Tutorial 10

Cao Sheng

1NF

1. Each attribute contains only one value.
2. All attribute values are atomic, which means they can't be broken down into anything smaller.

student	courses
Jane Smith	Databases, Mathematics
John Lipinsky	English Literature, Databases
Dave Beyer	English Literature, Mathematics

2NF

Definition: A relation schema R is in second normal form (2NF) if every **non-prime** attribute A in R is **fully functionally dependent** on every key of R

R = {<u>A, B</u>, C, D}

(A, B) is the key, so AB -> C (ok) , A -> D (not good)

Informally, the second normal form states that **all attributes must depend on the entire candidate key**.

All relations that have a single-attribute key are by definition in 2NF.

Bike parts warehouse

part	supplier	quantity	supplier country
Saddle	Bikeraft	10	USA
Brake lever	Tripebike	5	Italy
Top tube	UpBike	3	Canada
Saddle	Tripebike	8	Italy

part, supplier → *quantity*

supplier → *supplier country* **violates the 2NF**

3NF

A relation is in third normal form(3NF) if and only if:

All **non-prime** attributes are directly (non-**transitively**) dependent on the entire candidate key.

Another definition is:

A relation schema R is in third normal form (3NF) if whenever a FD $X \rightarrow A$ holds in R,then either:

(a) X is a superkey of R, or

(b) A is a prime attribute of R

order_id	date	customer	customer email
1/2020	2020-01-15	Jason White	white@example.com
2/2020	2020-01-16	Mary Smith	msmith@mailinator.com
3/3030	2020-01-17	Jacob Albertson	jasobal@example.com
4/2020	2020-01-18	Bob Dickinson	bob@fakemail.com

(order_id → date);

(order_id → customer),

(customer → customer email) violates the 3NF

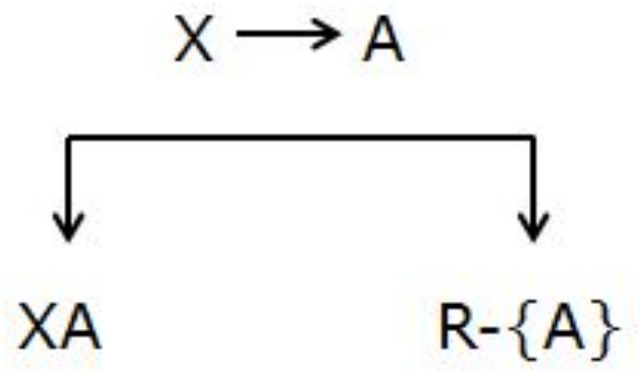
Normal Forms Defined Informally

- 1st normal form
  - All attributes depend on **the key**
- 2nd normal form
  - All attributes depend on **the whole key**
- 3rd normal form
  - All attributes depend on **nothing but the key**

BCNF

A relation schema R is in Boyce-Codd Normal Form (BCNF) if whenever an FD $X \rightarrow A$ holds in R, then

X is a superkey of R



Q1 SUPPLY (Supplier#, Part#, Date, Project, Quantity, Supp_name, Part_name)

Fd1: <u>Supplier#, Part#, Date </u>→ SUPPLY

Fd2: Supplier# → Supp_name

Fd3: Part# → Part_name.

decomposed into:

SUPPLY1 (Supplier#, Part#, Date, Project, Quantity, Part_name)

SUPPLIER (Supplier#, Supp_name)

A (i): Have we preserved the Fds?

Yes

In Supply 1: we have

Supplier#, Part#, Date -> Project, Quantity, Part_name
Part# -> Part_name

In SUPPLIER: we have

Supplier# -> Supp_name

Lossless Join

The join of two decomposed table should not generate spurious tuples

NJB (non-additive join test for binary decompositions):

- The f.d. ((R1 ∩ R2) -> (R1- R2)) is in F + , or
- The f.d. ((R1 ∩ R2) -> (R2 - R1)) is in F + .

A(ii) Is this decomposition non-additive (lossless) – why?

R1 = SUPPLY1;

R2 = SUPPLIER.

R1 ∩ R2 = Supplier#

R2 – R1 = Supp_name

Because R1 ∩ R2 → (R2 – R1), we conclude that the decomposition is non-additive.

A(iii) What NF is SUPPLY1 in?

In SUPPLY1 (<u>Supplier#, Part#, Date</u>, Project, Quantity, Part_name)

we have:

<u>Supplier#, Part#, Date</u> -> Project, Quantity, Part_name

Part# -> Part_name

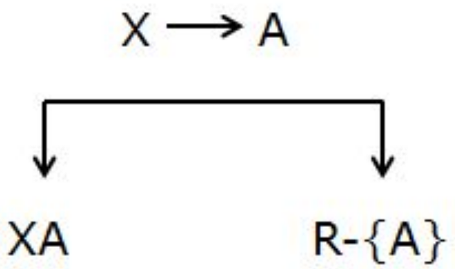
Show a further decomposition of SUPPLY1 and show that the decomposition is non-additive and achieves BCNF.

SUPPLY1 (Supplier#, Part#, Date, Project, Quantity, Part_name)

Part# -> Part_name violates the BCNF definition

R1: (Part#, Part_name).

R2: (Supplier#, Part#, Date, Project, Quantity)



Q2

STUDENT_COURSE (Stud#, Course#, St_name, Course_name, Course_credit_hr, Grade, Major_dept, Dept_phone_no)

F: {<u>Stud#, Course#</u> → St_name, Course_name, Course_credit_hr, Grade, Major_dept, Dept_phone_no;
Stud# → St_name, Major_dept, Dept_phone_no;
Course# → Course_name, Course_credit_hr
Major_dept → Dept_phone_no.}.

Find the 3NF

synthesis algorithm 15.4

1. Find the minimal cover
    a. M: {A-> B, A -> C, C-> D}
2. For each LHS attribute, create a reletion with X1 U {A1} U {A2} …
    a. we have R1 = {ABC), R2 = {CD}
3. Check if any relation contain a key, if not, create a R0 with the key
4. Remove any redundant relation

Minimal Cover
Stud# → St_name, Major_dept

Course# → Course_name, Course_credit_hr

Major_dept → Dept_phone_no

Stud#, Course# → Grade

Use some trick e.g. (A -> BCDE, B -> C, D-> E} -> {B -> C, D->E, A -> BD}

What is the 3NF decompostions?

R1(Stud#, St_name, Major_dept)
R2 (Course#, Course_name, Course_credit_hr)
R3 (Major_dept , Dept_phone_no)
R4(Stud#, Course# , Grade)

Q3

PATIENT_PROC (Patient#, Doctor#, Date, Doctor_name, Doctor_specialty, Procedure, Charge)

The Fds are:
FD1:Patient#, Doctor#, Date → PATIENT_PROC

FD2:Doctor# → Doctor_name, Doctor_specialty

FD3:Procedure → Doctor

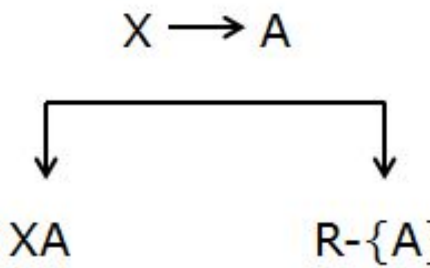
Follow the practice of successive normalization upto BCNF. For converting 3Nf to BCNF, apply the decomposition as per the decomposition in algorithm 15.5

1. Doctor# → Doctor_name, Doctor_specialty violates the BCNF

we will have

PP1 (Patient#, Doctor#, Date, Procedure, Charge)

DOCTOR (Doctor#, Doctor_name, Doctor_specialty)



2. Procedure → Doctor still violates? Keep decomposing PP1!

PP11(Patient#, Date, Procedure, Charge)
PP12 (Procedure, Doctor#) Procedure -> Doctor#
DOCTOR (Doctor#, Doctor_name, Doctor_specialty)