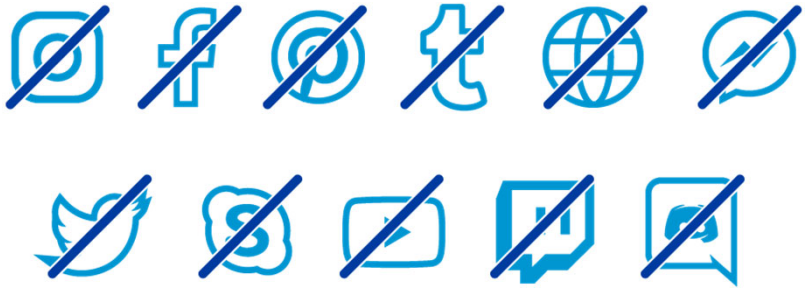
DATABASES

Source: https://en.itpedia.nl/2017/11/26/wat-is-een-database/

Prof. Dr. Ulrike Herster
Hamburg University of Applied Sciences

HAW HAMBURG

COPYRIGHT



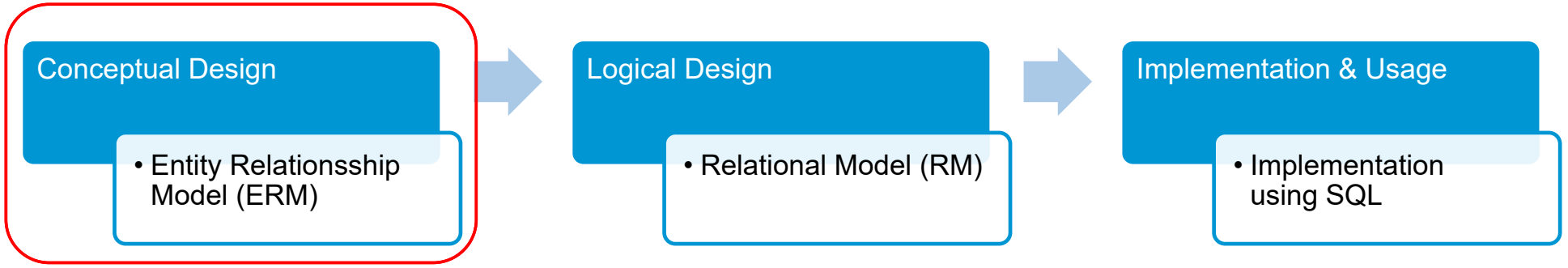
The publication and sharing of
slides, images and sound recordings of this
course is not permitted

ORGANIZATION
OUR JOURNEY IN THIS SEMESTER

- Integrity, Trigger & Security
- Database Applications
- Transactions
- Subqueries & Views
- More Features
- Notations & Guidelines
- Constraints
- **Relationships**
- Simple Entities and Attributes
- Basics

Source: Foto von Justin Kauffman auf Unsplash

RELATIONSHIPS
DATABASE DESIGN



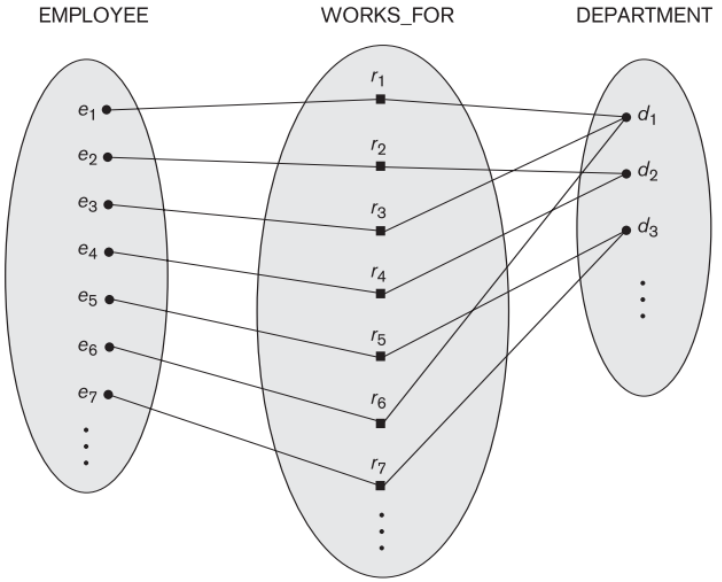
RELATIONSHIPS
ERM: RELATIONSHIP TYPES

- Describe relationships between entity types
- Characterized by a verb
  → Often 2 naming possibilities:
  - teaches vs. is taught by
  - Relationship has always two (or more) directions
- May have attributes
- Number of participating entity types (degree):
  1. Unary relationship type (e.g., Employee supervises another employee)
  2. Binary relationship type (e.g., Employee works for one department)
  3. Ternary relationship type (e.g., Lecturer recommends books for one specific course)
  4. Higher degrees...

RELATIONSHIPS
ERM: RELATIONSHIP TYPES

- Each relationship instance r_i in R is an association of entities, where the association includes exactly one entity from each participating entity type

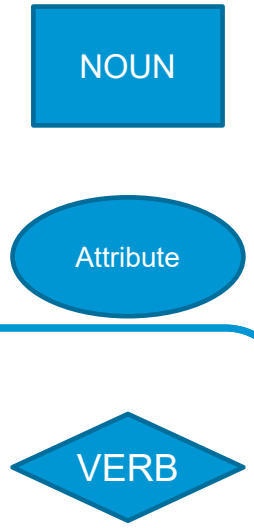
- In an ERM, relationship types are displayed as diamond-shaped boxes, which are connected by straight lines to the entity types



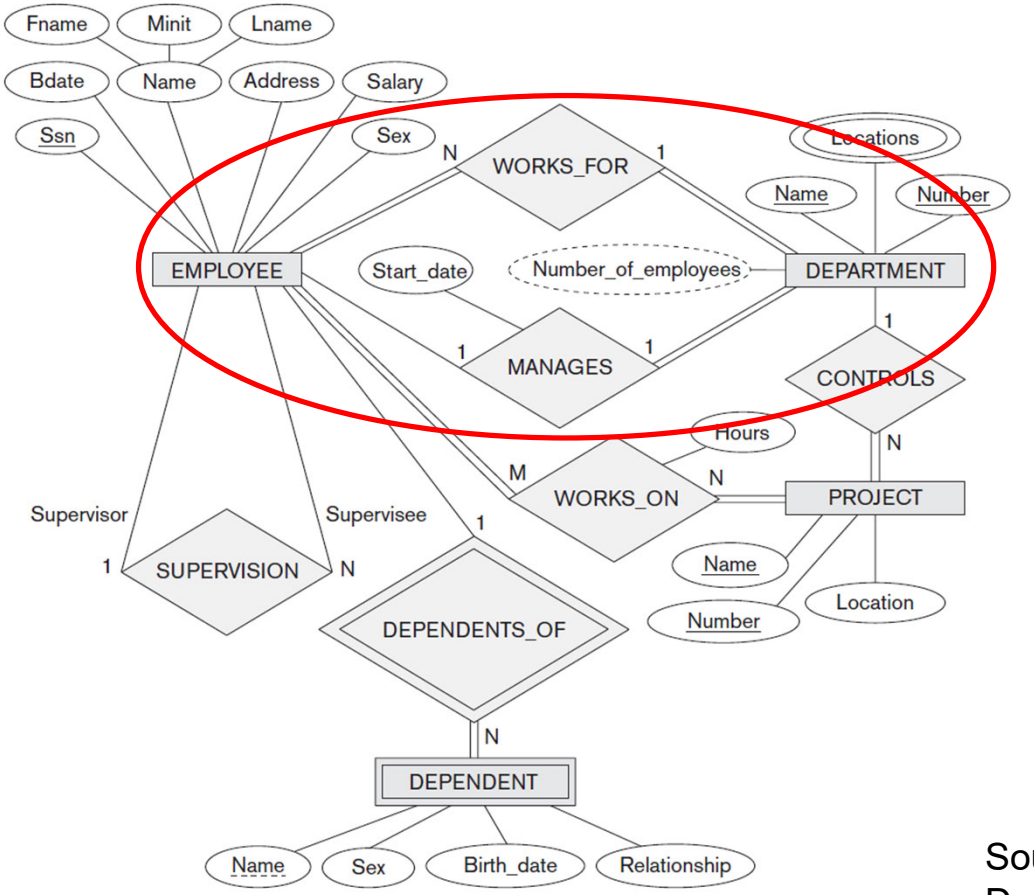
Source: Elmasri, Fundamentals of Database Systems, Page 212 ff

RELATIONSHIPS
ERM: RELATIONSHIP TYPES

- Entity Type
  - Represented as rectangle in ERM
  - Singular noun
- Attribut Type
  - Represented as ovals
  - Noun
- Relationship Type
  - Represented as diamond in ERM
  - Always between entity types
  - Verb
  - Has cardinalities

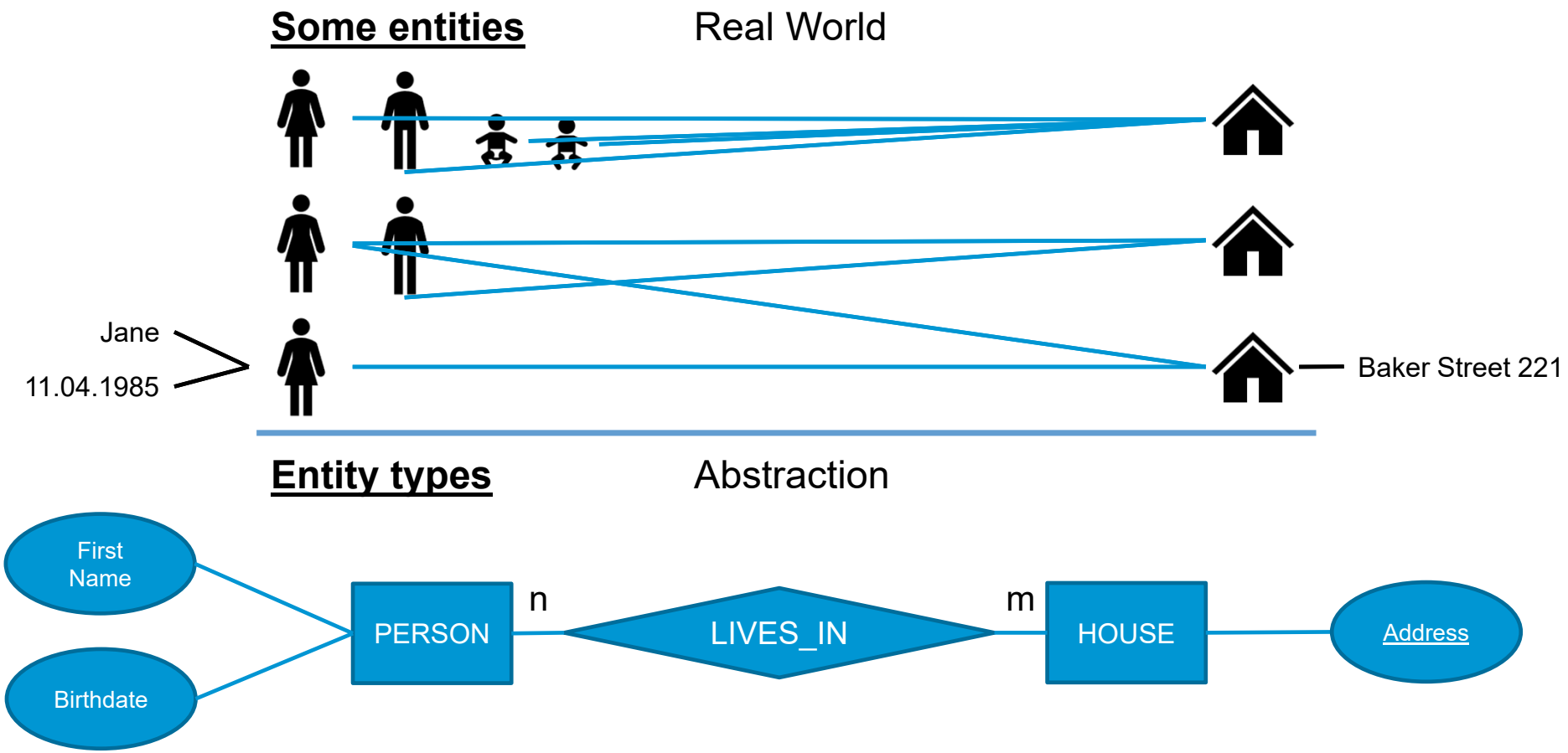


RELATIONSHIPS
ERM: EXAMPLE - COMPANY



Source: Elmasri, Fundamentals of Database Systems, Page 204 ff

RELATIONSHIPS
ERM: RELATIONSHIP TYPES - ABSTRACTION



RELATIONSHIPS
ERM: RELATIONSHIP TYPES – ROLE NAMES

- The **role name** signifies the role that a participating entity from the entity type plays in each relationship instance, and helps to explain what the relationship means

Source: Elmasri, Fundamentals of Database Systems, Page 212 ff

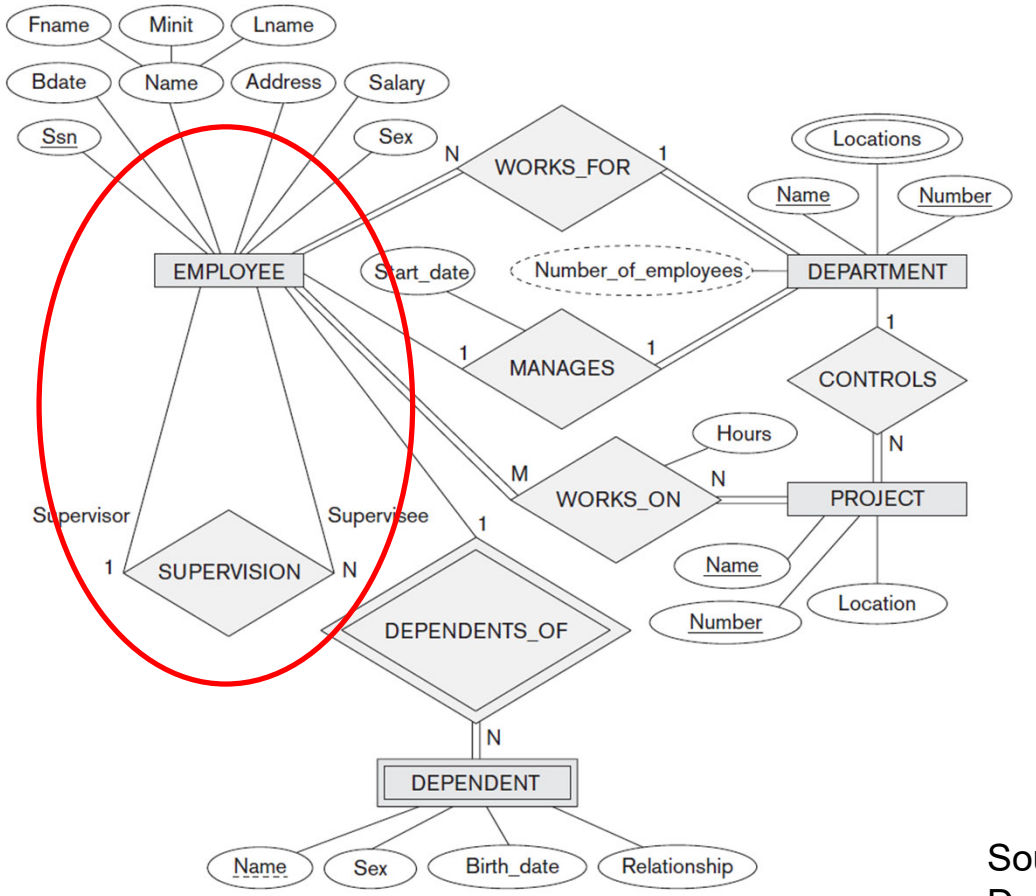
RELATIONSHIPS
ERM: RELATIONSHIP TYPES – RECURSIVE RELATIONSSHIP TYPES

- In some cases, the same entity type participates more than once in a relationship type in different roles
- In such cases the **role name** becomes essential for distinguishing the meaning of the role that each participating entity plays

→ Such relationship types are called **recursive relationship types**

Source: Elmasri, Fundamentals of Database Systems, Page 212 ff

RELATIONSHIPS
ERM: EXAMPLE - COMPANY

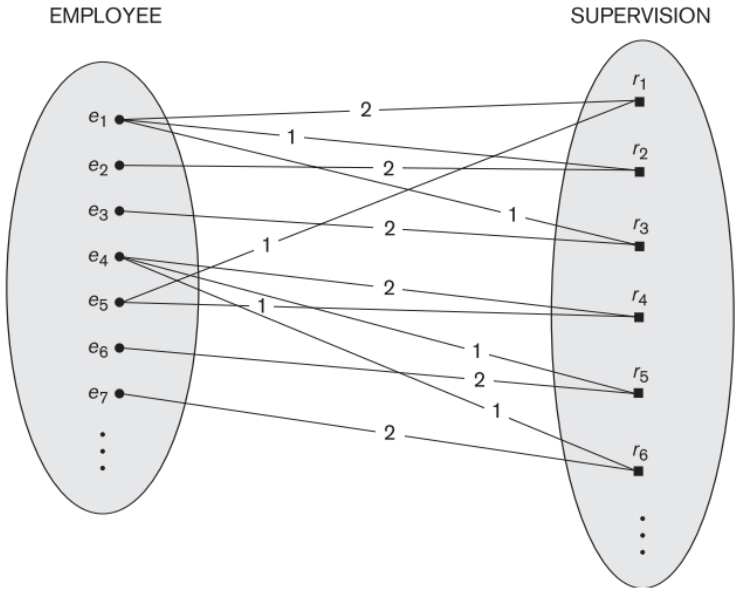


Source: Elmasri, Fundamentals of Database Systems, Page 204 ff

RELATIONSHIPS
ERM: RELATIONSHIP TYPES – EXAMPLE

Example: Employee in 2 roles

- Supervisor (boss) → role name 1
- Supervisee (subordinate) → role name 2



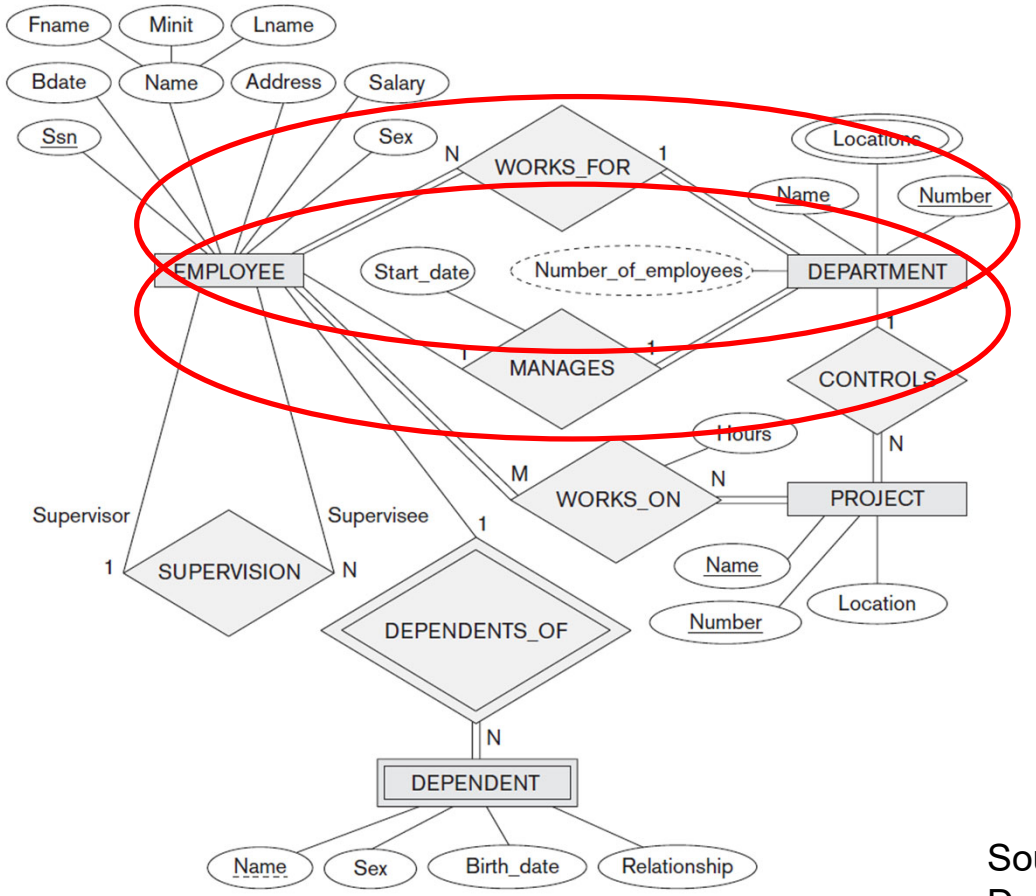
Source: Elmasri, Fundamentals of Database Systems, Page 212 ff

RELATIONSHIPS
ERM: RELATIONSHIP TYPES – CONSTRAINTS

- Cardinality
  - Specifies the maximum number of relationship instances that an entity can participate in
  - Cardinality ratios
    - 1:1
    - 1:N
    - M:N
  - Cardinality ratios for binary relationships are represented on ER diagrams by displaying 1, M, and N on the diamonds
  - Notice that in this notation, we can either specify no maximum (N) or a maximum of one (1) on participation

Source: Elmasri, Fundamentals of Database Systems, Page 212 ff

RELATIONSHIPS
ERM: EXAMPLE - COMPANY



Source: Elmasri, Fundamentals of Database Systems, Page 204 ff

RELATIONSHIPS
ERM: RELATIONSHIP TYPES – CONSTRAINTS

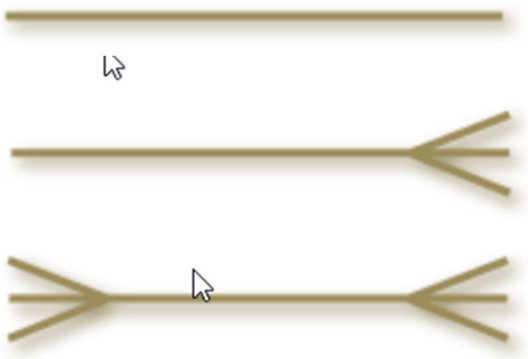
- Cardinality
  - (min,max) Notation
  - Example
    - A car has at least 3 and at most 5 wheels
      Every wheel is associated to exactly one car

  - Attention: In UML, (min,max) is placed on the opposite sites!
  - Problem: General case cannot be easily implemented in RDBMS

Source: Elmasri, Fundamentals of Database Systems, Page 212 ff

RELATIONSHIPS
ERM: RELATIONSHIP TYPES – CONSTRAINTS

- Other notations (e.g., in tools like draw.io)
  - 1:1
  - 1:N
  - M:N



RELATIONSHIPS
ERM: RELATIONSHIP TYPES – CONSTRAINTS

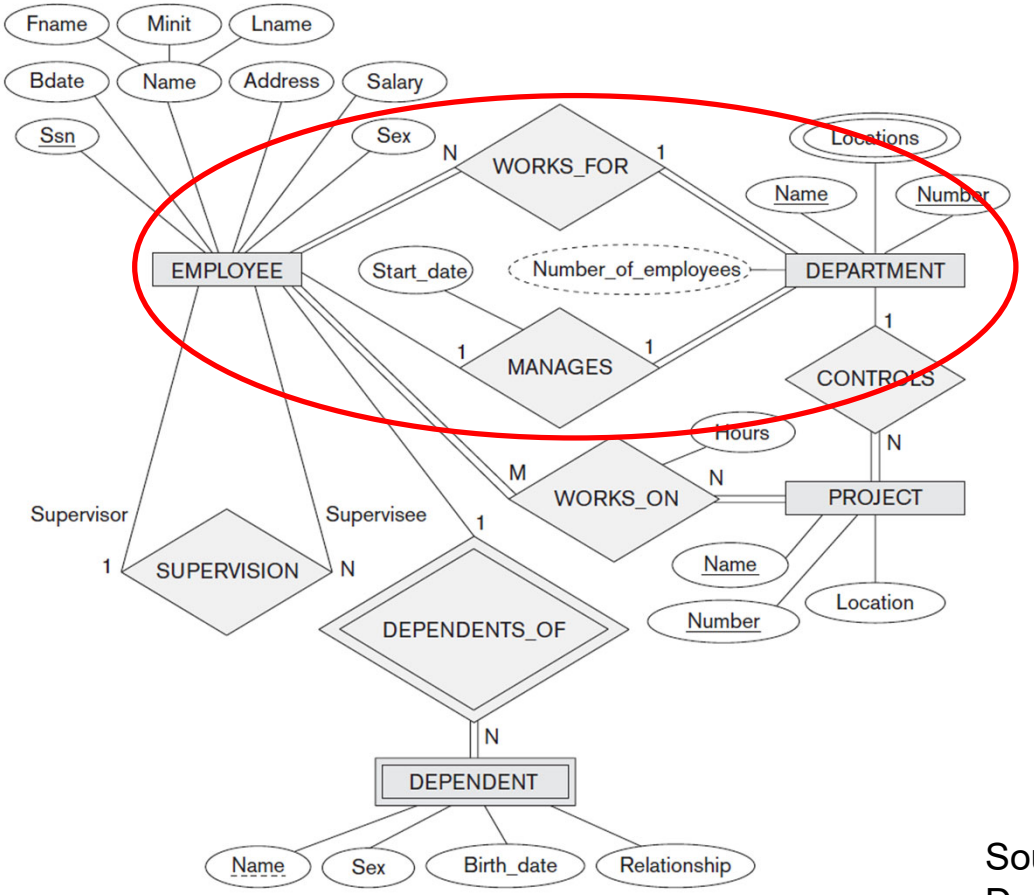
- Participation
  - Specifies whether the existence of an entity depends on its being related to another entity via the relationship type
  - Also called *minimum cardinality constraint*
  - Two types
    - Total: every entity in the total set of all entities of an entity type A must be related to an entity of entity type B via a relationship
      → Total participation is also called *existence dependency*
      - Is displayed as a double line connecting the participating entity type to the relationship
    - Partial: some or part of the entities of an entity type A are related to some entities of an entity type B via a relationship
      - Is displayed by a single line connecting the participating entity type to the relationship

Source: Elmasri, Fundamentals of Database Systems, Page 212 ff

RELATIONSHIPS
ERM: RELATIONSHIP TYPES – CONSTRAINTS

- Cardinality
  → specifies the maximum number of relationship instances that an entity can participate in
- Participation
  → specifies if the existence of an entity depends on its being related to another entity via the relationship type
  → *minimum cardinality constraint*

RELATIONSHIPS
ERM: EXAMPLE - COMPANY



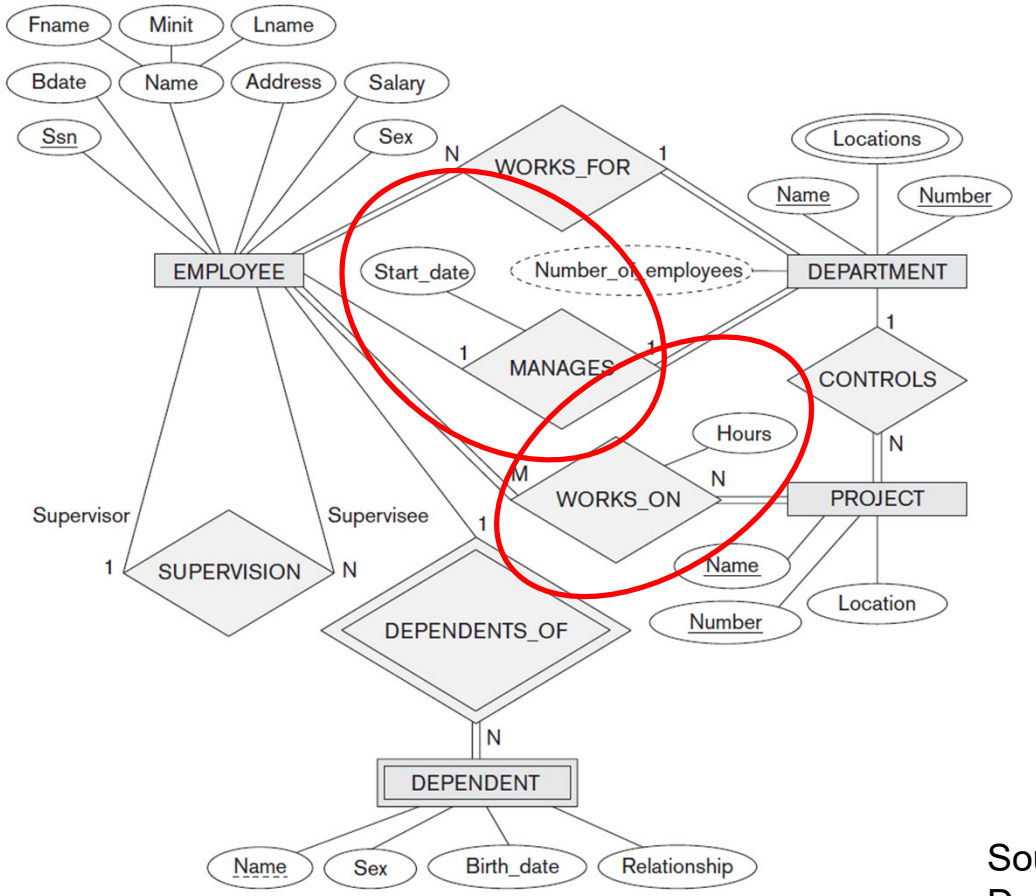
Source: Elmasri, Fundamentals of Database Systems, Page 204 ff

RELATIONSHIPS
ERM: RELATIONSHIP TYPES – ATTRIBUTES

- Relationship types can also have attributes
- Notice that attributes of 1:1 or 1:N relationship types can be migrated to one of the participating entity types
- For M:N relationship types, some attributes may be determined by the combination of participating entities in a relationship instance, not by any single entity
  → Such attributes must be specified as relationship attributes

Source: Elmasri, Fundamentals of Database Systems, Page 212 ff

RELATIONSHIPS
ERM: EXAMPLE - COMPANY

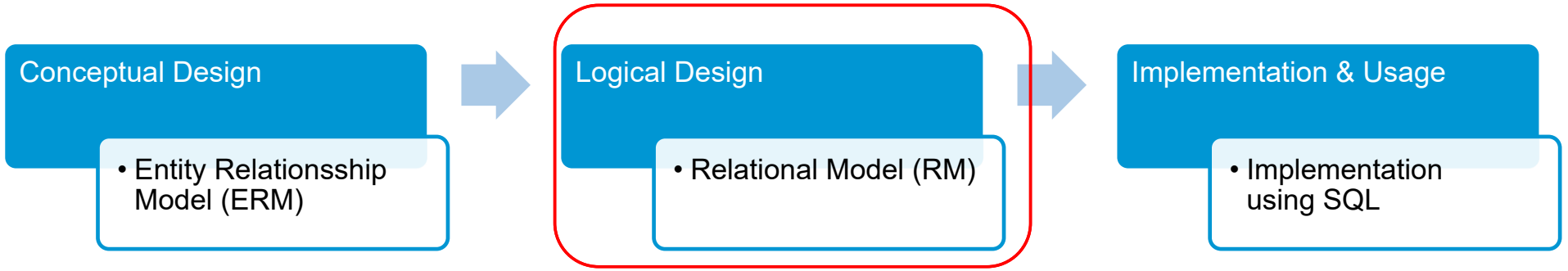


Source: Elmasri, Fundamentals of Database Systems, Page 204 ff

RELATIONSHIPS
ERM: RELATIONSHIP TYPES – HOW TO DEFINE THEM?

- Relationship between entity types
- Good naming
- More than one relationship?
  - Maybe different meanings, roles
  - Example for role: Supervisor, Supervisee
- Cardinalities
- Mandatory/optional
- Attributes for Relationship Type?

RELATIONSHIPS
DATABASE DESIGN



RELATIONSHIPS
RM: CONSTRAINTS

Three categories

1. Constraints that are inherent in the data model
   → *inherent model-based constraints* or *implicit constraints*
   Example: no duplicate tuples in a relation
2. Constraints that can be directly expressed in schemas of the data model
   → *schema-based constraints* or *explicit constraints*
   Example: Domain constraints, primary key (entity integrity constraints), constraints on NULL, and **referential integrity constraints**
3. Constraints that cannot be directly expressed in the schemas of the data model, and hence must be expressed and enforced by the application programs
   → *application-based* or *semantic constraints* or *business rules*

Source: Elmasri, Fundamentals of Database Systems, Page 59ff

RELATIONSHIPS
RM: CONSTRAINTS – REFERENTIAL INTEGRITY CONSTRAINT

- It is defined between two relations
- It is used to maintain the consistency among tules in the two relations: a tuple in one relation that refers to another relation must refer to an existing tuple in that relation
- *Foreign key*: a set of attributes FK in relation schema R_1 is a foreign key of R_1 that references relation R_2 if it satisfies the following rules:
  1. The attributes in FK have the same domain(s) as the primary key attributes PK of R_2 ; the attributes FK are said to *reference* or *refer to* the relation R_2
  2. A value of FK in a tuple t_1 of the current state $r_1(R_1)$ either occurs as a value of PK for some tuple t_2 in the current state $r_2(R_2)$ or is NULL. In the former case, we have $t_1[FK] = t_2[FK]$, and we say that the tuple t_1 references or refers to the tuple t_2

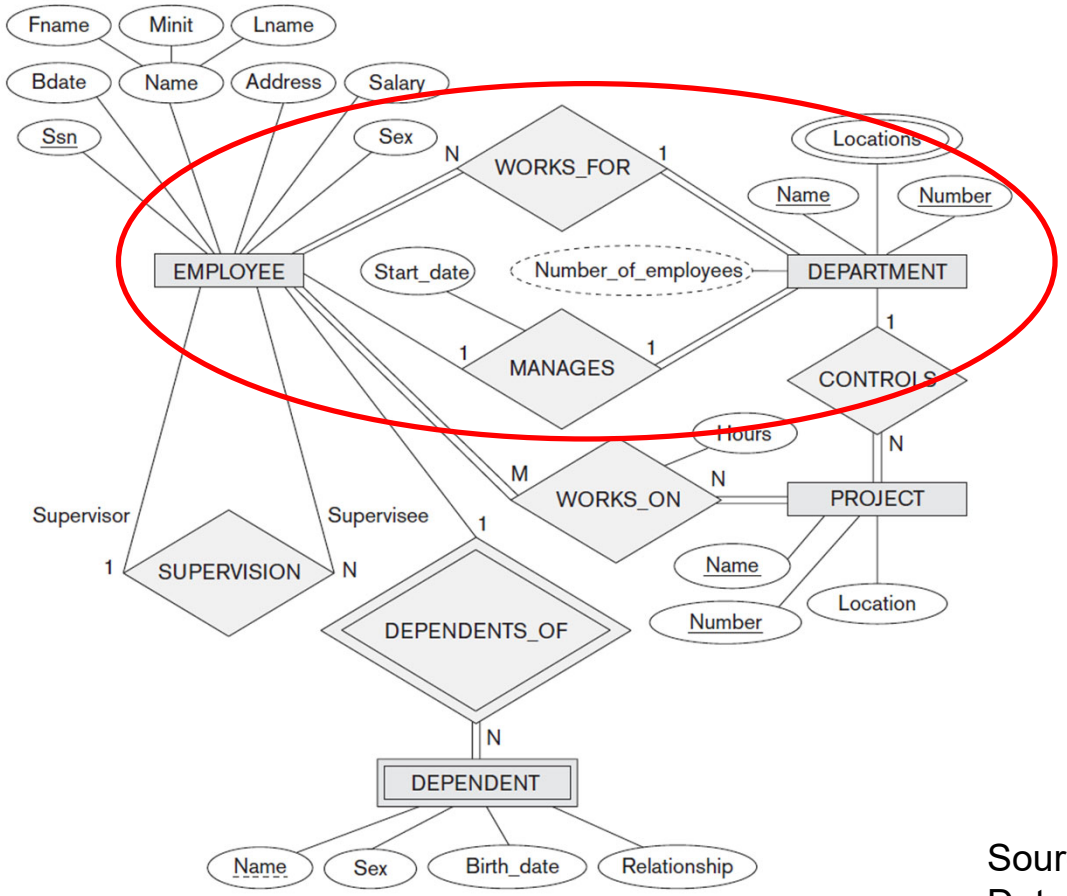
Source: Elmasri, Fundamentals of Database Systems, Page 59ff

RELATIONSHIPS
RM: CONSTRAINTS – REFERENTIAL INTEGRITY CONSTRAINT

- A foreign key can refer to its own relation
- Foreign keys are depicted with a directed arrow
  → The arrowhead may point to the primary key
- All integrity constraints can be defined with the DDL,
  thus the DBMS can automatically enforce them

Source: Elmasri, Fundamentals of Database Systems, Page 59ff

RELATIONSHIPS
ERM: EXAMPLE - COMPANY



Source: Elmasri, Fundamentals of Database Systems, Page 204 ff

RELATIONSHIPS
RM: CONSTRAINTS – EXAMPLE

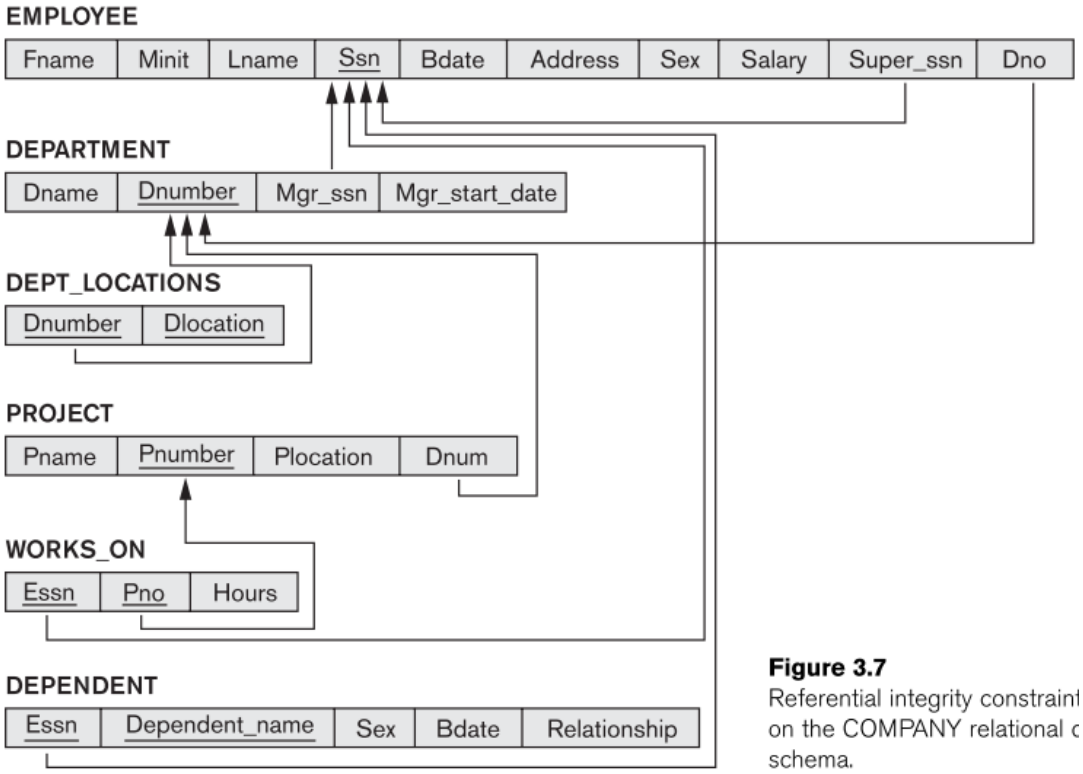


Figure 3.7
Referential integrity constraint
on the COMPANY relational c
schema.

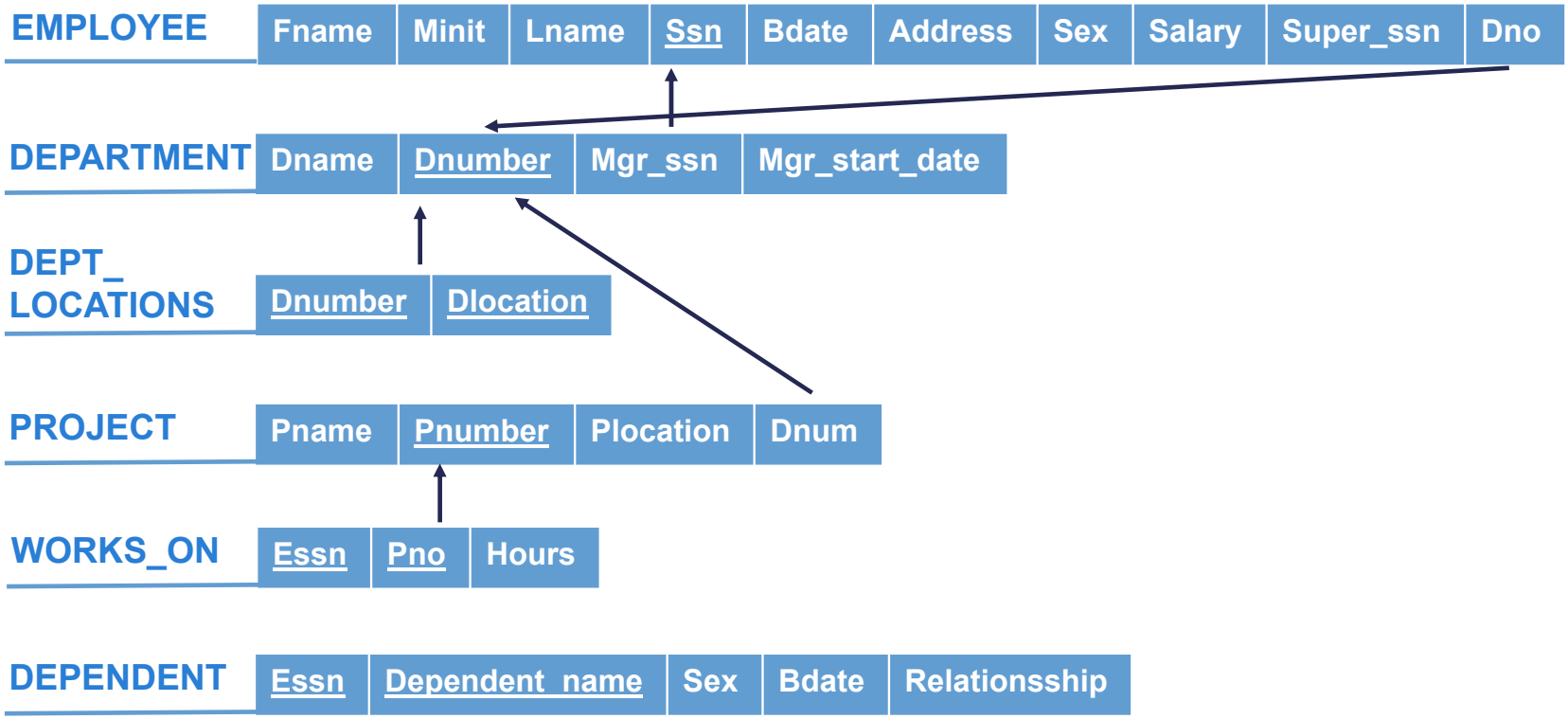
Source: Elmasri, Fundamentals of Database Systems, Page 59ff

RELATIONSHIPS
RM: NOTATION OF FOREIGN KEYS WITHIN A RELATIONAL SCHEMA

- There are several ways for the notation of relational schemas
  → especially for representing foreign keys, e.g.
  - Option 1:
    - Foreign Keys can be represented with arrows
    - This notation is used in the lecture slides and in the book „Fundamentals of Database Systems" from Elmasri and Navathe
    - Advantage: Each FK-arrow connects the referencing attribute and referenced attribute, so the involved relations are obvious
  - Option 2:
    - Foreign Keys can be represented with addition (FK) within the referencing attribute
    - This notation is used in the laboratory of Mr. Ocker
    - Advantage: This notation is more readable for large, complex schemas
  - Both notations are correct and may be used within the examination

RELATIONSHIPS
RM: NOTATION OF FOREIGN KEYS WITHIN A RELATIONAL SCHEMA

- Example: Some foreign Keys with option 1



RELATIONSHIPS
RM: NOTATION OF FOREIGN KEYS WITHIN A RELATIONAL SCHEMA

- Example: Some foreign Keys with option 2

EMPLOYEE	Fname	Minit	Lname	Ssn	Bdate	Address	Sex	Salary	Super_ssn	Dno

DEPARTMENT	Dname	Dnumber	Mgr_ssn (FK)	Mgr_start_date

DEPT_ LOCATIONS	Dnumber (FK)	Dlocation

PROJECT	Pname	Pnumber	Plocation	Dnum (FK)

WORKS_ON	Essn	Pno (FK)	Hours

DEPENDENT	Essn	Dependent_name	Sex	Bdate	Relationsship

RELATIONSHIPS
RM: MAPPING OF ERM TO RELATIONAL MODEL

Seven Steps

1. Mapping of regular entity types
2. Mapping of weak entity types
3. **Mapping of binary 1:1 relationships**
4. **Mapping of binary 1:n relationships**
5. **Mapping of binary m:n relationships**
6. Mapping of multivalued attributes
7. Mapping of n-ary relationships

Source: Elmasri, Fundamentals of Database Systems, Page 286ff

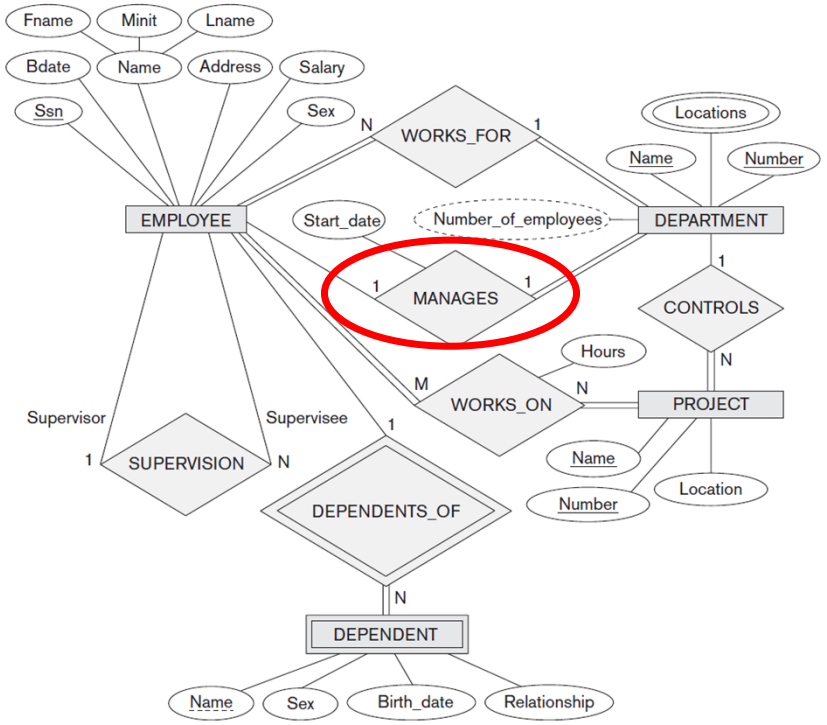
RELATIONSHIPS
RM: 3. MAPPING OF BINARY 1:1 RELATIONSHIPS

- For each binary 1:1 relationship type R in the ER schema,
  identify the relations S and T that correspond to the entity types participating in R
- There are three possible approaches:
  1. The foreign key approach
  2. The merged relationship approach
  3. The cross-reference or relationship relation approach

Source: Elmasri, Fundamentals of Database Systems, Page 286ff

RELATIONSHIPS
RM: 3. MAPPING OF BINARY 1:1 RELATIONSHIPS

1. The foreign key approach
   - Choose one of the relations S and include as a foreign key in S the primary key of T
   - It is better to choose an entity type with total participation in R in the role of S
   - Include all the simple attributes (or simple components of composite attributes) of the 1:1 relationship type R as attributes of S



Source: Elmasri, Fundamentals of Database Systems, Page 286ff

RELATIONSHIPS
RM: 3. MAPPING OF BINARY 1:1 RELATIONSHIPS

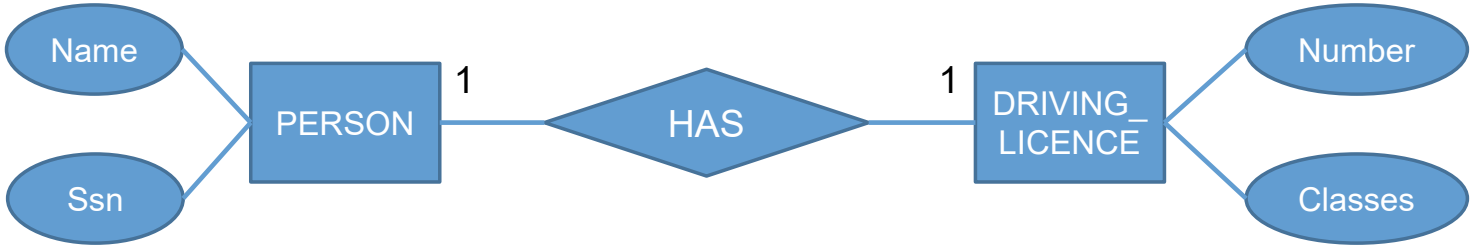
- Mapping of relationship type MANAGES
  - DEPARTMENT serves as S
  - EMPLOYEE serves as T
- Attribute Ssn is renamed in Mgr_ssn in DEPARTMENT
- Attribute Start_date is renamed in Mgr_start_date in DEPARTMENT
- It is also possible to include primary key of S as foreign key in T
- For the mapping, a **UNIQUE-Constraint** must be used!!!
  - Otherwise, an employee could manage several departments!

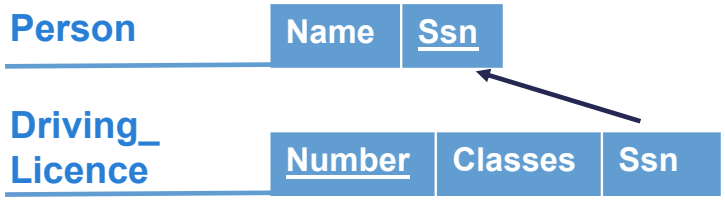
Source: Elmasri, Fundamentals of Database Systems, Page 286ff

RELATIONSHIPS
RM: 3. MAPPING OF BINARY 1:1 RELATIONSHIPS

- For the mapping, a **UNIQUE-Constraint** must be used!

- Example:





Source: Elmasri, Fundamentals of Database Systems, Page 286ff

RELATIONSHIPS
RM: 3. MAPPING OF BINARY 1:1 RELATIONSHIPS

2. Merged relation approach
   - Merge the two entity types and the relationship into a single relation
   - This is possible when both participations are total, as this would indicate that the two tables will always have the exact same number of tuples

EMPLOYEE_IN_DEPARTMENT	Fname	Minit	Lname	Ssn	Dname	Dnumber	…

Source: Elmasri, Fundamentals of Database Systems, Page 286ff

RELATIONSHIPS
RM: 3. MAPPING OF BINARY 1:1 RELATIONSHIPS

3. The cross-reference or relationship relation approach
   - Set up a third relation R for the purpose of cross-referencing the primary keys of the two relations S and T representing the entity types
   - This approach is required for binary M:N relationships
   - The relation R will include the primary key attributes of S and T as foreign keys to S and T
   - The primary key of R will be one of the two foreign keys, and the other foreign key will be a unique key of R

MANAGES	Ssn	Dnumber

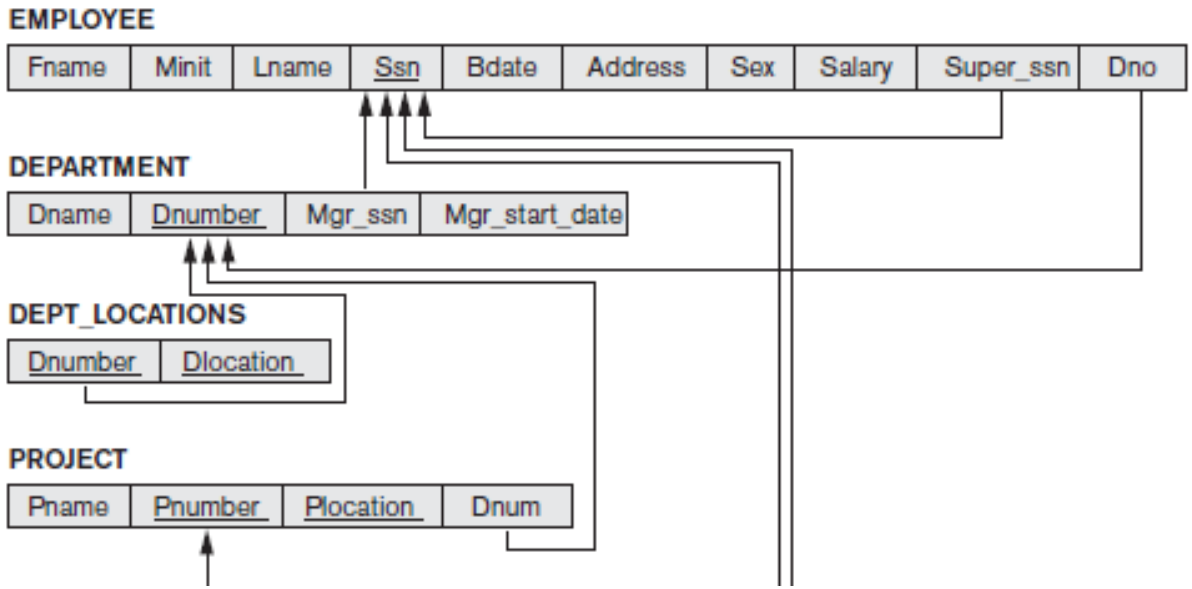
Source: Elmasri, Fundamentals of Database Systems, Page 286ff

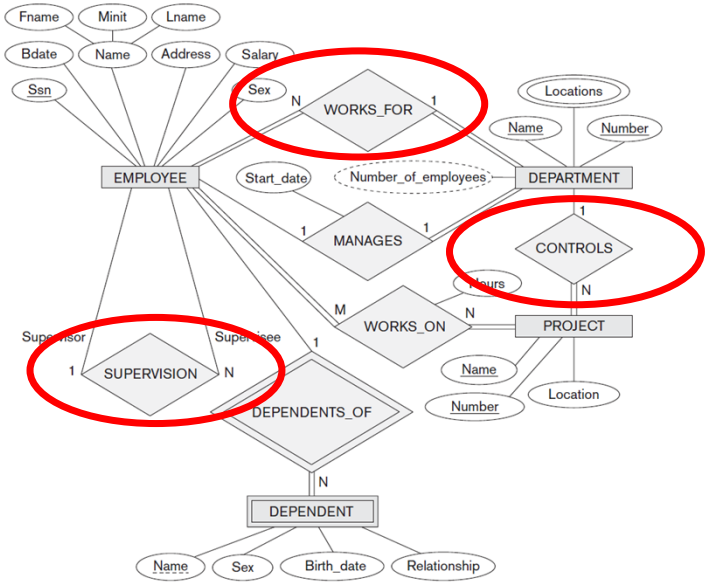
RELATIONSHIPS
RM: 4. MAPPING OF BINARY 1:N RELATIONSHIPS

- For each regular binary 1:N relationship type R, identify the relation S that represents the participating entity type at the N-side of the relationship type
- Include as foreign key in S the primary key of the relation T that represents the other entity type participating in R
- Include any simple attributes (or simple components of composite attributes) of the 1:N relationship type as attributes of S
- An alternative approach: use the relationship relation and create a separate relation

Source: Elmasri, Fundamentals of Database Systems, Page 286ff

RELATIONSHIPS
RM: 4. MAPPING OF BINARY 1:N RELATIONSHIPS





- Relationship type WORKS_FOR: Attribute Dno as foreign key in EMPLOYEE
- Relationship type SUPERVISION: Attribute Super_ssn as foreign key in EMPLOYEE
- Relationship type CONTROLS: Attribute Dnum as foreign key in PROJECT

Source: Elmasri, Fundamentals of Database Systems, Page 286ff

RELATIONSHIPS
RM: 4. MAPPING OF BINARY 1:N RELATIONSHIPS - TOTAL PARTICIPATION

Total and Partial Participation should be mapped as well

- For participation definitions on the ”1” side, a constraint assures the requirement
  - Total Participation 1:m → NOT NULL on FK
  - Partial Participation “0:m” → NULL on FK
- For participation definitions on the ”m” side, there is a problem
  - These types (1:n vs. “1:0n”) are not distinguishable in Relational Model
  - These types of Total Participation cannot be implemented / enforced using SQL-DDL!

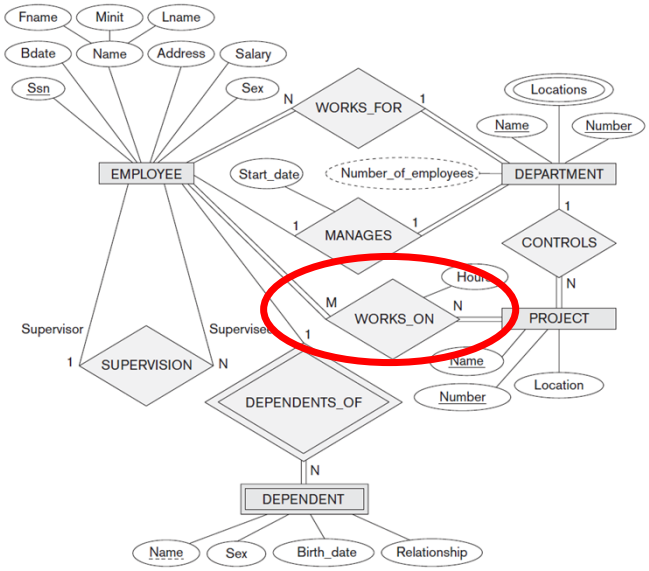
RELATIONSHIPS
RM: 5. MAPPING OF BINARY M:N RELATIONSHIPS

- For each binary M:N relationship type R, create a new relation S to represent R
- Include as foreign key attributes in S the primary keys of the relations that represent the participating entity types; their combination will form the primary key of S
- Also include any simple attributes of the M:N relationship type (or simple components of composite attributes) as attributes of S
- Notice that a M:N relationship type can not be represented by a single foreign key attribute in one of the participating relations (as in 1:1 or 1:N relationship types) because of the M:N cardinality ratio

Source: Elmasri, Fundamentals of Database Systems, Page 286ff

RELATIONSHIPS
RM: 5. MAPPING OF BINARY M:N RELATIONSHIPS

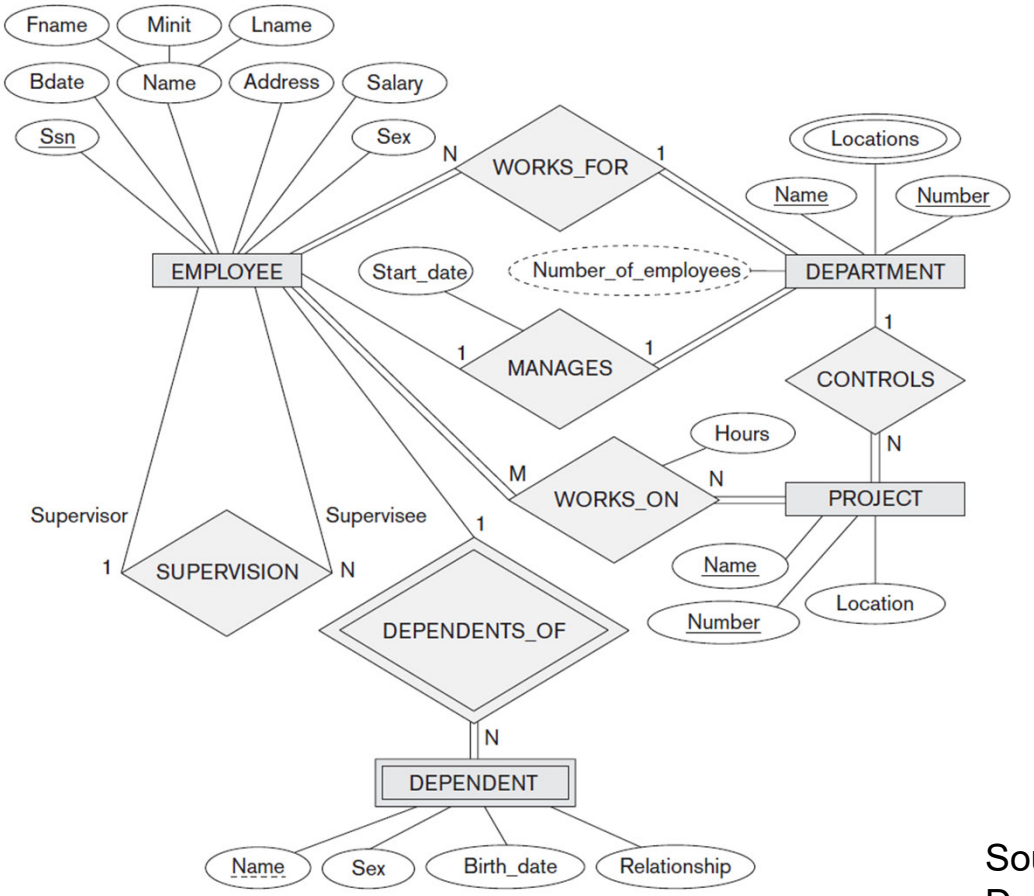
WORKS_ON	Essn	Pno	Hours



- Attribute Ssn is renamed in Essn in WORKS_ON
- Attribute Pname is renamed in Pno in DEPARTMENT
- Primary key is the combination {Essn, Pno}
- Note: The existence dependency between EMPLOYEE and PROJECT should be specified on the foreign keys in the relation corresponding to the relationship R (ON UPDATE and ON DELETE)

Source: Elmasri, Fundamentals of Database Systems, Page 286ff

RELATIONSHIPS
ERM: EXAMPLE - COMPANY



Source: Elmasri, Fundamentals of Database Systems, Page 204 ff

RELATIONSHIPS
RM: EXAMPLE - COMPANY

EMPLOYEE

Fname	Minit	Lname	Ssn	Bdate	Address	Sex	Salary	Super_ssn	Dno

DEPARTMENT

Dname	Dnumber	Mgr_ssn	Mgr_start_date

PROJECT

Pname	Pnumber	Plocation	Dnum

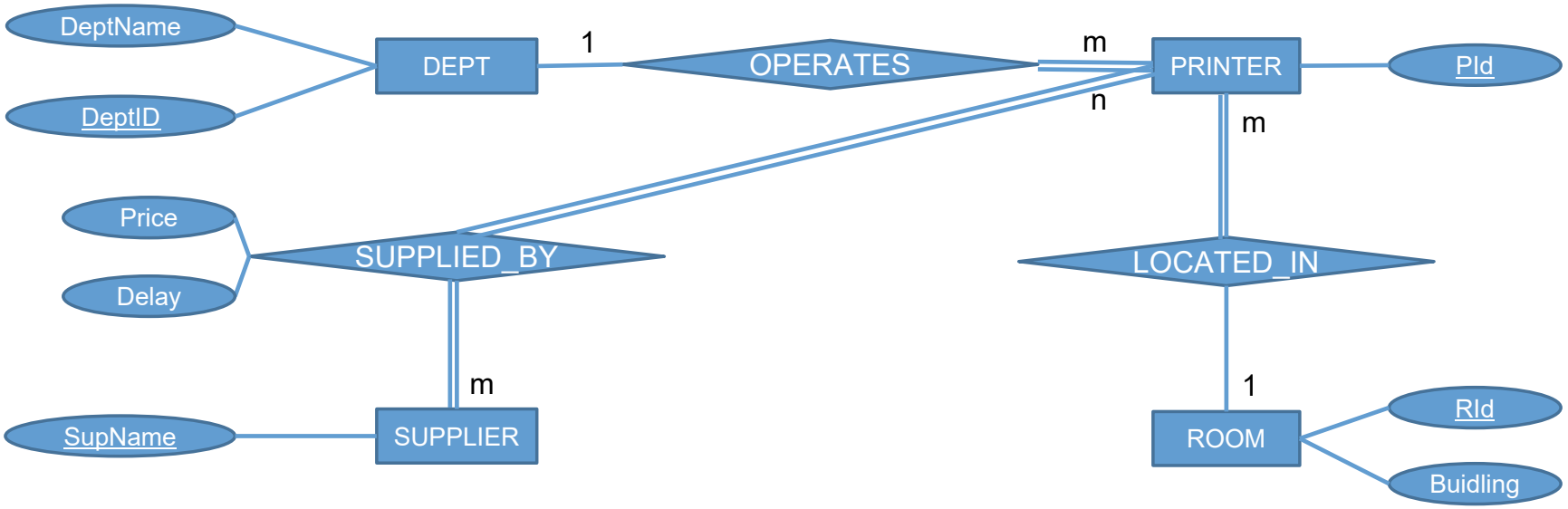
WORKS_ON

Essn	Pno	Hours

Source: Elmasri, Fundamentals of Database Systems, Page 59ff

RELATIONSHIPS
RM: ASSIGNMENT OFFICE - CONVERT THE ERD TO A RM

- Departments, identified by ID, operate a variety of printers, each located in a particular room in a particular building. Printers are supplied by a number of suppliers, identified by name, with each supplier charging a different price for a given printer, but also providing different delivery delays, measured in days. A given room can have any number of printers, including none.



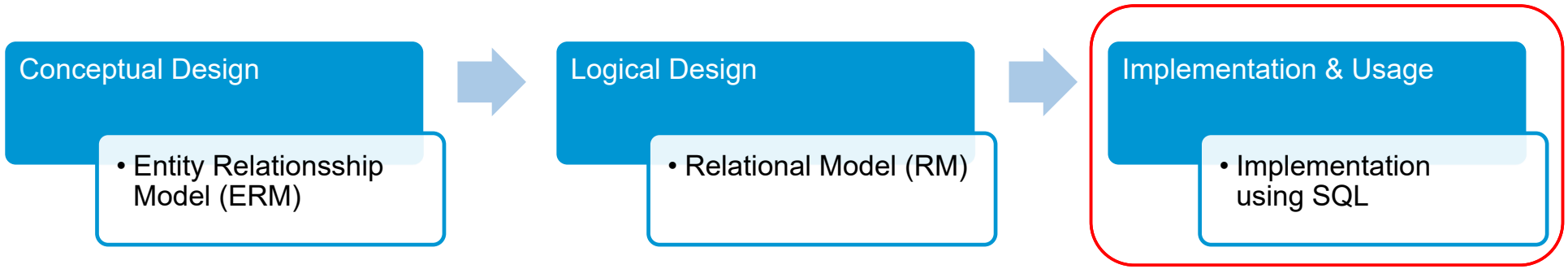
RELATIONSHIPS
RM: MAPPING OF ERM TO RELATIONAL MODEL

Seven Steps

1. ✓ Mapping of regular entity types
2. Mapping of weak entity types
3. ✓ Mapping of binary 1:1 relationships
4. ✓ Mapping of binary 1:n relationships
5. ✓ Mapping of binary m:n relationships
6. Mapping of multivalued attributes
7. Mapping of n-ary relationships

Source: Elmasri, Fundamentals of Database Systems, Page 286ff

RELATIONSHIPS
DATABASE DESIGN



RELATIONSHIPS
SQL: CREATE TABLE – COLUMN AND TABLE CONSTRAINTS: FOREIGN KEY

- Referential integrity is specified via the **`FOREIGN KEY`**
- FK relates two tables
- Referenced table must exist already
- Referenced column must be **`UNIQUE`**
  → Best to use PK
  → If not PK: need to specify (column)

Source: Elmasri, Fundamentals of Database Systems, Page 88ff

RELATIONSHIPS
SQL: CREATE TABLE – COLUMN AND TABLE CONSTRAINTS: FOREIGN KEY

Syntax:

- As Column Constraint
  → Only if the foreign key is one single attribute (and not combined)

```
[ CONSTRAINT < constraintname > ]
    REFERENCES < tablename >[( column )] [< action >]
```

- As Table Constraint

```
[ CONSTRAINT < constraintname >]
    FOREIGN KEY (< column list >)
    REFERENCES < tablename >[(< column list >)]
          [< action >]
```

RELATIONSHIPS
SQL: CREATE TABLE – COLUMN AND TABLE CONSTRAINTS: FOREIGN KEY

Example column constraint:

```
CREATE TABLE Department
( Dname             VARCHAR(15)  NOT NULL,
Dnumber             INT          NOT NULL,
Mgr_ssn             CHAR(9)      REFERENCES Employee(Ssn) ,
Mgr_start_date      DATE,
PRIMARY KEY (Dnumber) ,
UNIQUE (Dname));
```

Source: Elmasri, Fundamentals of Database Systems, Page 88ff

RELATIONSHIPS
SQL: CREATE TABLE – COLUMN AND TABLE CONSTRAINTS: FOREIGN KEY

Example table constraint:

```
CREATE TABLE Department
( Dname             VARCHAR(15)  NOT NULL,
Dnumber             INT          NOT NULL,
Mgr_ssn             CHAR(9)      NOT NULL,
Mgr_start_date      DATE,
PRIMARY KEY ( Dnumber ) ,
UNIQUE ( Dname ) ,
FOREIGN KEY ( Mgr_ssn ) REFERENCES Employee ( Ssn ) );
```

Source: Elmasri, Fundamentals of Database Systems, Page 88ff

RELATIONSHIPS
SQL: CREATE TABLE – COLUMN AND TABLE CONSTRAINTS: FOREIGN KEY

- < action >:
  → How to react on changes to the referenced table

- The default action: reject the update operation (**RESTRICT** option)

- Syntax

```
action ::= ON {UPDATE | DELETE}
     {NO ACTION | SET NULL | SET DEFAULT | CASCADE}
```

SET DEFAULT is not supported by MariaDB, mySQL!

ON UPDATE CASCADE not supported by Oracle! → Use DEFERRABLE constraints or triggers

RELATIONSHIPS
SQL: CREATE TABLE – COLUMN AND TABLE CONSTRAINTS: FOREIGN KEY

- Options:
  - **`SET NULL`** → Value of foreign key is set to NULL
  - **`SET DEFAULT`** → Value of foreign key is set to a default value
  - **`CASCADE`** → Value of foreign key is updated
- For example:
  - **`ON DELETE CASCADE`** → Delete all referencing tuples
  - **`ON UPDATE CASCADE`** → Change Value of the foreign key attribute(s)
- General Rule for using **`CASCADE`**:
  - For “relationship” relations
  - For multivalued attributes
  - For relations that represent weak entity types

Source: Elmasri, Fundamentals of Database Systems, Page 88ff

RELATIONSHIPS
SQL: CREATE TABLE – COLUMN AND TABLE CONSTRAINTS: FOREIGN KEY

```
CREATE TABLE Employee
( . . . ,
Dno           INT    NOT NULL      DEFAULT 1,
CONSTRAINT EMPPK
      PRIMARY KEY (Ssn),
CONSTRAINT EMPSUPERFK
      FOREIGN KEY (Super_ssn) REFERENCES Employee(Ssn)
      ON DELETE SET NULL
      ON UPDATE CASCADE,
CONSTRAINT EMPDEPTFK
      FOREIGN KEY(Dno) REFERENCES Department(Dnumber)
      ON DELETE SET DEFAULT
      ON UPDATE CASCADE);
```

Source: Elmasri, Fundamentals of Database Systems, Page 88ff

RELATIONSHIPS
SQL: ALTER TABLE

- For modifying an existing relation
  - COLUMN: **`ADD, DROP, MODIFY`**
  - CONSTRAINT: **`ADD, DROP`**
  - TABLE: **`RENAME`**
  - Vendor-specific extensions

RELATIONSHIPS
SQL: ALTER TABLE - COLUMN

Syntax for altering a table:

```
ALTER TABLE < relationname > . . .

ADD [ COLUMN ] < column > < type >
      [ < col\_constraint > [ . . .]

DROP [ COLUMN ] <column> [ RESTRICT | CASCADE ]

RENAME COLUMN <column> TO <new\_column>
```

RELATIONSHIPS
SQL: ALTER TABLE - COLUMN

Syntax for altering a table:

```
ALTER TABLE < relationname > . . .
```

Modification of columns vendor-specific:

Oracle:

```
... MODIFY < column > < type > [< col\_constraints > [...]]
```

MySQL:

```
... CHANGE [ COLUMN ] < column > < type > ...
```

RELATIONSHIPS
SQL: ALTER TABLE - COLUMN

- Example:

```
ALTER TABLE COMPANY.Employee ADD COLUMN Job VARCHAR(12);
```

- Inserting values for the new column:
  - Default is NULL → **NOT NULL** constraint is not allowed
  - Using default clause
  - Using **UPDATE** individually on each tuple

Source: Elmasri, Fundamentals of Database Systems, Page 88ff

RELATIONSHIPS
SQL: ALTER TABLE - CONSTRAINTS

Syntax for adding a new constraint:

```
ALTER table ADD  < tableconstraint > ;
```

- Example: Add a foreign key
  (instead of within create table statement)

```
ALTER TABLE DEPARTMENT
ADD CONSTRAINT DEPTMGRFK  FOREIGN KEY (Mgr_ssn) REFERENCES EMPLOYEE(Ssn)
ON DELETE RESTRICT ON UPDATE CASCADE;
```

RELATIONSHIPS
SQL: ALTER TABLE - CONSTRAINTS

Syntax for droping an existing constraint:

```
      ALTER TABLE < tablename > < alterstatement >

< alterstatement > ::=
      DROP PRIMARY KEY |
      DROP FOREIGN KEY < keyname > |
```

RELATIONSHIPS
SQL: ALTER TABLE - RENAME

Syntax for renaming an existing table:

- Oracle, MySQL
  ```
  RENAME TABLE < relationname > TO < newrelationname >
  ```

- PostgreSQL, MySQL
  ```
  ALTER TABLE < name > RENAME TO < new_name >
  ```

HOMEWORK

- Company Example
  - Implement all relationship types from the ERM in your database
  - Think also about the cardinalities and participation constraints of these relationship types
  - What should be the behavior of these relations if data changes?
  - Try SQL statements for inserting, updating, and deleting data
- Implement the printer example in your database
- Think about your own, individual example (e.g., contact list)
  - Implement all relationship types from the ERM in your database
  - Think also about the cardinalities and participation constraints
  - What should be the behavior of these relations if data changes?
  - Try SQL statements for inserting, updating, and deleting data

Source: Foto von K8 auf Unsplash