# Mapping Issues

**From Ternary Relationship to Relational Tables: A Case Against Common Beliefs**

Le Thi Minh Chau

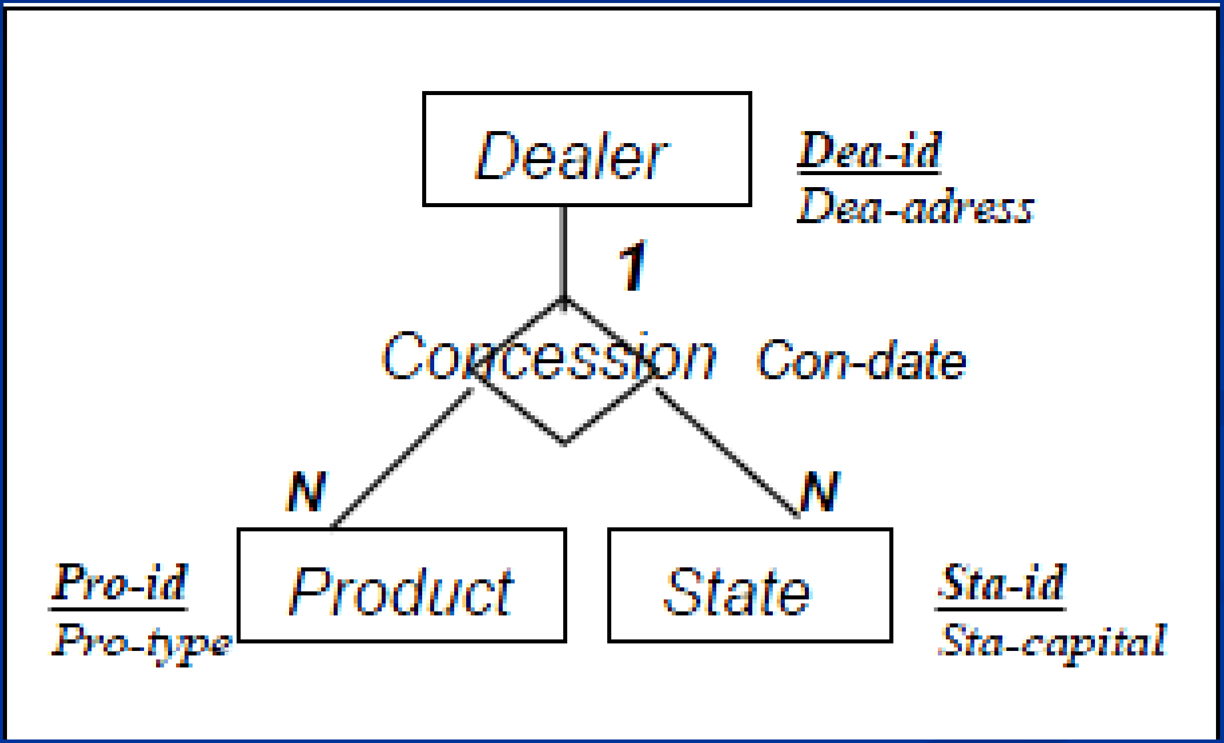
## Content

- Introduction
- Example
- Solution

## Introduction

- Transformations to relational tables from some ternary patterns are ignored in the literature.
- Analyzing the transformation from one of the ternary cases to relational table.

# Example


Dealer
Dea-id
Dea-adress
1
Concession
Con-date
N
N
Pro-id
Pro-type
Product
State
Sta-id
Sta-capital

## Example (cont)

- The semantics of the case:
  - A dealer company can distribute several products.
  - A product can be distributed by several dealers, but in each state a product can not be distributed by more than one dealer.
  - Each concession of distribution to a dealer of a product into a state.
  - Each instance of the ternary relationship has a date of concession (Con-date).

## Example (cont)

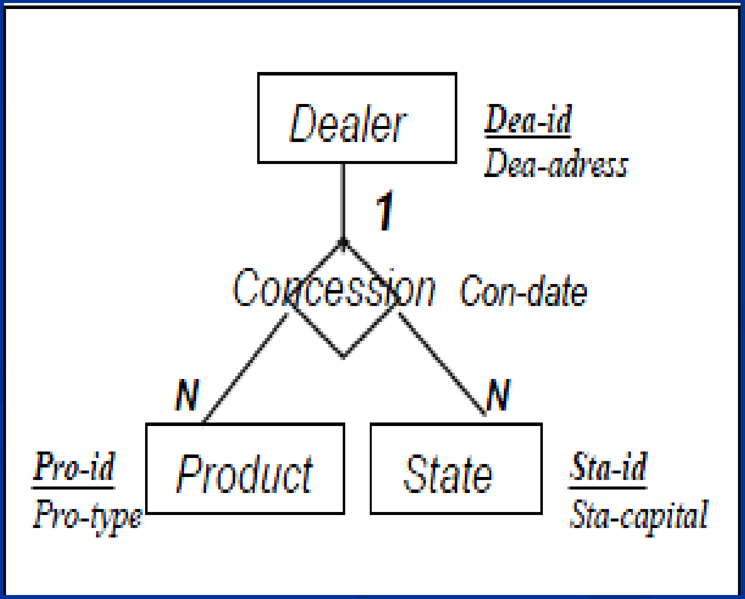
- The meaning of the cardinalities:
  - Each pair of one dealer and one state, can be associated to many products.
  - Each pair of one state and one product can not be associated to more than one dealer.
  - Each pair of product dealer can be associated to more than one state.

## Example (cont)

- Functional dependencies:
  - (Pro-id, Sta-id) → Dea-id
  - (Pro-id, Sta-id, Dea-id) → Con-date

↓

  - (Pro-id, Sta-id) → (Dea-id, Con-date)

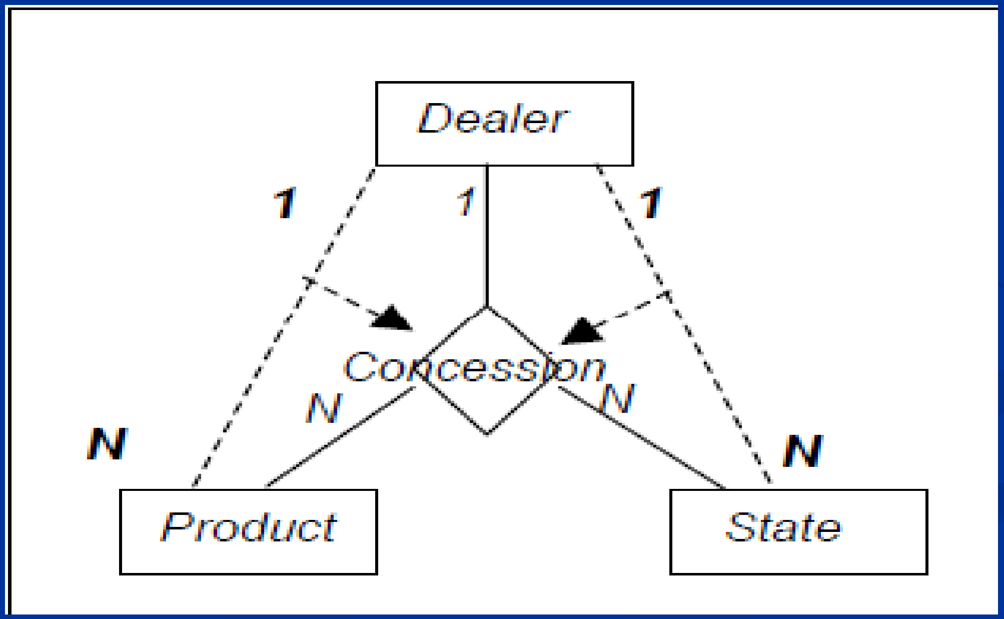


## Example (cont)

- Mapping
  - State (**Sta-id**, Sta-capital)
  - Product (**Pro-id**, Pro-type)
  - Dealer (**Dea-id**, Dea-address)
  - Concession (**Pro-id, Sta-id**, **Dea-id**, Con-date)

## Example (cont)

- Adding more FDs to the example
  - Each product is distributed only by one dealer.
  - In each state where it exists concessions, only one dealer exists.

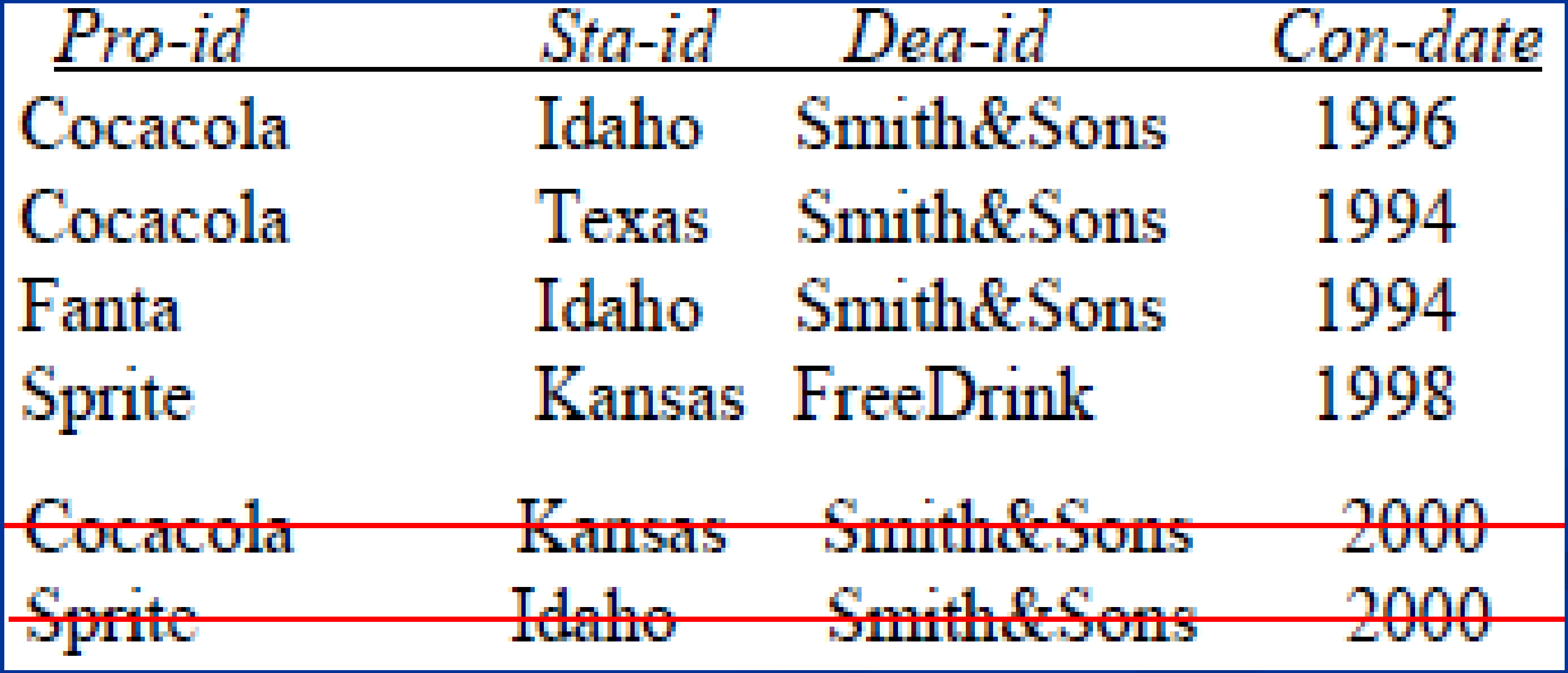


## Example (cont)

- Note
  - A product can still be distributed in several state.
  - In a state is still possible to distribute several products.

## Example (cont)

- A valid set of instances for the new relationship Concession



| *Pro-id* | *Sta-id* | *Dea-id* | *Con-date* |
|---|---|---|---|
| Cocacola | Idaho | Smith&Sons | 1996 |
| Cocacola | Texas | Smith&Sons | 1994 |
| Fanta | Idaho | Smith&Sons | 1994 |
| Sprite | Kansas | FreeDrink | 1998 |
| ~~Cocacola~~ | ~~Kansas~~ | ~~Smith&Sons~~ | ~~2000~~ |
| ~~Sprite~~ | ~~Idaho~~ | ~~Smith&Sons~~ | ~~2000~~ |

## Example (cont)

- Reason:
  - Cocacola can not have a dealer in Kansas because Kansas has a different dealer – FreeDrink.
  - Sprite can not be distributed by  Smith&Sons because FreeDrink has already distributed Sprite.

## Example (cont)

- FDs
  - (Pro-id, Sta-id) → Dea-id
  - **Pro-id → Dea-id**
  - **Sta-id → Dea-id**
  - (Pro-id, Sta-id) → Con-date
  - Pro-id →Pro-type
  - Sta-id → Sta-capital
  - Dea-id → Dea-address

## Example (cont)

- New schema:
  - State (**Sta-id**, **Dea-id,** Sta-capital)
  - Product (**Pro-id**, **Dea-id,** Pro-type)
  - Dealer (**Dea-id**, Dea-address)
  - Concession (**Pro-id, Sta-id**, **Dea-id**, Con-date)

# Example (cont)

- Issues
  - Concession (Pro-id, Sta-id, Dea-id, Con-date) is not in 2NF
    - Pro-id → Dea-id but Pro-id is not a key.
    - Sta-id → Dea-id but Sta-id is not a key.
  - Schema does not fully represent the semantics of ternary relationship
    - Pro-id → Dea-id, Sta-id → Dea-id should be defined as part of the ternary but in schema they are not (they are independent of Concession).

## Solution 1

- To express the full semantic, we could add the following inclusion dependencies:

$$\prod_{Sta-id,Dea-id} Concession \subseteq \prod_{Sta-id,Dea-id} State \quad (1)$$

$$\prod_{Sta-id,Dea-id} Concession \supseteq \prod_{Sta-id,Dea-id} State \quad (2)$$

$$\prod_{Pro-id,Dea-id} Concession \subseteq \prod_{Pro-id,Dea-id} Product \quad (3)$$

$$\prod_{Pro-id,Dea-id} Concession \supseteq \prod_{Pro-id,Dea-id} Product \quad (4)$$

## Solution 1 (cont)

- (1) and (3) can be easily expressed by adding keys and foreign keys:
  - Table Product: **UNIQUE(Pro-id, Dea-id)**
  - Table State: **UNIQUE(Sta-id, Dea-id)**
  - Table Concession:

**FOREIGN KEY (Pro-id, Dea-id) REFERENCES Product (Product-id, Dea-id)**

**FOREIGN KEY (State-id, Dea-id) REFERENCES State (State-id, Dea-id)**

## Solution 1 (cont)

$$\prod_{Sta-id, Dea-id} Concession \subseteq \prod_{Sta-id, Dea-id} State \quad (1)$$

$$\prod_{Sta-id, Dea-id} Concession \supseteq \prod_{Sta-id, Dea-id} State \quad (2)$$

$$\prod_{\text{Pro}-id, Dea-id} Concession \subseteq \prod_{\text{Pro}-id, Dea-id} \text{Product} \quad (3)$$

$$\prod_{\text{Pro}-id, Dea-id} Concession \supseteq \prod_{\text{Pro}-id, Dea-id} \text{Product} \quad (4)$$

## Solution 1 (cont)

(2) and (4) are little more complex:

- Table State:
  - **CHECK ((Sta-id,Dea-id) MATCH (SELECT Sta-id,Dea-id FROM Concession))**
- Table Product:
  - **CHECK ((Pro-id,Dea-id) MATCH (SELECT Pro-id,Dea-id FROM Concession))**

## Solution 2

- In order to normalize **Concession**, we can take out attribute Dea-id because the additional FDs represented in tables Product and State already imply the dependence (Pro-id, Sta-id) → Dea-id:
  - State (**<u>Sta-id</u>**, **Dea-id,** Sta-capital)
  - Product (**<u>Pro-id</u>**, **Dea-id,** Pro-type)
  - Dealer (**<u>Dea-id</u>**, Dea-address)
  - Concession (**<u>Pro-id</u>**, **<u>Sta-id</u>**, Con-date)

## Solution 2 (cont)

- The schema is normalized in 3NF (and also in BCNF) without lost of data and preserving all the FDs.
- Issues:
  - Schema is not representing the inclusion of the binary restrictions in the ternary relationship.
- Add restriction to enforce that Concession is not independent of the two binary FDs (Pro-id → Dea-id, Sta-id → Dea-id)

# Solution 2 (cont)

- State (**Sta-id**, **Dea-id,** Sta-capital)
- Product (**Pro-id**, **Dea-id,** Pro-type)
- Dealer (**Dea-id**, Dea-address)
- Concession (**Pro-id, Sta-id**, Con-date)

$$(a)\ \prod_{Pro-id,Sta-id} Concession \subseteq \prod_{Pro-id,Sta-id} Product * State$$

$$(b)\ \prod_{Pro-id} Product \subseteq \prod_{Pro-id} Concession \quad if\ Product.Dea-id\ is\ not\ null$$

$$\prod_{Sta-id} State \subseteq \prod_{Sta-id} Concession \quad if\ State.Dea-id\ is\ not\ null$$

## Solution 2 (cont)

- (a)
  - **CHECK ((Pro-id,Sta-id) MATCH (SELECT Pro-id,Sta-id FROM (Product NATURAL JOIN State)**
- (b)
  - **CHECK (NOT EXISTS (SELECT Pro-id FROM Product WHERE Dea-id IS NOT NULL) EXCEPT (SELECT Pro-id FROM Concession))**

    **AND NOT EXISTS (SELECT Sta-id FROM State WHERE Dea-id IS NOT NULL) EXCEPT (SELECT Sta-id FROM Concession)))**

# Solution 3

- Take out Dea-id from Product (or State symmetrically)
- New schema
  - Product (**Pro-id**, Pro-type)
  - State (**Sta-id**, Dea-id, Sta-capital)
  - Dealer (**Dea-id**, Dea-address)
  - Concession (**Pro-id**, **Sta-id**, Con-date)

# Solution 3 (cont)

- Product (**Pro-id**, Pro-type)
- State (**Sta-id**, Dea-id, Sta-capital)
- Dealer (**Dea-id**, Dea-address)
- Concession (**Pro-id**, **Sta-id**, Con-date)

- FD (Pro-id $\rightarrow$Dea-id) is not enforced
  - **CHECK (UNIQUE (SELECT Pro-id FROM (SELECT DISTINCT Pro-id,Dea-id FROM (Concession NATURAL JOIN State))))**

## Solution 3 (cont)

- Issuses: Concession is ternary, the instances of State associated to instances of Product must be exactly the same ones that are associated to instances of Dealer. That can be enforced by an SQL check for equality (remember that State-id from table Concession is already defined as foreign key of State)
  - **CHECK ((SELECT COUNT(*) FROM State WHERE Dea-id IS NOT NULL) = (SELECT COUNT(DISTINCT Sta-id) FROM Concession WHERE Sta-id NOT IN (SELECT Sta-id FROM State WHERE Dea-id IS NULL)))**

## Solution 4

- We can also leave the schema as original
  - State (Sta-id, Sta-capital)
  - Product (Pro-id, Pro-type)
  - Dealer (Dea-id, Dea-address)
  - Concession (Pro-id, Sta-id, Dea-id, Con-date)

## Solution 4 (cont)

- Issues: there is no intertable redundancy but table Concession is not in 3NF. We must enforce Pro-id → Dea-id and Sta-id → Dea-id.
  - **CHECK ((UNIQUE (SELECT Pro-id FROM (SELECT DISTINCT Pro-id,Dea-id FROM Concession))) AND (UNIQUE (SELECT State-id FROM (SELECT DISTINCT State-id, Dea-id FROM Concession))))**

## References

- Rafael Camps Paré, “From Ternary Relationship to Relational Tables: A Case Against Common Beliefs “
- R.Elmasri and S.B.Navathe, “Fundamentals of Database Systems”
- R.Camps, "Transforming N-ary Relationships to Database Schemas: An Old and Forgotten Problem"

## Thank you