CSE2DBF – CSE4DBF

Normalization

Reading:
Elmasri and Navathe, "Fundamentals of Database Systems, Chapters 1 & 2", Pearson, 2016.
Ebook: *https://ebookcentral-proquest-com.ez.library.latrobe.edu.au/lib/latrobe/detail.action?docID=5573709*

Relational Database Design

Two Approaches in Relational Database Design

- From Data Modeling (eg. ER Model) to Relational Logical Model for implementation.

TOP DOWN DESIGN

- Normalization of Relations

BOTTOM UP DESIGN

Today's lecture looks at **bottom-up design.**

Normalization

- Normalization Of ‘User View’ Relations (Bottom-Up Design)

- The Steps
  - Represent all user views (e.g. forms, reports etc.) as a collection of relations.
  - Normalize these relations, user view by user view.
  - Combine relations that have exactly the same primary key/s.

Normalization

Functional Dependencies

A relational table is one way of describing the way in which several attributes interact, or depend on each other. The simplest kind of dependency is called *functional dependency* (FD).

LHS ⟶ **RHS**

For example, *LecturerID* → *LecturerName*

is a valid FD because:

For each LecturerID there is at most one LecturerName, or

LecturerName is determined by LecturerID , or

LecturerName is uniquely determined by LecturerID , or

LecturerName depends on LecturerID .

Each of the above statements is equivalent.

Normalization

The FD **X** → **Y** is a ***full dependency*** if no attribute can be removed from X.

LabDate, SubjectCode → Tutor is full, that is, Tutor is fully dependent on both LabDate and SubjectCode.

The FD **X** → **Y** is a ***partial dependency*** if an attribute can be removed from X.

LecturerID, SubjectCode → LecturerName is partial, that is, LecturerName is partially dependent on LecturerID and SubjectCode.

Normalization

Dependencies can be ***transitive***.

For example, if one lecturer can teach one subject and each subject only has one tutor, then we might have the dependencies

LecturerID $\rightarrow$ SubjectCode

SubjectCode $\rightarrow$ Tutor

and, transitively LecturerID $\rightarrow$ Tutor.

Normalization

Functional dependencies can be used to decide whether a schema is well designed.

For example, in the following relation:

LecturerSubject (LecturerID, LecturerName, SubjectCode, SubjectName, Term)

Anomalies?:

- If there is a new subject which has not been allocated a lecturer, can you record the details of this subject in the above table? (Insert Anomaly)
- If an existing subject changes the name, can you do the changes to one instance only? (Update Anomaly)
- If a lecturer resigns and the details are to be deleted, would there be a chance that some subjects will be removed permanently and we won't have any track record of those subjects anymore? (Delete Anomaly)

Normalization

- Design errors in relations, such as the potential for certain kinds of ***anomalies***, can be categorised.
- These categories of error can be successively eliminated by decomposing relations into ***normal forms***.

- The major/main normal forms are first (**1NF**), second (**2NF**), third (**3NF**), and BoyceCodd (**BCNF**). Higher/advanced normal forms including fourth (**4NF**), and fifth (**5NF**).

- These forms are increasingly strict, that is, increasingly error-free. Advanced normal forms are based on complex kinds of dependency.

Because problems with 4NF and 5NF rarely occur, moreover database designers in industry normally do not need to use the highest possible NF for practical reasons, in this subject we will focus on satisfying 3NF and BCNF level.

Normalization

NORMAL FORMS

Unnormalized Form(UNF)

- First Normal Form(1NF)

A relation is in 1NF if :

- there are no repeating groups.
- a unique key has been identified for each relation.
- all attributes are functionally dependent on all or part of the key.

- Second Normal Form (2NF)

A relation is in 2NF if :

- the relation is in 1 NF
- all non-key attributes are fully functionally dependent on the entire key (partial dependency has been removed).

Normalization

NORMAL FORMS

- Third Normal Form(3NF)

A relation is in 3NF if :

- the relation is in 2NF
- all transitive dependencies have been removed.
  Transitive dependency: non-key attribute dependent on another non-key attribute.

- Boyce-Codd Normal Form (BCNF)

A relation is in BCNF if :

- the relation is in 3NF
- any remaining anomalies that result from functional dependencies have been removed.

Normalization

Example:

Transform the ORDER form below into BCNF relations.

ORDER FORM

Order # 5258
Customer # 32
Customer Name Computer Training Centre
Customer Address 1 Plenty Rd
City-State-Postcode Bundoora Victoria 3086
Order Date 27 March 2013

Product #	Description	Quantity	Unit Price
P123	Bookcase	4	200
P234	Cabinet	2	500
P345	Table	1	150

Normalization

Example Solution:

From the ORDER FORM (user view) we can derive ORDER relation:

- Currently in UNF (Un-normalized Form)

  ORDER
  (Order #, Customer #, Customer Name, Customer Address, City State PostCode, Order Date, (Product #, Description, Quantity, Unit Price))

  the order form is not in1NF because there is a repeating group (Product#, Description….).

To convert the above relation into 1NF, **the repeating group must be removed** by **creating a new relation** based on the repeating group along with the primary key of the main relation.

Normalization

Example Solution (cont.):

- **1NF:**

  ORDER
  (Order#, Customer#, Customer Name, Customer Address,
  CityStatePostCode, OrderDate)
  ORDER_PRODUCT
  (*Order#*, Product#, Description, Quantity, Unit Price)

  Anomalies:

  - **Insertion Anomalies**: cannot insert a new product until there is an order for that product.
  - **Deletion Anomalies**: if an order is deleted the whole detail of the product will also be deleted.
  - **Update Anomalies**: if the detail of a particular product needs to be updated, each order that contains that product has to be updated.

Normalization

Example Solution (cont.):

- **2NF:**

  The ORDER_PRODUCT relation is not in 2NF because **not all non-key attributes are fully dependent on the entire key**.

  To convert the ORDER_PRODUCT relation into 2NF, **a new relation must be created** which consists of part of the keys (becomes the primary key of the new relation) and all non key attributes that are dependent on the partial key.

  ORDER_PRODUCT
  (*Order#*, *Product#*, Quantity)

  PRODUCT
  (Product#, Description, Unit_Price)

  The ORDER relation is already in 2NF as there are no non key attributes that are dependent on partial key (ORDER only has a single key).

Normalization

Example Solution (cont.):

- **2NF:**

  Anomalies:

  - **Insert Anomalies**: a new customer cannot be inserted until he/she has an order.
  - **Delete Anomalies**: if an order is deleted, the whole information of the customer is also deleted.
  - **Update Anomalies**: if a customer detail is to be updated, all orders for that customer need to be updated.

Normalization

Example Solution (cont.):

- **3NF:**

The ORDER relation is not in 3NF because there is a transitive dependency (**non-key attribute dependent on another non-key attribute**).

To convert the relation into 3NF, **a new relation must be created** for the non-key attributes that are dependent to another non-key attribute.

CUSTOMER
(Customer#, Customer Name, Customer Address, CityStatePostCode)

ORDER
(Order#, *Customer#*, Order Date)

Both the order ORDER_PRODUCT and the PRODUCT relations are already in 3NF.

ORDER_PRODUCT(*Order#*, *Product#*, Quantity)
PRODUCT (Product#, Description, Unit Price)

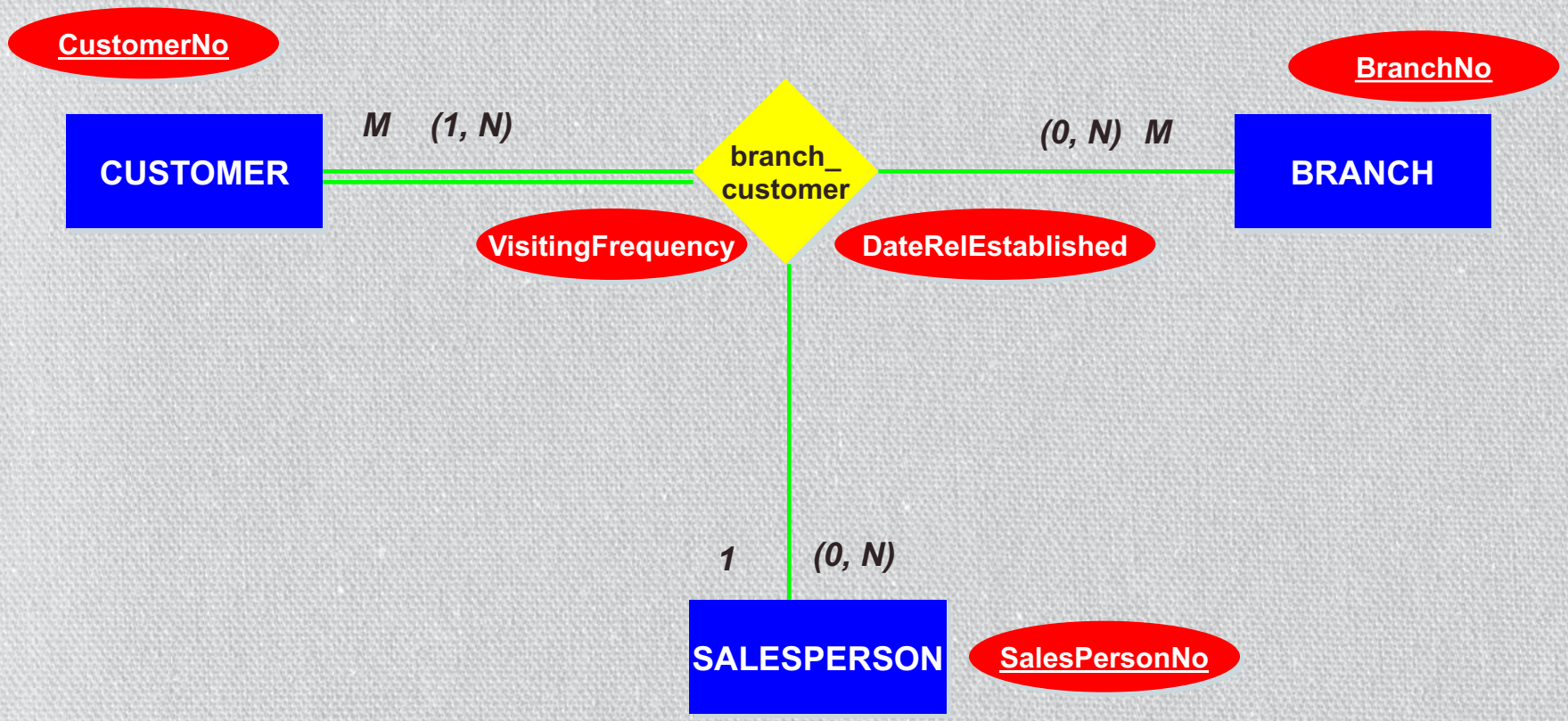
Normalization

Example Solution: Final Relations in 3NF and BCNF

ORDER
(Order#, *Customer#*, OrderDate)
CUSTOMER
(Customer#, CustomerName, CustomerAddress, CityStatePostCode)
ORDER_PRODUCT
(*Order#*, *Product#*, Quantity)
PRODUCT
(Product#, Description, UnitPrice)

- Note that only in rare situations that a relation in 3NF is not in 4NF or 5NF. Furthermore, 4NF and 5NF are harder to identify. Hence current practice of database design often disregard them. Also, the higher the level of normal form, the more tables (more splitting) need to be generated. Most relations that are in 3NF are also in BCNF.
- **The above 3NF relations are all in BCNF already.**
- In the following slides, we will go through some examples where 3NF relations may not be in BCNF.

BCNF (Boyce Codd Normal Form) – an example



(**Source:** Graeme Simsion and Graham Witt, Data Modeling Essentials - Analysis, Design and Innovation, International Thomson Publishing, 1996, Second Edition (revised and updated by Witt and Simsion) published in 2000 by Coriolis.

Boyce Codd Normal Form

The table is in 3NF (there are no repeating groups, no partial and transitive dependencies).

BRANCH-CUSTOMER
(*CustomerNo, BranchNo*, *SalespersonNo*, VisitingFrequency, DateRelationshipEstablished)

1. The table enforces the rule that each branch will serve a customer through only one salesperson, as there is only one SalespersonNo for each combination of CustomerNo and BranchNo.
2. If each salesperson works for one branch only, the table still has some normalization problems. The fact that a particular salesperson belongs to a particular branch can appear in more than one row. In fact, it will appear in every row for that salesperson.

The underlying reason for the normalization problems is that there is a dependency between SalespersonNo and BranchNo (SalespersonNo is a ***determinant*** of BranchNo). In another word, **there is a non-key that determines partial of the keys (violating BCNF rule)**.

Boyce Codd Normal Form

Problem :

BRANCH-CUSTOMER
(*CustomerNo, BranchNo, SalespersonNo*, VisitingFrequency,
DateRelationshipEstablished)

Solution (?) :

CUSTOMER-SALESPERSON (*CustomerNo, SalesPersonNo*, VisitingFrequency,
DateRelationshipEstablished, *BranchNo*)

This table is not even in 2NF

Solution:

CUSTOMER-SALESPERSON (*CustomerNo, SalespersonNo*, VisitingFrequency,
DateRelationshipEstablished)
SALESPERSON (SalespersonNo, *BranchNo*)

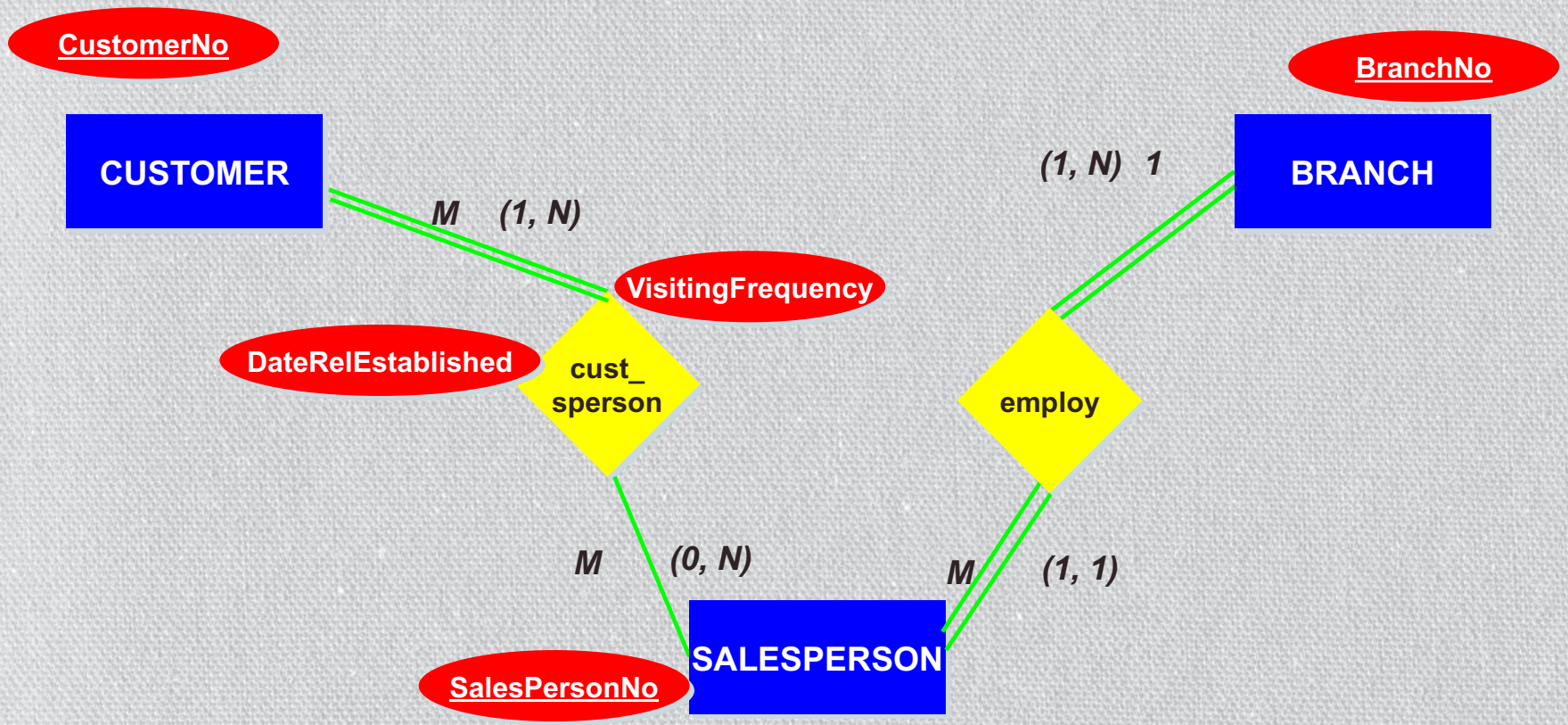
Boyce Codd Normal Form

BCNF

- Generic Format:
  - 3NF but not BCNF: **R1 (A, B, C)** where C may determine B
  - Converted to BNCF: **R11 (A, C)** and **R12 (C, B)**

- Implications for the previous example:
  - In the original table we enforced the rule that a given customer was only served by one salesperson from each branch.
  - Our new model no longer enforces that rule. It is now possible for a customer to be supported by several salespersons from the same branch.

Boyce Codd Normal Form



Boyce Codd Normal Form

NOTE:

- As an exercise, study the example of 3NF relations that are NOT in BCNF in the Elmasri text book Chapter 14 pages 536 – 538 (Figure 14.13, and Figure 14.14).

Normalization Exercise

Normalization of a single form:

- Quality Car Imports maintains a service log for all of its customers' vehicles. This log can be represented by the following form.
- Task: derive the relational tables for the form.

QUALITY CAR IMPORTS					
Customer Number:					
Customer Name:					
Customer Address:					
Customer Town:			Customer Postcode:		
Vehicle Registration Number:					
Vehicle Engine Number:					
Vehicle Description:					
Vehicle Colour:					
Date Purchased:					
Service Date	Mechanic Number	Mechanic Name	Description of Work Carried Out	Speedo Reading	Service Cost

Normalization Exercise

Solutions to be discussed during lecture.

Normalization Exercise

Normalization of multiple reports:

- Derive the relational tables for the following 3 reports (subject, lecturer and student)

SUBJECT REPORT

SUBJECTS CURRENTLY APPROVED

Subject Code	Subject Name	Subject Description	Subject Credit
CS830	Introduction to Databases	Database Technology	15
CS577	Object-Oriented Programming	C++ Programming	15
CS670	Computer Programming for Technologist	C Programming	10
CS825	Software Engineering Analysis & Design	Analysis & Design	10

Normalization Exercise

LECTURER REPORT

LECTURER DETAILS

LECTURER'S NUMBER	AS200
LECTURER'S NAME	GIUSEPPE BLOGGS
LECTURER'S OFFICE No.	Bldg 63 Room 130
LECTURER'S PHONE No.	52246

LECTURING:

Subject Code	Subject Name
CS830	Introduction to Databases
CS825	Software Engineering Analysis & Design

Note: A given subject may have several lecturers.

Normalization Exercise

STUDENT REPORT

STUDENT DETAILS

STUDENT NO.	S1234567
STUDENT NAME	Poindexter Jones
STUDENT ADDRESS	23 Wide Road, Kew, 3101
COURSE ENROLLED	BSc
MODE OF STUDY	Internal
LECTURER NUMBER	AS200
LECTURER NAME	Guiseppe Bloggs

ACADEMIC RECORD:

Subject Code	Subject Name	Year/Semester	Grade
CS830	Introduction to Databases	2013/1	A
CS891	Computing Fundamentals	2013/1	B

Normalization Exercise

Solutions to be discussed during lecture.

Next Lecture

Data Manipulation using Relational Algebra

Reading:

Elmasri and Navathe, "Fundamentals of Database Systems, Chapters 1 & 2", Pearson, 2016.

Ebook: *https://ebookcentral-proquest-com.ez.library.latrobe.edu.au/lib/latrobe/detail.action?docID=5573709*