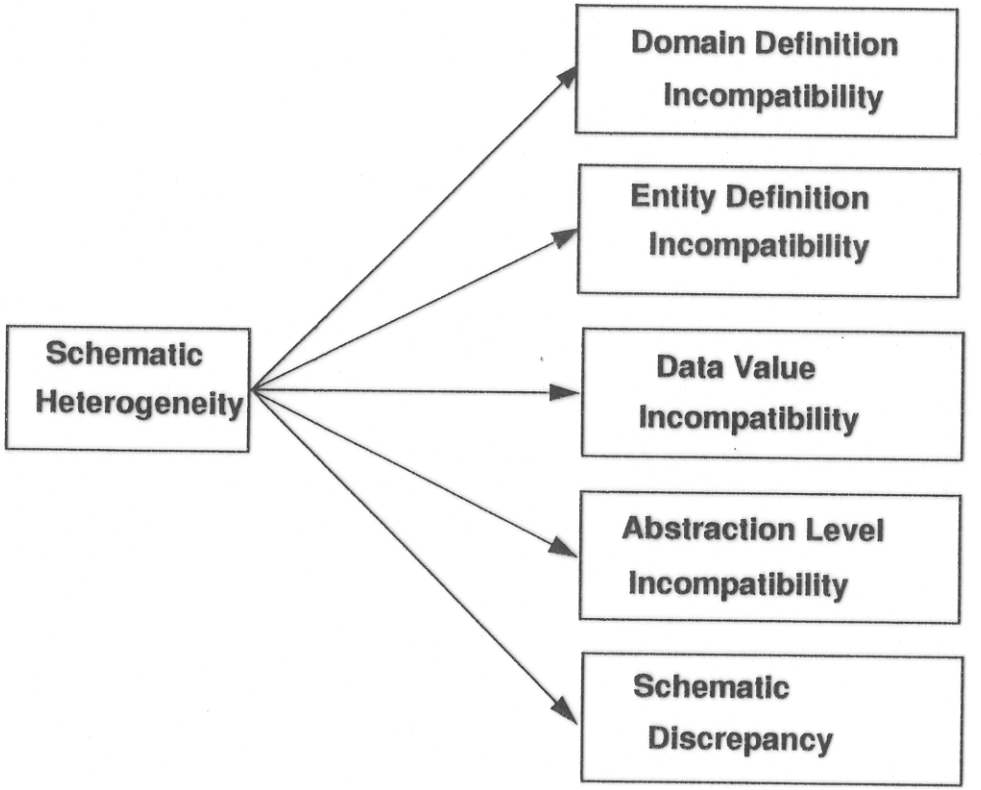
Problems in Enterprise Integration:

Schematic Heterogeneities between Semantically similar objects

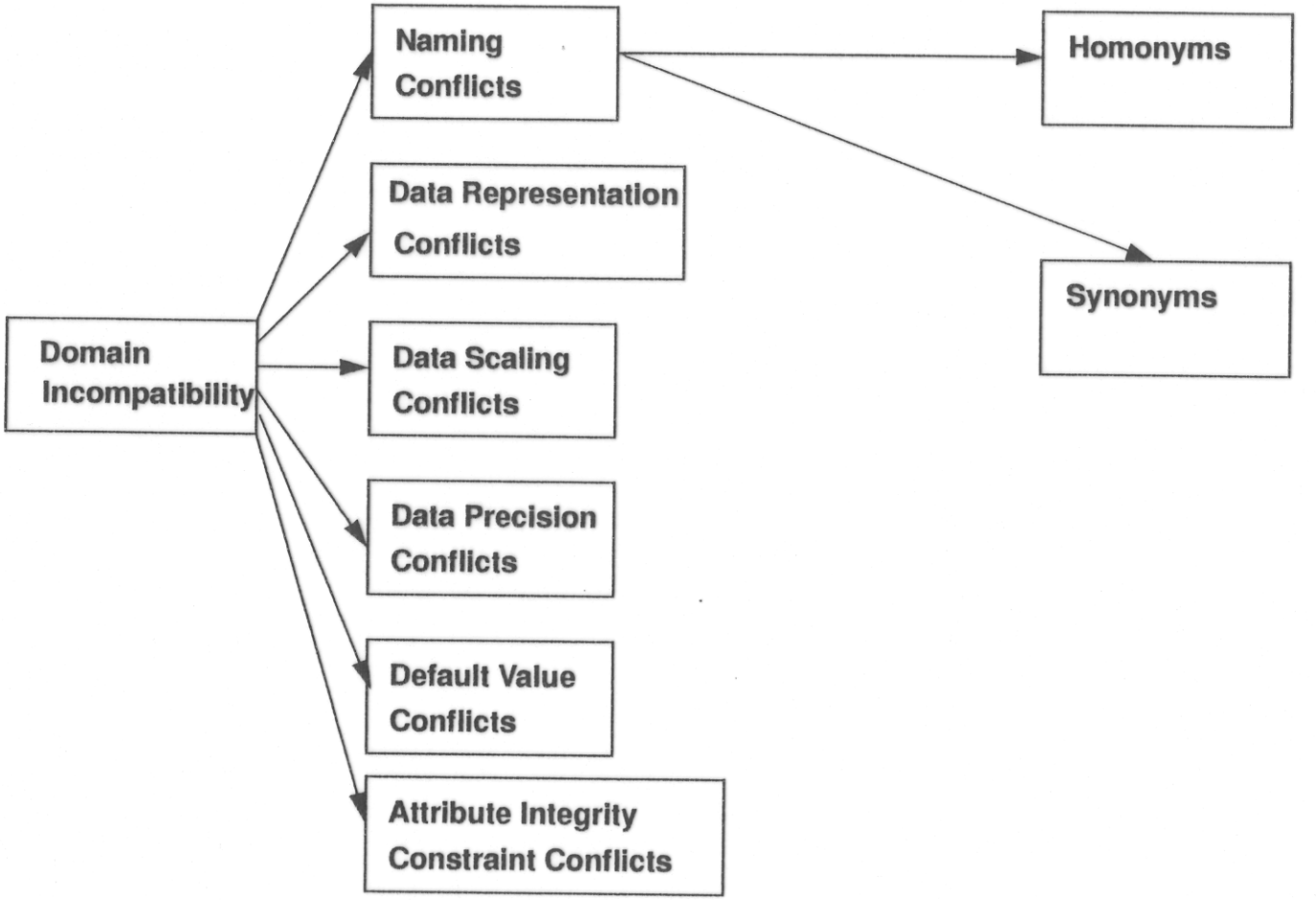
Based On:

A. Sheth and V. Kashyap, So Far (Schematically) yet So Near (Semantically), Invited paper in Proceedings of the IFIP TC2/WG2.6 Conference on Semantics of Interoperable Database Systems, DS-5, November 1992

Schematic Heterogeneities



Domain definition Incompatibilities



Domain definition Incompatibilities

Naming Conflicts

- Semantically alike attributes may have different names and are called **synonyms**

  STUDENT(Id#, Name, Address) in database DB1
  GRADUATE(SS#, Name, Address) in database DB2
  STUDENT.Id# and GRADUATE.SS# are synonyms

- Semantically unrelated attributes may have the same names and are called **homonyms**

  STUDENT(Id#, Name, Address) in database DB1
  BOOK(Id#,Title, Author) in database DB2
  STUDENT.Id# and BOOK.Id# are homonyms

  Solution:
  Synonyms can be renamed to the same concept.
  Homonyms can be renamed to different concepts.
  The associations are kept in a catalog

Domain definition Incompatibilities

Data Representation Conflicts

Semantically alike attributes may have different data types or representations.

STUDENT(Id#, Name, Address) in database DB1

GRADUATE(SS#, Name, Address) in database DB2

STUDENT.Id# is defined as INTEGER

GRADUATE.SS# is defined as Char(9)

Solution: The type of SS# is coerced to be an integer. A virtual class U_STUDENT is defined as follows:

```
CREATE VCLASS U_STUDENT
        (Id: INTEGER, Name: Char(20), Address: Char(50))
AS SELECT Id#, Name, Address
     FROM DB1.STUDENT,
   SELECT SS# AS INTEGER, Name, Address
     FROM DB2.GRADUATE
```

Domain definition Incompatibilities

Data Scaling Conflicts

Semantically alike attributes represented using different units and measures

EMPLOYEE(Name, SS#, Salary) in database DB1
Salary is stored using British pounds
EMPLOYEE(Name,SS#,Salary) in database DB2
Salary is stored using US dollars

Solution: Use of appropriate conversion expressions. Assuming 1 British pound = 1.6 US dollars. A virtual class U_EMPLOYEE is defined:

```
CREATE U_EMPLOYEE
(Name: char(20), SS#: INTEGER, Salary: REAL)
AS SELECT Name, SS#, 1.6*Salary
      FROM DB1.EMPLOYEE,
   SELECT Name, SS#, Salary
     FROM DB2.EMPLOYEE
```

Domain definition Incompatibilities

Data Precision Conflicts

Semantically alike attributes represented using different precisions.

Lower	Upper	Grade
81	100	A
61	80	B
41	60	C
21	40	D
0	20	F

STUDENT(Name,SS#,Marks) in database DB1
STUDENT(Name,SS#,Grade) in database DB2
Grade is represented at a **coarser** level of precision when compared to Marks
=> Need to Map Marks to Grade represented in the Table

Solution: Let the table MAP(Lower,Upper,Grade) be defined in DB1
A Virtual Class U_STUDENT can be defined:

```
CREATE VCLASS U_STUDENT
      (Name: Char(20), SS#: INTEGER, Grade: CHAR(1))
AS SELECT Name, SS#, MAP.Grade
     FROM DB1.STUDENT, MAP
     WHERE Marks <= MAP.Upper and Marks >= MAP.Lower
   SELECT Name, SS#, Grade
     FROM DB2.STUDENT
```

Domain definition Incompatibilities

Default Value Conflicts

Semantically alike attributes may have different default values in different databases.

EMPLOYEE(Name,SS#,Salary,Bonus) in database DB1
where default(Bonus) = 10%
EMPLOYEE(Name,SS#,Salary,Bonus) in database DB2
where default(Bonus) = 30%

Solution: Default value for attribute of virtual class is used.
Update of attribute value not propagated to databases.

Suppose default(Bonus) for VCLASS U_EMPLOYEE is 20%
Query retrieving values for bonus attribute will return a value corresponding to 20%

However, update of Bonus value will not be propagated to DB1 or DB2

Domain definition Incompatibilities

Attribute Intergrity Constraint Conflicts

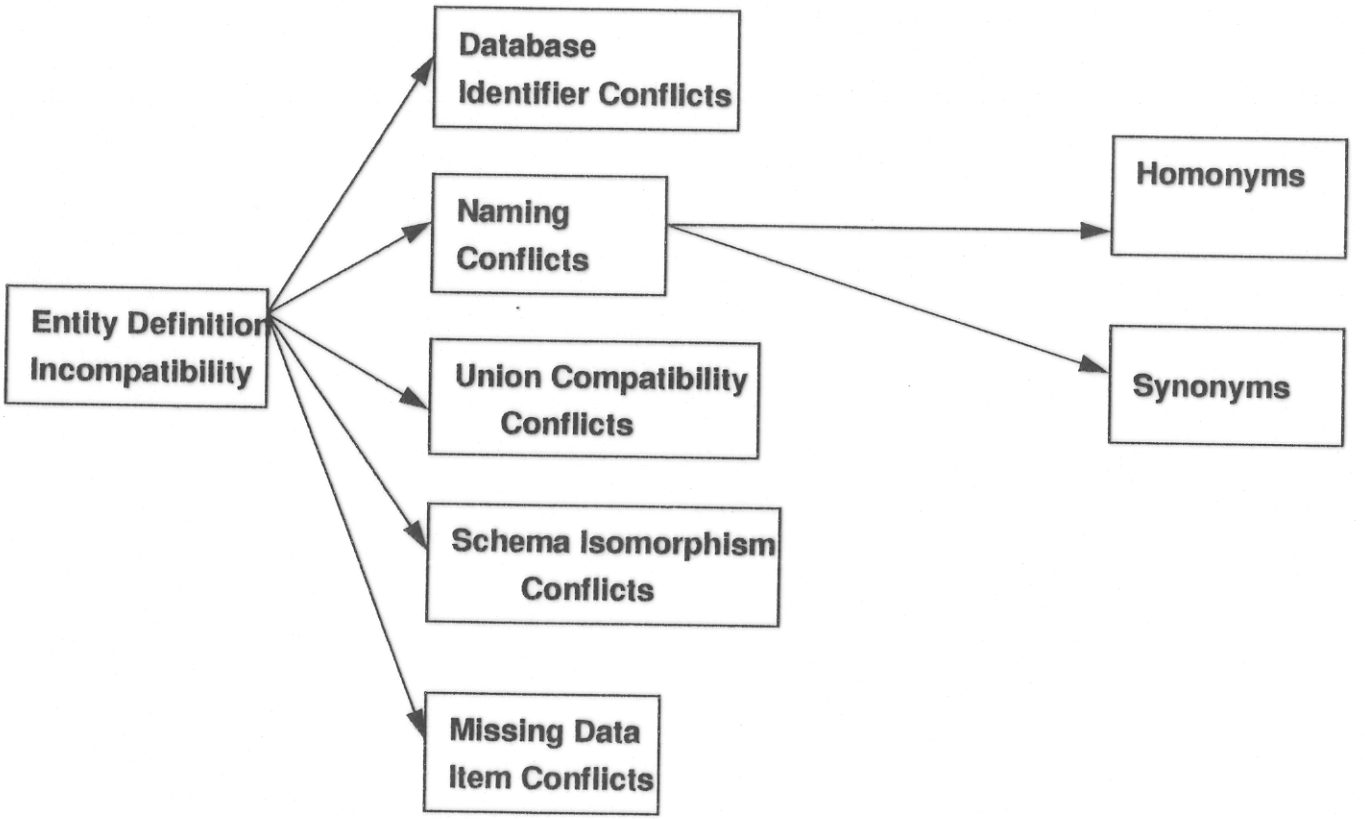
Semantically alike attributes might be restricted by constraints that are not consistent with each other. Consider the attribute Age in the databases DB1 and DB2:

C1: Age < 18 defined in database DB1

C2: Age > 21 defined in database DB2

C1 and C2 defined on attribute Age are inconsistent

Entity definition Incompatibilities



Entity definition Incompatibilities

Database Identifier Conflicts

- Entity descriptions use identifier records that are semantically different
- Also known as the **key equivalence** problem

STUDENT(SS#, Course, Grades) in database DB1

STUDENT(Name, Course, Grades) in database DB2

SS# in DB1and Name in DB2 are semantically different identifiers

Entity definition Incompatibilities

Naming Conflicts

- Semantically alike entities may have different names and are called **synonyms**

  EMPLOYEE(SS#, Name, Address) in database DB1
  WORKER(SS#, Name, Address) in database DB2
  EMPLOYEE and WORKER are synonyms

- Semantically unrelated entities may have the same names and are called **homonyms**

  TICKET(Movie, Theater, Date, Timing) in database DB1

  TICKET(Date, Offense, Officer, CourtDate) in database DB2

  The entities TICKET in DB1 and DB2 are homonyms

  Solution:

  Synonyms can be renamed to the same concept.

  Homonyms can be renamed to different concepts.

  The associations are kept in a catalog

Entity definition Incompatibilities

Union Compatibility Conflicts

Descriptors of semantically alike entities may not have equivalent signatures.

STUDENT(SS#, Name, DateOfBirth) in database DB1
STUDENT(SS#, Name, Age) in database DB2

The attributes Age and DateOfBirth are not equivalent but can be made equivalent after some transformations

```
CREATE VCLASS U_STUDENT
      (SS#: INTEGER, Name: Char(20), Age: INTEGER)
AS SELECT SS#, Name, round(getdate()-DateofBirth)
       FROM DB1.STUDENT,
      SELECT SS#, Name, Age
       FROM DB2.STUDENT
```

Entity definition Incompatibilities

Missing Data Item Conflict

Descriptors of semantically alike entities may have different number of attributes

PROFESSOR(Name, OffPhone, HomePhone) in database DB1
PROFESSOR(Name, Phone) in database DB2

The attribute Phone in DB2 can be assumed to have both the Office and the Home Phone numbers

```
CREATE VCLASS U_PROFESSOR
      (Name: Char(20), Phone: INTEGER)
AS SELECT Name, OffPhone
      FROM DB1.PROFESSOR,
  SELECT Name, HomePhone
      FROM DB1.PROFESSOR,
 SELECT Name, Phone
      FROM DB2.PROFESSOR
```

Entity definition Incompatibilities

Missing Data Item with Implicit Value

Descriptor of a similar entity may have a missing data item whose value is implicit

STUDENT(SS#, Name, Address, Type) in database DB1
GRADUATE(SS#, Name, Address) in database DB2

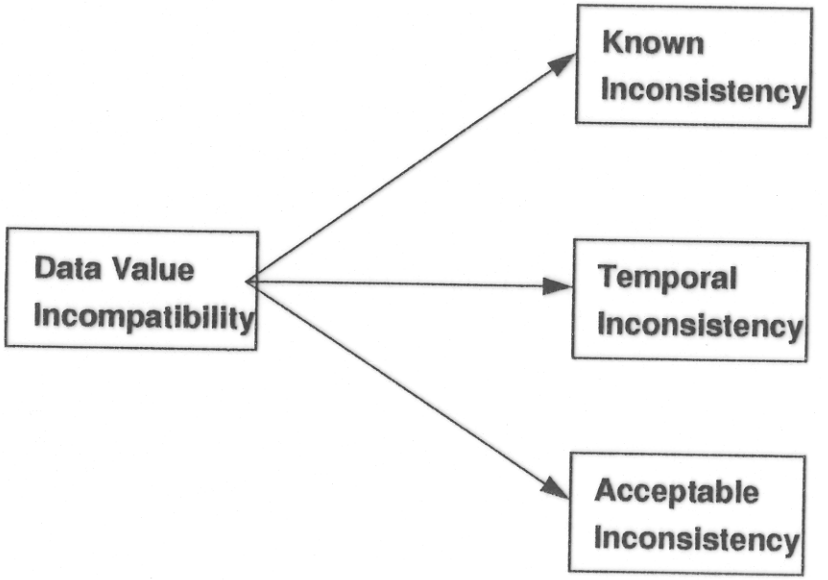
STUDENT.Type can have the values “Grad” or “UnderGrad”
GRADUATE has a missing data item whose value is implicit, i.e.
GRADUATE.Type = “Grad”

```
CREATE VCLASS U_STUDENT
 (SS#: INTEGER, Name: Char(20), Address: Char(50), Type: Char(9))
AS SELECT SS#, Name, Address, Type
      FROM DB1.STUDENT,
      SELECT SS#, Name, Address, Type
       FROM DB2.GRADUATE
       WHERE Type = “Grad”
```

Data Value Incompatibilities

Incompatibilities due to different values for the same entity

Consider the entity SHIP(Id#,Name,Weight)
SHIP(123,USSEnterprise,100) in database DB1
SHIP(123,USSEnterprise,200) in database DB2



Data Value Incompatibilities

Known Inconsistency

When the source of Inconsistency is identified

Solution: Choose the more reliable database

In the previous example:

If DB2 is identified as the more reliable database

=> weight(USSEnterprise) = 200

Data Value Incompatibilities

Temporal Inconsistency

- Inconsistency of a temporary nature.
- Specified by using a temporal consistency predicate.

In the previous example, the two weights may be considered temporarily inconsistent with DB1 updating it's value subject to the following constraints:

- At a particular date and/or at a specific point in time:
  AT 9:00 o'clock on 27th April 1995
- Before or after a specific instance of time or date:
  BEFORE 8:00 a.m. on 27th April 1995
- During intervals of time and/or date:
  INTERVAL(27th April 1995 AT 17:00 - 28th April 1995 AT 8:00)
- Periodically, when certain amount of time has elapsed:
  Every Day AT 12:00

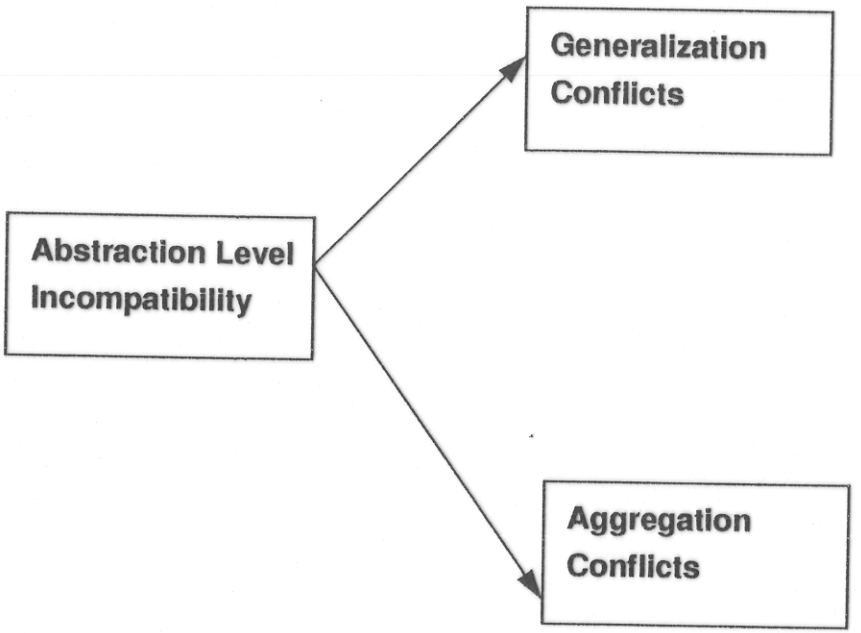
Data Value Incompatibilities

Acceptable Inconsistency

Depending on the type of the query, the inconsistency between the values from different databases might be within an acceptable range.

- **Numerical Inconsistency:**
  QUERY: Find the Tax Bracket of an Employee
  INCONSISTENCY: If **abs(Salary1 - Salary2) < 0.99**
  The inconsistency is ignored
- **Non-numerical Inconsistency:**
  QUERY: Find the State of residence of an Employee
  INCONSISTENCY: If **Residence1 IN State1** and **Residence2 IN State2** and **State1 = State2**
  The inconsistency is ignored

Abstraction Level Incompatibilities



Abstraction Level Incompatibilities

Generalization Conflicts

When the same entity is represented at different levels of generalization

STUDENT(SS#, Name, Major) in database DB1
GRAD-STUDENT(SS#, Name, Major, Advisor) in database DB2

A Virtual Class U_STUDENT is defined as follows:

```
CREATE VCLASS U_STUDENT
 (SS#: INTEGER, Name: Char(20), Major: Char(20), Advisor: Char(20))
AS SELECT SS#, Name, Major, NA
      FROM DB1.STUDENT
      WHERE SS# NOT IN (SELECT SS#
                              FROM DB2.GRAD-STUDENT),
      SELECT SS#, Name, Major, Advisor
      FROM DB2.GRAD-STUDENT
```

Abstraction Level Incompatibilities

Aggregation Conflicts

When an entity in one database corresponds to a set of entities (or any other aggregation) in another

CONVOY(Id#, AvgWeight) in database DB1
SHIP(Id#, InConvoy,Weight)in database DB2

A Virtual Class U_CONVOY is defined as follows:

```
CREATE VCLASS U_CONVOY
      (Id#: INTEGER,AvgWeight: REAL)
AS SELECT DB2.CONVOY.Id#, AvgWeight, ShipSet
      FROM DB1.CONVOY
      WHERE Exists (SELECT AvgWeight = avg(Weight)
                           FROM DB2.SHIP
                           WHERE SHIP.InConvoy = CONVOY.Id#)
```

Schematic Discrepancies

Schematic Discrepancies

Data Value Attribute Conflict

When attributes in one database corresponds to the value of the attributes in another

Database DB1:
relation STOCKS(Date, StkCode, ClsPrice)

Database DB2:
relation STOCKS(Date, Stk1, Stk2, ...)

where Values(StkCode) = Stk1, Stk2, ...
and Values (Stk1, Stk2, ...) = (ClsPrice1, ClsPrice2, ...)

Schematic Discrepancies

Attribute Entity Conflict

When attributes in one database corresponds to entities in another

Database DB2:
relation STOCKS(Date, Stk1, Stk2)

Database DB3:
relations Stk1(Date, ClsPrice),
Stk2(Date, ClsPrice)

```
CREATE VCLASS U_STOCKS
    (Date: Char(7), StkCode: Char(4), ClsPrice: REAL)
AS SELECT Date, "Stk1", Stk1
    FROM STOCKS,
    SELECT Date, "Stk2", Stk2
    FROM STOCKS,
    SELECT Date, "Stk1", ClsPrice
    FROM Stk1,
    SELECT Date, "Stk2", ClsPrice
    FROM Stk2
```

Schematic Discrepancies

Data Value Entity Conflict

When entities in one database correspond to values of attributes in another

Database DB1:
relation STOCKS(Date, StkCode, ClsPrice)
where values(StkCode) = Stk1, Stk2, ...

Database DB2:
relations Stk1(Date, ClsPrice)
Stk2(Date, ClsPrice) .....