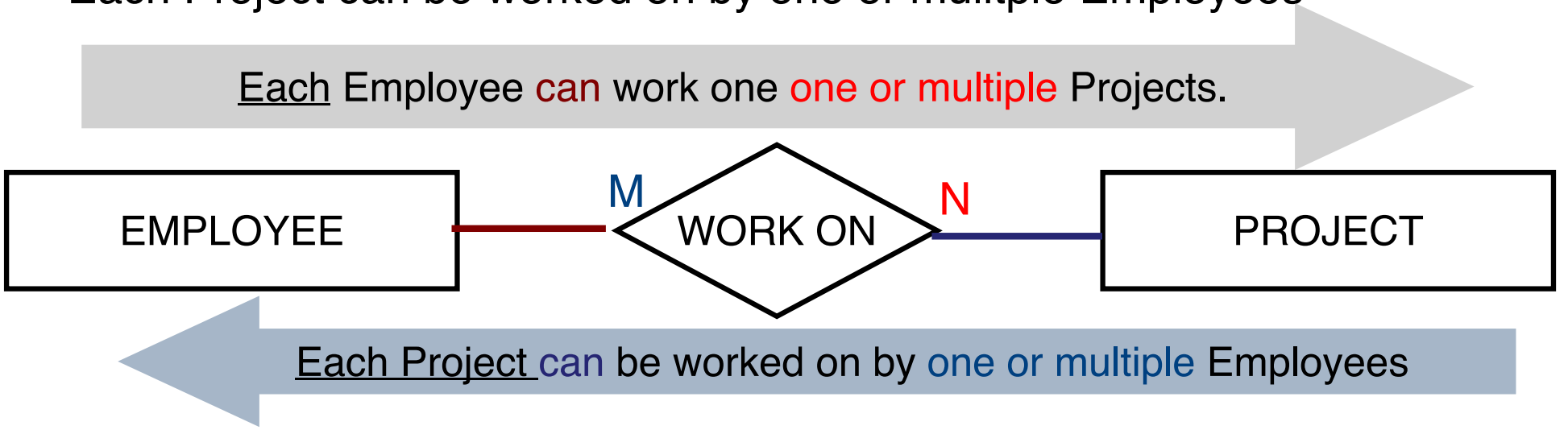
Decoding requirements text 2

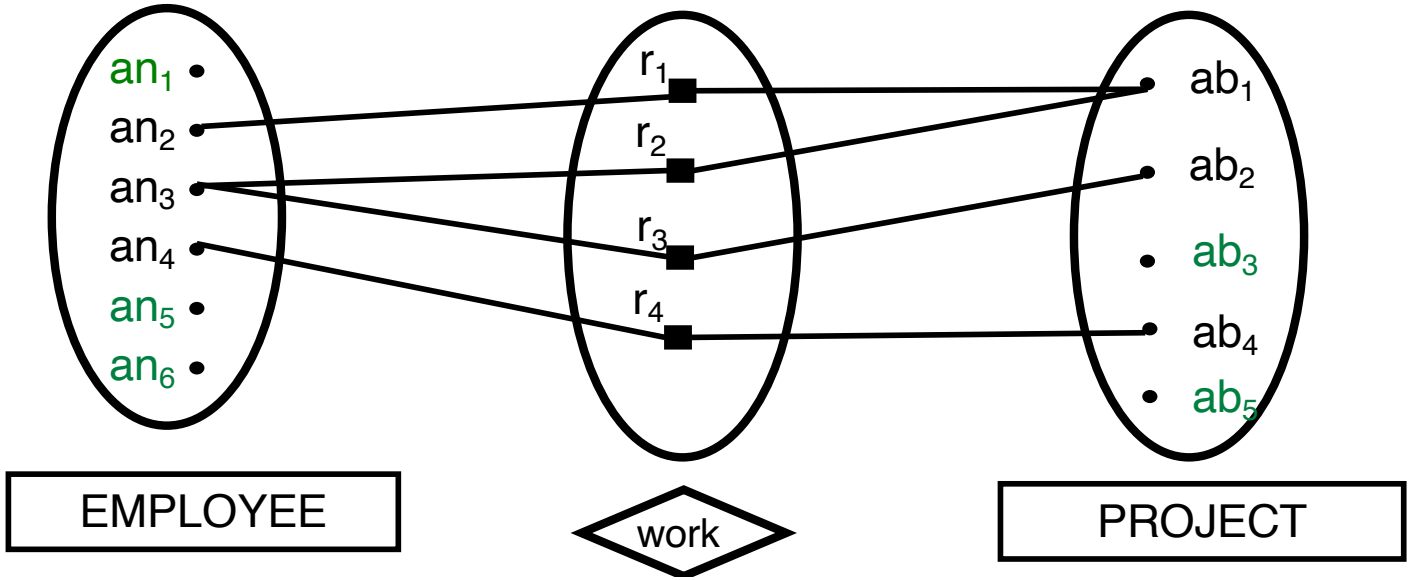
- Each Person must live in a single Place
- Each Place can be lived in by one or multiple Persons



Decoding requirements text 3

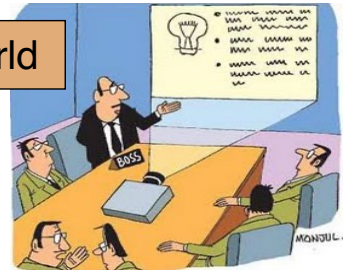
- Each Employee can work on one or multiple Projects
- Each Project can be worked on by one or mulitple Employees





DB Design

Real World

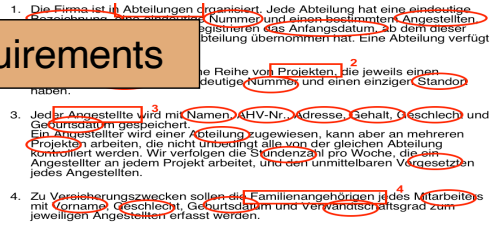


Entities in the real world: Department head Müller, Employee Muster, Project 'Budget 2014'

Lecture 3 — Requirements Analysis

Interview of users, study of documentation

Database Requirements



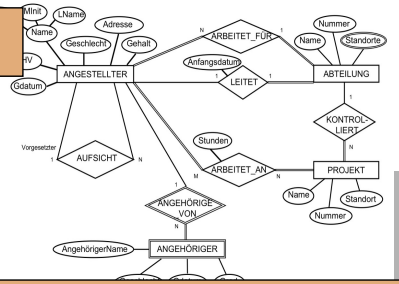
1. Die Firma ist in Abteilungen organisiert. Jede Abteilung hat eine eindeutige Nummer und einen bestimmten Angestellten registrieren das Anfangsdatum, ab dem dieser Abteilung übernommen hat. Eine Abteilung verfügt
2. Reihe von Projekten, die jeweils einen deutige Nummer und einen einziger Standor haben.
3. Jeder Angestellte wird mit Namen, AHV-Nr., Adresse, Gehalt, Geschlech und Geburtsdatum gespeichert. Ein Angestellter wird einer Abteilung zugewiesen, kann aber an mehreren Projekten arbeiten, die nicht unbedingt alle von der gleichen Abteilung kontrolliert werden. Wir verfolgen die Stundenzahl pro Woche, die ein Angestellter an jedem Projekt arbeitet, und den unmittelbaren Vorgesetzten jedes Angestellten.
4. Zu Versicherungszwecken sollen die Familienangehörigen jedes Mitarbeiters mit Vorname, Geschlecht, Geburtsdatum und Verwandtschaftsgrad zum jeweiligen Angestellten erfasst werden.

Written concise description of requirements (data reqs and functional [operations] reqs)

Lecture 4 — Conceptual DB design

Text analysis, Entities, Relationships

ER-Model



Entity types with attributes and relationships

DBMS-independent
DBMS-specific

Lecture 5 — Logical DB design (data model mapping)

- translation of model to implementation
- Validation through normalization

DB Schema

Relational data model	Object-relational data model	Other: network data model, …

PNAME | PNUMMER | PSTANDORT | ABTNR

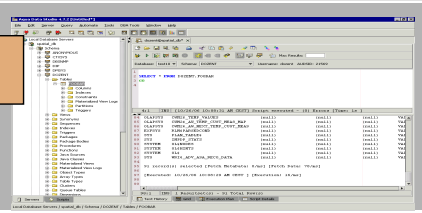
ANGEHÖRIGER

EAHV | ANGEHÖRIGER_NAME | GESCHLECHT | GDATUM | GRAD

Lecture 5 — Physical DB design

Description of database implementation (HW, indices, …)

Database



Implemented DB design, Tables (ordered) with rows and columns

➲ Elmasri & Navathe (2014): Ch 7-9

Learning objectives 5

- ✓ You will gain an understanding of entity integrity and referential integrity
- ✓ You will learn how to translate (“map”) an ER diagram into a relational model – translate entity/relationship types into logical relations (tables).
- ✓ You understand the problem of redundancy and potential anomalies.

Contents

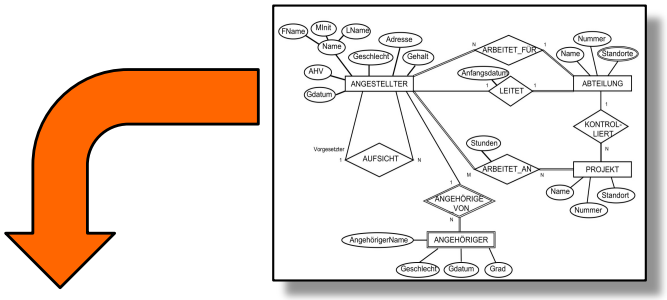
1. **Logical DB design**
    1. **Concepts of the relational model (Relation, Attribute, Tuple)**
    2. DB schema integrity
    3. Mapping rules
    4. Redundancy and anomalies (→ Normalisation)
2. Physical DB design

Logical DB design

Aim: mapping of the conceptual model (our case: ER diagram)
Into a logical model (our case relational model, thus performing a relational DB design).

- Systematic definition of the schemas of the relations
- Structures assuring consistency and integrity

The main goal of the relational DB design is to logically group attributes into relations



Relational data model of DB Company

EMPLOYEE

EFNAME	INITAL	ENAME	SSNO	EBIRTHD
EADRESS	ESEX	SALARY	SUPERSSNO	EDEPTNO

DEPARTMENT

DEPTNAME	DEPTNO	MGRSSNO	MGRLEADSDATE

DEPTLOC

LOCNR	DEPTNO

LOCATION

LOCNR	LOCNAME

WORKSON

ESSNO	PNO	STUNDEN

PROJECT

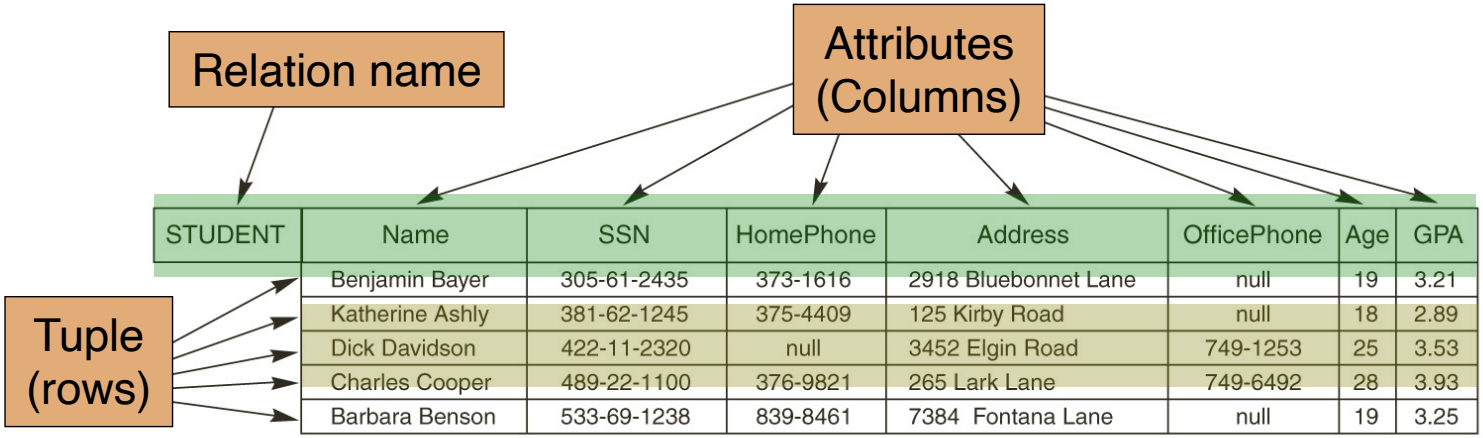
PNAME	PROJNO	STAONR	PDEPTNO

DEPENDENT

ESSNO	DEPENDNAME	DSEX	DBIRTHD	DEGREE

Relation

- The relational model represents the DB as a collection of relations
  - Each relation/table has a name, and contains a number of **attributes (=fields, or columns)**
  - Each attribute has a **domain** (=**data type**), which defines what kinds of data the attribute can contain (e.g., integer, string, ...)
  - A **tuple** (= **record**, or **row**) is a set of values from the domains of attributes $A_1..A_n$ from a given relation.
  - Each tuple consists of values for each attribute.
  - One tuple describes one entity or one relationship.



STUDENT	Name	SSN	HomePhone	Address	OfficePhone	Age	GPA
	Benjamin Bayer	305-61-2435	373-1616	2918 Bluebonnet Lane	null	19	3.21
	Katherine Ashly	381-62-1245	375-4409	125 Kirby Road	null	18	2.89
	Dick Davidson	422-11-2320	null	3452 Elgin Road	749-1253	25	3.53
	Charles Cooper	489-22-1100	376-9821	265 Lark Lane	749-6492	28	3.93
	Barbara Benson	533-69-1238	839-8461	7384 Fontana Lane	null	19	3.25

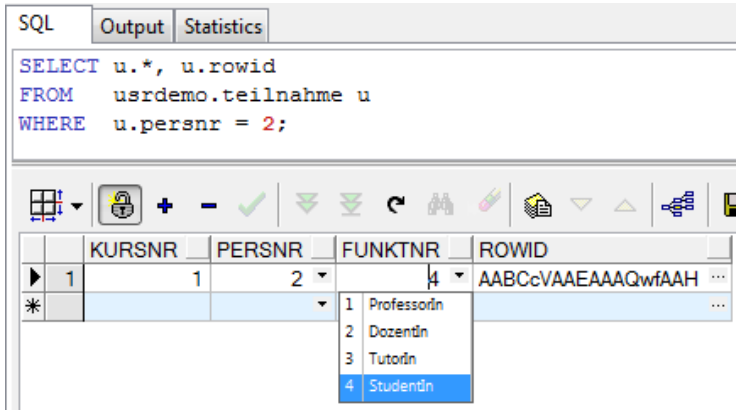
Contents

Goal: DB with Consistency and Integrity

- **Consistency**: a DB is consistent if it satisfies all constraints specified in the model (implicit) or the schema (explicit) – the DB is then free of data conflicts (valid state):
  - No redundancy (avoid multiplicity of storage, do not store the same data several times)
  - No conflicting entries
- **Integrity constraints:** ensuring the consistency of all entries in the database - both data elements and relationships
  - E.g., Primary key cannot be NULL
  - E.g., Attributes are not null; Attribute values in a range.

i. Domain constraints

- Specify, that within each tuple in a given relation, the value of an attribute *A* is always from the domain *dom(A)* of the attribute value domain.
  - The data types usually associated with domains
    - Numbers, e.g., numeric (n,m)
    - Characters, e.g., varchar(n)
    - Date, time, currency
    - Value subsets of a data type (Age 16-65)
  - Enumerated data types (key-value pairs: 1=small, 2=medium, 3=large; or PLZ)
  - Enumeration (e.g., PLZ) best implemented through referential integrity constraints; see iv.



ii. Key constraints

- Key attributes must uniquely identify tuples in a relation
  - All tuples in a relation must be distinct - there cannot be two tuples with the same value combination for all its attributes
  - Key attributes are used for this
  - They can be used for linking tables (joins while querying)
  - Primary key attributes used in a different relation B is called **Foreign Key** in B.

Data level:

PERSON

PersNr	LastName	FirstName	PLZ
1	Von Gunten	Reto	3000
2	Stofer	Christian	6330
3	Stofer	Silvia	6330
4	Ginzler	Christian	9000
5	Burghardt	Dirk	8001

Primary key PersNr

Foreign key

LOCATION

PLZ	Name
6330	Cham
9000	St. Gallen
8002	Zürich
6300	Zug
8400	Winterthur
8003	Zürich
3000	Bern
8001	Zürich
8051	Zürich

Primary key PLZ

Relational model:

PERSON

PersNr	LastName	FirstName	PLZ

LOCATION

PLZ	Name

iii. Entity integrity

- The **Entity integrity constraint** states, that no primary key can be NULL

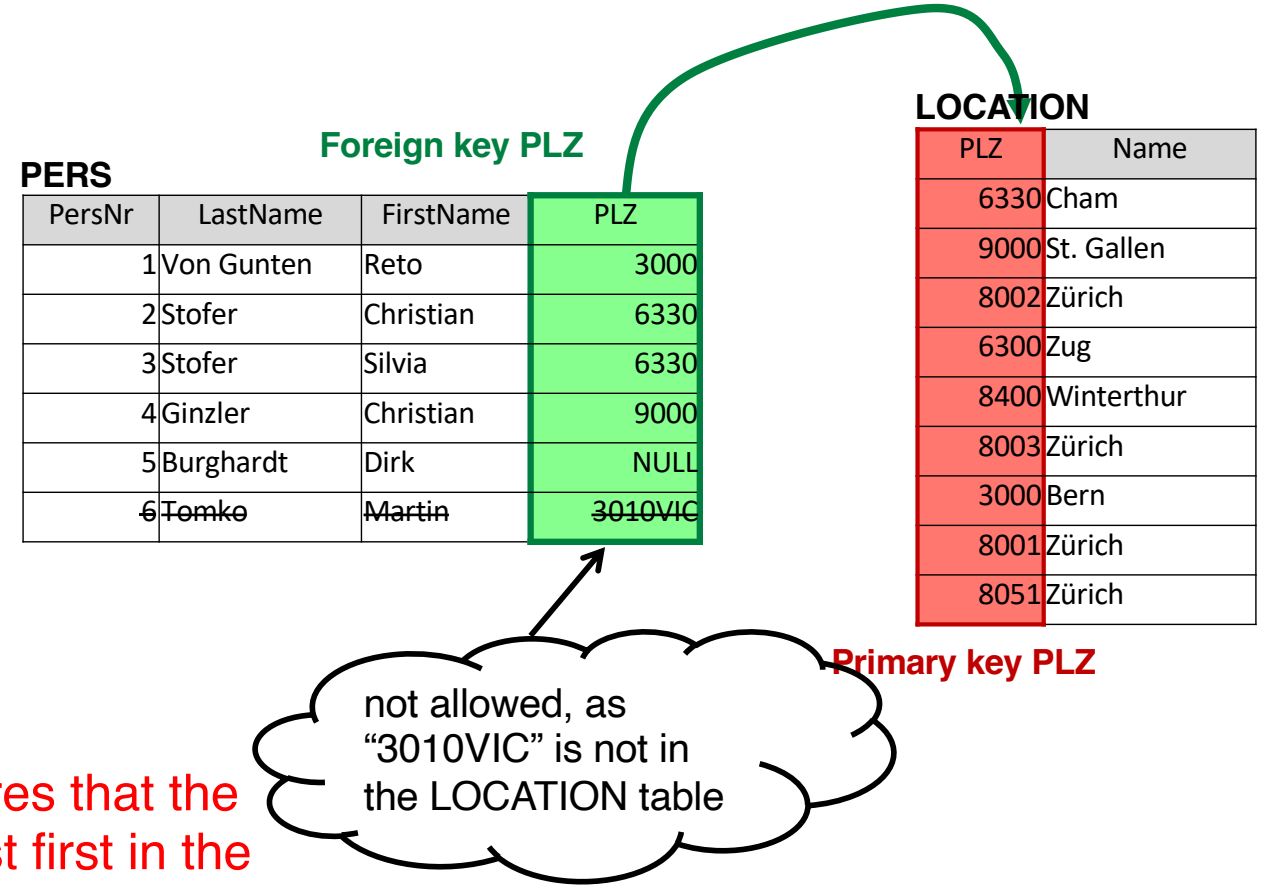
EMPLOYEE

ID	Surname	Department
1	Purves	GIS
2	Schmid	GIS
3	Reichenbacher	GIVA

It is possible to not fill all the attributes in the table, but the primary key must always be filled.

iv. Referential integrity

- **Referential integrity constraints** are specified between two relations and maintain consistency between tuples in the two relations
  - Referential integrity relates to the linkage of the tables through primary and foreign keys
- Values of the foreign key in PERS (**PLZ**) **can be only taken** from the existing value set of the primary key (**PLZ**) in the referred relation LOCATION, **or** NULL.
- A foreign key simply requires that the value in that field must exist first in the referred relation/table, or NULL.



Contents

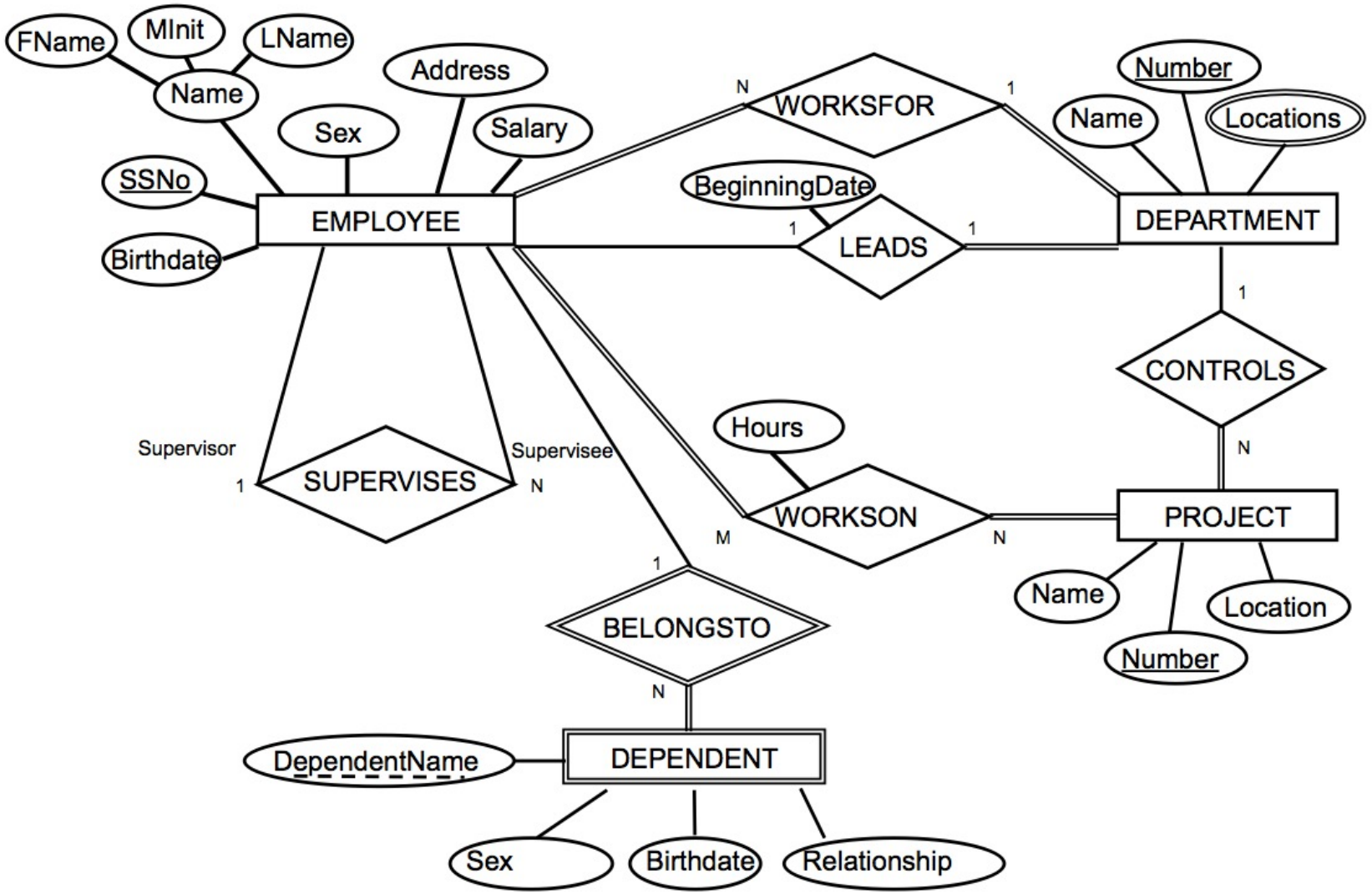
Logical DB design

Aim: mapping of the conceptual model (our case: ER diagram)
into a logical model (our case relational model, thus performing a relational DB design).

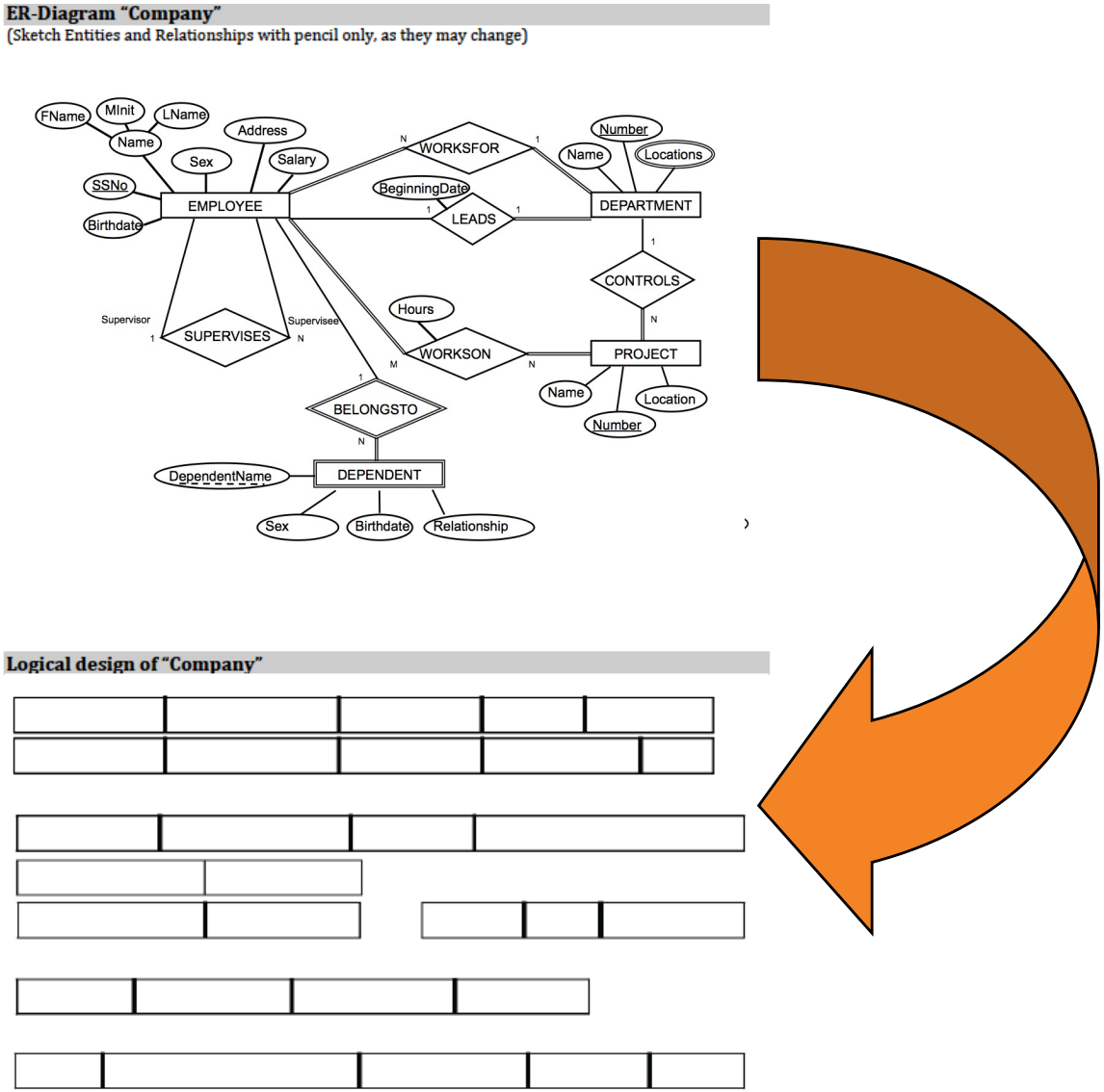
- Systematic definition of the schemas of the relations
- Structures assuring consistency and integrity

The main goal of the relational DB design is to logically group attributes/information into relations

Example: the ER Diagram of DB Company



Mapping the Diagram to Tables

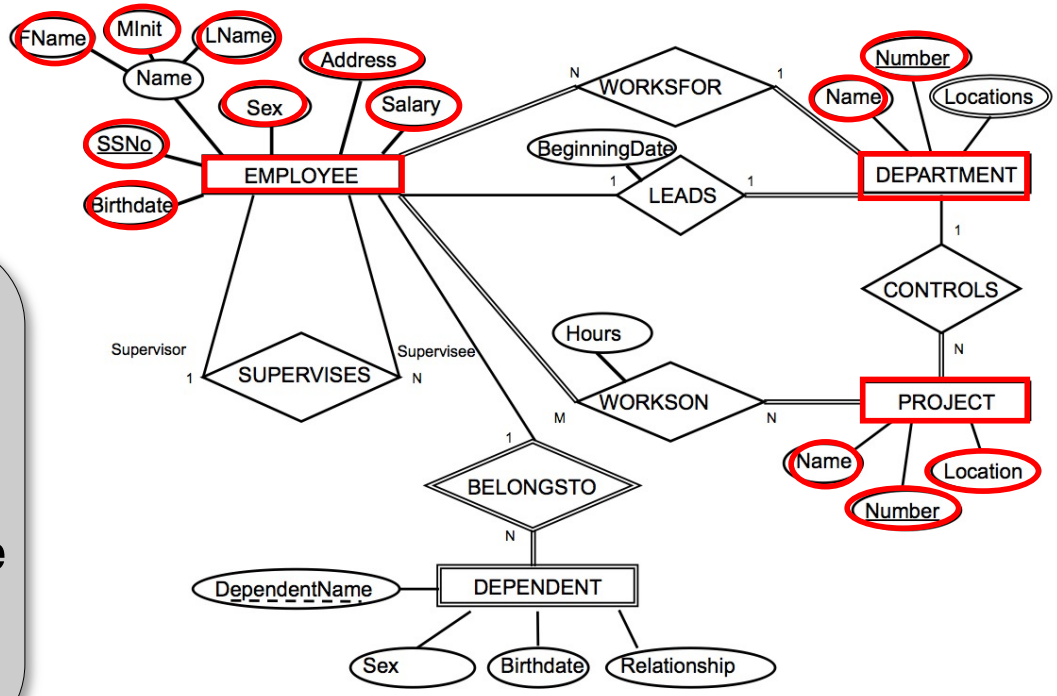


➲ Elmasri & Navathe (2014): Chapter 9, pp. 287-296.

Step 1

Regular (strong) entity types become relations

- Each strong entity type will be mapped into a relation/table with all its simple attributes
- For the composite attributes, map only their simple components into the entity type‘s attributes
- Choose the primary key attributes



DB Company

- Define the relation for the EMPLOYEE, DEPARTMENT and PROJECT
- SSNo (Employee), Number (Department) and Number (Project) are mapped to primary keys
- Attributes renamed while transforming ER -> Relation

EMPLOYEE

			SSNO	

DEPARTMENT

	DEPTNO		

PROJECT

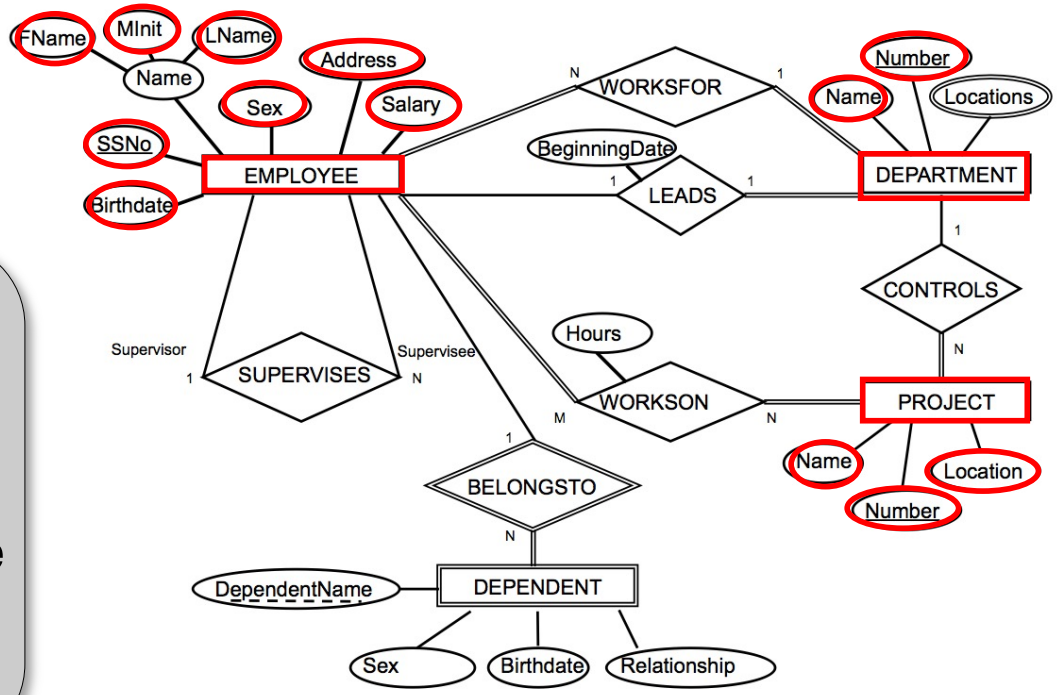
	PROJNO		

➲ Elmasri & Navathe (2014): Chapter 9, pp. 287-296.

Step 1

Regular (strong) entity types become relations

- Each strong entity type will be mapped into a relation/table with all its simple attributes
- For the composite attributes, map only their simple components into the entity type‘s attributes
- Choose the primary key attributes



DB Company

- Define the relation for the EMPLOYEE, DEPARTMENT and PROJECT
- SSNo, Number (Department) and Number (Project) are mapped to primary keys
- Attributes renamed while transforming ER -> Relation

EMPLOYEE

EFNAME	INITAL	ENAME	SSNO	EBIRTHD
EADRESS	ESEX	SALARY		

DEPARTMENT

DEPTNAME	DEPTNO		

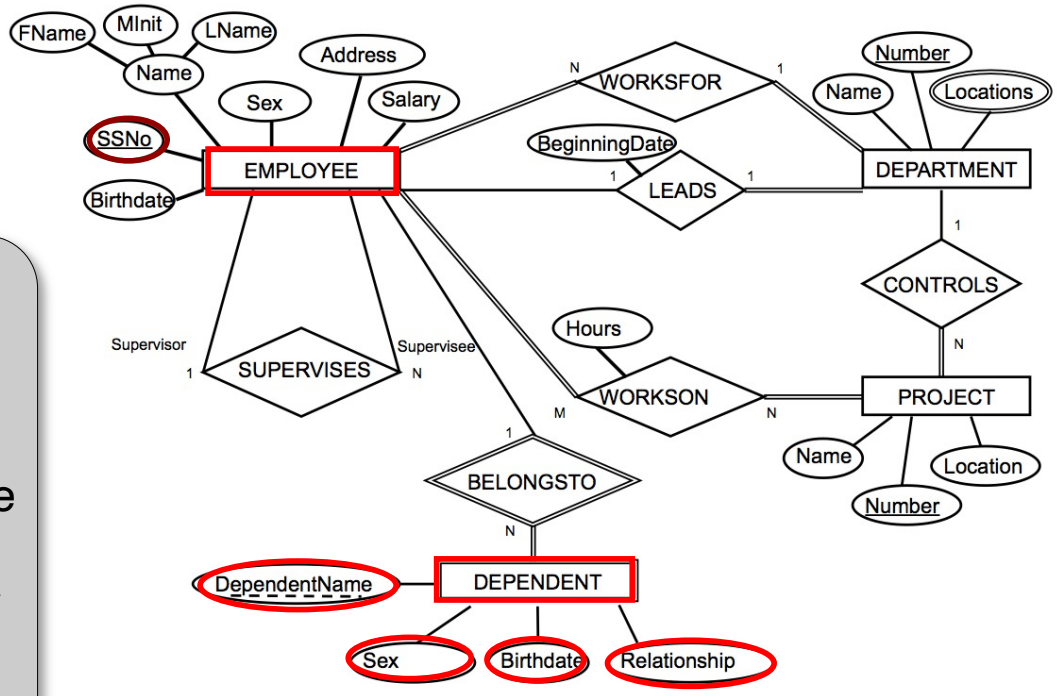
PROJECT

PNAME	PROJNO	PLOCATION	

➲ Elmasri & Navathe (2014): Chapter 9, pp. 287-296.

Step 2 Weak entity types also become relations with foreign key attr.

- A relation/table is defined for each weak entity type
- Simple attributes are directly mapped
- Additionally, add primary key attribute of the identifying entity (becoming foreign key in the current relation)
- Primary key? Combination of the foreign key attribute and the partial key of the weak entity



DB Company

- Design the relation for DEPENDENT
- Add new attribute ESSNO (duplicate of SSNO in EMPLOYEE)
- Primary key of the relation DEPENDENT is the combination {ESSNO, DEPENDNAME}, where DEPENDNAME is the partial key of entity type DEPENDENT

EMPLOYEE — **PK.attr**

EFNAME	INITAL	ENAME	SSNO	EBIRTHD
EADRESS	ESEX	SALARY		

DEPARTMENT

DEPTNAME	DEPTNO		

PROJECT

PNAME	PROJNO	PLOCATION	

DEPENDENT

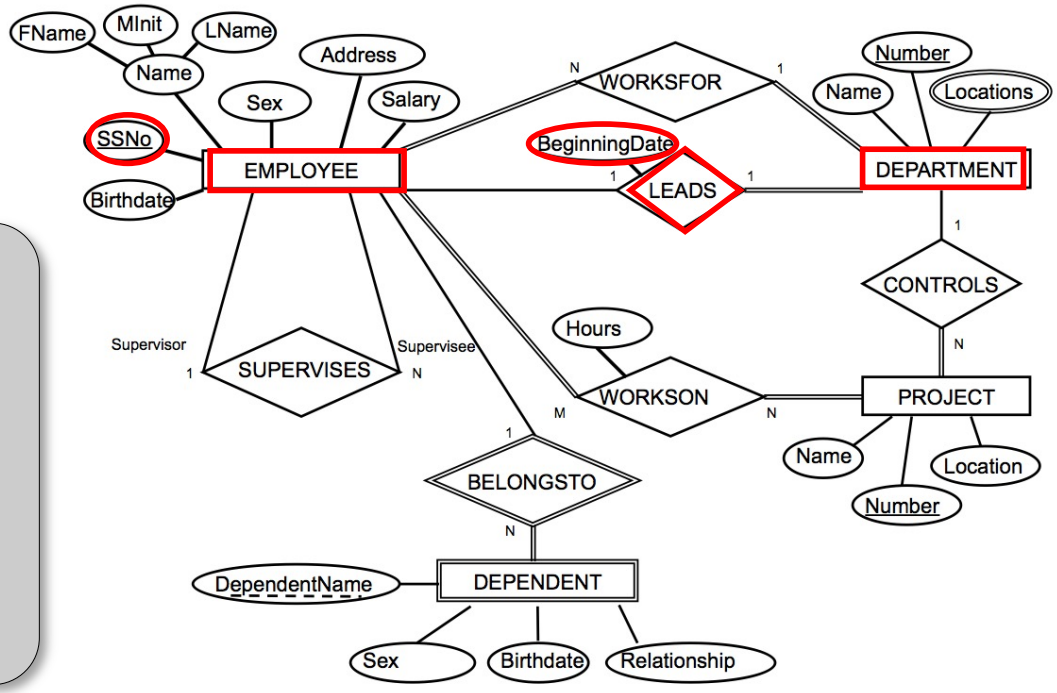
as **FK.attr**

ESSNO	DEPENDNAME	DSEX	DBIRTHD	DEGREE

Step 3

1:1-relationship based on primary key reference

- For each binary 1:1 relationship type
  - Include foreign key attribute on the side of the entity type with total participation
  - Relationship attributes map into the relation representing the entity with total participation
  - If no total participation, just either of the participating entity types.



DB Company

- Primary key attribute of the relation EMPLOYEE goes as foreign key attribute into the relation DEPARTMENT (here: SSNO as MGRSSNO)
- We map the attribute BEGINNINGDATE of the relationship type LEADS into the relation DEPARTMENT (as MGRLEADSDATE)

EMPLOYEE

EFNAME	INITAL	ENAME	SSNO	EBIRTHD
EADRESS	ESEX	SALARY		

DEPARTMENT

DEPTNAME	DEPTNO		

PROJECT

PNAME	PROJNO	PLOCATION	

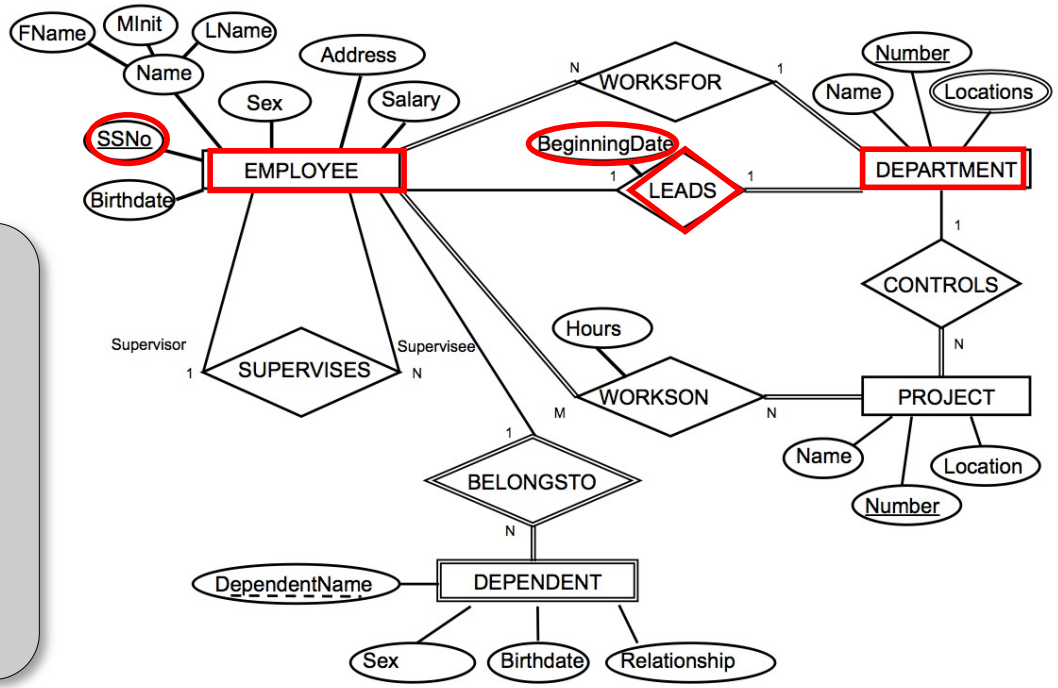
DEPENDENT

ESSNO	DEPENDNAME	DSEX	DBIRTHD	DEGREE

Step 3

1:1-relationship based on primary key reference

- For each binary 1:1 relationship type
  - Include foreign key attribute on the side of the entity type with total participation
  - Relationship attributes map into the relation representing the entity with total participation
  - If no total participation, just either of the participating entity types.



DB Company

- Primary key attribute of the relation EMPLOYEE goes as foreign key attribute into the relation DEPARTMENT (here: SSNO as MGRSSNO)
- We map the attribute BEGINNINGDATE of the relationship type LEADS into the relation DEPARTMENT (as MGRLEADSDATE)

Why on the total participation side?

– to prevent many NULL values

EMPLOYEE

			PK.attr	
EFNAME	INITAL	ENAME	SSNO	EBIRTHD
EADRESS	ESEX	SALARY		

DEPARTMENT

as **FK.attr**

DEPTNAME	DEPTNO	**MGR**SSNO	MGRLEADSDATE

The ***foreign key*** sits on DEPARTMENT using the ***foreign key attribute*** MGRSSNO and referencing SSNO

PROJECT

PNAME	PROJNO	PLOCATION	

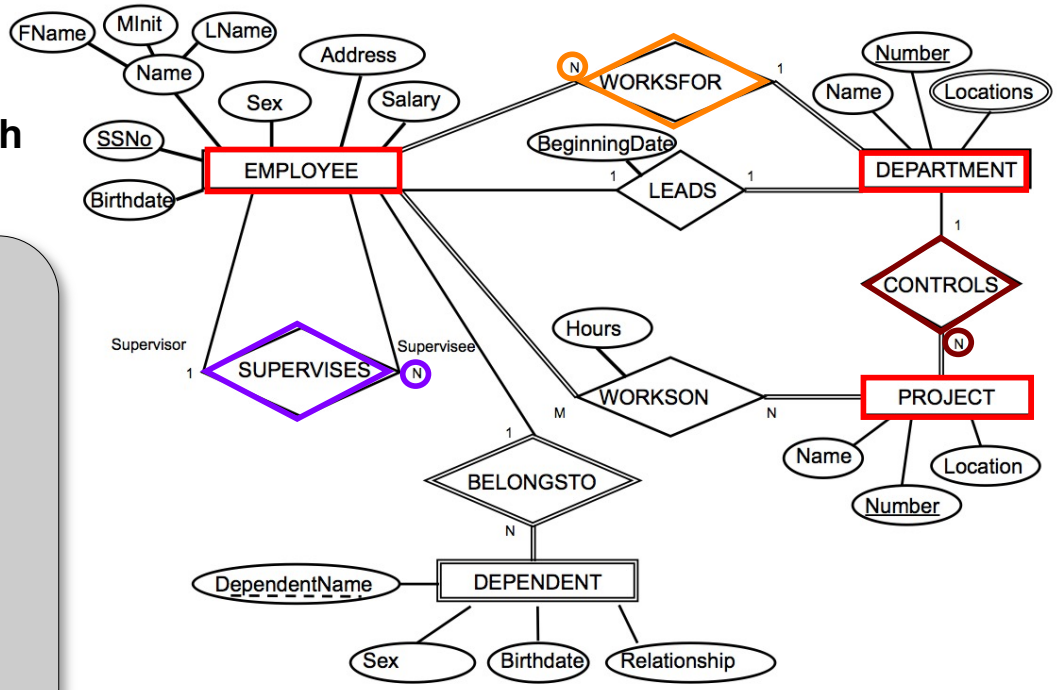
DEPENDENT

ESSNO	DEPENDNAME	DSEX	DBIRTHD	DEGREE

Step 4

Binary 1:N relationship through primary key reference

- For all binary 1:N relationship types we build foreign key references
  - Add the PK of the relation on the 1-side to the relation on N-side (become FK on the N-side)
  - Include any simple attributes of the 1:N relationship in the relation on the N-side



DB Company

- For the 1:N relationship types WORKSFOR, CONTROLS and SUPERVISES we design primary-foreign key references.
- In practice: we introduce the necessary foreign key attributes.

EMPLOYEE

EFNAME	INITAL	ENAME	SSNO	EBIRTHD
EADRESS	ESEX	SALARY	SUPERSSNO	EDEPTNO

PK.attr (SSNO)

as FK.attr (SUPERSSNO)

as FK.attr (EDEPTNO)

DEPARTMENT

DEPTNAME	DEPTNO	MGRSSNO	MGRLEADSDATE

PK.attr (DEPTNO)

PK.attr (DEPTNO)

PROJECT

PNAME	PROJNO	PLOCATION	PDEPTNO

as FK.attr (PDEPTNO)

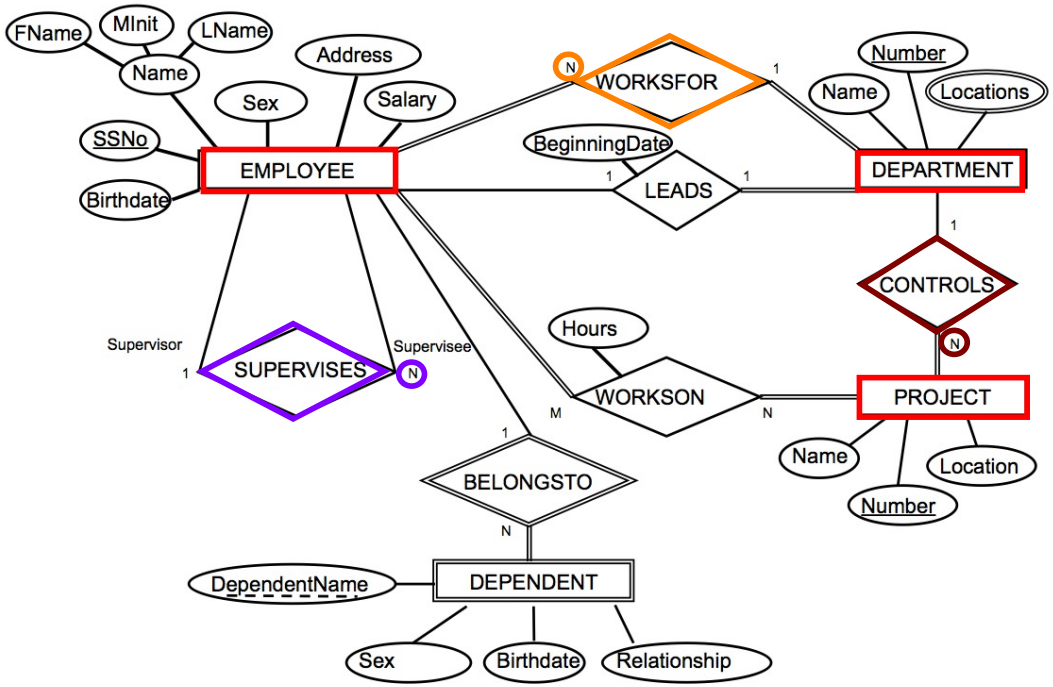
DEPENDENT

ESSNO	DEPENDNAME	DSEX	DBIRTHD	DEGREE

Step 4

Binary 1:N relationship through primary key reference

- For 1:N relationship type WORKSFOR, CONTROLS and SUPERVISES we create foreign key attributes on the N side (and later foreign keys in the DB):



	1-side		N-side	
Relationship	**Primary key attribute (PK)**	**Of entity...**	**Foreign key attribute (FK)**	**Declared in entity**
WORKSFOR	DEPTNO	DEPARTMENT	EDEPTNO	EMPLOYEE
CONTROLS	DEPTNO	DEPARTMENT	PDEPTNO	PROJECT
SUPERVISES	SSNO	EMPLOYEE	SUPERSSNO	EMPLOYEE

This is the process of generating foreign key attributes on connected relations

PK.attr ⋯⋯▶ As **FK.attr**

This is the foreign key finally to be generated in the database pointing from one table (relation) to another

➲ Elmasri & Navathe (2014): Chapter 9, pp. 287-296.

Lecture 2 (SQL): Definition of tables

```
CREATE TABLE user44.DEPARTMENT(
      DEPTNAME          VARCHAR(128)    NOT NULL,
      DEPTNO            NUMERIC(4)      NOT NULL,
      MGRSSNO           NUMERIC(16)     NOT NULL,
      MGRLEADSDATE      DATE            NOT NULL,
       PRIMARY KEY(DEPTNO)
);
```

```
ALTER TABLE user44.DEPARTMENT
ADD CONSTRAINT FOREIGN KEY
(MGRSSNO)
REFERENCES
user44.EMPLOYEE (SSRN);
```

DEPARTMENT

DEPTNAME	DEPTNO	MGRSSNO	MGRLEADSDATE

PK. attr

FK in PROJECT refers to PK in DEPARTMENT

PROJECT

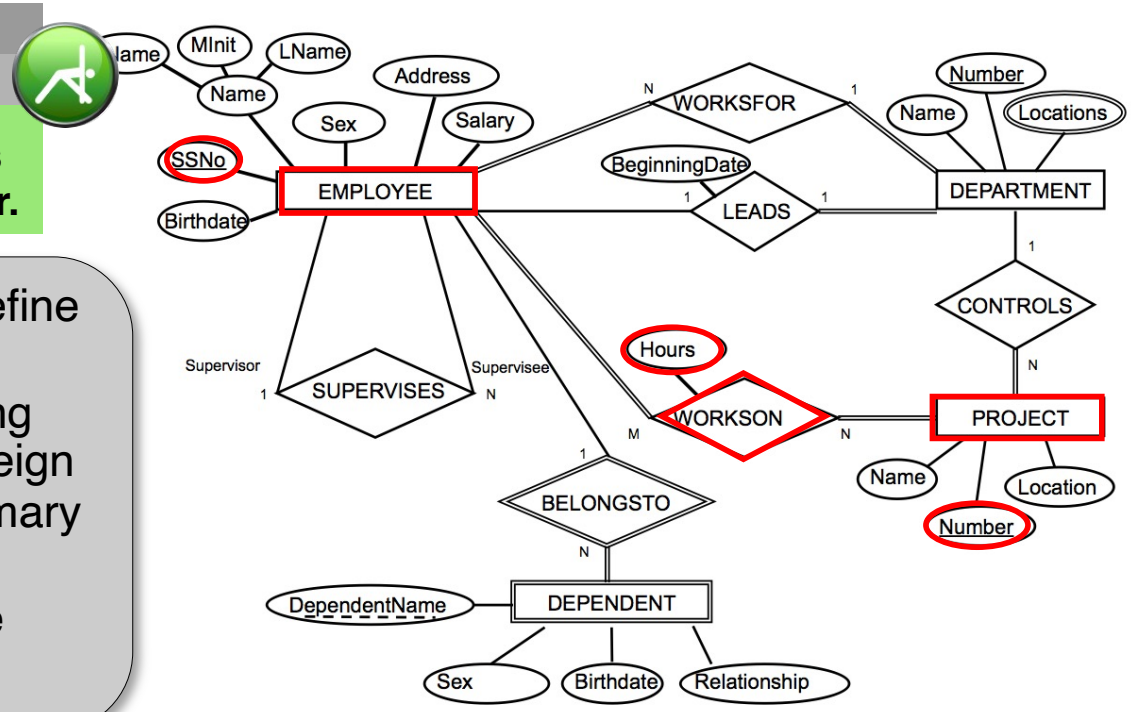
as **FK.attr**

PNAME	PROJNO	PLOCATION	PDEPTNO

```
CREATE TABLE user44.PROJECT (
        PROJNO          NUMERIC(7)      NOT NULL,
        PNAME           VARCHAR(25)     NOT NULL,
        PLOCATION       VARCHAR(25)     NOT NULL,
        PDEPTNO         NUMERIC(4)      NOT NULL,
        PRIMARY KEY(PROJNO),
        FOREIGN KEY(PDEPTNO) REFERENCES user44.DEPARTMENT (DEPTNO)
);
```

Step 5

Binary M:N relationship as a relation with two FK refer.



- For each binary M:N relationship we define a new relation
- The primary keys of the two participating relations will be referenced through foreign keys and will make up a composite primary key of the new relation
- Simple attributes of the relationship are added to the relation

DB Company

- For the M:N relationship type WORKSON we define a relation WORKSON
- PKs of the relations PROJECT and EMPLOYEE will be added in WORKSON (new Attribute names: PNO and ESSNO, becoming FK)
- Attribute HOURS will be stored in WORKSON

EMPLOYEE

EFNAME	INITAL	ENAME	SSNO	EBIRTHD
EADRESS	ESEX	SALARY	SUPERSSNO	EDEPTNO

DEPARTMENT

DEPTNAME	DEPTNO	MGRSSNO	MGRLEADSDATE

WORKSON

PROJECT

PNAME	PROJNO	PLOCATION	PDEPTNO

DEPENDENT

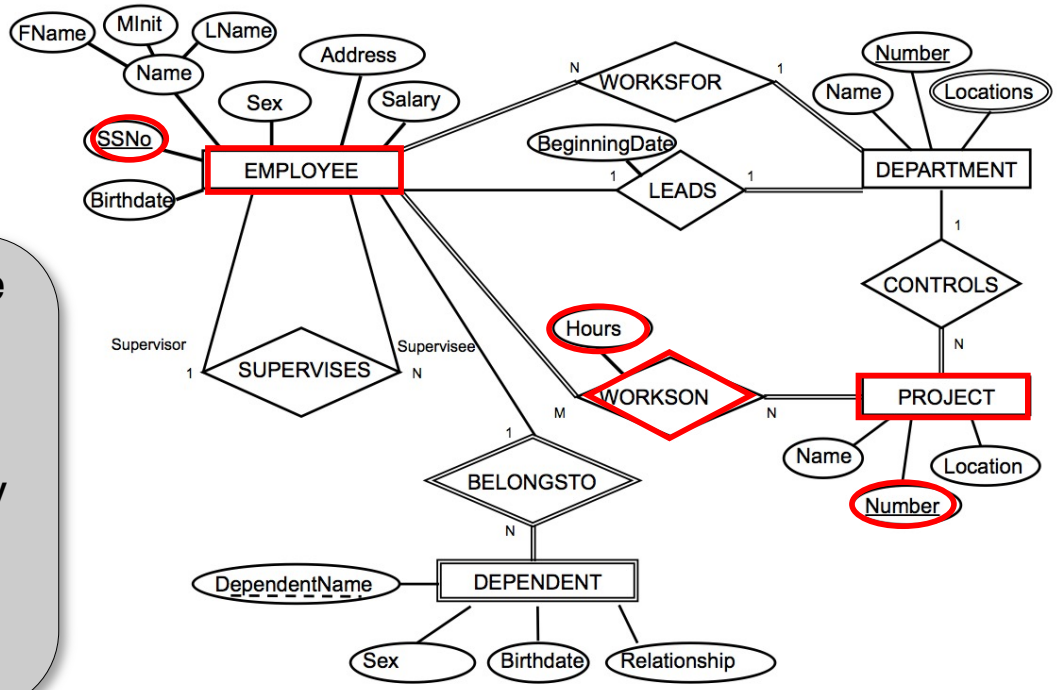
ESSNO	DEPENDNAME	DSEX	DBIRTHD	DEGREE

➲ Elmasri & Navathe (2014): Ch 9, pp. 287-296.

Step 5

Binary M:N relationship as a relation with two FK refer.

- For each binary M:N relationship we define a new relation
- The primary keys of the two participating relations will be referenced through foreign keys and will make up a composite primary key of the new relation
- Simple attributes of the relationship are added to the relation



DB Company

- For the M:N relationship type WORKSON we define a relation WORKSON
- PKs of the relations PROJECT and EMPLOYEE will be added in WORKSON (new Attribute names: PNO and ESSNO, becoming FK)
- Attribute HOURS will be stored in WORKSON

EMPLOYEE

EFNAME	INITAL	ENAME	SSNO	EBIRTHD
EADRESS	ESEX	SALARY	SUPERSSNO	EDEPTNO

PK.attr

DEPARTMENT

DEPTNAME	DEPTNO	MGRSSNO	MGRLEADSDATE

WORKSON

as FK.attr	ESSNO	PNO	HOURS

as FK.attr

PROJECT

PK.attr

PNAME	PROJNO	PLOCATION	PDEPTNO

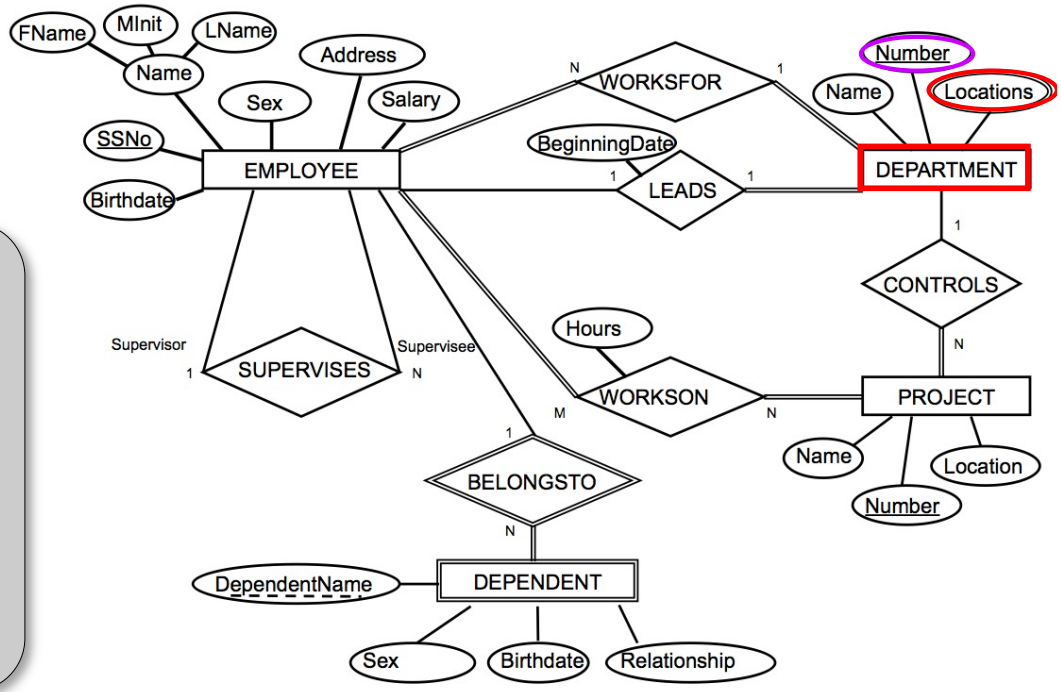
DEPENDENT

ESSNO	DEPENDNAME	DSEX	DBIRTHD	DEGREE

➲ Elmasri & Navathe (2014): Ch 9, pp. 287-296.

Step 6 Multi-valued attributes modeled as Entity types

- For each multi-valued attribute A, create a new relation
  - Add an attribute corresponding to A,
  - Add the PK of its owning entity type
  - If the multi-valued attribute is composite, add its simple components



DB Company

- Define relation DEPT_LOCATION
- Add a corresponding attribute
- Add the PK of DEPARTMENT

- PK of DEPT_LOCATION:
  - DEPTNO, DLOCATION

EMPLOYEE

EFNAME	INITAL	ENAME	SSNO	EBIRTHD
EADRESS	ESEX	SALARY	SUPERSSNO	EDEPTNO

PK.attr

DEPARTMENT

DEPTNAME	DEPTNO	MGRSSNO	MGRLEADSDATE

DEPT_LOCATIONS

DEPTNO	DLOCATION

as **FK.attr**

WORKSON

ESSNO	PNO	STUNDEN

PROJECT

PNAME	PROJNO	LOCNR	PDEPTNO

DEPENDENT

ESSNO	DEPENDNAME	DSEX	DBIRTHD	DEGREE

Step 7

For n-ary relationship types (n>2) new relations are introduced

(see practical 5)

- For each *n*-ary relationship type with *n*>2 we introduce a new relation
- Include as FK in the new relation the PK attributes of the relations representing participating entities
- The PK of the new relation is usually the combination of all the FKs of the participating relations.
  - If the cardinality constraints on any of the entity types E participating in R is 1, then the primary key of the new relation should not include the foreign key attribute that references the relation corresponding to E.
- Simple attributes of the n-ary relationship are represented as attributes of the relation

➲ Elmasri & Navathe (2014): Ch 9, pp. 287-296.

Summary 5.1

Correspondence between ER- and relational models

ER model	Step	Relational model
Entity type	1,2	Relation
Binary 1:1- or 1:N relationship type	3,4	FK (or a standalone relation also possible)
Binary M:N relationship type	5	Relation and two FKs
n-ary relationship type	7	Relation with n FKs
Simple attribute	1, ...	Attribute
Composite attribute	1, ...	Set of simple attributes
Multivalued Attribute	6	Relation and FK

➲ Elmasri & Navathe (2014): Ch 9, pp. 287-296.

Contents

1. **Logical DB design**
    1. Concepts of the relational model (Relation, Attribute, Tuple)
    2. DB schema integrity
    3. Mapping rules
    4. **Redundancy and anomalies** (→ Normalisation)
2. Physical DB design

Redundancy

Redundancy: the same information is stored multiple times in the tables (and thus in the DB)

Main rule for a relational database:

> § Each piece of information should only be stored once in a database!

Why?

- Reduce storage space
- Reduce inconsistencies (by avoiding update anomalies, see later)
- Reduce maintenance and update effort

Example

Very often seen in database...

EMP_DEPT

Ename	SNr	UnitNr	UnitName
Müller, Jens	12345678	1	Geographische Informationssysteme
Berger, Katrin	08155555	1	Geografische Informationssysteme
Schmid, Beat	35552661	1	Geographische Informationssysteme
Meier, Lorenz	12323434	2	Humangeographie

Redundancy and inconsistency (anomaly)

Problems of Redundancy: update anomalies

- Insertion anomalies
  - Occurs when a new inconsistent tuple is stored in a relation

- Deletion anomalies
  - Means that during the deletion of only one part of the stored information, an entire tuple with independent data is deleted

- Modification anomalies
  - Occurs, when the modification of one piece of information entails the modification of a number of values in other relations (e.g., geografische Informationssysteme)

➲ Elmasri & Navathe (2014): Ch 15, Section 1.2, pp. 514-516

Exercise 5.3

- Identify possibilities where update anomalies could occur in the relation EMP_DEPT

EMP_DEPT

Ename	SNr	UnitNr	UnitName	Floor
Müller, Jens	12345678	1	GIS	J
Berger, Katrin	08155555	1	GIS	J
Schmid, Beat	35552661	1	GIS	J
Meier, Lorenz	12323434	2	GIVA	L

Exercise 5.3

Insertion Anomalies

- Insert new employee – might easily insert inconsistent info (e.g., with incorrect floor value).
- Impossible to insert a new dept without employee – breaks PK rule

EMP_DEPT

Ename	SNr	UnitNr	UnitName	Floor
Müller, Jens	12345678	1	GIS	J
Berger, Katrin	08155555	1	GIS	J
Schmid, Beat	35552661	1	GIS	J
Meier, Lorenz	12323434	2	GIVA	L
Nova, Peter	54879504	**2**	GIS	**M**
NULL	**NULL**	3	PHYS	K

Integrity constraint
Violation: no PK – will fail

Can insert
Incorrect values
(should be avoided)

Exercise 5.3

Deletion anomaly

- Deletion of the last employee of a unit will cause a loss of the information concerning the unit.

EMP_DEPT

Ename	SNr	UnitNr	UnitName	Floor
Müller, Jens	12345678	1	GIS	J
Berger, Katrin	08155555	1	GIS	J
Schmid, Beat	35552661	1	GIS	J
Meier, Lorenz	12323434	2	GIVA	L
Nova, Peter	54879504	1	GIS	L
~~Stein, Hans~~	~~65843654~~	~~3~~	~~PHYS~~	~~K~~

Data for Unit 3 are lost

Exercise 5.3

Modification anomaly

- Change of an attribute in one record will cause a update to many tuples in the table
  - Change “Floor” info of unit “GIS” needs to update all employees who work in that unit.
  - Might easily cause inconsistency

EMP_DEPT

Ename	SNr	UnitNr	UnitName	Floor
Müller, Jens	12345678	1	GIS	J
Berger, Katrin	08155555	1	GIS	J
Schmid, Beat	35552661	1	GIS	J
Meier, Lorenz	12323434	2	GIVA	**L**
Nova, Peter	54879504	1	GIS	**L**

Inconsistent values

Exercise 5.3: Split the table into smaller ones to avoid the above problems

EMP_DEPT

Ename	SNr	UnitNr	UnitName	Floor
Müller, Jens	12345678	1	GIS	J
Berger, Katrin	08155555	1	GIS	J
Schmid, Beat	35552661	1	GIS	J
Meier, Lorenz	12323434	2	GIVA	L
Nova, Peter	54879504	2	GIVA	L
Stein, Hans	65843654	3	PHYS	K

...or rather:

EMPLOYEE

Ename	SNr	UnitNr
Müller, Jens	12345678	1
Berger, Katrin	08155555	1
Schmid, Beat	35552661	1
Meier, Lorenz	12323434	2
Nova, Peter	54879504	2
Stein, Hans	65843654	3

DEPARTMENT

UnitNr	UnitName	Floor
1	GIS	J
2	GIVA	L
3	PHYS	K
4	HUM	M

foreign key UnitNr

If the conceptual DB design (Requirement Text → ER-Diagram) and logical DB design (ER-Diagram → relational model) are properly done, the above redundancy and anomalies can be avoided.

Normalization

- Systematic analysis of the dependencies among attributes in a relation helps to achieve a high quality design of relations
  - **Aim**: the quality of a DB design process should be measurable and certifiable
    - Minimize redundancy of data stored in a table
    - Minimize update anomalies
  - Achieved through the **process** of normalization
    - Test the relations based on their PKs and FKs to achieve the progressively stronger properties of relations
    - Alter attribute grouping by splitting relations into smaller ones, in order to progress to higher normal forms satisfying certain conditions
    - A series of tests that can be carried out by the DB designer on relation schemas to certify that they satisfy a given normal form

Normalization

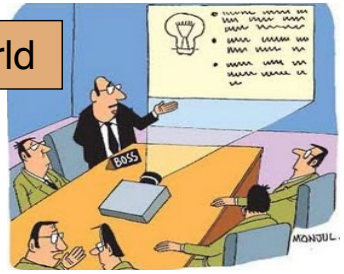
- 1. Normal form
- 2. Normal form
- 3. Normal form
- 4. Normal form
- 5. Normal form

- Please refer to the appendix slides for more details.
- We will not cover this part in the exercise and exam.

Contents

DB Design

Real World



Entities in the real world: Department head Müller, Employee Muster, Project 'Budget 2014'

Lecture 3 Requirements Analysis

Interview of users, study of documentation

Database Requirements

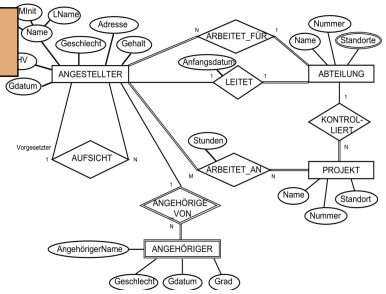
1. Die Firma ist in Abteilungen organisiert. Jede Abteilung hat eine eindeutige Beziehung, eine eindeutige Nummer und einen bestimmten Angestellten registrieren das Anfangsdatum, ab dem dieser Abteilung übernommen hat. Eine Abteilung verfügt
2. ne Reihe von Projekten, die jeweils einen deutige Nummer und einen einziger Standor haben.
3. Jeder Angestellte wird mit Namen, AHV-Nr., Adresse, Gehalt, Geschlech und Geburtsdatum gespeichert. Ein Angestellter wird einer Abteilung zugewiesen, kann aber an mehreren Projekten arbeiten, die nicht unbedingt alle von der gleichen Abteilung kontrolliert werden. Wir verfolgen die Stundenzahl pro Woche, die ein Angestellter an jedem Projekt arbeitet, und den unmittelbaren Vorgesetzten jedes Angestellten.
4. Zu Versicherungszwecken sollen die Familienangehörigen jedes Mitarbeiters mit Vorname, Geschlecht, Geburtsdatum und Verwandtschaftsgrad zum jeweiligen Angestellten erfasst werden.

Written concise description of requirements (data reqs and functional [operations] reqs)

Lecture 4 Conceptual DB design

Text analysis, Entities, Relationships

ER-Model



Entity types with attributes and relationships

DBMS-independent
DBMS-specific

Lecture 5 Logical DB design (data model mapping)

- translation of model to implementation
- Validation through normalisation

DB Schema (e.g., relational data model)

GESTELLTER

VNAME	INITAL	NNAME	AHV	GDATUM
DRESSE	GESCHLECHT	GEHALT	SUPERAHV	ANR

TEILUNG

ANAME	ABTNUMMMER	MGRAHV	MGR_ANFANGSDATUM

T_STANDORTE

BTNUMMMER	ASTANDORT

ARBEITET_AN

EAHV	PNR	STUNDEN

PROJEKT

PNAME	PNUMMER	PSTANDORT	ABTNR

ANGEHÖRIGER

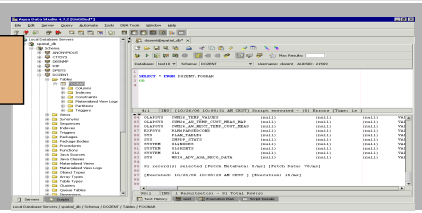
EAHV	ANGEHÖRIGER_NAME	GESCHLECHT	GDATUM	GRAD

Set of relations with attributes (incl. data types + value domains)

Lecture 5 Physical DB design

Description of database implementation (HW, indices, …)

Database



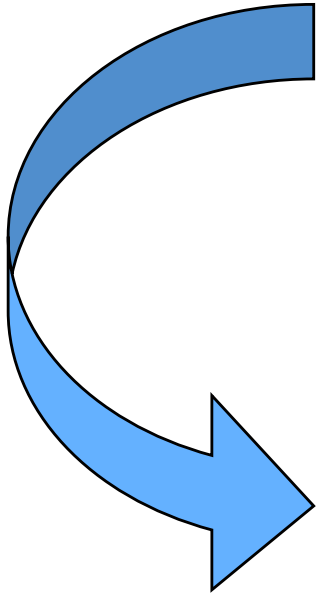
Implemented DB design, Tables (ordered) with rows and columns

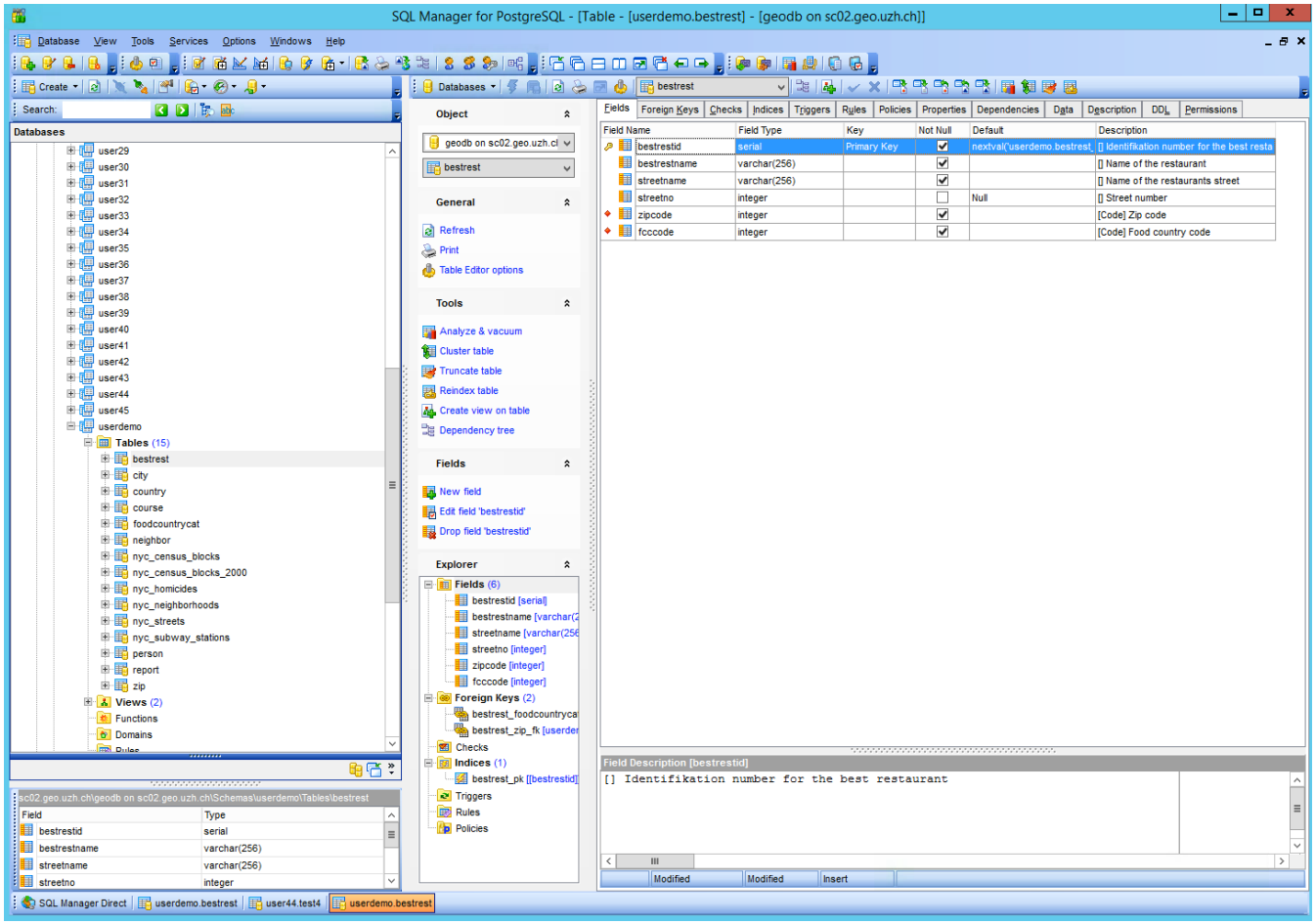
➲ Elmasri & Navathe (2007): ch3.1, p.59f., part 3 (ch 10,11)

Physical DB design

EMPLOYEE

EFNAME	INITAL	ENAME	SSNO	EBIRTHD
EADRESS	ESEX	SALARY	SUPERSSNO	EDEPTNO





Physical DB design

- Conversion of the logical design (relations, attributes) into a physical DB design that allows the DBS to be implemented
  - Translation of the relational data model into the target DBMS system (e.g., PostgreSQL)
    - Definition of tables through SQL commands or the Graphic UI
    - E.g., from the **relation** EMPLOYEE we get the **TABLE** EMPLOYEE
    - Choose the right data types
  - Design of the physical realization (e.g., storage, data structures, access rights, indexes, …)
  - Design of the security mechanisms
  - Monitoring and tuning of the system

- Practices at the lab in next week

Physical DB design

- Designing Fields/attributes
  - Choosing data types that
    - Minimize storage space
    - Represent all possible values
    - Improve data integrity (eliminate illegal values)
    - Support all data manipulation
    - Examples of data types: integer, numeric, varchar, date/time, …
  - Integrity: default value, range control, null value control, referential integrity (foreign-key), …

```
CREATE TABLE user01.Person (
     PersNr    numeric(7) PRIMARY KEY,
     LastName  varchar(25) NOT NULL,
     FirstName varchar(25) NOT NULL,
     PLZ       numeric(4) REFERENCES Ort (Zip)
)
```

Physical DB design

- Rules for using indexes
  - Use on larger tables
  - Index the primary key of each table (default in PostgreSQL)
  - Index fields that frequently appear in the WHERE clause
  - Fields often appearing in ORDER BY and GROUP BY commands
  - Avoid use of indexes for fields with long values
  - For fields with >100 unique values but not for fields with < 30 unique values
  - DBMS may have limit on number of indexes per table

Summary

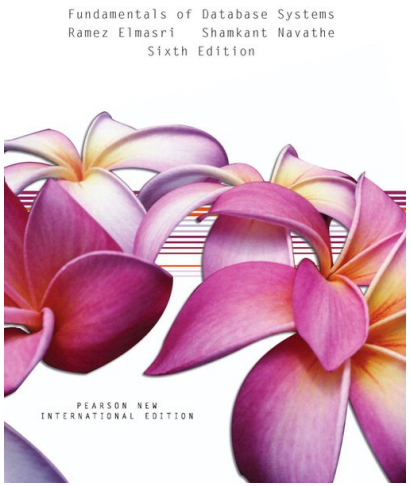
- Logical DB design
  - DB schema integrity
  - Translate/Map an ER-diagram into a relational DB model
    - Mapping rules
  - Redundancy: anomalies
- Physical DB design

Resources

- Ramez A. Elmasri, Shamkant B. Navathe (2014). Fundamentals of Database Systems, 6th edition, ISBN: 978-1-292-02560-5, Pearson, 1081 pages.

  Ch 9: ER-to-Relational Mapping

  Ch 15: Functional Dependencies and Normalization for Relational Database



Later, in lab

- 10:15-12:00
- Y25-J-09, Y25-J-10
- **Logical DB design (with pencil and paper)**