UTS:ENGINEERING AND INFORMATION TECHNOLOGY

UTS UNIVERSITY OF TECHNOLOGY, SYDNEY

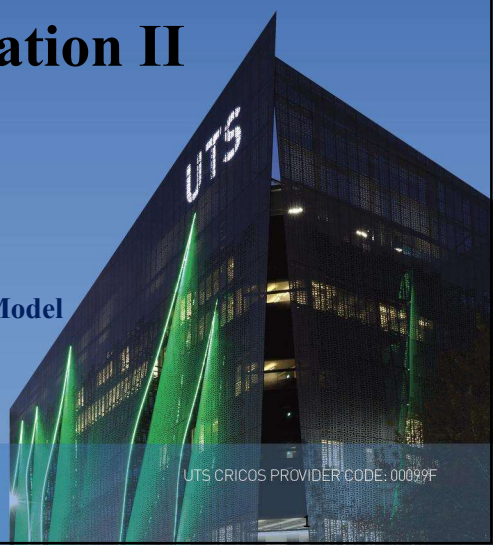
lecture 6: Normalization II

Modern Database Management

11th Edition, International Edition

Chapter 4: Logical Database Design and the Relational Model

Jeffrey A. Hoffer, V. Ramesh, Heikki Topi



Innovation in practice

eng.uts.edu.au • it.uts.edu.au

1

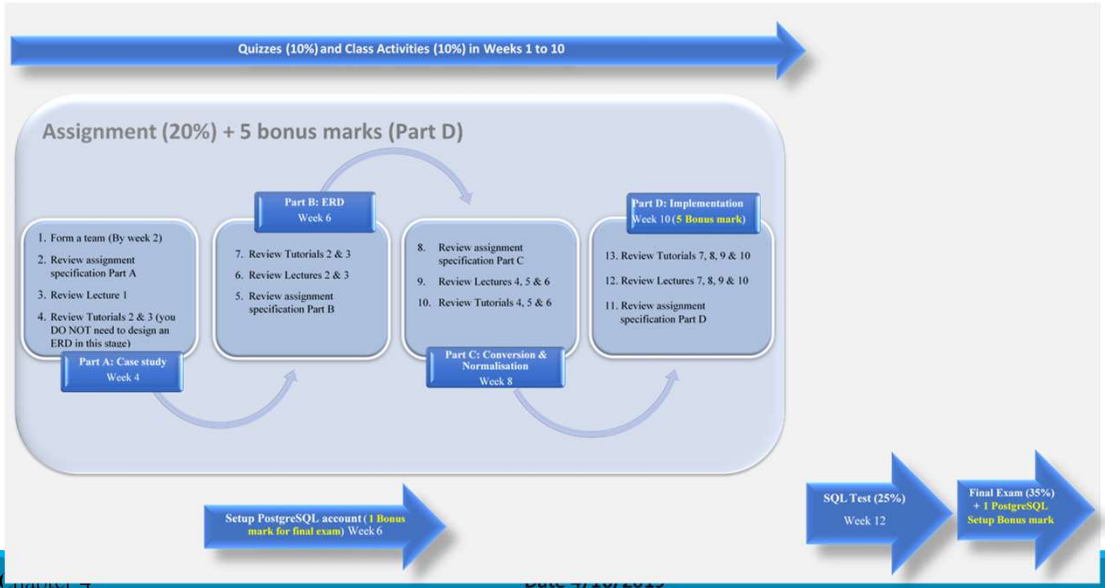
Participations and Discussions

If you have any question and you don't want to share it now, send it to us via UTSOnline/Discussion Board.

However, it is better to speak out ☺

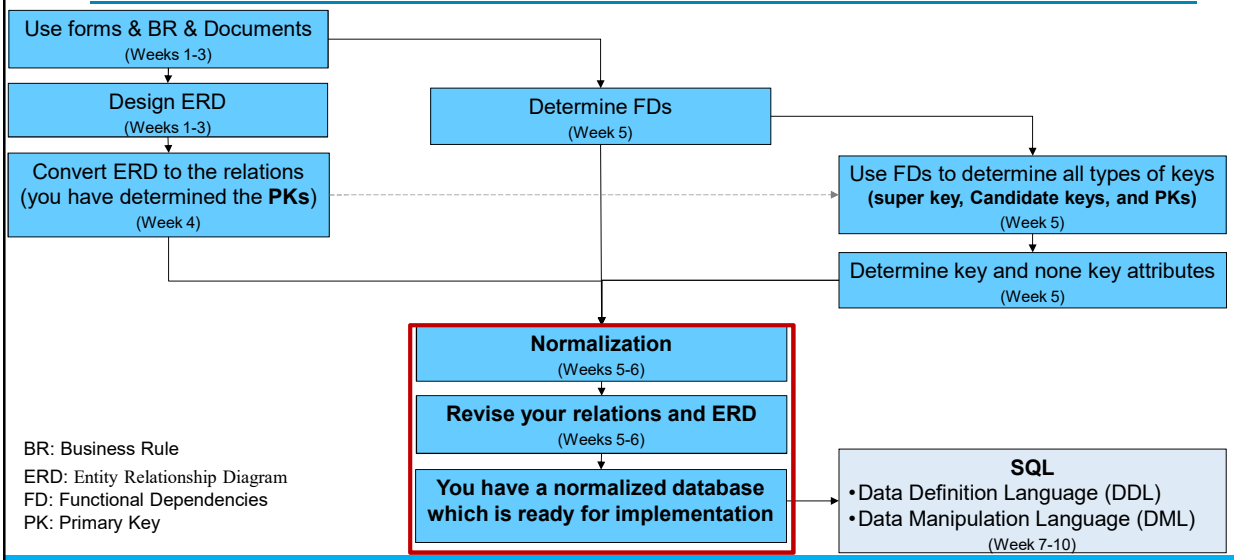
2

Assessment Chart and Knowledge Guideline



3

Subject Flowchart



BR: Business Rule
ERD: Entity Relationship Diagram
FD: Functional Dependencies
PK: Primary Key

4

DF Learning Plan

Description: we will have collaborative lecture at the beginning of the class. You need to do some tasks during the lecture as part of your class activities. Then you will do a quiz of what you have learned, then the tutorial will start. you will work in groups during the class.

Please be aware that the lecture slides with Blue title are designed for your self study.

Workshop Timetable:

Activity	Duration	Comments
Lecture	1 hour and 20 minutes	15 minutes lecture followed by 1 hour working on section one of the **Online Camera Shop Case Study**
Rest	10 minutes	Have fun
Review	10 minutes	Please ask your questions if you have any.
Tutorial	50 Minutes	Have even more fun :D
Quiz (Open Book)	5 minutes	On today's content. Will be run before or after the tutorial. Do your best ;)
Leave the class	5 minutes	Don't forget to review what you have learn in this class, and check the information that is provided on **UTSOnline/Learning Material/Week 6**

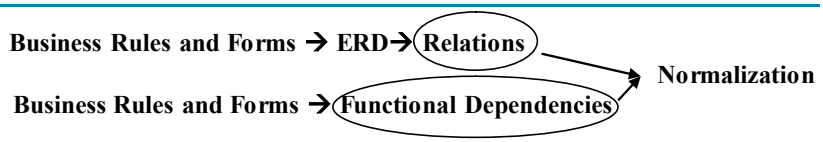
Subject Overview

- **Design Entity Relationship Diagram (ERD)**
  - Week 1: Data Modelling I (Conceptual Level)
  - Week 2: Data Modelling II (Conceptual Level)
  - Week 3: Data Modelling III (Conceptual Level)
  - Week 4: Convert ERD to Relations (Logical Level)
  - Week 5: Functional Dependencies
  - Week 5: Normalization I
  - **Week 6: Normalization II**
- **Data manipulation**
  - **Week 7: Simple Query**
  - **Week 8: Multiple Table Queries**
  - **Week 9: Subquery**
  - **Week 10: Correlated Subquery**

Objectives

1. Review of 1NF, 2NF and 3NF.
2. Boyce Codd Normal Form (BCNF) → Optional
   - 1.1. BCNF Example 1
   - 1.2. BCNF Example 2
3. Creating New Relations in a Higher Normal Form → Optional
4. Role of Normalization → Optional
5. Advantages of Refinement (Top-Down) Approach → Optional
6. Class Activity 6 – Section one

Top-Down Normalization Process:



NOTE: Please DO NOT use your ERD or Relations to determine FDs.

1. Review: First Normal Form

I. **No derived attribute** (Derived attribute can be calculated or derived using some business rule from other attributes)

- **Example:** In the following relation StuAge is a derived attribute and should be removed from the relation

- Student(StudentID, StuDateOfBirth, **StuAge**, StuAddress)
- Student(StudentID, StuDateOfBirth, StuAddress)

II. **Every attribute value is atomic** (Atomic attributes can't be divided into subparts)

- **Example:** In the following relation StuAddress is a non-atomic attribute and should be divided to smaller parts

- Student(StudentID, StuDateOfBirth, **StuAddress**)
- Student(StudentID, StuDateOfBirth, **StuUnitNumber, StuStreet, StuSuburb, StuState**)

Note: in the following slides "Customer Address" has been ASSUMED as an atomic attribute.

III. **No multivalued attributes** (Multivalued attributes can have more than one value at a time)

Based on this, a relation is in first normal form if:

- There are no repeating groups in the relation.
- A Primary key has been defined, which uniquely identifies each row in the relation

- **Example:** In the next slides

1. Review-Second Normal Form

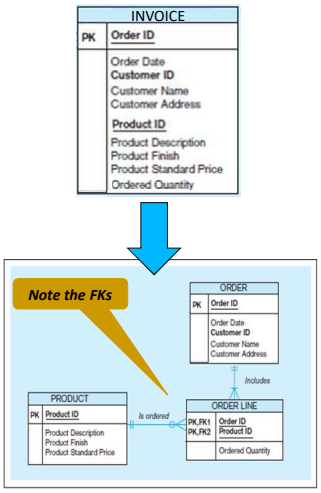
➢ **1NF PLUS every non-key attribute is fully functionally dependent on the ENTIRE primary key**

- Every non-key attribute must be defined by the **entire key**, not by only part of the key
- No partial functional dependencies

➢ **Solution:**

1. Create new relation for each Primary Key (PK) and move non-key attributes that are only dependent on this PK.
2. Consider this PK as a Foreign Key (FK) in the original table (relation)

Removing Partial Dependencies (Different Format for the Relations)



INVOICE (OrderID, ProductID, OrderDate, CustomerID, CustomerName, CustomerAddress, ProductDescription, ProductFinish, ProductStandardPrice, OrderQuantity)

OrderID, ProductID → OrderQuantity
ProductID → ProductDescription, ProductFinish, ProductStandardPrice
OrderID → OrderDate, CustomerID, CustomerName, CustomerAddress
CustomerID → CustomerName, CustomerAddress

- **New Relations:** Getting it into Second Normal Form

PRODUCT (ProductID, ProductDescription, ProductFinish, ProductStandardPrice)
ORDER (OrderID, OrderDate, CustomerID, CustomerName, CustomerAddress)
ORDER-LINE (OrderID*, ProductID*, OrderQuantity)

Note the FKs

Partial dependencies are removed, but there are still transitive dependencies in ORDER relation.

1- Review: Third Normal Form

- **2NF PLUS no transitive dependencies** (functional dependencies on non-primary-key attributes)
- Note: This is called transitive, because the primary key is a determinant for another attribute, which in turn is a determinant for a third

- **Solution:**
  1. Non-key determinant with transitive dependencies go into a new table;
  2. Non-key determinant becomes primary key in the new table, and
  3. Stays as foreign key in the old table (relation)

Removing Transitive Dependencies (Different Format for the Relations)

Original Relation

ORDER (OrderID, OrderDate, CustomerID, CustomerName, CustomerAddress)

- **FDs:**

OrderID → OrderDate, CustomerID, CustomerName, CustomerAddress

CustomerID → CustomerName, CustomerAddress

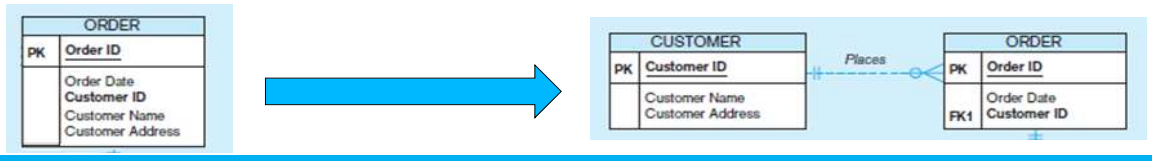
Getting it into third Normal Form

- **New Relations:**

ORDER (OrderID, OrderDate, CustomerID*)

CUSTOMER (CustomerID, CustomerName, CustomerAddress)

Note the FKs



Optional Content _Start

2. Boyce Codd Normal Form (BCNF)

- A table is already in 3NF.
- Every determinant must be a **candidate** key

- Simpler definition: A relation is not in BCNF if:
  - **Non-key → part of key**
  - **Part of key → Part of key**

- **Special case that are not covered by 3NF (not common) Often occurs when there is more than one composite candidate key and they overlap**

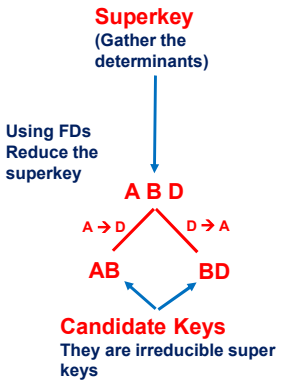
2.1. BCNF Example 1: UnivTable4

A	B	C	D
StdSSN	**OfferNo**	EnrGrade	Email
S1	O1	3.5	joe@bigu
S1	O2	3.6	joe@bigu
S2	O1	3.8	mary@bigu
S2	O3	3.5	mary@bigu

A → D
AB → C
D → A
DB → C

FD's:
StdSSN → Email
StdSSN, OfferNo → EnrGrade
Email → StdSSN
Email, OfferNo → EnrGrade

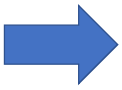
- another way to find the candidate keys (for your information)



Our Method to Determine Candidate Keys Using FDs

A B C D
UnivTable4 (StdSSN, OfferNo, EnrGrade, Email) OR UnivTable4 (ABCD)

FD's:
StdSSN → Email
StdSSN, OfferNo → EnrGrade
Email → StdSSN
Email, OfferNo → EnrGrade



Left	Middle	Right
OfferNo	**StdSSN, Email**	**EnrGrade**

(OfferNo)+ = {OfferNo}

(OfferNo, StdSSN)+ = {OfferNo, StdSSN, EnrGrade, Email} = R

(OfferNo, Email)+ = {OfferNo, Email, EnrGrade, StdSSN} = R

Therefore, (OfferNo, StdSSN) and (OfferNo, Email) are candidate keys.

2.1. BCNF Example 1

- **A relation is in BCNF if:**
  - A table is already in 3NF.
  - Every determinant must be a **candidate** key

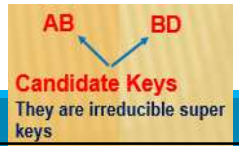
In the sample, we found two candidate keys: **(OfferNo, StdSSN)** and **(OfferNo, Email)**

Based on this, the following **FD's violate BCNF**

- **StdSSN → Email**
- **Email → StdSSN**

because they show that **StdSSN** and **Email** are determinants but **neither is a candidate key by itself** for UnivTable4, which is thus not in BCNF, so the relation should be decomposed

	FD's:
A → D	StdSSN → Email
AB → C	StdSSN, OfferNo → EnrGrade
D → A	Email → StdSSN
DB → C	Email, OfferNo → EnrGrade



2.1. BCNF Example 1: Decompose The Relation to BCNF

R1 UnivTable4 (StdSSN, OfferNo, EnrGrade, Email)

1- Start with one of the FDs that violates BCNF like:

StdSSN → Email

2- Then create a relation base on this FD:

UnivTable4-1(StdSSN, Email)

3- Then create another relation that contains the PK of UnivTable4 (R1) and the rest of the attributes in the original relation

UnivTable4-2 (StdSSN, OfferNo, EnrGrade)

These two relations cover all attributes in original relation, and can not be decomposed further based on the FDs.

	FD's:
A → D	StdSSN → Email
AB → C	StdSSN, OfferNo → EnrGrade
D → A	Email → StdSSN
DB → C	Email, OfferNo → EnrGrade

These are from the previous slide for reference

2.1. BCNF Example 1: Decompose The Relation to BCNF (cont.)

4- Make the determinant (LHS) of the chosen FD in step one (StdSSN → Email) a Foreign Key in the original relation (UnivTable4-2)

A B C

UnivTable4-2 (**StdSSN*, OfferNo**, EnrGrade) Note the FK

UnivTable4-1 (**StdSSN**, Email)

A D

5- Compute FDs for the new relations

UnivTable4-1(StdSSN, Email)

A → D **FD: StdSSN → Email**

UnivTable4-2 (StdSSN*, OfferNo, EnrGrade)

AB → C **FD: StdSSN, OfferNo → EnrGrade**

6- Compute keys for the new relations:

A is the key for **UnivTable4-1, AB** is the key for **UnivTable4-2**

	FD's:
A → D	StdSSN → Email
AB → C	StdSSN, OfferNo → EnrGrade
D → A	Email → StdSSN
DB → C	Email, OfferNo → EnrGrade

These are from the previous slide for reference

2.1. BCNF Example 1: Conclusion

Now we have two relations that are in BCNF i.e. they are in 3rd NF and the determinants of the related FDs are candidate keys.

A B C
UnivTable4-2 (**StdSSN*, OfferNo**, EnrGrade) Note the FK
UnivTable4-1 (**StdSSN**, Email)
A D

Determinant *"Email, OfferNo"* no longer appear in any relation, so it does not need to be considered when checking each one for BCNF.

	FD's:
A → D	StdSSN → Email
AB → C	StdSSN, OfferNo → EnrGrade
D → A	Email → StdSSN
DB → C	Email, OfferNo → EnrGrade

These are from the previous slide for reference

2.2. BCNF Example 2: Solution Boyce-Codd Normal Form (BCNF)

Original Relation

UnivTable4 (StdSSN, OfferNo, EnrGrade, Email)

	FD's:
A → D	StdSSN → Email
AB → C	StdSSN, OfferNo → EnrGrade
D → A	Email → StdSSN
DB → C	Email, OfferNo → EnrGrade

A	B	C	D
StdSSN	OfferNo	EnrGrade	Email
S1	O1	3.5	joe@bigu
S1	O2	3.6	joe@bigu
S2	O1	3.8	mary@bigu
S2	O3	3.5	mary@bigu

A B C
Grade: UnivTable4-2 (**StdSSN*, OfferNo**, EnrGrade) Note the FK
Student: UnivTable4-1 (**StdSSN**, Email)
A D

Grade

B	A	C
OfferNo	StdSSN	EnrGrade
O1	S1	3.5
O2	S1	3.6
O1	S2	3.8
O3	S2	3.5

Student

A	D
StdSSN	Email
S1	Joe@bigu
S2	Mary@bigu

2.2. BCNF Example 2: 3NF But With Redundancy

University Table (UnivTable5)

A	B	C	D
StdSSN	AdvisorNo	Major	Status
S1	A1	IS	COMPLETED
S1	A2	FIN	PENDING
S2	A1	IS	PENDING
S2	A3	FIN	COMPLETED

FDs

AC→ D	• StdSSN, Major → Status
B → C	• AdvisorNo → Major
BA → D	• AdvisorNo, StdSSN → Status
C → B	• Major → AdvisorNo

Left	Middle	Right
A	B, C	D

A+ ={A}
AB+ = {ABDC}
AC+ = {ACDB}

AB and AC are the candidate keys

AB and AC are the candidate keys
B → C
C → B

i.e. Part of key → Part of key

The following FDs violate BCNF:
B → C
C → B

2.2. BCNF Example 2: Decompose The Relation to BCNF

R2 UnivTable5 (StdSSN, AdvisorNo, Major, Status)

1- Start with one of the FDs that violates BCNF like:
AdvisorNo → Major

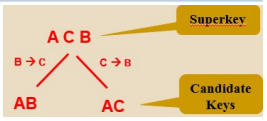
2- Then create a relation base on this FD:
UnivTable5-1(AdvisorNo, Major)

3- Then create another relation that contains the PK of UnivTable5 (R2) and the rest of the attributes in the original relation
UnivTable5-2 (StdSSN, AdvisorNo, Status)

These two relations cover all attributes in original relation, and can not be decomposed further based on the FDs.

AC→ D	StdSSN, Major → Status
B → C	AdvisorNo → Major
BA → D	AdvisorNo, StdSSN → Status
C → B	Major → AdvisorNo

These are from the previous slide for reference



2.2. BCNF Example 2: Decompose The Relation to BCNF (cont.)

4- Make the determinant (LHS) of the chosen FD in step on (AdvisorNo → Major) a Foreign Key in the original relation (UnivTable5-2)

A B D
UnivTable5-2 (**StdSSN, AdvisorNo***, Status)
UnivTable5-1 (**AdvisorNo**, Major)
B C

Note the FK

5- Compute FDs for the new relations

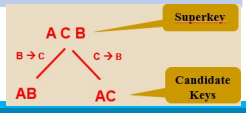
B → C **UnivTable5-1(AdvisorNo, Major)**
FD: AdvisorNo → Major

AB → D **UnivTable5-2 (StdSSN, AdvisorNo*, Status)**
FD: StdSSN, AdvisorNo → Status

6- Compute keys for the new relations:
B is the key for **UnivTable5-1**, AB is the key for **UnivTable5-2**

AC→ D	StdSSN, Major → Status
B → C	AdvisorNo → Major
BA → D	AdvisorNo, StdSSN → Status
C → B	Major → AdvisorNo

These are from the previous slide for reference



2.2. BCNF Example 2: Conclusion

Now we have two relations that are in BCNF i.e. they are in 3rd NF and the determinants of the related FDs are candidate keys.

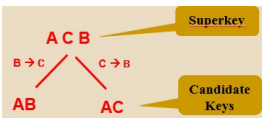
A B D
UnivTable5-2 (**StdSSN, AdvisorNo***, Status)
UnivTable5-1 (**AdvisorNo**, Major)
A C

Note the FK

Determinant *StdSSN, Major* no longer appear in any relation, so it does not need to be considered when checking each one for BCNF.

AC→ D	StdSSN, Major → Status
B → C	AdvisorNo → Major
BA → D	AdvisorNo, StdSSN → Status
C → B	Major → AdvisorNo

These are from the previous slide for reference



2.2. BCNF Example 2: Solution Boyce-Codd Normal Form (BCNF)

Original Relation

UnivTable5 (StdSSN, AdvisorNo, Major, Status)

AC → D	StdSSN, Major → Status
B → C	AdvisorNo → Major
BA → D	AdvisorNo, StdSSN → Status
C → B	Major → AdvisorNo

A	B	C	D
StdSSN	AdvisorNo	Major	Status
S1	A1	IS	COMPLETED
S1	A2	FIN	PENDING
S2	A1	IS	PENDING
S2	A3	FIN	COMPLETED

A B D

UnivTable5-2 (**StdSSN, AdvisorNo***, Status) Note the FK

UnivTable5-1 (**AdvisorNo**, Major)

A C

Advise

B	A	D
AdvisorNo	StdSSN	Status
A1	S1	COMPLETED
A2	S1	PENDING
A1	S2	PENDING
A3	S2	COMPLETED

Advisor

B	C
AdvisorNo	Major
A1	IS
A2	FIN
A3	FIN

3. Creating New Relations in a Higher Normal Form

When decomposing a relation to create new relations in a higher normal form you should:

1. Create a new relation for the FD group causing a violation
2. Create another new relation correspond to the original relation.
3. Copy the PK of the original relation to this new relation as its primary key
4. Move the dependencies (RHS attributes) of the chosen FD in step one **out of** the original relation. Then put the rest of the none key attributes in the original relation into the new relation (that is created is Step 2) as its non-key attributes
5. Make the determinant (LHS) of the chose FD in step 1 a Foreign Key in the relation that is created in Step 2 (remember a copy was left behind)
6. Compute FDs for the new relations
7. Compute keys for the new relations:
8. Repeat for each FD until highest normal form is achieved (the aim is 3NF or BCNF)

4. Role of Normalization

- **Refinement (Top Down Design)**
  - Use after ERD
  - Apply to table design or ERD
- **Initial design (Bottom Up Design)**
  - Record attributes and determine FDs
  - No initial ERD

5. Advantages of Refinement (Top-Down) Approach

- ✓ Easier to translate requirements into an ERD than list of FDs
- ✓ Fewer FDs to specify
- ✓ Fewer tables to split
- ✓ Easier to identify relationships especially M-N relationships without attributes

6. Summary

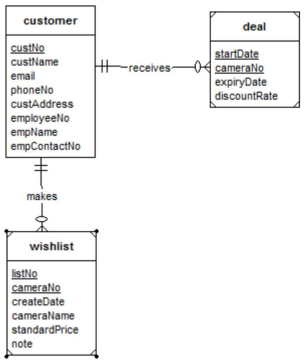
- Beware of unwanted redundancies
- FDs are important constraints
- Strive for 3NF or BCNF
- Focus on the normalization objective

Optional Content _End

7. Class Activities
Online Camera Shop Case Study

Online Camera Shop Case Study

The following ERD has been designed for an Online Digital Camera Shop. The database keeps records of customers and their wish lists for managing market campaign.

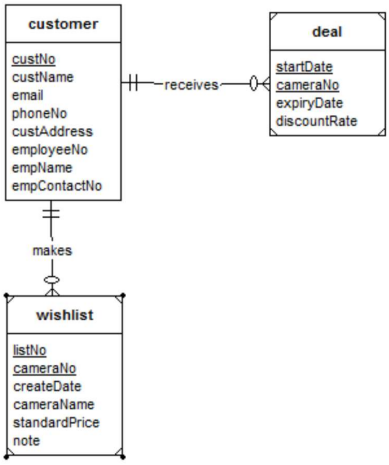


Class Activity 6.1: Determine the BRs of the case study.

- Every customer needs to **register** on the site by providing his/her personal information including name, home address, email address and contact phone number. The system will generate a **unique customer number** for each customer when his/her registration is successfully completed.
- Customers can **create their own private wish lists**, which contain their favorite cameras they wish to purchase. The system will automatically **generate a unique number for a wish list** of a customer in order to trace all the wish lists. The system also **stores** the date that the wish list of the customer has been created. Within a wish list, customer can **add as many cameras as he or she likes**. For each entry in a wish list, which is for **one and only one camera**, customer can make a **brief note**, for example, "this is my first choice". A camera is described by a unique camera number, a camera name and a standard price.
- A customer **may be allocated to an employee** of the shop. This employee will help them with all future purchases. An employee is described by a unique employee no, a name and a contact phone number. In order to **attract more sales**, the top manager will **regularly propose promotion deals for customers** to buy specific cameras. Each promotion deal is described by a start date, an expiry date and a discount rate for a specific camera model. **Some employees also act as supervisors for other employees.**

Class Activity 2: Convert "wishlist" entity to the relation.

Answer:



Class Activity 3: Determine FDs for the following business rules.

BR 4: The system will automatically generate a unique number for a wish list of a customer in order to trace all the wish lists.

BR 5: The system also stores the date that the wish list of the customer has been created.

Class Activity 3: Determine FDs for the following business rules.

BR 6: A Customer can add as many cameras as he or she likes to their wish list.

BR 7: For one and only one camera, customer can make a brief note. for example, “this is my first choice”.

BR 8: A camera is described by a unique camera number, a camera name and a standard price.

Class Activity 4: is wishList relation in 1NF, 2NF, 3NF and BCNF (optional)? If not normalize it.

wishlist (listNo, cameraNo, custNo* createDate, cameraName, standardPrice, note)

FK custNo references customer

FDs: custNo, ListNo→createDate

custNo, ListNo, camearNo → note

camearNo→cameraName, standardPrice

Is the relation in 1NF?

Class Activity 4: is wishList relation in 2NF, 3NF and BCNF (optional)? If not normalize it.

wishlist (listNo, cameraNo, custNo* createDate, cameraName, standardPrice, note)

FK custNo references customer

FDs: custNo, ListNo→createDate

custNo, ListNo, camearNo → note

camearNo→cameraName, standardPrice

Is the relation in 2NF?

Class Activity 4 (cont.): Is the wishlist relation in 2NF?

wishlist (listNo, cameraNo, custNo* createDate, cameraName, standardPrice, note)
FK custNo references customer

FDs: custNo, ListNo→createDate
custNo, ListNo, camearNo → note
camearNo→cameraName, standardPrice

Is the wishlist relation in 2NF? (cont.)

Class Activity 4 (cont.): Is the wishlist relation in 2NF? (cont.)

wishlist (listNo, cameraNo, custNo* createDate, cameraName, standardPrice, note)
FK custNo references customer

FDs: custNo, ListNo→createDate
custNo, ListNo, camearNo → note
camearNo→cameraName, standardPrice

43

Class Activity 4 (cont.): Is the wishlist relation in 2NF? (cont.)

wishlist (listNo, cameraNo, custNo* createDate, cameraName, standardPrice, note)
FK custNo references customer

FDs: custNo, ListNo→createDate
custNo, ListNo, camearNo → note
camearNo→cameraName, standardPrice

44

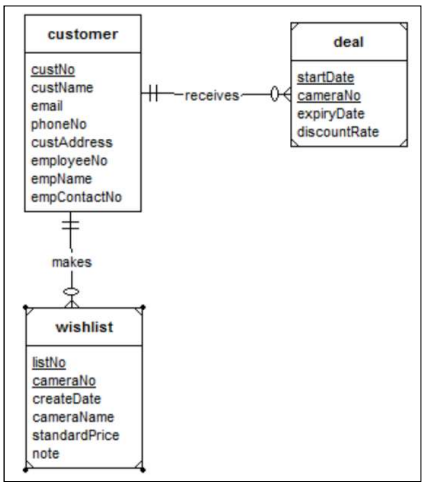
Class Activity 4 (cont.): Is the wishlist relation in 3NF?

Is the relation in 3NF?

The new created relations should be in 1NF, 2NF, 3NF and BCNF (optional) as well:

The new created relations should be in 1NF, 2NF, 3NF and BCNF (optional) as well:

d. Change the ERD based on the normalization results.



Business Rules

Class Activity 5: Change the ERD based on the normalization results.