Part IV

Database Design

Database Design

Database Design

Database Design

Database Design

Database Design

Database Design

Database Design

Educational Objective for Today . . .

- Goals and steps of the database design process

Educational Objective for Today . . .

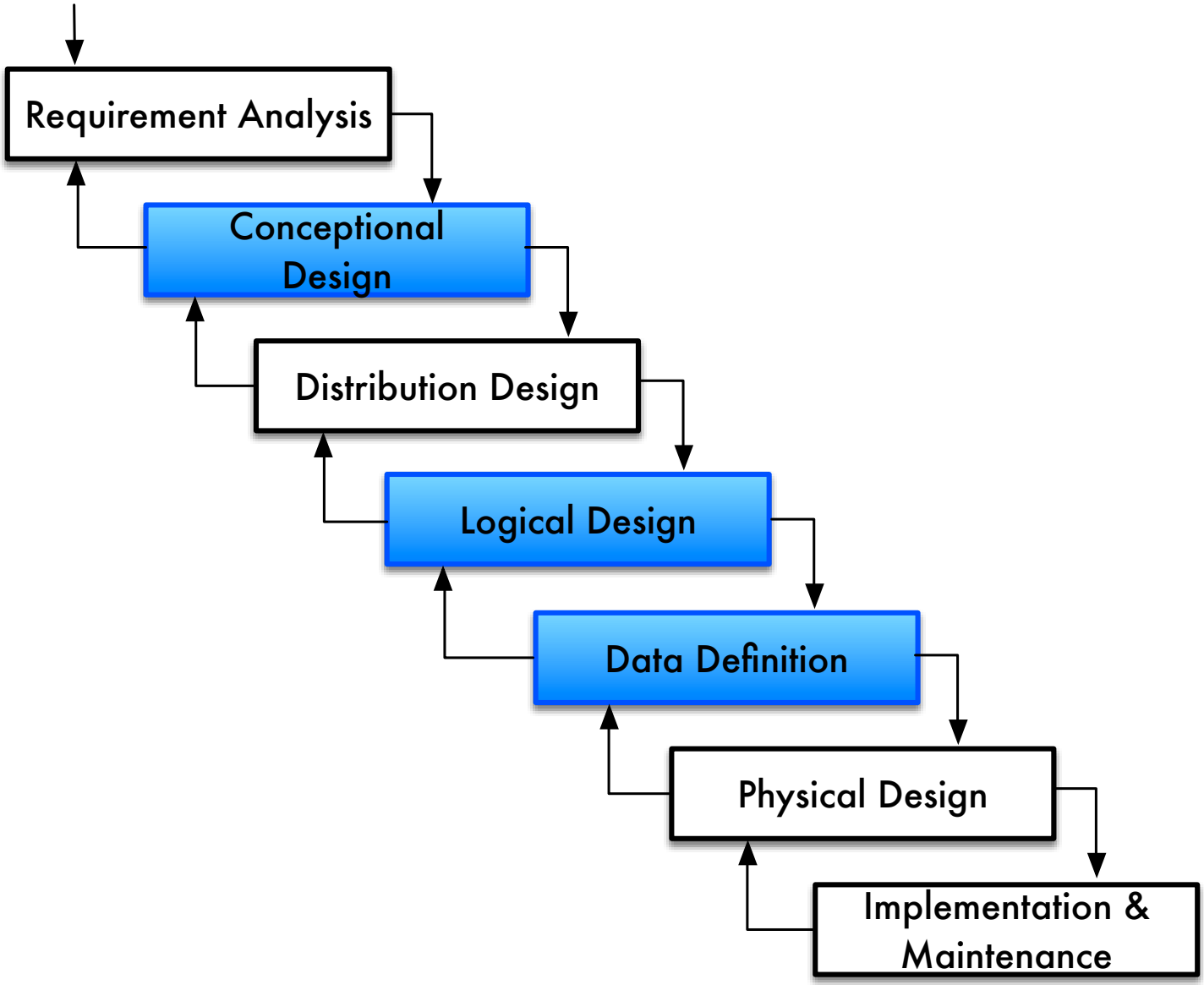
- Goals and steps of the database design process
- Rules to transform ER schemata into relational schemata

Phases of Database Design

Goal of Design

- Data management for multiple application systems, for multiple years
- Therefore: special importance
- Design requirements
  - For each application, it should be possible to derive application data from stored data in the database — efficiently!
  - Only store “sensible” (actually needed) data
  - Avoid redundancies

Phase Model


Requirement Analysis
Conceptional Design
Distribution Design
Logical Design
Data Definition
Physical Design
Implementation & Maintenance

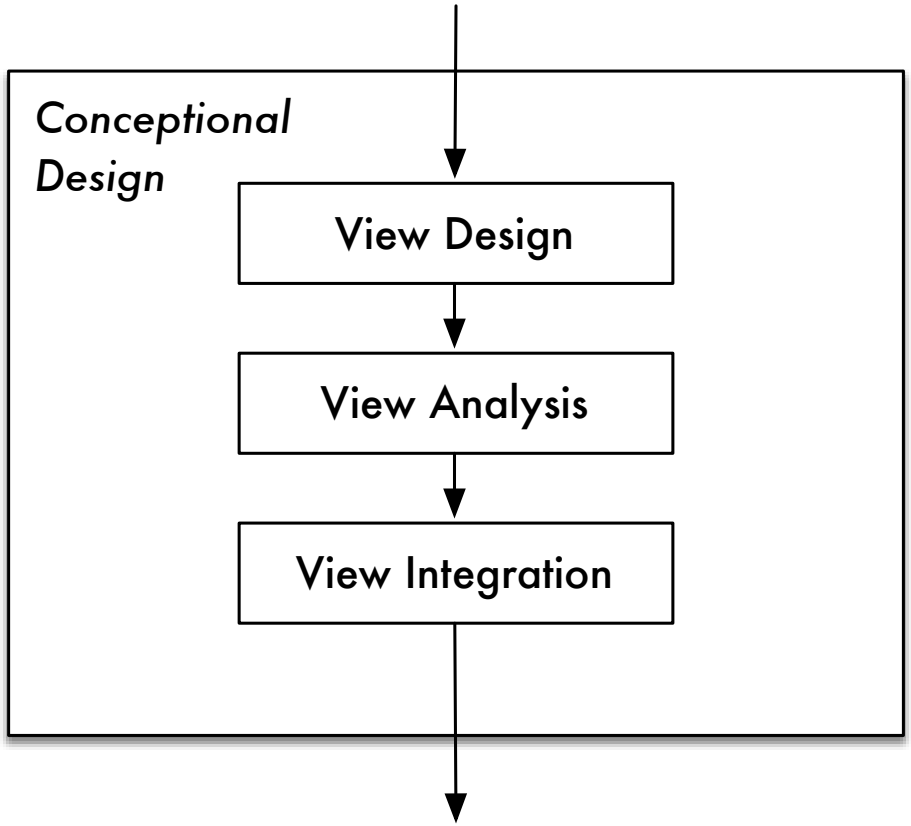
Requirements Analysis

- Approach: Collecting information needs from all specialist divisions
- Result:
  - Informal description (text, tabular lists, forms, etc.) of the problem domain
  - Separation of the information about data (data analysis) from the information about functions (functional analysis)
- “Classical” DB design:
  - Only data analysis and following steps
- Functional design:
  - See methods of software engineering

Conceptual Design

- First formal description of the problem domain
- Language means: semantical data model
- Process:
  - Modeling of views, e.g., for different specialist divisions
  - Analysis of existing views with respect to conflicts
  - Integration of views into a full schema
- Result: full conceptual schema, e.g., ER diagram

Phases of Conceptual Design



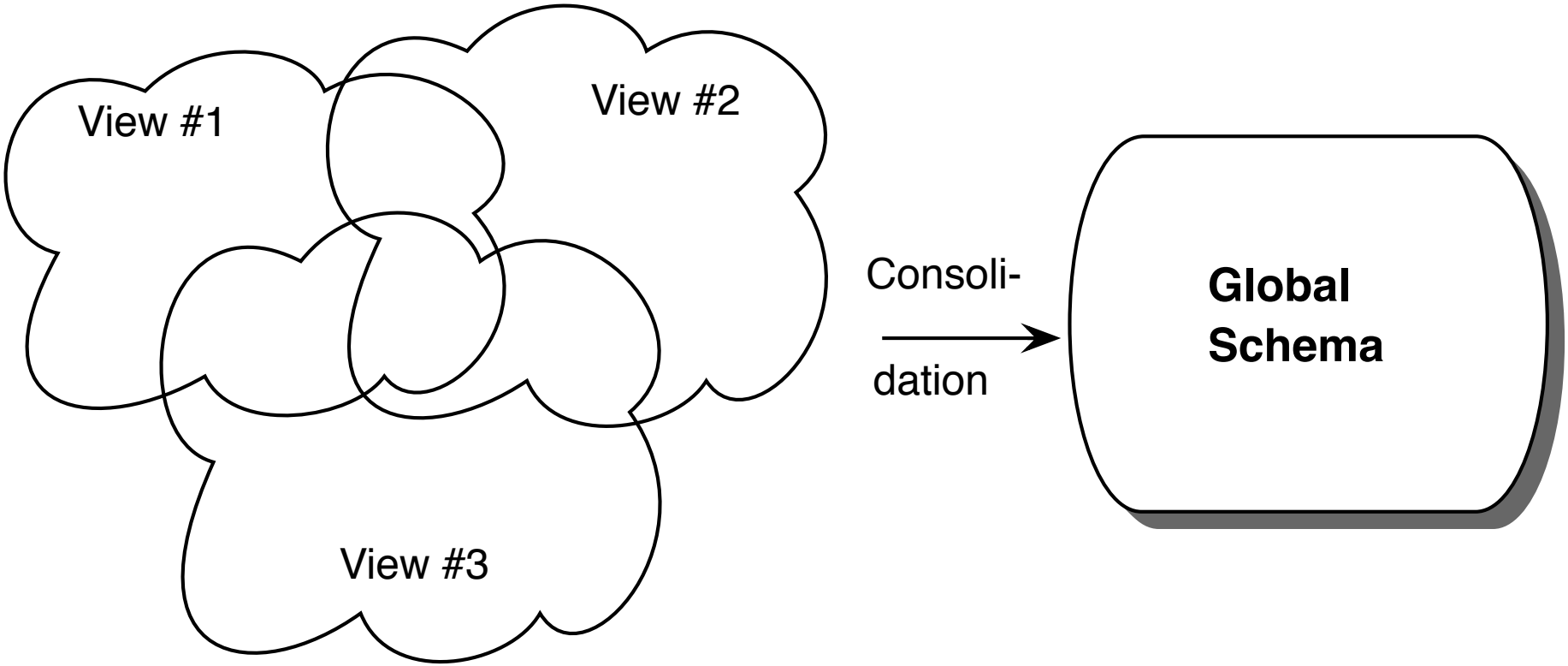
Further Steps During Design

Further Steps During Design

- ER modeling of different views of the complete information, e.g., for different specialist divisions of a company $\rightsquigarrow$ conceptual design
  - Analysis and integration of views
  - Result: full conceptual schema
- Distribution design when using distributed storage
- Transformation to concrete implementation model (e.g., relational model) $\rightsquigarrow$ logical design
- Data definition, implementation and maintenance $\rightsquigarrow$ physical design

View Integration

- Analysis of existing view with respect to conflicts
- Integration of views into a full schema



Integration Conflicts

- **Naming conflicts**: Homonyms / synonyms
  - Homonyms: bank (money / river); order (command / request for goods)
  - Synonyms: car, vehicle, automobile
- **Typing conflicts**: different structures for the same element
- **Domain mismatch**: different domains for an element
- **Identifier conflicts**: e.g., different keys for the same element
- **Structural conflicts**: same fact expressed in different ways

Distribution Design

- If data should be distributed to several machines, a way of distributed storage must be determined
- E.g., for a relation

  `CUSTOMER (CNo, Name, Address, Zipcode, Account)`
  - Horizontal distribution:

    ```
    CUSTOMER_1 (CNo, Name, Address, Zipcode, Account)
    where Zipcode < 50 000
    CUSTOMER_2 (CNo, Name, Address, Zipcode, Account)
    where Zipcode >= 50 000
    ```
  - Vertical distribution (connection via attribute `CNo`):

    ```
    CUSTOMER_Adr (CNo, Name, Address, Zipcode)
    CUSTOMER_Account (CNo, Account)
    ```

Logical Design

- Language means: Data model of the chosen “implementation” DBMS, e.g., relational model
- Process:
  1. (Automatical) transformation of the conceptual schema, e.g., ER $\rightarrow$ relational model
  2. Improvement of the relational schema based on quality criteria (normalization, see Chapter 5):
     Design goals: avoid redundancies, . . .
- Result: logical schema, e.g., collection of relation schemata

Data Definition

- Translation of logical schema into a concrete schema
- Language means: DDL and DML of DBMS (e.g., Oracle, DB2, SQL Server)
  - Database declaration in the DDL of the DBMS
  - Realization of integrity constraints
  - Definition of views

Physical Design

- Supplement physical design with support for efficient access, e.g., by defining indexes
- Index
  - Access path: data structure for additional, key-based access to tuples ($\langle$*key attribute value, tuple address*$\rangle$)
  - Usually implemented as a B*-tree
- Language means: *storage structure (definition) language* SSL

Indexes in SQL

```
create [ unique ] index indexname
    on relname (
        attrname [ asc | desc ],
        attrname [ asc | desc ],
    ...
)
```

- Example

```
create index WineIdx on WINES (Name)
```

Necessity of Access Paths

- Example: Table with 100 GB of data, hard disk transfer rate of ca. 50 MB/s
- Operation: Search for a tuple (selection)
- Implementation: Sequential search
- Cost: $102.400/50 = 2.048$ sec. ≈ 34 min.

Implementation and Maintenance

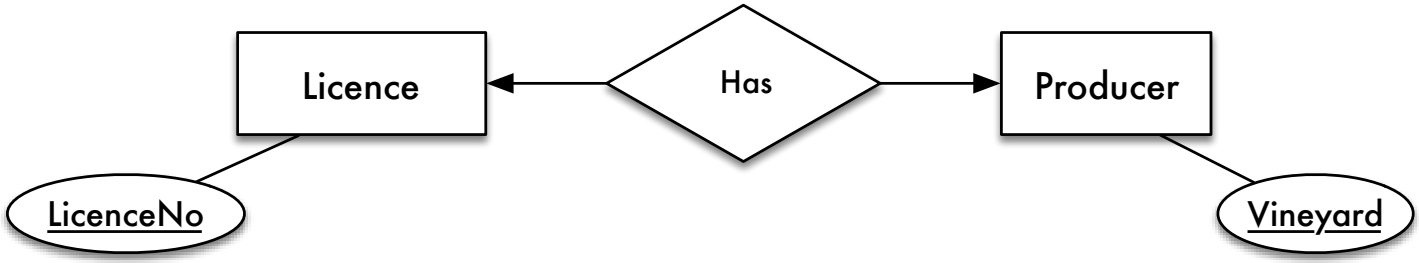
- Phases of . . .
  - Maintenance,
  - Further optimization of the physical layer,
  - Adaptation to new requirements or operating system platforms,
  - Porting to new database management systems
  - etc.

Capacity-preserving Transformations

Transformation of the Conceptual Schema

- Translation to logical schema
  - Example: ER $\rightarrow$ RM
  - Correct?
  - Quality of transformation?
- Preservation of *information capacity*
  - Is it possible, after the transformation, to store exactly the same data as before?
  - . . . or more?
  - . . . or less?

Capacity-increasing Transformation



- Transformation into

$$R = \{\texttt{LicenseNo}, \texttt{Vineyard}\}$$

with exactly one key

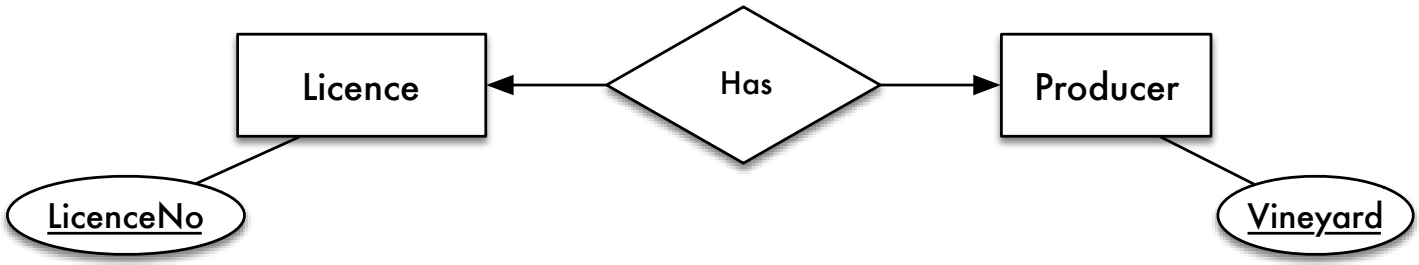
$$K = \{\{\texttt{LicenseNo}\}\}$$

- Possible invalid relation:

Has

LicenseNo	**Vineyard**
007	Helena
42	Helena

Capacity-preserving Transformation



- Correct instantiation

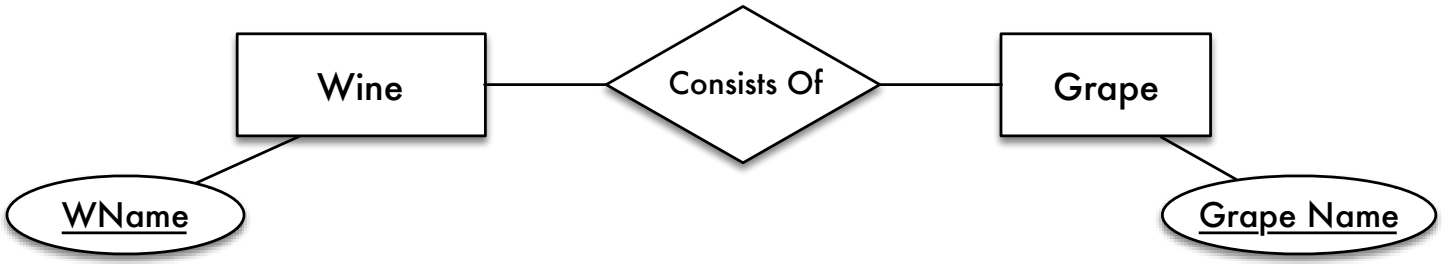
Has

LicenseNo	Vineyard
007	Helena
42	Mueller

- Correct set of keys

$$K = \{\{\texttt{LicenseNo}\}, \{\texttt{Vineyard}\}\}$$

Capacity-decreasing Transformation



- Relation schema with one key $\{\texttt{WName}\}$
- Instantiation that is no longer valid:

ConsistsOf

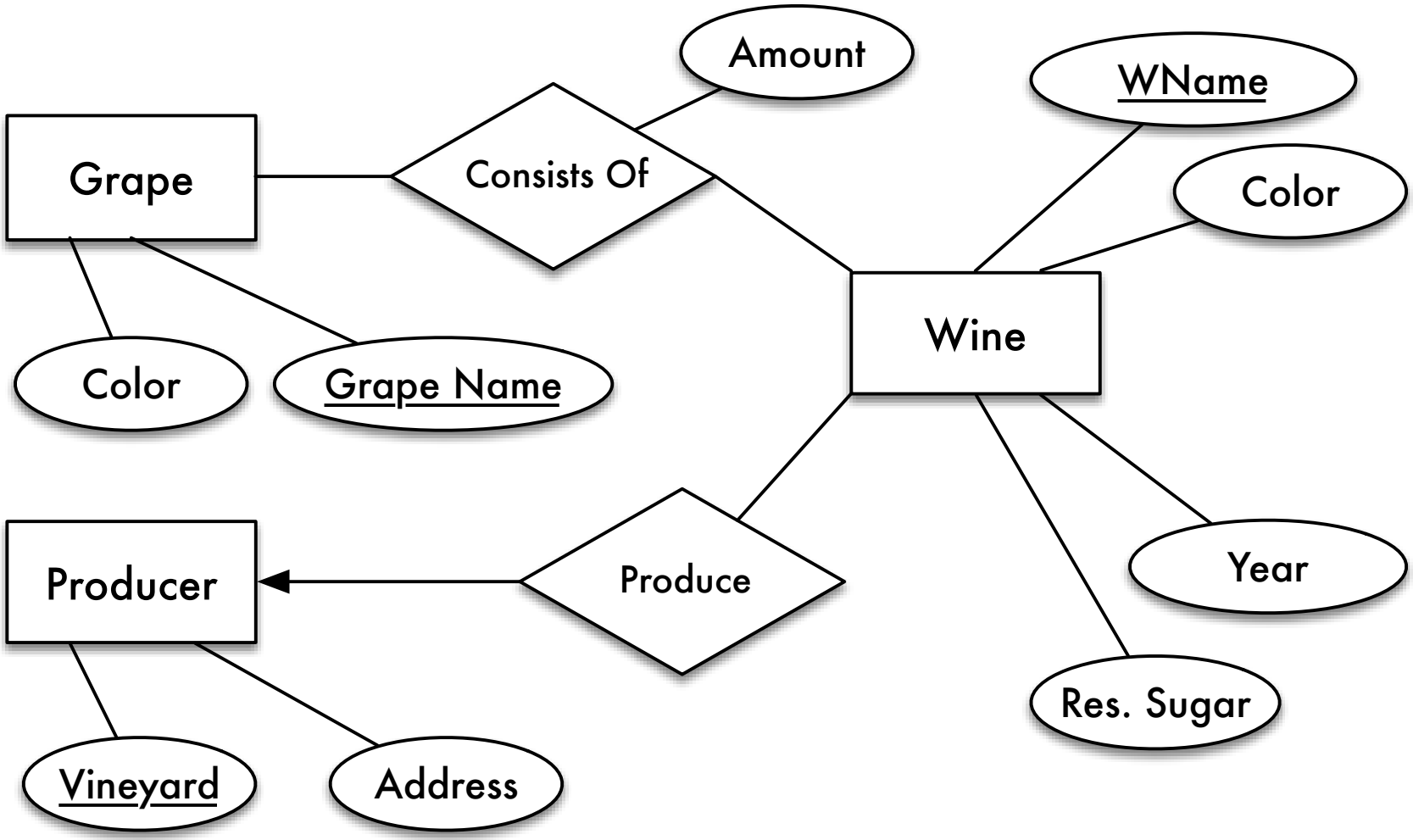
WName	**GrapeName**
Zinfandel Red Blossom	Zinfandel
Bordeaux Blanc	Cabernet Sauvignon
Bordeaux Blanc	Muscadelle

- Capacity-preserving when using the keys of both entity types as the new key in the relation schema

$$K = \{\{\texttt{WName}, \texttt{GrapeName}\}\}$$

ER-to-RM Transformation

Example Transformation ER-RM: Input



Example Transformation ER-RM: Result

1. $\texttt{GRAPE} = \{\texttt{Color}, \texttt{GrapeName}\}$ with $K_{\texttt{GRAPE}} = \{\{\texttt{GrapeName}\}\}$
2. $\texttt{ConsistsOf} = \{\texttt{GrapeName}, \texttt{WName}, \texttt{Amount}\}$ with $K_{\texttt{ConsistsOf}} = \{\{\texttt{GrapeName}, \texttt{WName}\}\}$
3. $\texttt{WINE} = \{\texttt{Color}, \texttt{WName}, \texttt{Vintage}, \texttt{Res.Sugar}\}$ with $K_{\texttt{WINE}} = \{\{\texttt{WName}\}\}$
4. $\texttt{PRODUCE} = \{\texttt{WName}, \texttt{Vineyard}\}$ with $K_{\texttt{PRODUCE}} = \{\{\texttt{WName}\}\}$
5. $\texttt{PRODUCER} = \{\texttt{Vineyard}, \texttt{Address}\}$ with $K_{\texttt{PRODUCER}} = \{\{\texttt{Vineyard}\}\}$

ER Transformation into Relations

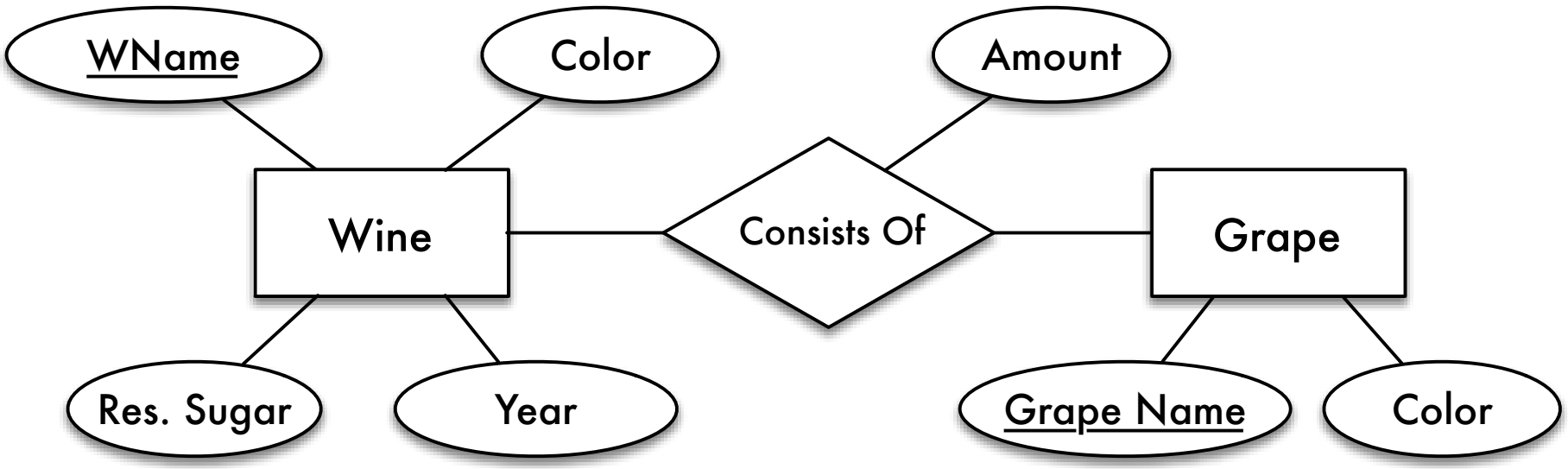
- Entity types and relationship types: both transformed into relation schemata
- Attributes: attributes of the relation schema, keys are adopted
- Cardinalities of the relationships: expressed in respective relation schemata by choice of keys
- In some cases: merge of the relation schemata of entity and relationship types
- Introduce foreign key constraints between the remaining relation schemata

Transformation of Relationships

Transformation of Relationship Types

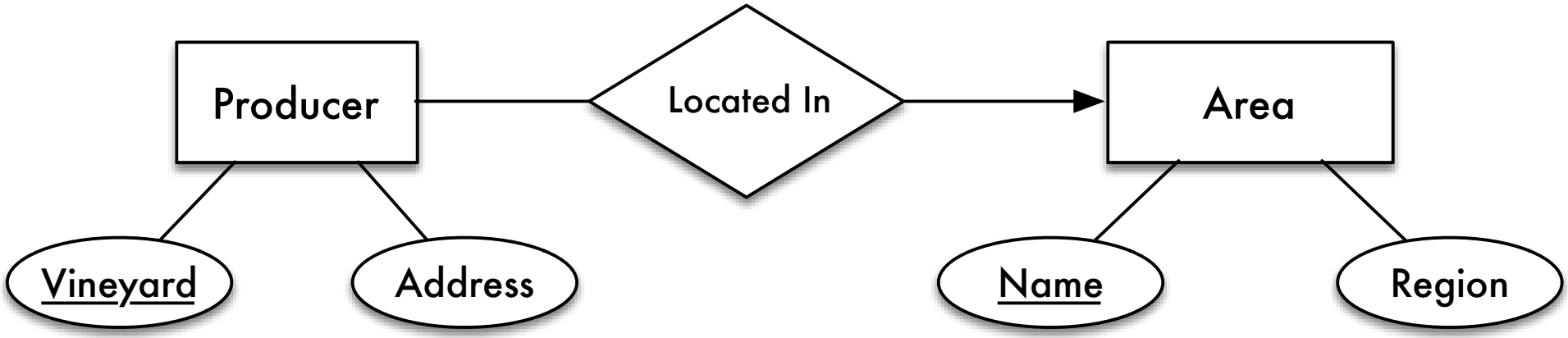
- New relation schema with all attributes of the relationship type; additionally, adopt all primary keys of the participating entity types
- **Determining keys**:
  - m:n relationship: both primary keys together form the key in the new relation schema
  - 1:n relationship: primary keys of the n-side (in the functional notation, this is the side without the arrowhead) form key in the new relation schema
  - 1:1 relationship: both primary keys become a key in the new relation schema; the primary key is then chosen from these keys

n:m Relationships



- Transformation
  1. $\texttt{GRAPE} = \{\texttt{Color}, \texttt{GrapeName}\}$ with $K_{\texttt{GRAPE}} = \{\{\texttt{GrapeName}\}\}$
  2. $\texttt{ConsistsOf} = \{\texttt{GrapeName}, \texttt{WName}, \texttt{Amount}\}$ with $K_{\texttt{ConsistsOf}} = \{\{\texttt{GrapeName}, \texttt{WName}\}\}$
  3. $\texttt{WINE} = \{\texttt{Color}, \texttt{WName}, \texttt{Vintage}, \texttt{Res.Sugar}\}$ with $K_{\texttt{WINE}} = \{\{\texttt{WName}\}\}$
- Attributes `GrapeName` and `WName` together are key

1:n Relationships

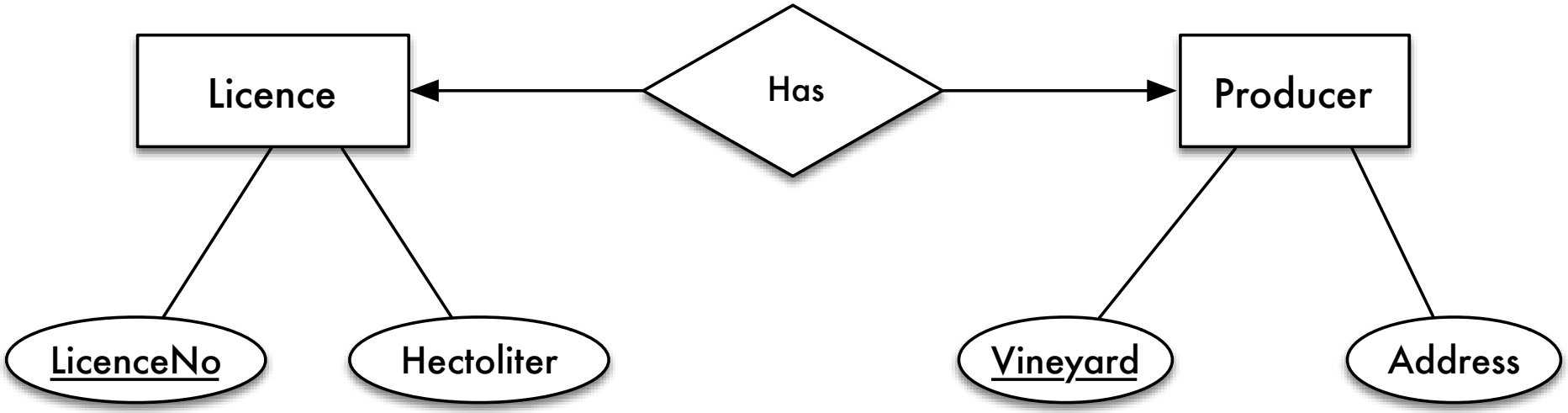


- (Preliminary) transformation
  - `PRODUCER` with the attributes `Vineyard` and `Address`,
  - `AREA` with the attributes `Name` and `Region`, and
  - `LocatedIn` with the attributes `Vineyard` and `Name` and the primary key of the n-side `Vineyard` as primary key of this schema.

Possible Merges

- Optional relationships ([0,1] or [0,n]) are not merged
- With cardinalities [1,1] or [1,n] (mandatory relationships), merge is possible:
  - 1:n relationship: the entity-relation schema of the n-side can be integrated into the relation schema of the relationship
  - 1:1 relationship: both entity-relation schemata can be integrated into the relation schema of the relationship

1:1 Relationships



- (Preliminary) transformation
  - `PRODUCER` with the attributes `Vineyard` and `Address`
  - `LICENSE` with the two attributes `LicenseNo` and `Hectoliters`
  - `Has` with the primary keys of both participating entity types each as key of this schema, that is `LicenseNo` and `Vineyard`

1:1 Relationships: Merge

- Transformation with merge
  - Merged relation:

PRODUCER

Vineyard	Address	LicenseNo	Hectoliters
Rotkäppchen	Freiberg	42-007	10 000
Vineyard Müller	Dagstuhl	42-009	250

  - Producers without license require null values:

PRODUCER

Vineyard	Address	LicenseNo	Hectoliters
Rotkäppchen	Freiberg	42-007	10 000
Vineyard Müller	Dagstuhl	⊥	⊥

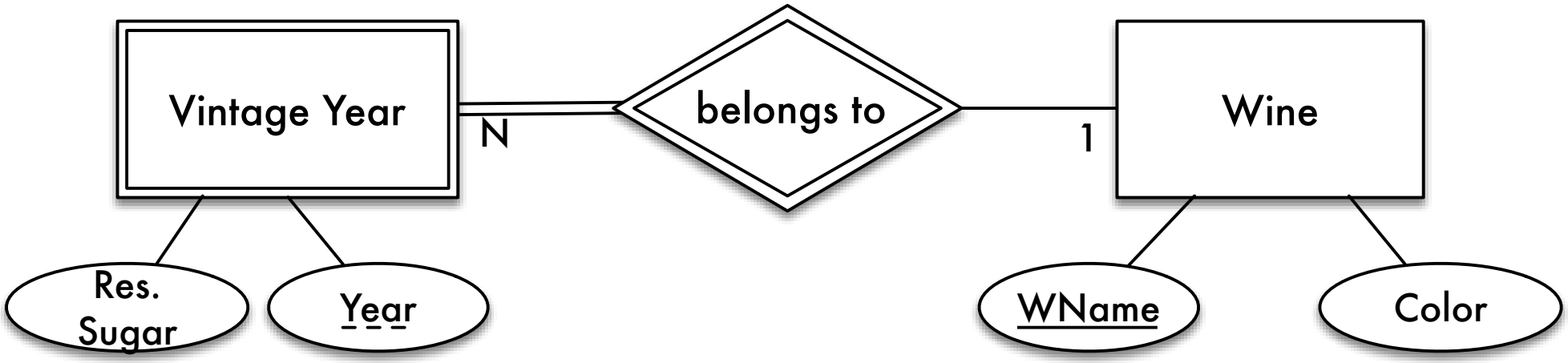
  - Free Licenses lead to additional null values:

PRODUCER

Vineyard	Address	LicenseNo	Hectoliters
Rotkäppchen	Freiberg	42-007	10 000
Vineyard Müller	Dagstuhl	⊥	⊥
⊥	⊥	42-003	100 000

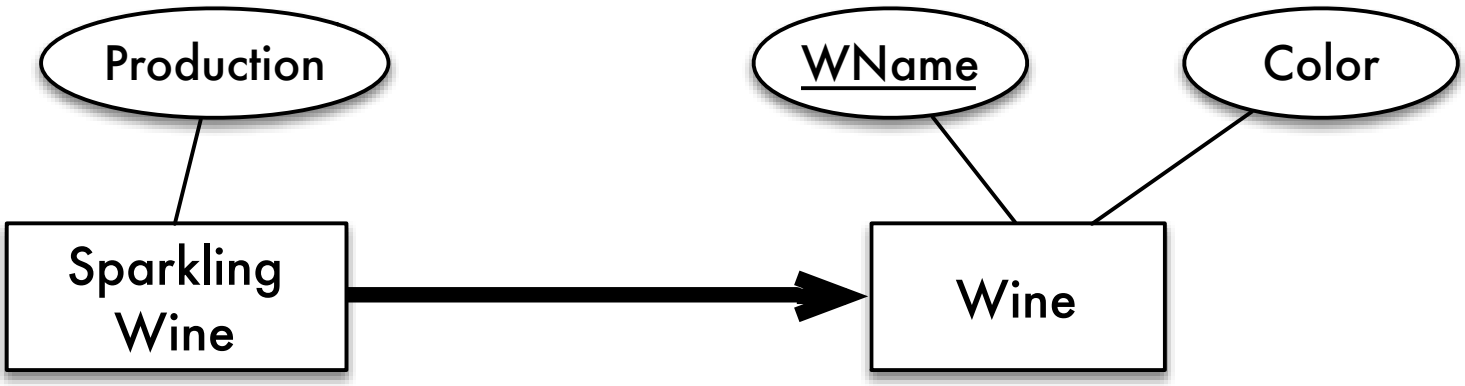
Special Cases for Transformation

Dependent Entity Types



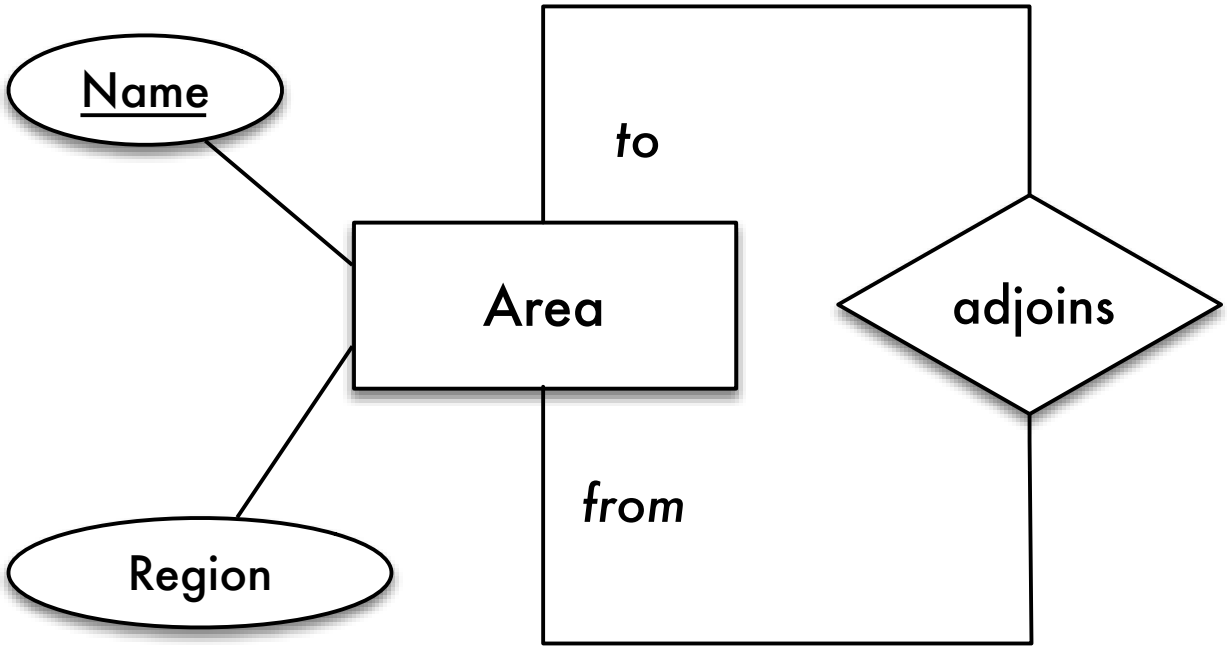
- Transformation
  1. $\texttt{WINEVINTAGE} = \{\texttt{WName}, \texttt{Res.Sugar}, \texttt{Year}\}$ with $K_{\texttt{WINEVINTAGE}} = \{\{\texttt{WName}, \texttt{Year}\}\}$
  2. $\texttt{WINE} = \{\texttt{Color}, \texttt{WName}\}$ with $K_{\texttt{WINE}} = \{\{\texttt{WName}\}\}$
     - Attribute WName in WINEVINTAGE is foreign key to relation WINE

IS-A Relationship



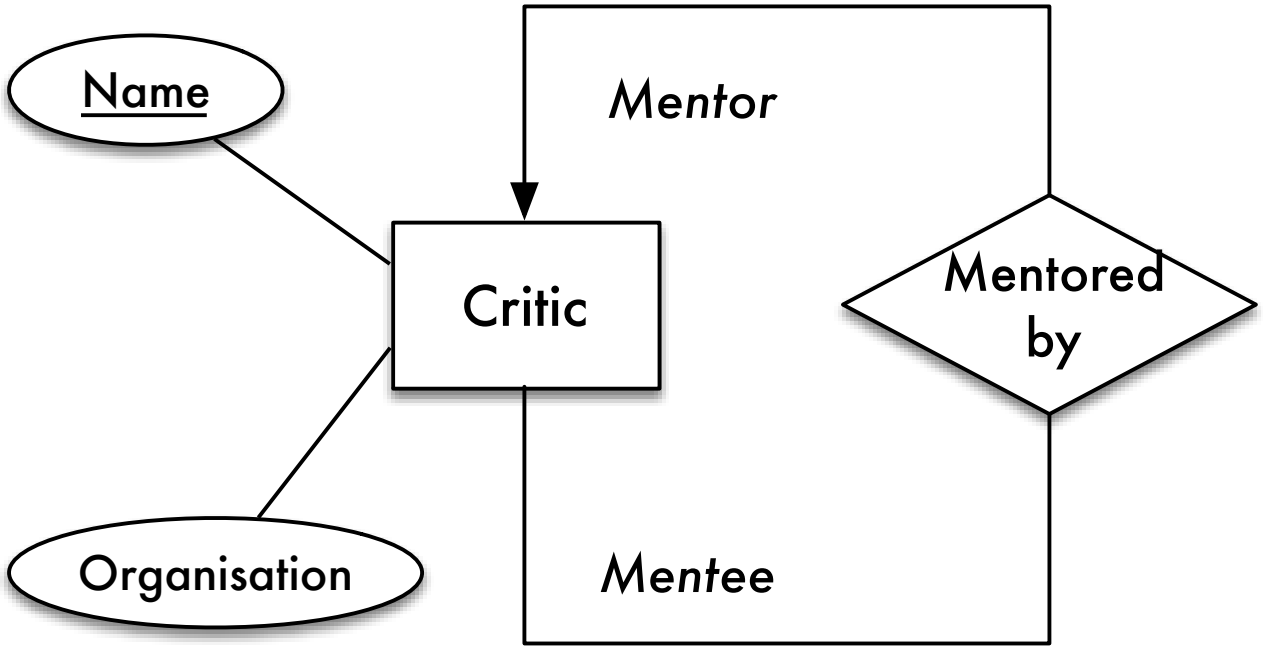
- Transformation
  1. WINE $= \{$Color, WName$\}$ with $K_{\texttt{WINE}} = \{\{\texttt{WName}\}\}$
  2. SPARKLING_WINE $= \{$WName, Production$\}$ with $K_{\texttt{SPARKLING\_WINE}} = \{\{\texttt{WName}\}\}$
     - WName in SPARKLING_WINE is foreign key with respect to relation WINE

Recursive Relationships



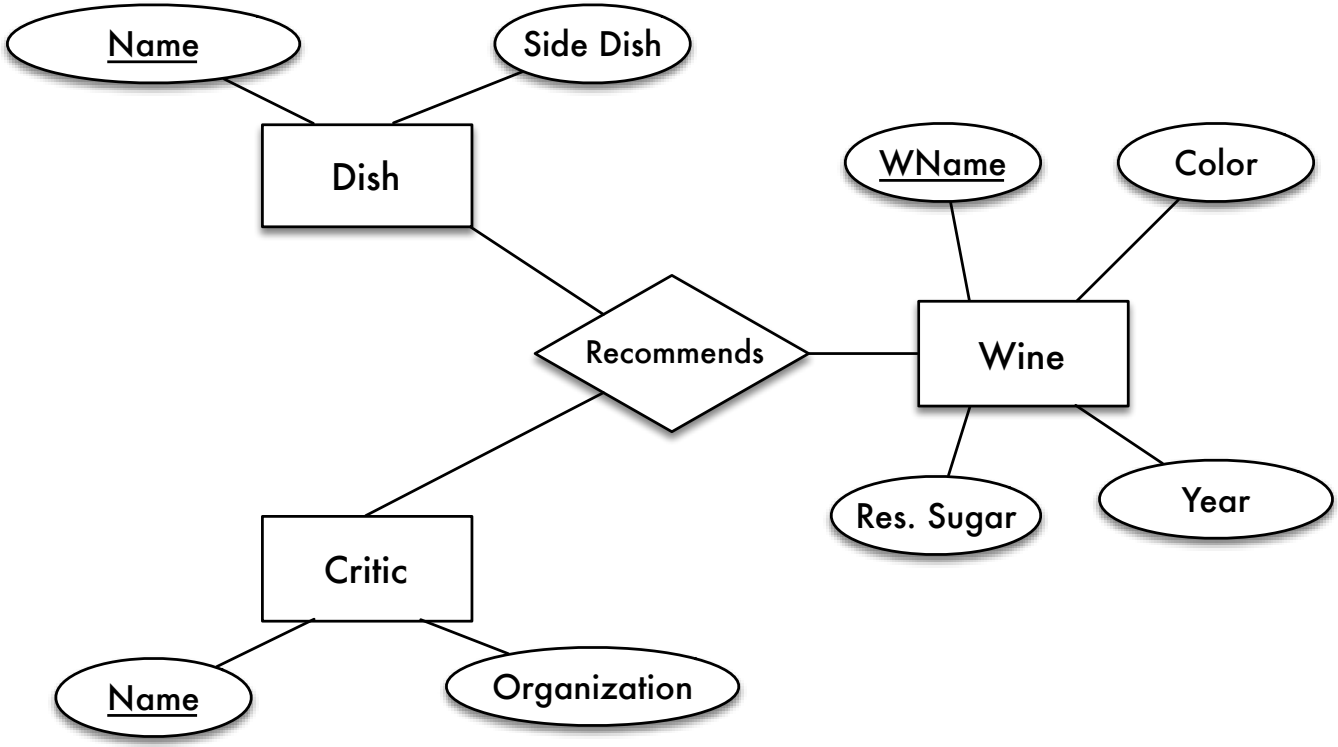
- Transformation
  1. AREA $= \{\texttt{Name}, \texttt{Region}\}$ with $K_{\texttt{AREA}} = \{\{\texttt{Name}\}\}$
  2. ADJOINS $= \{\texttt{to}, \texttt{from}\}$ with $K_{\texttt{ADJOINS}} = \{\{\texttt{to}, \texttt{from}\}\}$

Recursive Functional Relationships



- Transformation
  1. $\texttt{CRITIC} = \{\texttt{Name}, \texttt{Organization}, \texttt{Mentorname}\}$ with $K_{\texttt{CRITIC}} = \{\{\texttt{Name}\}\}$
     - `Mentorname` is foreign key to attribute `Name` of relation `CRITIC`.

N-ary Relationships



- Every participating entity type is treated according to the rules stated above
- For relationship `Recommends`, the primary keys of the three participating entity types are included in the resulting relation schema
- Relationship has a generic type (k:m:n relationship): all primary keys together form the key

N-ary Relationships: Result

1. $\texttt{RECOMMENDS} = \{\texttt{WName}, \texttt{DName}, \texttt{Name}\}$ with $K_{\texttt{RECOMMENDS}} = \{\{\texttt{WName}, \texttt{DName}, \texttt{Name}\}\}$
2. $\texttt{DISH} = \{\texttt{DName}, \texttt{Side\_Dish}\}$ with $K_{\texttt{DISH}} = \{\{\texttt{DName}\}\}$
3. $\texttt{WINE} = \{\texttt{Color}, \texttt{WName}, \texttt{Vintage}, \texttt{Res\_Sugar}\}$ with $K_{\texttt{WINE}} = \{\{\texttt{WName}\}\}$
4. $\texttt{CRITIC} = \{\texttt{Name}, \texttt{Organization}\}$ with $K_{\texttt{CRITIC}} = \{\{\texttt{Name}\}\}$

- The three key attributes of RECOMMENDS are foreign keys to the respective source relations (CRITIC, WINE, DISH).

Overview and Summary

Overview of Transformations

ER Concept	Is Translated into Relational Concept
Entity type E_i	Relation schema R_i
Attributes of E_i	Attributes of R_i
Primary key P_i	Primary key P_i
Relationship type	Relation schema
	Attributes: P_1, P_2
Its attributes	Further attributes
$1:n$	P_2 becomes primary key of the relationship
$1:1$	P_1 and P_2 become key of the relationship
$m:n$	$P_1 \cup P_2$ becomes primary key of the relationship
IS-A relationship	R_1 gets an additional key P_2

E_1, E_2: Entity types participating in a relationship,

P_1, P_2: Their primary keys,

$1:n$ relationship: E_2 is n-side,

IS-A relationship: E_1 is a special entity type

Summary

- Phases of database design

Summary

- Phases of database design
- Capacity-preserving transformations

Summary

- Phases of database design
- Capacity-preserving transformations
- Transformation ER $\rightarrow$ relational

Control Questions

- Which steps does the database design process comprise?

Control Questions

- Which steps does the database design process comprise?
- Which requirements do the transformations between each design step have to fulfill? Why?

Control Questions

- Which steps does the database design process comprise?
- Which requirements do the transformations between each design step have to fulfill? Why?
- How are concepts of the ER model translated into concepts of the relational model?

Control Questions

- Which steps does the database design process comprise?
- Which requirements do the transformations between each design step have to fulfill? Why?
- How are concepts of the ER model translated into concepts of the relational model?
- How are the different cardinalities of relationship types accounted for during transformation?