Abbildungen zwischen Schemata unterschiedlicher Datenmodelle

Mastering the Information Explosion -
Information Integration and Information Quality

Susanne Braun
07.07.2006

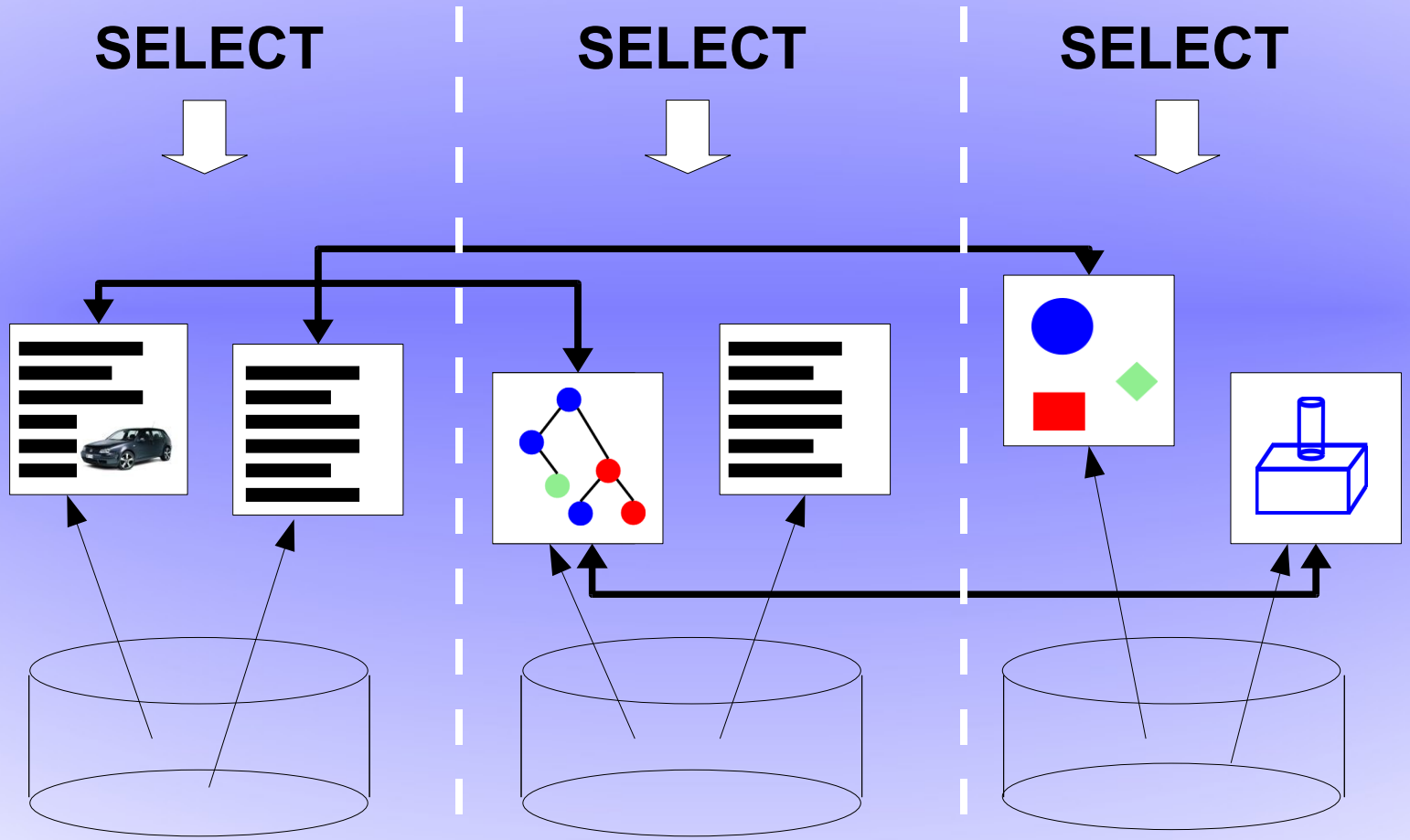
Inhalt

1. Motivation
2. Die 5-Ebenen-Schema-Architektur
3. Semantische Kongruenz
4. Probleme bei der Schemaintegration und -transformation
5. Probleme bei der Abbildung auf objektorientierte Daten

Inhalt

6. Probleme bei der Abbildung auf XML-Daten
7. BRIITY
8. Pegasus
9. SQL/XML

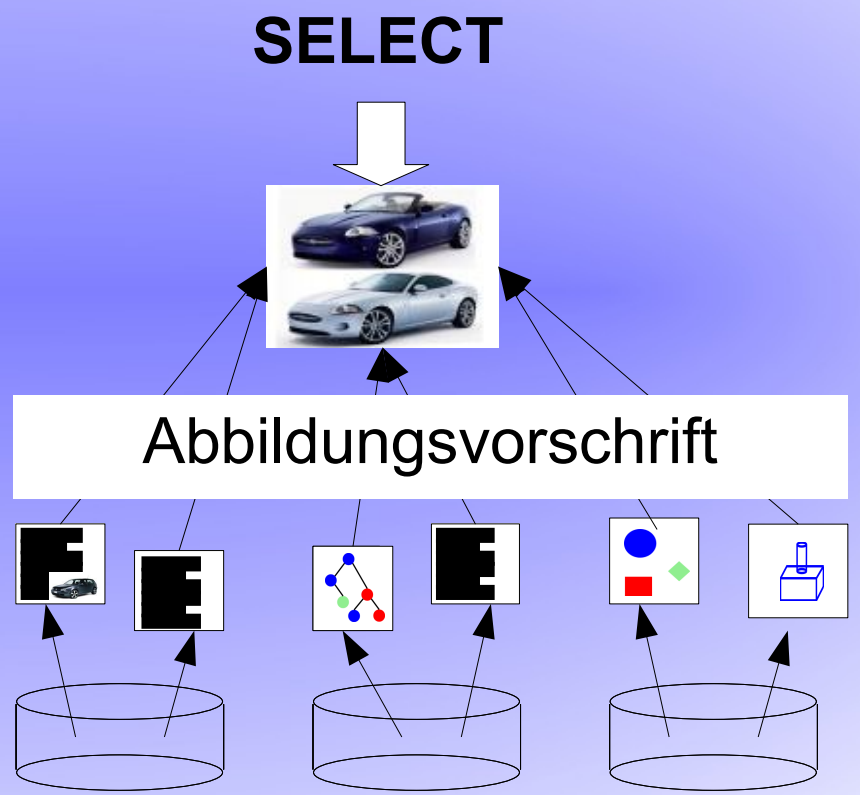
Motivation



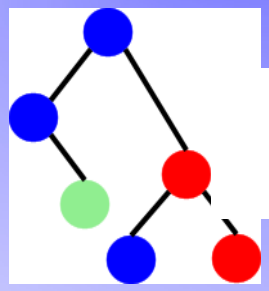
Motivation

Integriertes Schema:

- Kapselung
- Heterogenitäts- und Ortstransparenz
- Daten-unabhängigkeit
- Dokumentation



Motivation



Produktstruktur Fahrzeug x

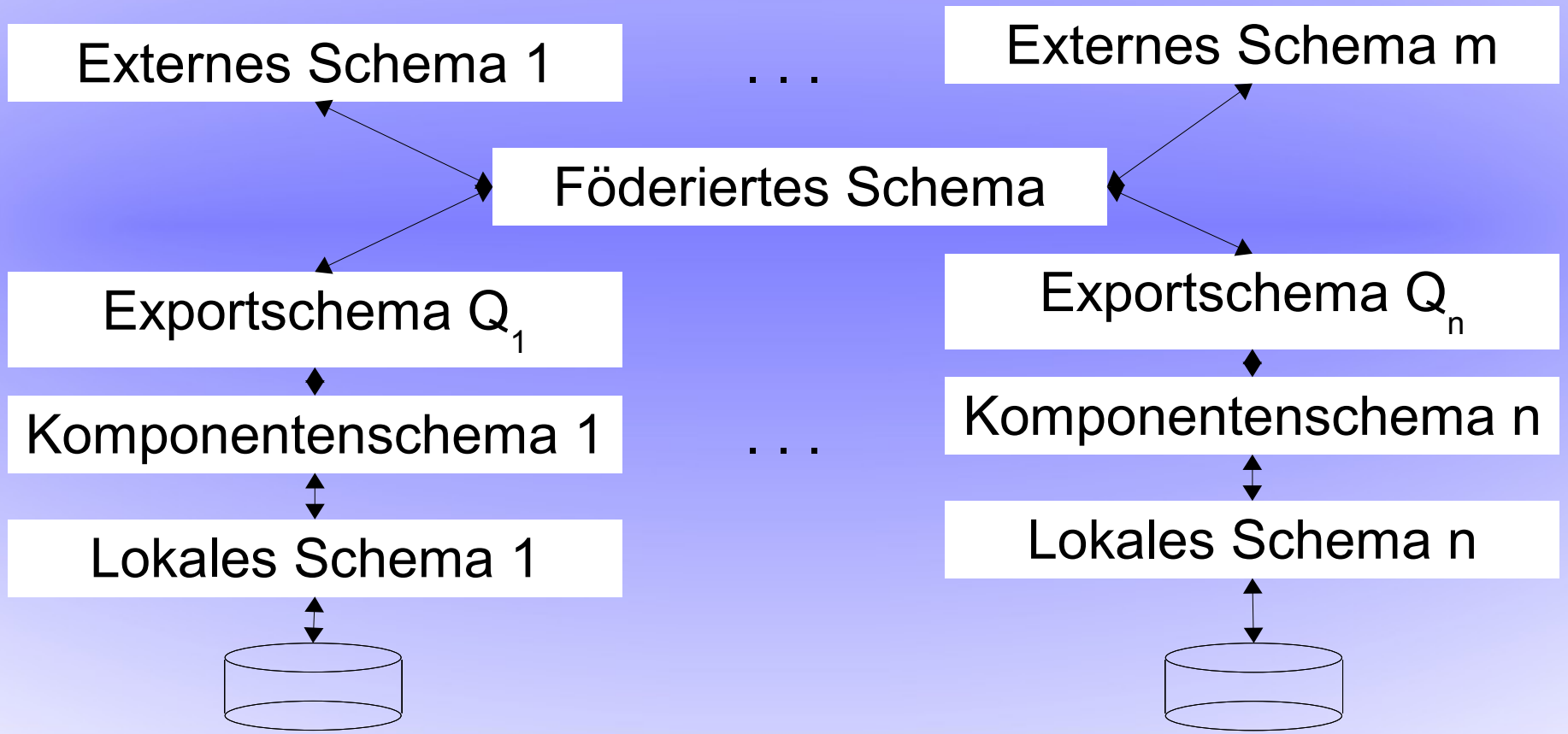
Fahrzeug x

- Antrieb — Motor
- Kraftübertragung — Getriebe
- Fahrwerk
- Elektronik
- Karosserie

Fahrzeug x

- Motor
- Getriebe
- Fahrwerk
- Elektronik
- Karosserie

5-Ebenen-Schema-Architektur



Semantische Kongruenz

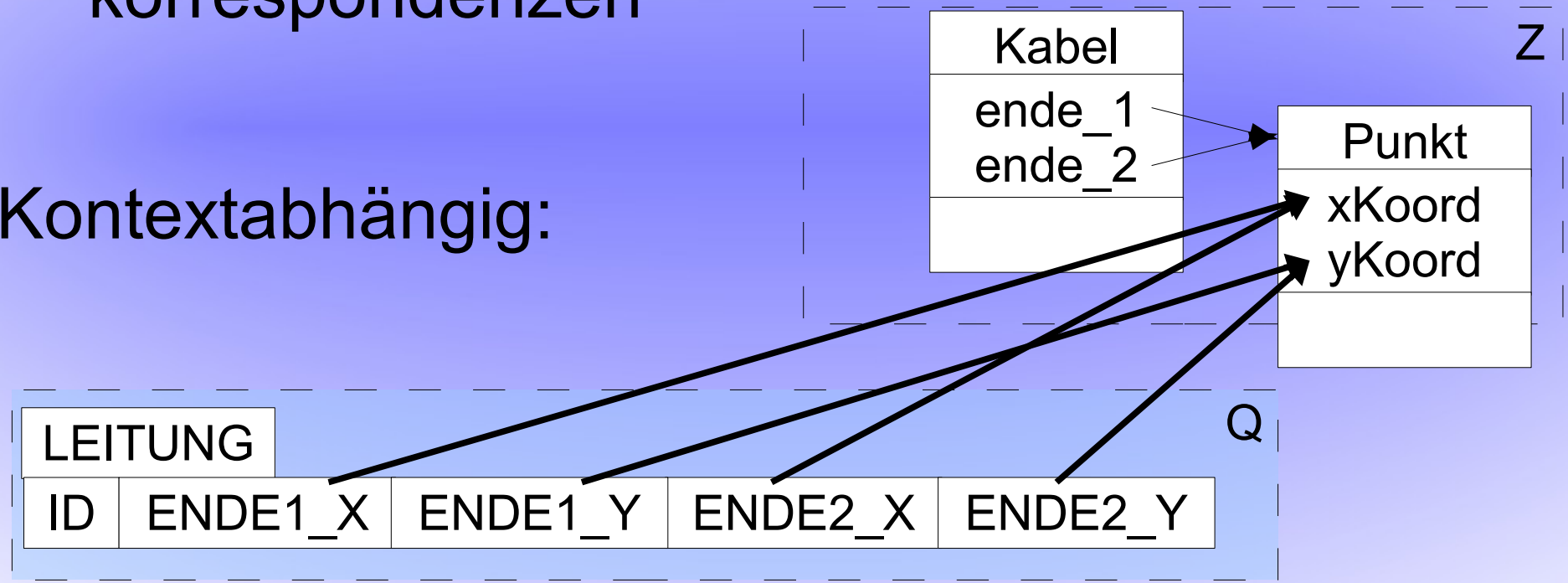
„Übereinstimmung des betrachteten Ausschnitts der realen Welt.“ (Sauter)

- Identischer Weltauschnitt, unterschiedliche Anwendungsbereiche
- Überlappende Weltauschnitte

Objekttypkorrespondenzen

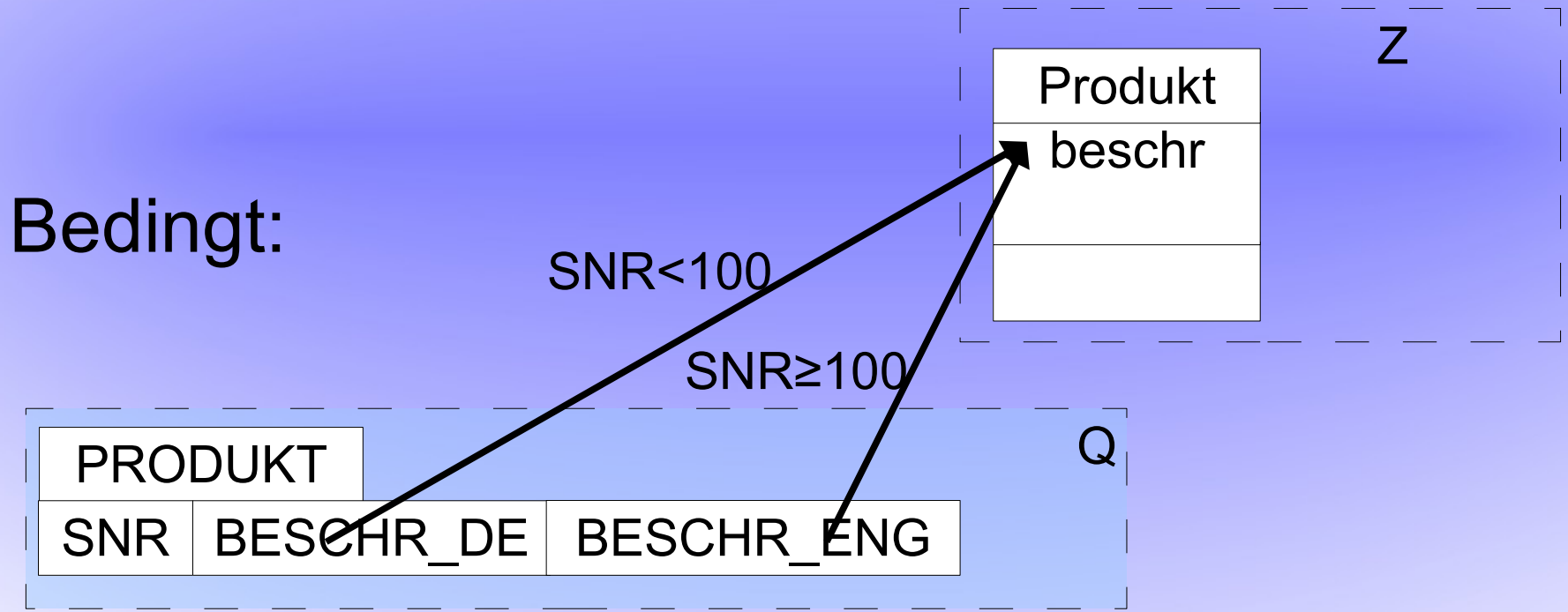
- 1:1-, 1:n-, n:1- und n:m-Objekttyp-korrespondenzen

Kontextabhängig:



Attributkorrespondenzen

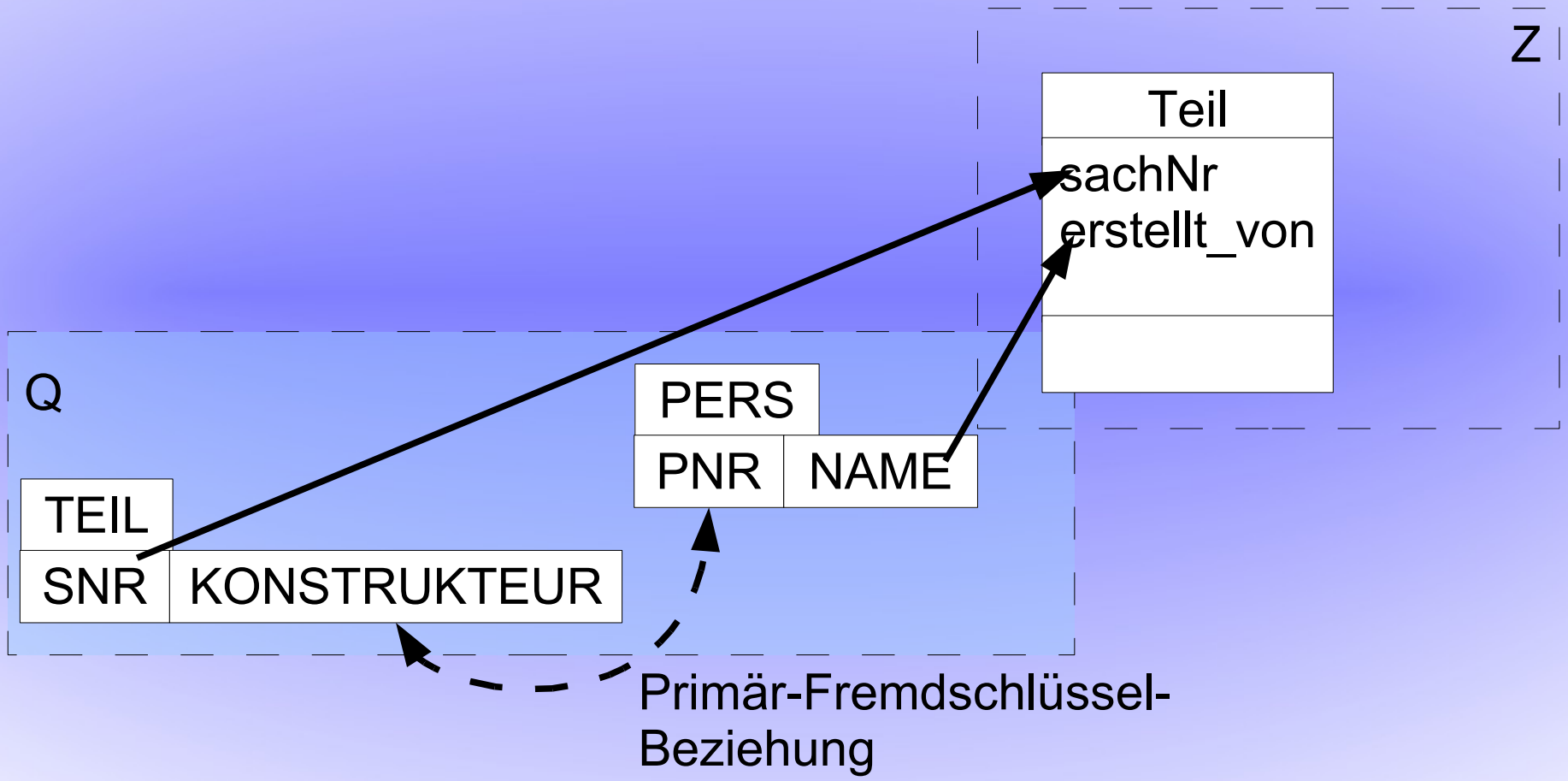
- 1:1-, 1:n-, n:1- und n:m-Attributkorrespondenzen



Bidirektionale Abbildungen

- Schreibende Operationen auf den Daten des Zielschemas erfordern Abbildung $Z \rightarrow Q_i$ ($i=1,...,n$, Z Zielschema, Q_i Quellschema)
- View-Update-Problem: Abbildung $Q_i \rightarrow Z$ nicht eindeutig umkehrbar
- Möglichkeit zur expliziten Definition der Umkehrabbildung erwünscht

Bidirektionale Abbildungen



Vernetzung von Objekttypen

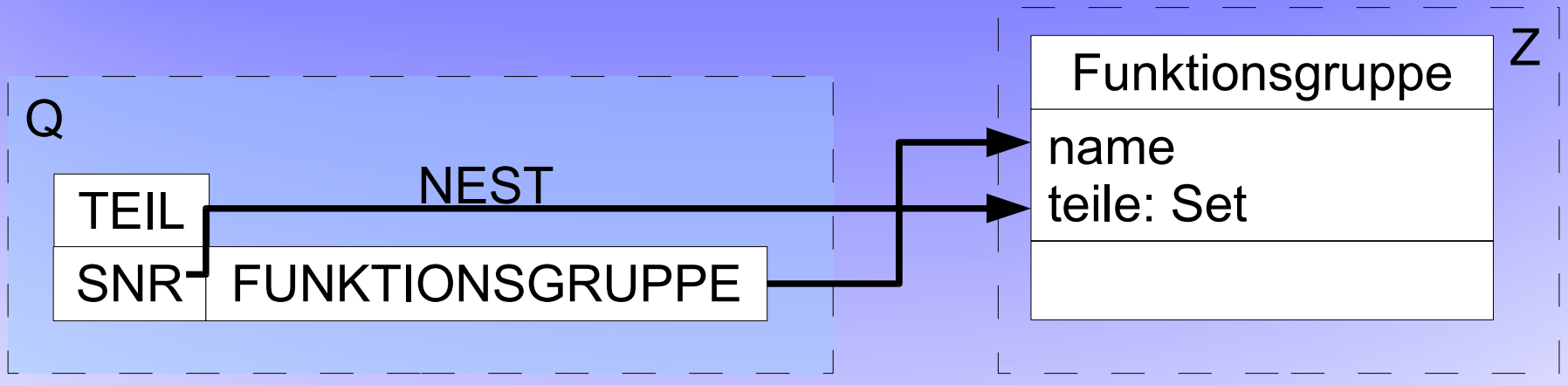
- Abbilden von Referenzen und Schlüsselbeziehungen
- Semantische Anreicherung:
  - Beziehungen die bisher nicht dargestellt wurden (werden konnten), sollen im Zielschema nachgebildet werden.
- Referentielle Integrität bei bidirektionalen Abbildungen

Weitere elementare Probleme

- Schemakardinalität:
  - Systemübergreifende Verbundoperationen
  - Horizontale semantische Korrespondenz
  - Konsistenz
- Bezeichner
- Datentypkorrespondenzen

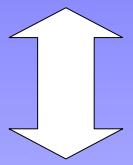
Umsetzung des objektorientierten Paradigmas

- mengenwertige Attribute: Nest- und Unnest-Operation



Umsetzung des objektorientierten Paradigmas

- Objektidentität:
  - systemweit eindeutig für die gesamte Lebensdauer
  - unabhängig von Attributwerten

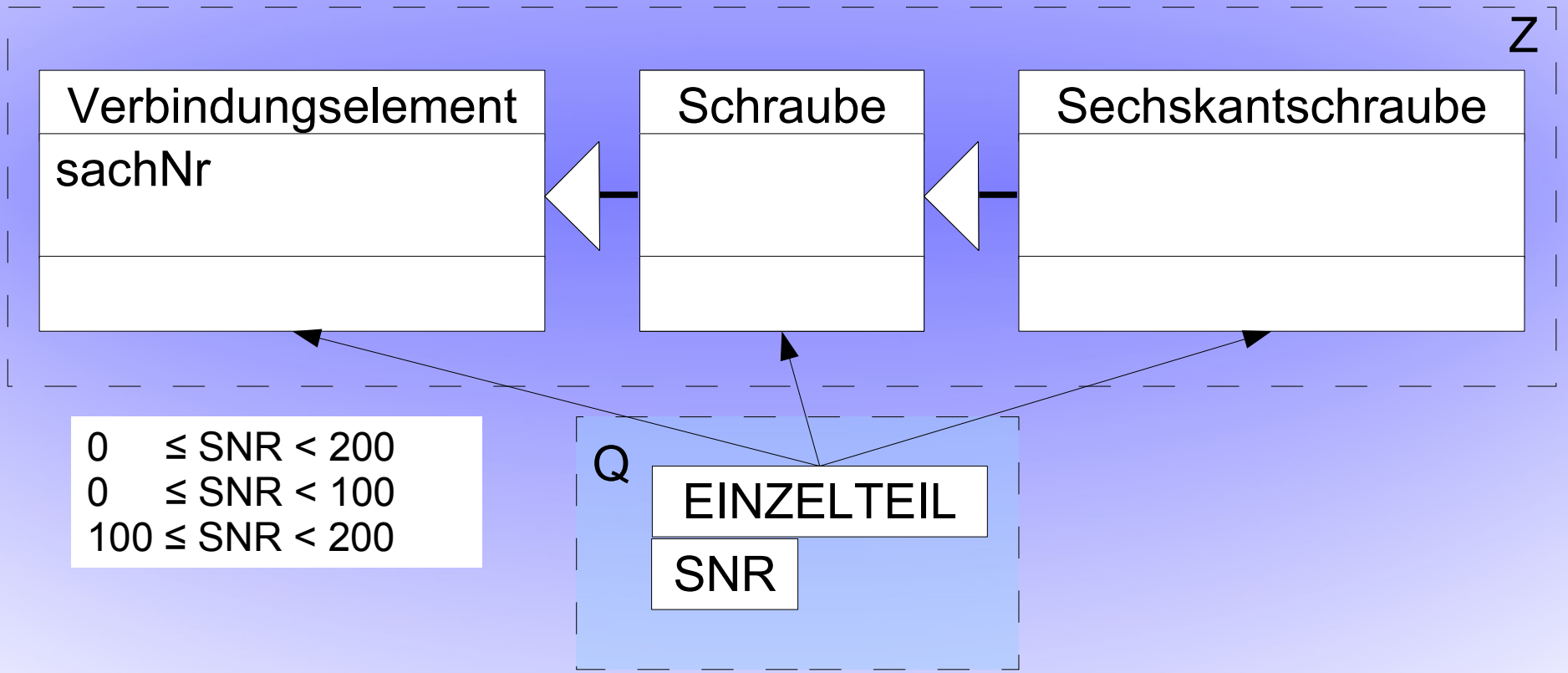


  - Relationale Identifikation anhand von Attributwerten
  - Primärschlüssel nur innerhalb einer Relation eindeutig

Umsetzung des objektorientierten Paradigmas

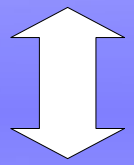
- Spezialisierungs- und Generalisierungsbeziehungen:
  - OO → Rel: Vertikale Partitionierung, Hausklassenmodell, Volle Redundanz
  - Rel → OO: (Sub-)Typ muss anhand von Prädikaten über Attributen eindeutig feststellbar sein.

Umsetzung des objektorientierten Paradigmas



XML-Daten

- Reihenfolge von XML-Elementen



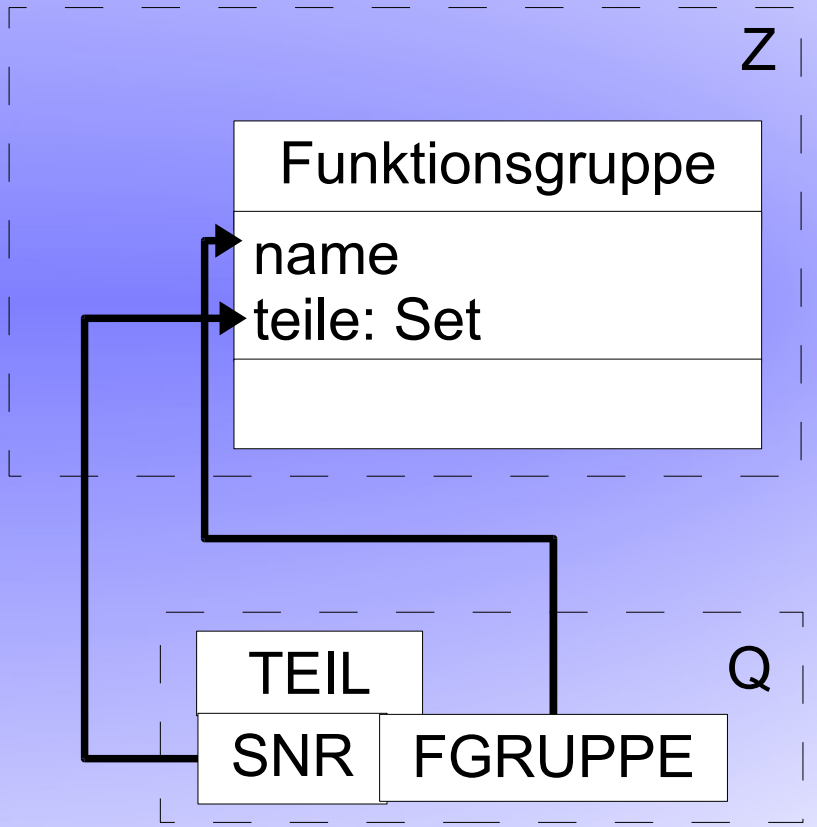
  Mengenwertigkeit von Relationen
- Hierarschiche Struktur und mengenwertige Elemente
  - Nest- und Unnest-Operation

BRIITY (Bridging Heterogeneity)

- Deklarativ
- Anhlehnung an SQL
- Integration relationaler und objektorientierter Datenbanksysteme
- Relationales oder objektorientiertes Zielschema
- mächtige Abbildungssprache

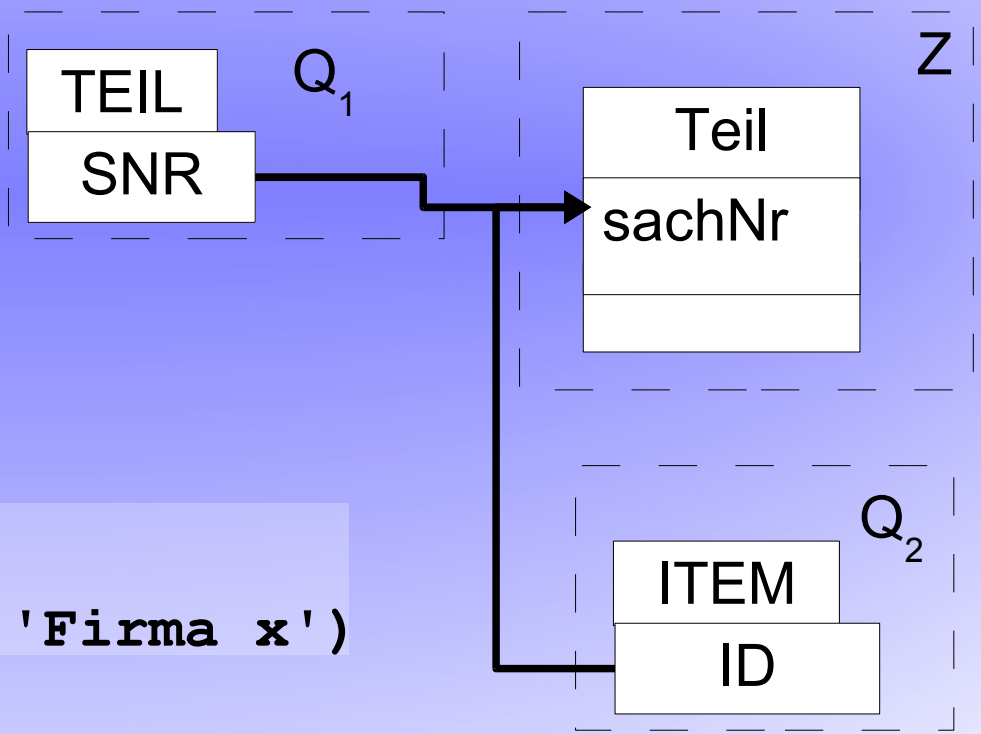
BRIITY: Nesting & Objektidentität

```
MAP FUNKTIONSGRUPPE
FROM t:= Q.TEIL
ON_RETRIEVE
    name= t.FGRUPPE;
    teile= NEST(t.SNR);
    GROUPED_BY t.FGRUPPE;
    IDENTIFIED_BY(t.FGRUPPE);
    . . .
END_MAP;
```



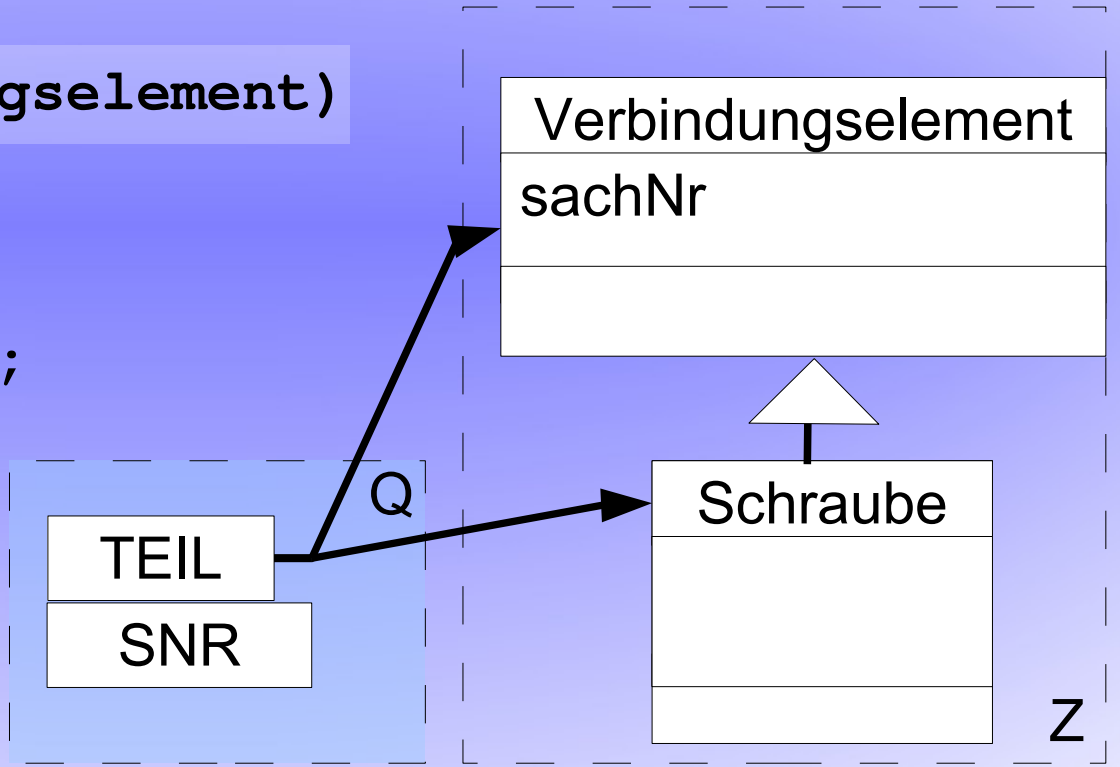
BRIITY: Globale Objektidentität & semantische Kongruenz

```
MAP Teil
FROM t:= Q1.TEIL;
     i:= Q2.ITEM;
PARTITION par_q1
   ON_RETRIEVE
      sachNr= t.SNR;
      IDENTIFIED_BY(t.SNR);
      GLOBAL_IDENTITY(t.SNR,'Firma x')
PARTITION par_q2 . . .
```



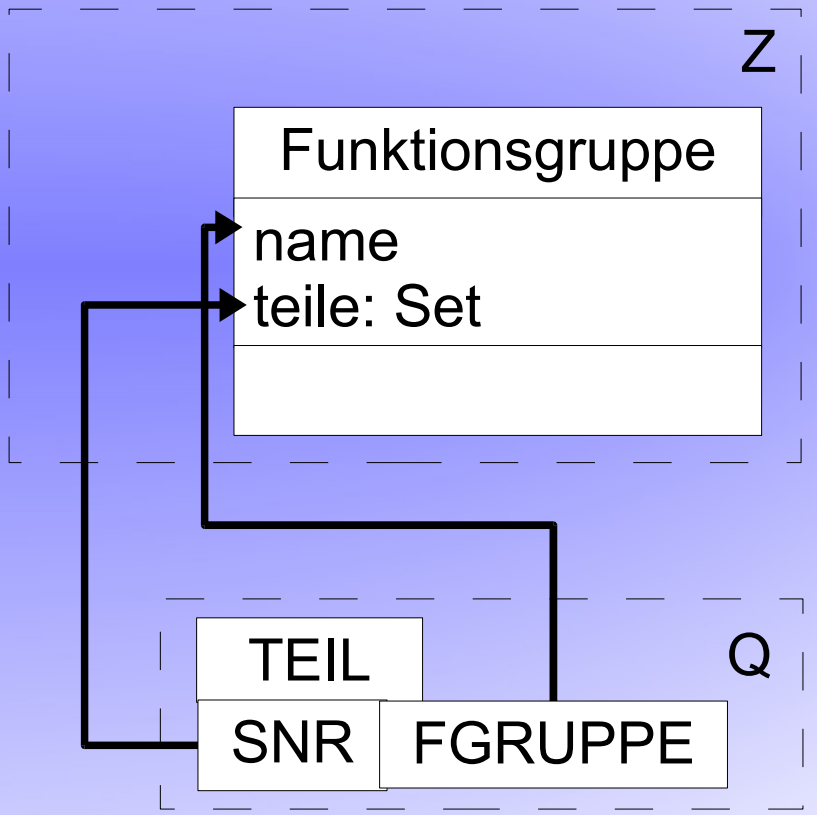
BRIITY: Generalisierungshierarchie

```
MAP Schraube
   SUBTYPE_OF(Verbindungselement)
FROM t:= Q.TEIL;
ON_RETRIEVE
   IDENTIFIED_BY(t.SNR);
   WHERE t.SNR ≥ 100
   AND t.SNR < 200;
. . .
```



BRIITY: Bidirektionale Abbildungen

```
ON_INSERT
   FOR EACH_ELEMENT_VALUE t
   OF Funktionsgruppe.teile
   DO ASSIGN IS_INSTANCE(
      Q.TEIL :
         SNR= t,
         FGRUPPE=
         Funktionsgruppe.name
   );
```

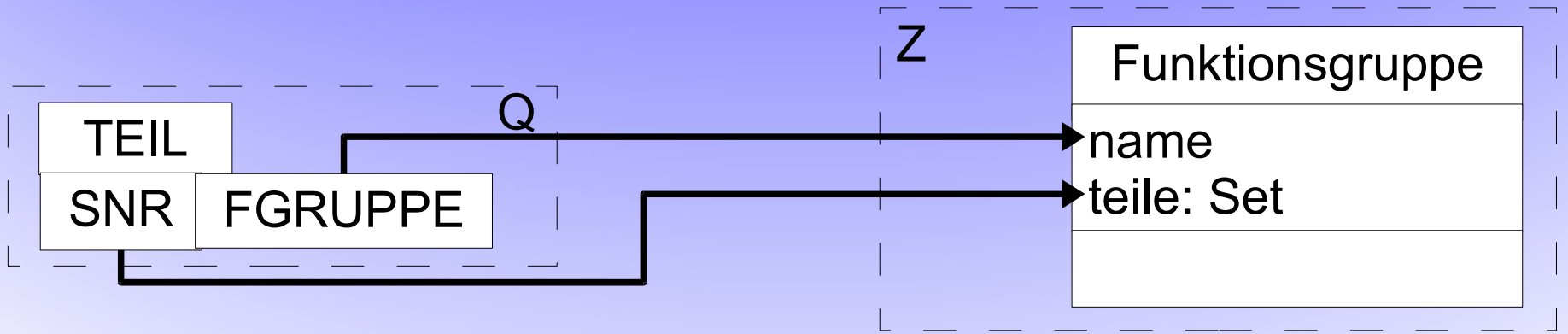


Pegasus

- Deklarative und imperative Elemente
- Erweiterung von SQL: HOSQL
- Integration relationaler und objektorientierter Datenbanksysteme
- Objektorientiertes Zielschema
- Nur lesender Zugriff

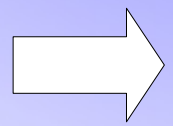
Pegasus: Importieren von Quellschemata

```
CREATE TYPE Funktionsgruppe AS
IMPORTED FROM . . . TEIL
PRODUCING_BY(FGRUPPE)
FUNCTIONS(name STRING UNIQUE AS MATCHING FGRUPPE);
```



Pegasus: Nest-Operation

```
CREATE FUNCTION teile(Funktionsgruppe f)
-> SETTYPE(STRING s) AS
IMPORTED FROM . . . TEIL(
   name(f) AS MATCHING FUNKTIONSGRUPPE;
   s AS MATCHING SNR );
```



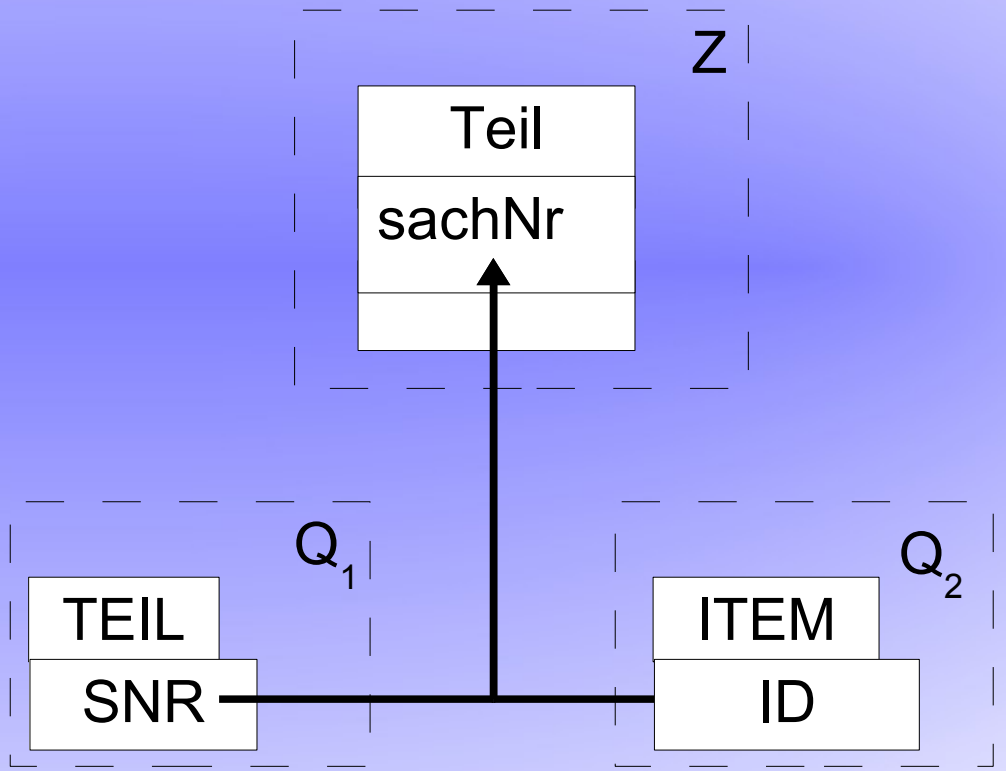
Abbildungs- und Integrationsprobleme werden hauptsächlich innerhalb von Funktionen aufgelöst.

Pegasus: Semantische Kongruenz

```
CREATE TYPE Teil_Q1
AS IMPORTED . . . ;

CREATE TYPE Teil_Q2
AS IMPORTED . . .;

CREATE TYPE TEIL
AS COVERING SUPERTYPE
OF Teil_Q1, Teil_Q2;
```



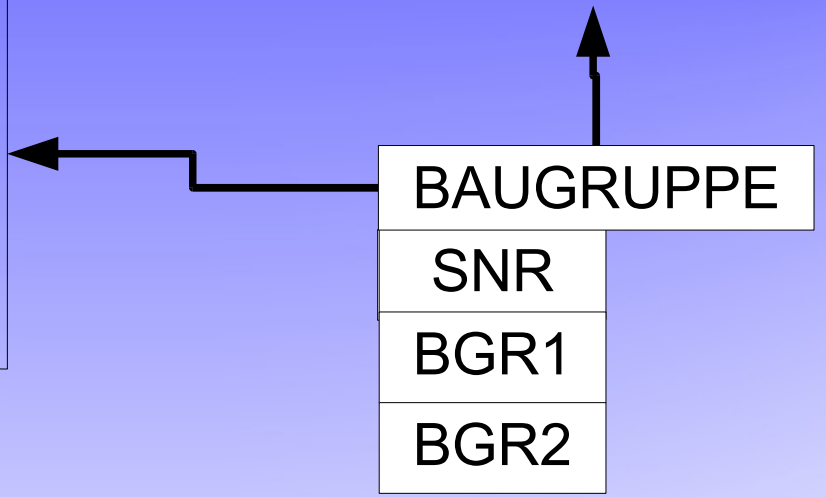
SQL/XML

- SQL-Datentypen auf XML-Schema-Datentypen abbilden
- Erzeugen von XML-Daten aus relationalen Daten
- Publishing Paradigm:
  - standardisierte Ausgabe relationaler Daten als XML-Daten
  - XML-Sichten auf relationale Daten

SQL/XML: standardisierte Ausgabe

```
<BAUGRUPPE>
   <row>
      <SNR>BGR1</SNR>
   </row>
   <row>
      <SNR>BGR2</SNR>
   </row>
</BAUGRUPPE>
. . .
```

Beschreibung des Zielschemas durch XML Schema

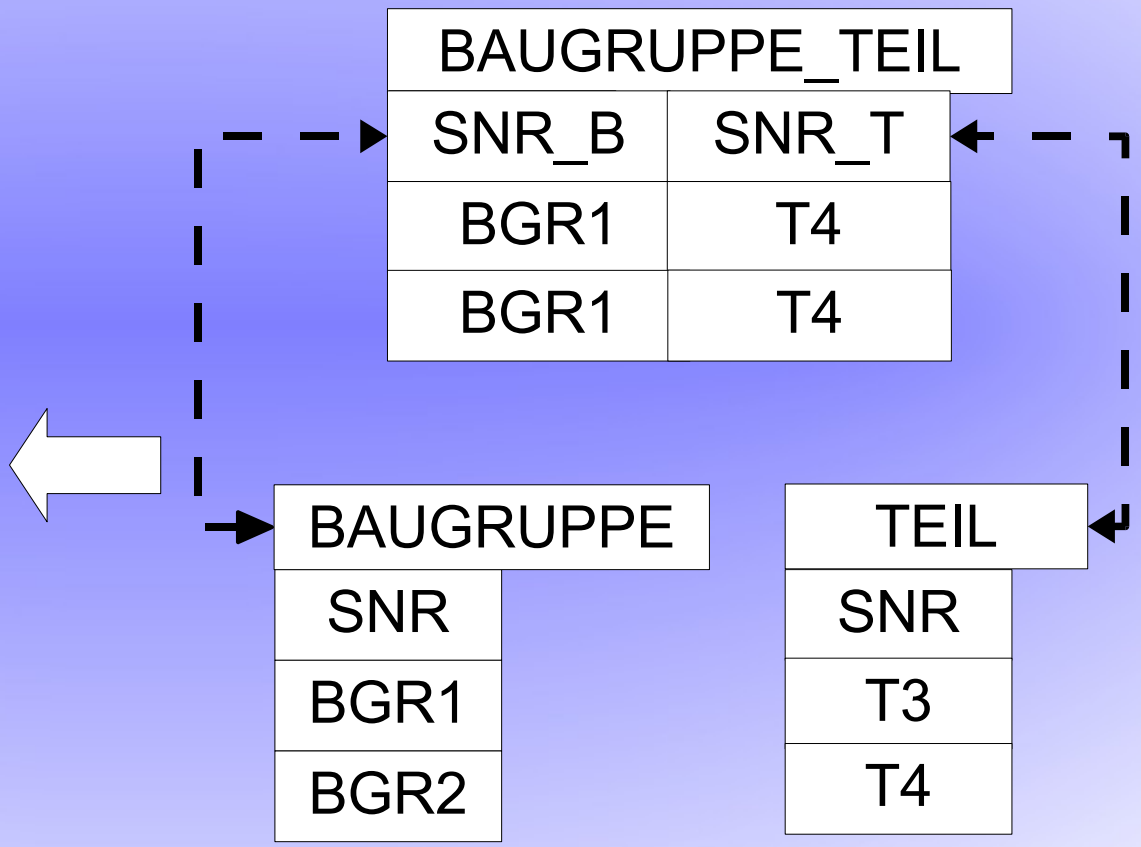


SQL/XML: Erzeugen von XML-Daten

- SQL um zusätzliche Funktionen erweitern:
  - XMLELEMENT
  - XMLATTRIBUTE
  - XMLFOREST
  - XMLCONCAT
  - XMLAGG

SQL/XML: Beispiel

```
<BAUGRUPPE SNR='BGR1'>
   <Teil SNR=T4/>
   <Teil SNR=T3/>
</BAUGRUPPE>
. . .
```



SQL/XML: SQL-Erweiterung

```
SELECT XMLELEMENT(
   NAME 'BAUGRUPPE',
   . . .
   XMLAGG(
      XMLELEMENT(
         NAME 'TEIL',
          . . .
)))
FROM BAUGRUPPE B, BAUGRUPPE_TEIL BG
WHERE B.SNR = BT.SNR_B
GROUP BY B.SNR;
```

SQL/XML: SQL und XQuery

```
CREATE VIEW BAUGRUPPEN AS
FOR $B IN TABLE('BAUGRUPPE')/BAUGRUPPE/row RETURN
<BAUGRUPPE SNR='{data($B/SNR)}'>

   FOR $BT IN TABLE('BAUGRUPPE_TEIL')/BAUGRUPPE_TEIL/row
      WHERE $B/SNR = $BT/SNR_B RETURN
         <Teil SNR='{data($BT/SNR_T)}'/>

</BAUGRUPPE>
```

Vielen Dank für die Aufmerksamkeit

Fragen?