L14: Normalization

CS3200 Database design (sp18 s2)

https://course.ccs.neu.edu/cs3200sp18s2/

3/1/2018

Announcements!

- Keep bringing your name plates ☺
- Outline today
  - More Normalization
  - Project 1 discussion & Transactions after Spring Break

Closures, Superkeys, and (Candidate) Keys

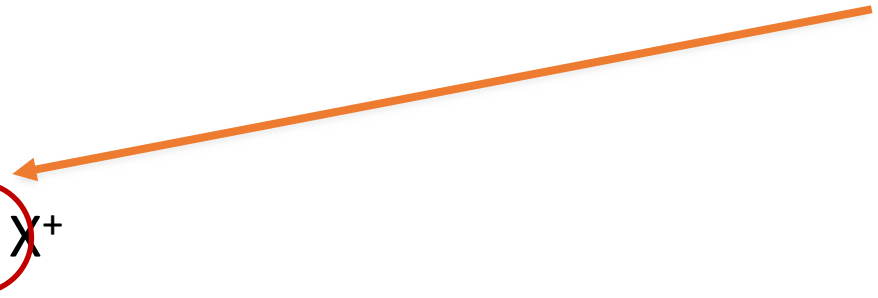
What we will see next

- Closures Part 2
- Superkeys & Keys
- Practice: The key or a key?

Why Do We Need the Closure?

- With closure we can find all FD's easily

- To check if $X \rightarrow A$
  - Compute X^+
  - Check if $A \in X^+$



Note here that **X** is a *set* of attributes, but **A** is a *single* attribute. Why does considering FDs of this form suffice?

Recall the **Split/combine** rule:
$X \rightarrow A_1, \ldots, X \rightarrow A_n$
implies
$X \rightarrow \{A_1, \ldots, A_n\}$

Using Closure to Infer ALL FDs

Step 1: Compute X^+, for every set of attributes X:

Example:
Given F =

$\{A,B\} \rightarrow C$
$\{A,D\} \rightarrow B$
$\{B\} \rightarrow D$

$\{A\}^+ = \{A\}$
$\{B\}^+ = \{B,D\}$
$\{C\}^+ = \{C\}$
$\{D\}^+ = \{D\}$
$\{A,B\}^+ = \{A,B,C,D\}$
$\{A,C\}^+ = \{A,C\}$
$\{A,D\}^+ = \{A,B,C,D\}$
$\{A,B,C\}^+ = \{A,B,D\}^+ = \{A,C,D\}^+ = \{A,B,C,D\}$
$\{B,C,D\}^+ = \{B,C,D\}$
$\{A,B,C,D\}^+ = \{A,B,C,D\}$

No need to compute all of these- why?

Using Closure to Infer ALL FDs

Example:

Given F =

$\{A,B\} \rightarrow C$
$\{A,D\} \rightarrow B$
$\{B\} \rightarrow D$

Step 1: Compute X^+, for every set of attributes X:

$\{A\}^+ = \{A\}$, $\{B\}^+ = \{B,D\}$, $\{C\}^+ = \{C\}$, $\{D\}^+ = \{D\}$, $\{A,B\}^+ = \{A,B,C,D\}$, $\{A,C\}^+ = \{A,C\}$, $\{A,D\}^+ = \{A,B,C,D\}$, $\{A,B,C\}^+ = \{A,B,D\}^+ = \{A,C,D\}^+ = \{A,B,C,D\}$, $\{B,C,D\}^+ = \{B,C,D\}$, $\{A,B,C,D\}^+ = \{A,B,C,D\}$

Step 2: Enumerate all FDs X → Y, s.t. $Y \subseteq X^+$ and $X \cap Y = \emptyset$:

$\{A,B\} \rightarrow \{C,D\}$, $\{A,D\} \rightarrow \{B,C\}$,
$\{A,B,C\} \rightarrow \{D\}$, $\{A,B,D\} \rightarrow \{C\}$,
$\{A,C,D\} \rightarrow \{B\}$

Using Closure to Infer ALL FDs

Example:

Given F =

$\{A,B\} \rightarrow C$
$\{A,D\} \rightarrow B$
$\{B\} \rightarrow D$

Step 1: Compute X^+, for every set of attributes X:

$\{A\}^+ = \{A\}$, $\{B\}^+ = \{B,D\}$, $\{C\}^+ = \{C\}$, $\{D\}^+ = \{D\}$, $\{A,B\}^+ = \{A,B,C,D\}$, $\{A,C\}^+ = \{A,C\}$, $\{A,D\}^+ = \{A,B,C,D\}$, $\{A,B,C\}^+ = \{A,B,D\}^+ = \{A,C,D\}^+ = \{A,B,C,D\}$, $\{B,C,D\}^+ = \{B,C,D\}$, $\{A,B,C,D\}^+ = \{A,B,C,D\}$

Step 2: Enumerate all FDs X → Y, s.t. $Y \subseteq X^+$ and $X \cap Y = \emptyset$:

"Y is in the closure of X"

$\{A,B\} \rightarrow \{C,D\}$, $\{A,D\} \rightarrow \{B,C\}$, $\{A,B,C\} \rightarrow \{D\}$, $\{A,B,D\} \rightarrow \{C\}$, $\{A,C,D\} \rightarrow \{B\}$

Using Closure to Infer ALL FDs

Example:

Given F =

$\{A,B\} \rightarrow C$
$\{A,D\} \rightarrow B$
$\{B\} \rightarrow D$

Step 1: Compute X^+, for every set of attributes X:

$\{A\}^+ = \{A\}$, $\{B\}^+ = \{B,D\}$, $\{C\}^+ = \{C\}$, $\{D\}^+ = \{D\}$, $\{A,B\}^+ = \{A,B,C,D\}$, $\{A,C\}^+ = \{A,C\}$, $\{A,D\}^+ = \{A,B,C,D\}$, $\{A,B,C\}^+ = \{A,B,D\}^+ = \{A,C,D\}^+ = \{A,B,C,D\}$, $\{B,C,D\}^+ = \{B,C,D\}$, $\{A,B,C,D\}^+ = \{A,B,C,D\}$

Step 2: Enumerate all FDs $X \rightarrow Y$, s.t. $Y \subseteq X^+$ and $X \cap Y = \emptyset$:

The FD $X \rightarrow Y$ is non-trivial

$\{A,B\} \rightarrow \{C,D\}$, $\{A,D\} \rightarrow \{B,C\}$,
$\{A,B,C\} \rightarrow \{D\}$, $\{A,B,D\} \rightarrow \{C\}$,
$\{A,C,D\} \rightarrow \{B\}$

Keys and Superkeys

A **superkey** is a set of attributes $A_1, \ldots, A_n$ s.t. for *any other* attribute **B** in R, we have $\{A_1, \ldots, A_n\} \rightarrow B$

I.e. all attributes are *functionally determined* by a superkey

A **key** is a *minimal* superkey (also called "candidate key")

This means that no subset of a key is also a superkey (i.e., dropping any attribute from the key makes it no longer a superkey)

Finding Keys and Superkeys

- For each set of attributes X
  - Compute X^+
  - If X^+ = set of all attributes then X is a superkey
  - If X is minimal, then it is a key

Example of Finding Keys

```
Product(name, price, category, color)
```

```
{name, category} → price
{category} → color
```

What is a key?

Example of Finding Keys

```
Product(name, price, category, color)
```

```
{name, category} → price
{category} → color
```

$\{name, category\}^+ = \{name, price, category, color\}$

= the set of all attributes

$\Longrightarrow$ this is a **superkey**

$\Longrightarrow$ this is a **key**, since neither `name` nor `category` alone is a superkey

Practice

- Activity-21.ipynb

Complete Normalization Practice!

Parking Tickets: Original List

TABLE 4-6 Parking Tickets at Millennium College

Parking Ticket Table

St ID	L Name	F Name	Phone No	St Lic	Lic No	Ticket #	Date	Code	Fine
38249	Brown	Thomas	111-7804	FL	BRY 123	15634	10/17/10	2	$25
						16017	11/13/10	1	$15
82453	Green	Sally	391-1689	AL	TRE 141	14987	10/05/10	3	$100
						16293	11/18/10	1	$15
						17892	12/13/10	2	$25

Parking Tickets: Original List

TABLE 4-6 Parking Tickets at Millennium College

Parking Ticket Table

St ID	L Name	F Name	Phone No	St Lic	Lic No	Ticket #	Date	Code	Fine
38249	Brown	Thomas	111-7804	FL	BRY 123	15634	10/17/10	2	$25
						16017	11/13/10	1	$15
82453	Green	Sally	391-1689	AL	TRE 141	14987	10/05/10	3	$100
						16293	11/18/10	1	$15
						17892	12/13/10	2	$25

ST ID	L Name	F Name	Phone No	St Lic	Lic No	Ticket #	Date	Code	Fine
38249	Brown	Thomas	111-7804	FL	BRY 123	15634	10/17/10	2	$25
38249	Brown	Thomas	111-7804	FL	BRY 123	16017	11/13/10	1	$15
82453	Green	Sally	391-1689	AL	TRE 141	14987	10/05/10	3	$100
82453	Green	Sally	391-1689	AL	TRE 141	16293	11/18/10	1	$15
82453	Green	Sally	391-1689	AL	TRE-141	17892	12/13/10	2	$25

Parking Tickets: Original List

TABLE 4-6 Parking Tickets at Millennium College

Parking Ticket Table

St ID	L Name	F Name	Phone No	St Lic	Lic No	Ticket #	Date	Code	Fine
38249	Brown	Thomas	111-7804	FL	BRY 123	15634	10/17/10	2	$25
						16017	11/13/10	1	$15
82453	Green	Sally	391-1689	AL	TRE 141	14987	10/05/10	3	$100
						16293	11/18/10	1	$15
						17892	12/13/10	2	$25

ST ID	L Name	F Name	Phone No	St Lic	Lic No	Ticket #	Date	Code	Fine
38249	Brown	Thomas	111-7804	FL	BRY 123	15634	10/17/10	2	$25
38249	Brown	Thomas	111-7804	FL	BRY 123	16017	11/13/10	1	$15
82453	Green	Sally	391-1689	AL	TRE 141	14987	10/05/10	3	$100
82453	Green	Sally	391-1689	AL	TRE 141	16293	11/18/10	1	$15
82453	Green	Sally	391-1689	AL	TRE-141	17892	12/13/10	2	$25

Parking Tickets: Relation in 1NF

TABLE 4-6 Parking Tickets at Millennium College

Parking Ticket Table

St ID	L Name	F Name	Phone No	St Lic	Lic No	Ticket #	Date	Code	Fine
38249	Brown	Thomas	111-7804	FL	BRY 123	15634	10/17/10	2	$25
						16017	11/13/10	1	$15
82453	Green	Sally	391-1689	AL	TRE 141	14987	10/05/10	3	$100
						16293	11/18/10	1	$15
						17892	12/13/10	2	$25

ParkingTickets

STID	LName	FName	PhoneNo	StLic	LicNo	Ticketnr	Date	Code	Fine
38249	Brown	Thomas	111-7804	FL	BRY 123	15634	10/17/10	2	$25
38249	Brown	Thomas	111-7804	FL	BRY 123	16017	11/13/10	1	$15
82453	Green	Sally	391-1689	AL	TRE 141	14987	10/05/10	3	$100
82453	Green	Sally	391-1689	AL	TRE 141	16293	11/18/10	1	$15
82453	Green	Sally	391-1689	AL	TRE-141	17892	12/13/10	2	$25

Parking Tickets: Relation in 1NF

TABLE 4-6 Parking Tickets at Millennium College

Parking Ticket Table

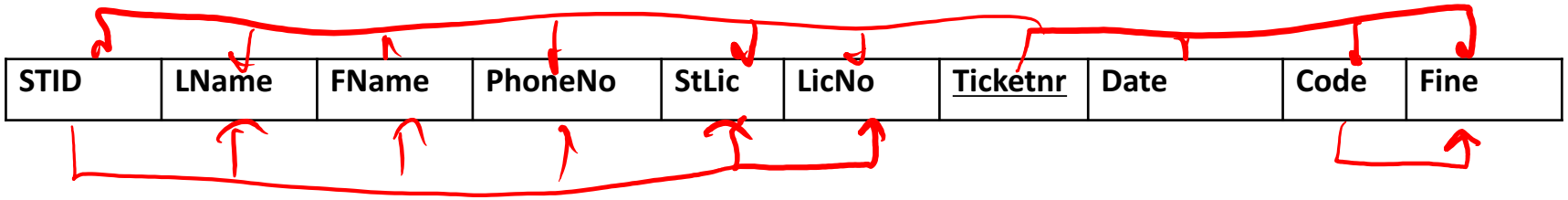
St ID	L Name	F Name	Phone No	St Lic	Lic No	Ticket #	Date	Code	Fine
38249	Brown	Thomas	111-7804	FL	BRY 123	15634	10/17/10	2	$25
						16017	11/13/10	1	$15
82453	Green	Sally	391-1689	AL	TRE 141	14987	10/05/10	3	$100
						16293	11/18/10	1	$15
						17892	12/13/10	2	$25

ParkingTickets

STID	LName	FName	PhoneNo	StLic	LicNo	Ticketnr	Date	Code	Fine
38249	Brown	Thomas	111-7804	FL	BRY 123	15634	10/17/10	2	$25
38249	Brown	Thomas	111-7804	FL	BRY 123	16017	11/13/10	1	$15
82453	Green	Sally	391-1689	AL	TRE 141	14987	10/05/10	3	$100
82453	Green	Sally	391-1689	AL	TRE 141	16293	11/18/10	1	$15
82453	Green	Sally	391-1689	AL	TRE-141	17892	12/13/10	2	$25

Parking Tickets: Dependency diagram

Assume, each student can have maximal one car:

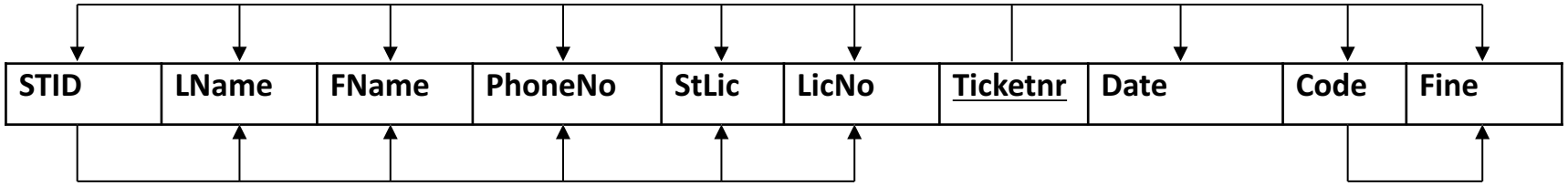


ParkingTickets

STID	LName	FName	PhoneNo	StLic	LicNo	Ticketnr	Date	Code	Fine
38249	Brown	Thomas	111-7804	FL	BRY 123	15634	10/17/10	2	$25
38249	Brown	Thomas	111-7804	FL	BRY 123	16017	11/13/10	1	$15
82453	Green	Sally	391-1689	AL	TRE 141	14987	10/05/10	3	$100
82453	Green	Sally	391-1689	AL	TRE 141	16293	11/18/10	1	$15
82453	Green	Sally	391-1689	AL	TRE-141	17892	12/13/10	2	$25

Parking Tickets: Dependency diagram

Assume, each student can have maximal one car:

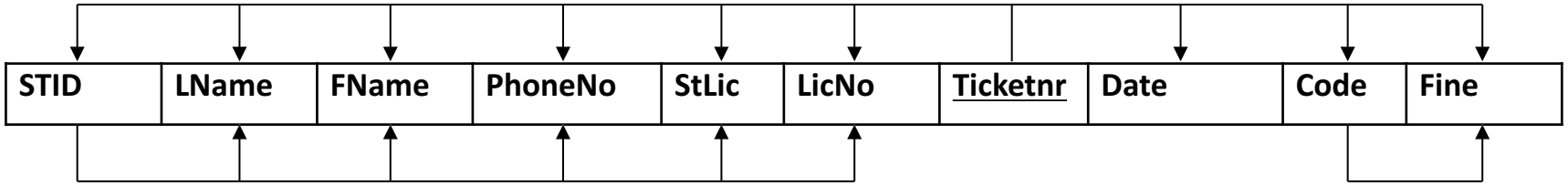


ParkingTickets

STID	LName	FName	PhoneNo	StLic	LicNo	Ticketnr	Date	Code	Fine
38249	Brown	Thomas	111-7804	FL	BRY 123	15634	10/17/10	2	$25
38249	Brown	Thomas	111-7804	FL	BRY 123	16017	11/13/10	1	$15
82453	Green	Sally	391-1689	AL	TRE 141	14987	10/05/10	3	$100
82453	Green	Sally	391-1689	AL	TRE 141	16293	11/18/10	1	$15
82453	Green	Sally	391-1689	AL	TRE-141	17892	12/13/10	2	$25

Parking Tickets: Dependency diagram

Assume, each student can have maximal one car:



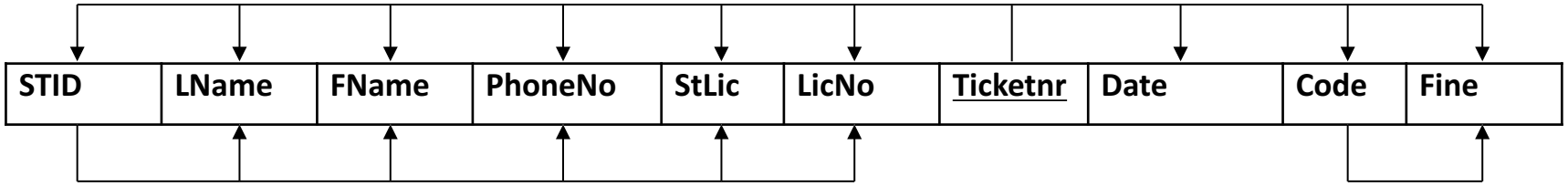
Next assume, students can have more than one car:

ParkingTickets

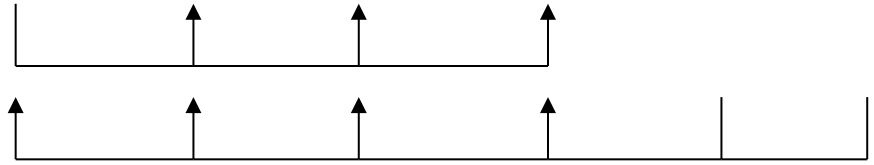
STID	LName	FName	PhoneNo	StLic	LicNo	Ticketnr	Date	Code	Fine
38249	Brown	Thomas	111-7804	FL	BRY 123	15634	10/17/10	2	$25
38249	Brown	Thomas	111-7804	FL	BRY 123	16017	11/13/10	1	$15
82453	Green	Sally	391-1689	AL	TRE 141	14987	10/05/10	3	$100
82453	Green	Sally	391-1689	AL	TRE 141	16293	11/18/10	1	$15
82453	Green	Sally	391-1689	AL	TRE-141	17892	12/13/10	2	$25

Parking Tickets: Dependency diagram

Assume, each student can have maximal one car:



Next assume, students can have more than one car:

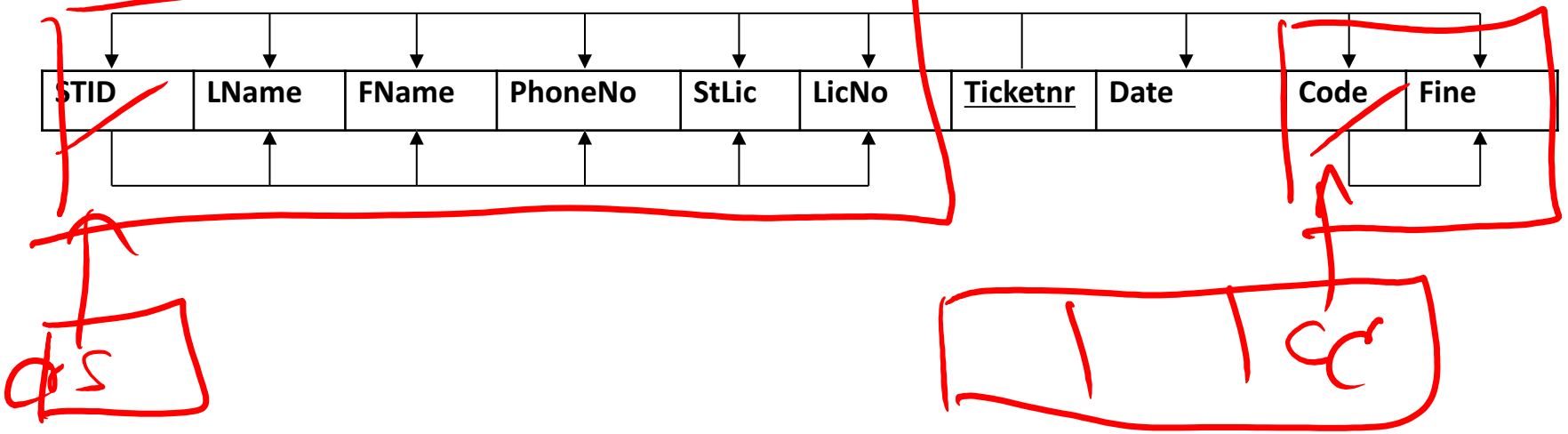


ParkingTickets

STID	LName	FName	PhoneNo	StLic	LicNo	Ticketnr	Date	Code	Fine
38249	Brown	Thomas	111-7804	FL	BRY 123	15634	10/17/10	2	$25
38249	Brown	Thomas	111-7804	FL	BRY 123	16017	11/13/10	1	$15
82453	Green	Sally	391-1689	AL	TRE 141	14987	10/05/10	3	$100
82453	Green	Sally	391-1689	AL	TRE 141	16293	11/18/10	1	$15
82453	Green	Sally	391-1689	AL	TRE-141	17892	12/13/10	2	$25

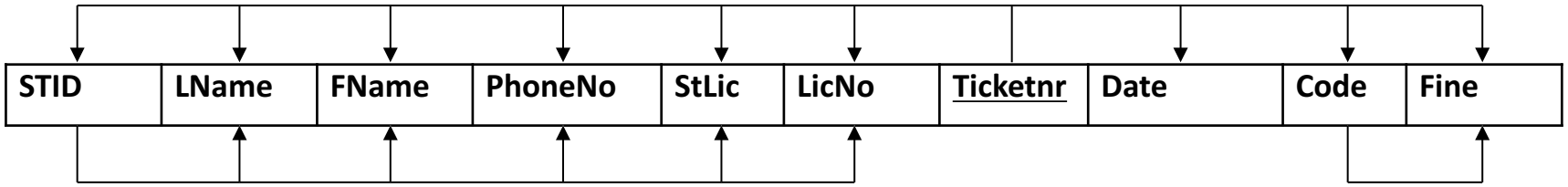
Parking Tickets: Relations in 3NF

Assume, each student can have ~~maximal one car~~:

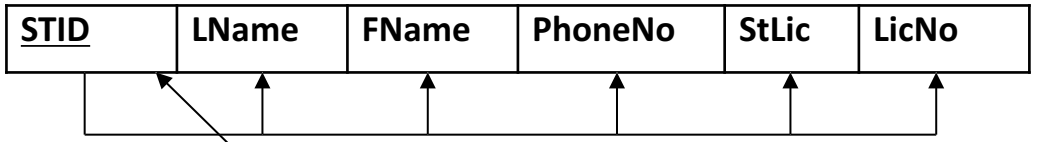


Parking Tickets: Relations in 3NF

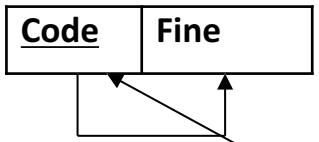
Assume, each student can have maximal one car:



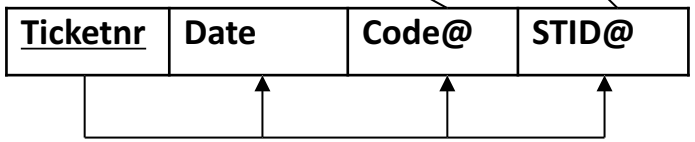
Student



TicketCode

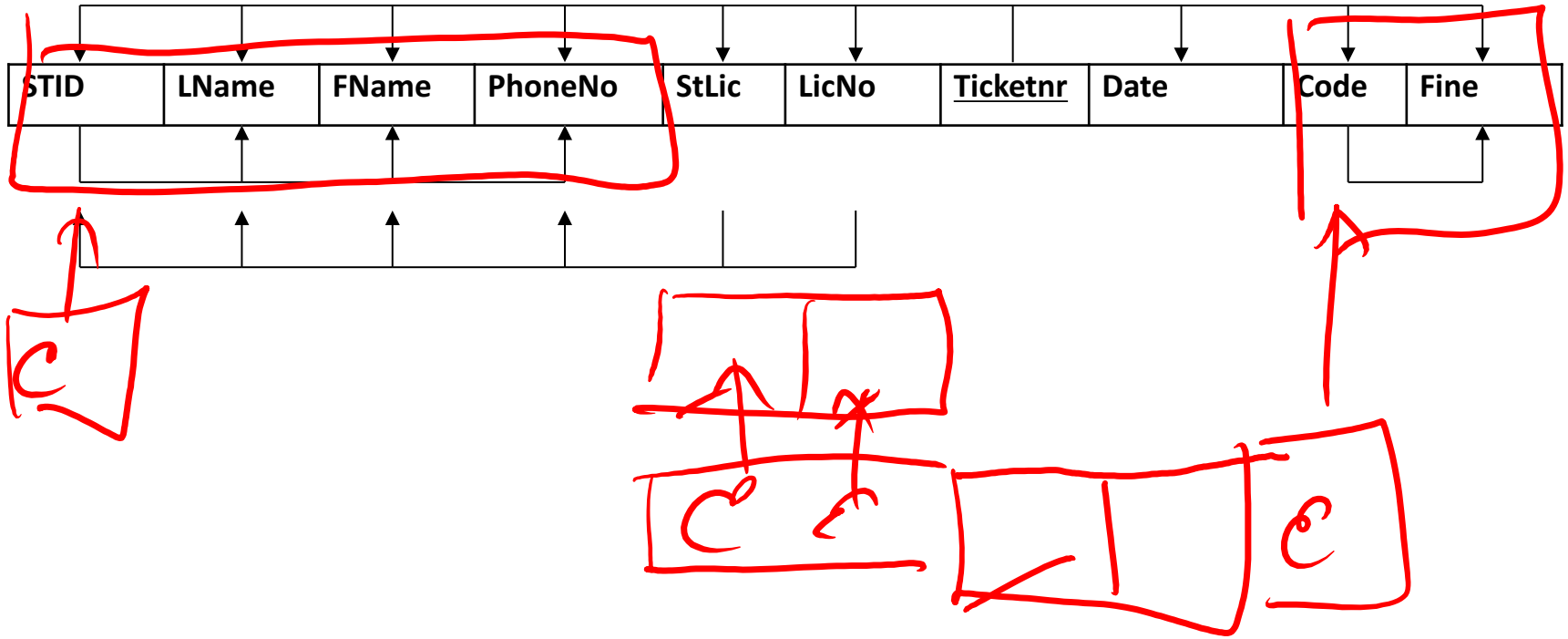


Ticket



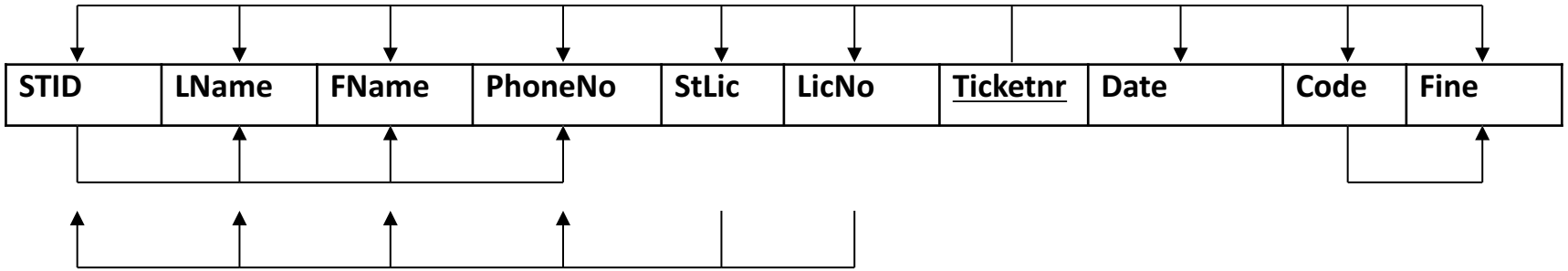
Parking Tickets: Relations in 3NF

Next assume, students can have more than one car:



Parking Tickets: Relations in 3NF

Next assume, students can have more than one car:



Student

STID	LName	FName	PhoneNo

Car

StLic	LicNo	STID@

TicketCode

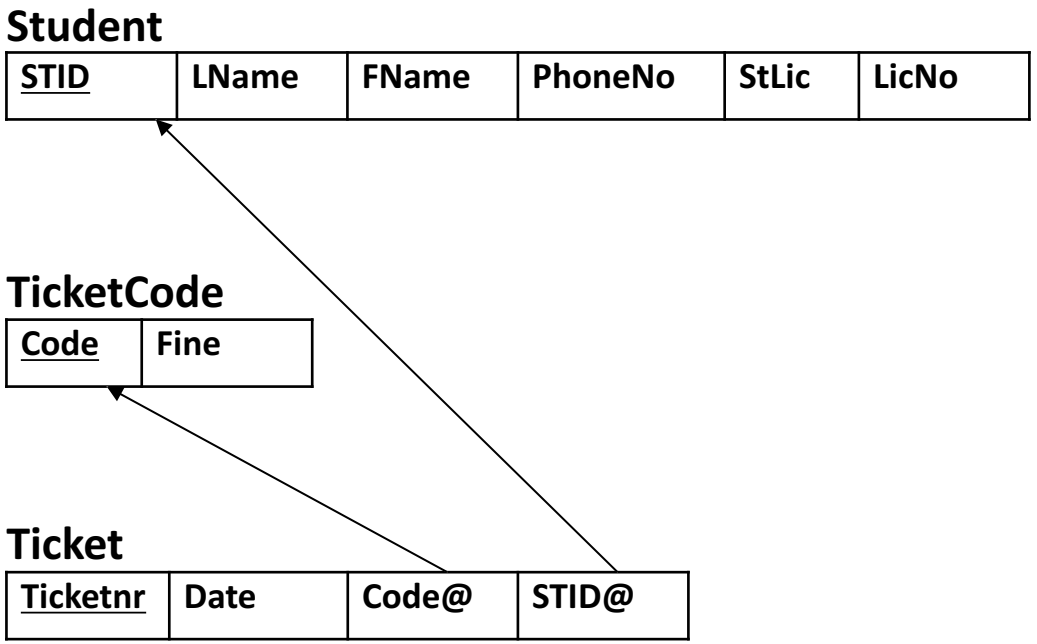
Code	Fine

Ticket

Ticketnr	Date	Code@	StLic@	LicNo@

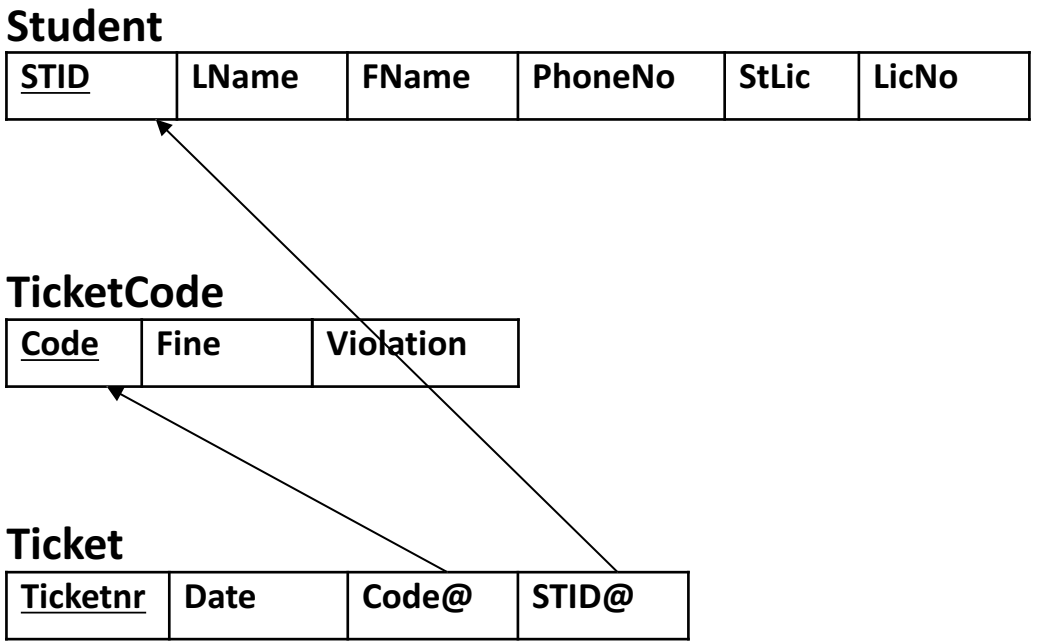
Parking Tickets: Adding Violation

Assume, each student can have maximal one car:

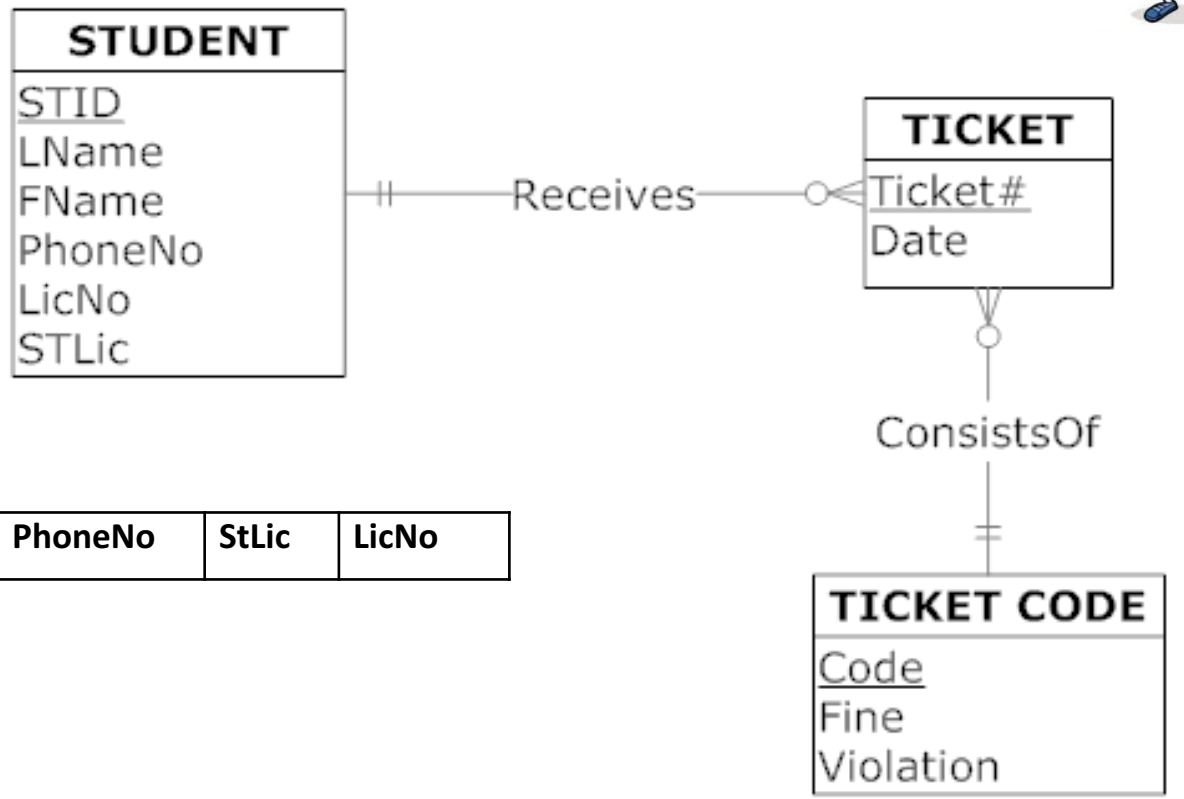


Parking Tickets: Adding Violation

Assume, each student can have maximal one car:



Parking Tickets: ER Diagram



Student

STID	LName	FName	PhoneNo	StLic	LicNo

TicketCode

Code	Fine	Violation

Ticket

Ticketnr	Date	Code@	STID@

ER diagrams do not show foreign keys!

Complete Normalization Practice!

Example: DreamHome Rental

StaffPropertyInspection

propertyNo	pAddress	iDate	iTime	comments	staffNo	sName	carReg
PG4	6 Lawrence St, Glasgow	18-Oct-03	10:00	need to replace crockery	SG37	Ann Beech	M231 JGR
		22-Apr-04	09:00	in good order	SG14	David Ford	M533 HDR
		1-Oct-04	12:00	damp rot in bathroom	SG14	David Ford	N721 HFR
PG16	5 Novar Dr, Glasgow	22-Apr-04	13:00	replace living room carpet	SG14	David Ford	M533 HDR
		24-Oct-04	14:00	good condition	SG37	Ann Beech	N721 HFR

Can a database store this information?
Is it in 1NF?

Members of DreamHome inspect properties

- When staff are required to undertake these inspections, they are allocated a company car for use on the day of the inspections. (One car per person & day)
- However, a car may be allocated to several members of staff as required throughout the working day.
- A member of staff may inspect several properties on a given date, but a property is only inspected once on a given date.

Source: Connolly, Begg: Database systems, 4th ed, p. 423, 2005.

Example: DreamHome Rental

StaffPropertyInspection

propertyNo	iDate	iTime	pAddress	comments	staffNo	sName	carReg
PG4	18-Oct-03	10:00	6 Lawrence St, Glasgow	need to replace crockery	SG37	Ann Beech	M231 JGR
PG4	22-Apr-04	09:00	6 Lawrence St, Glasgow	in good order	SG14	David Ford	M533 HDR
PG4	1-Oct-04	12:00	6 Lawrence St, Glasgow	damp rot in bathroom	SG14	David Ford	N721 HFR
PG16	22-Apr-04	13:00	5 Novar Dr, Glasgow	replace living room carpet	SG14	David Ford	M533 HDR
PG16	24-Oct-04	14:00	5 Novar Dr, Glasgow	good condition	SG37	Ann Beech	N721 HFR

No! Only now a database can store the information: 1NF
But we still need a primary key

Members of DreamHome inspect properties

- When staff are required to undertake these inspections, they are allocated a company car for use on the day of the inspections. (One car per person & day)
- However, a car may be allocated to several members of staff as required throughout the working day.
- A member of staff may inspect several properties on a given date, but a property is only inspected once on a given date.

Source: Connolly, Begg: Database systems, 4th ed, p. 423, 2005.

Example: DreamHome Rental

StaffPropertyInspection

propertyNo	iDate	iTime	pAddress	comments	staffNo	sName	carReg
PG4	18-Oct-03	10:00	6 Lawrence St, Glasgow	need to replace crockery	SG37	Ann Beech	M231 JGR
PG4	22-Apr-04	09:00	6 Lawrence St, Glasgow	in good order	SG14	David Ford	M533 HDR
PG4	1-Oct-04	12:00	6 Lawrence St, Glasgow	damp rot in bathroom	SG14	David Ford	N721 HFR
PG16	22-Apr-04	13:00	5 Novar Dr, Glasgow	replace living room carpet	SG14	David Ford	M533 HDR
PG16	24-Oct-04	14:00	5 Novar Dr, Glasgow	good condition	SG37	Ann Beech	N721 HFR

Now 1NF + PK

Members of DreamHome inspect properties

- When staff are required to undertake these inspections, they are allocated a company car for use on the day of the inspections. (One car per person & day)
- However, a car may be allocated to several members of staff as required throughout the working day.
- A member of staff may inspect several properties on a given date, but a property is only inspected once on a given date.

Source: Connolly, Begg: Database systems, 4th ed, p. 423, 2005.

Example: DreamHome Rental

StaffPropertyInspection

propertyNo	iDate	iTime	pAddress	comments	staffNo	sName	carReg

Draw all FDs

Members of DreamHome inspect properties

- When staff are required to undertake these inspections, they are allocated a company car for use on the day of the inspections. (One car per person & day)
- However, a car may be allocated to several members of staff as required throughout the working day.
- A member of staff may inspect several properties on a given date, but a property is only inspected once on a given date.

Source: Connolly, Begg: Database systems, 4th ed, p. 423, 2005.

Example: DreamHome Rental

StaffPropertyInspection

propertyNo	iDate	iTime	pAddress	comments	staffNo	sName	carReg

(full, PK)

Members of DreamHome inspect properties

- When staff are required to undertake these inspections, they are allocated a company car for use on the day of the inspections. (One car per person & day)
- However, a car may be allocated to several members of staff as required throughout the working day.
- A member of staff may inspect several properties on a given date, but a property is only inspected once on a given date.

Source: Connolly, Begg: Database systems, 4th ed, p. 423, 2005.

Example: DreamHome Rental

StaffPropertyInspection

propertyNo	iDate	iTime	pAddress	comments	staffNo	sName	carReg

(full, PK)

(partial)

(transitive)

Members of DreamHome inspect properties

- When staff are required to undertake these inspections, they are allocated a company car for use on the day of the inspections. (One car per person & day)
- However, a car may be allocated to several members of staff as required throughout the working day.
- A member of staff may inspect several properties on a given date, but a property is only inspected once on a given date.

Source: Connolly, Begg: Database systems, 4th ed, p. 423, 2005.

Example: DreamHome Rental

StaffPropertyInspection

propertyNo	iDate	iTime	pAddress	comments	staffNo	sName	carReg

(full, PK)

(partial)

(transitive)

(other)

Members of DreamHome inspect properties

- When staff are required to undertake these inspections, they are allocated a company car for use on the day of the inspections. (One car per person & day)
- However, a car may be allocated to several members of staff as required throughout the working day.
- A member of staff may inspect several properties on a given date, but a property is only inspected once on a given date.

Source: Connolly, Begg: Database systems, 4th ed, p. 423, 2005.

Example: DreamHome Rental

StaffPropertyInspection

propertyNo	iDate	iTime	pAddress	comments	staffNo	sName	carReg

(full, PK)

(partial)

(transitive)

(other)

(Candidate K)

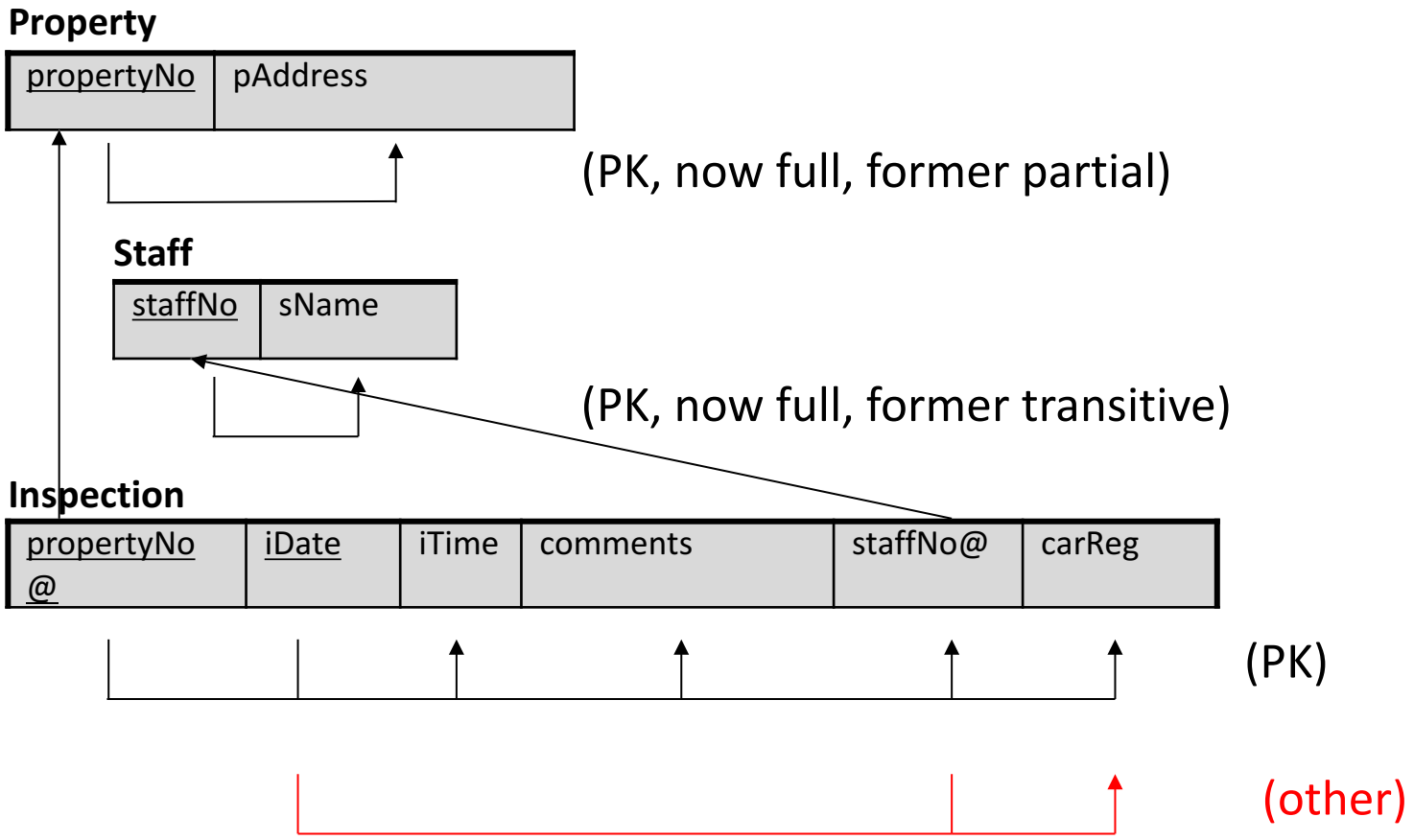
(Candidate K)

Members of DreamHome inspect properties

- When staff are required to undertake these inspections, they are allocated a company car for use on the day of the inspections. (One car per person & day)
- However, a car may be allocated to several members of staff as required throughout the working day.
- A member of staff may inspect several properties on a given date, but a property is only inspected once on a given date.

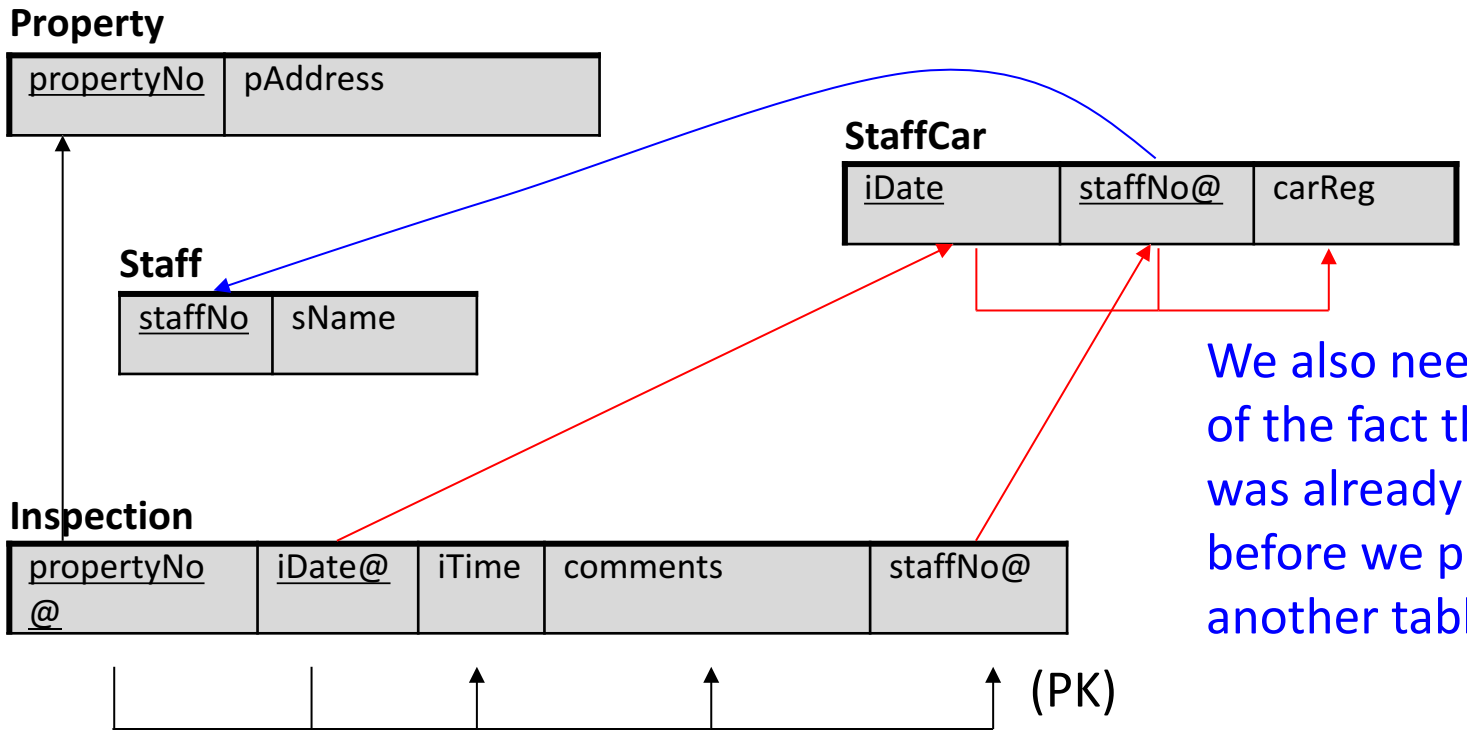
Source: Connolly, Begg: Database systems, 4th ed, p. 423, 2005.

Example: DreamHome Rental



Source: Connolly, Begg: Database systems, 4th ed, p. 423, 2005.

Example: DreamHome Rental

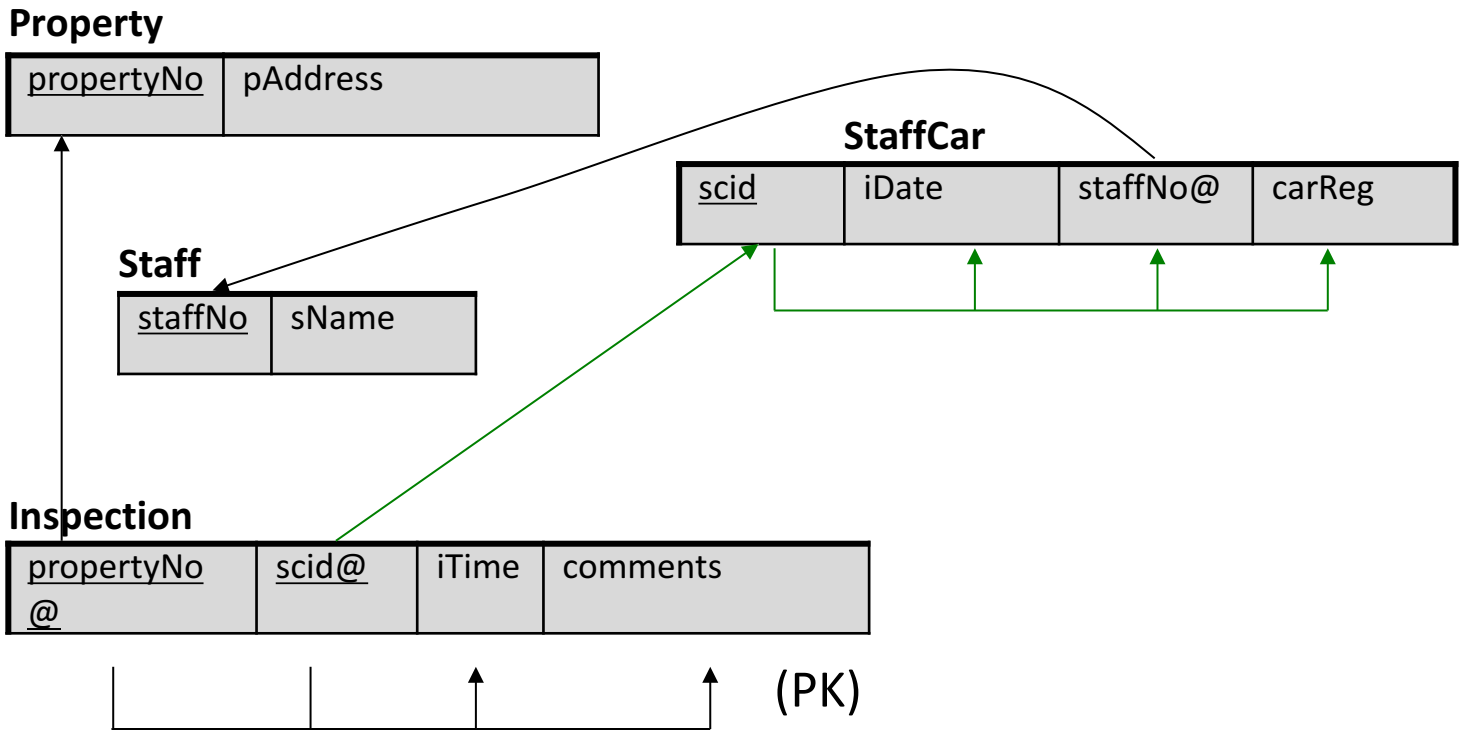


We also need to keep track of the fact that "staffno@" was already a foreign key before we put it into another table

Extra question: We now have a composite FK (idate, staffno) from INSPECTION to STAFFCAR. Thus (idate, staffno) is a composite PK in STAFFCAR.
Assume we like to replace it with a surrogate key. How would the resulting completely normalized tables look like?

Source: Connolly, Begg: Database systems, 4th ed, p. 423, 2005.

Example: DreamHome Rental



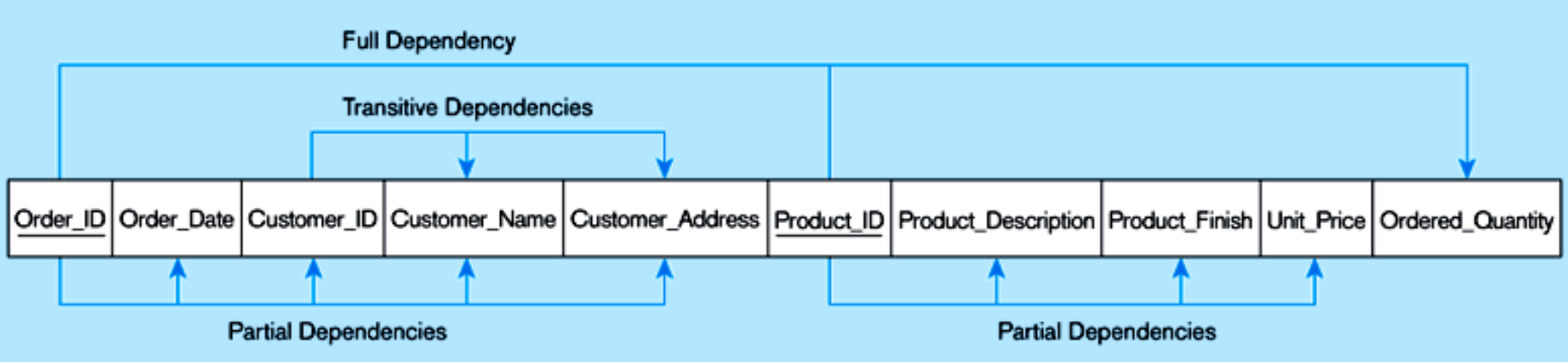
This is now fully normalized.
Downside: we need to join INSPECTION with STAFFCAR every time we like to find out about when a property (by "properyNo") was last inspected

Source: Connolly, Begg: Database systems, 4th ed, p. 423, 2005.

Boyce-Codd Normal Form

Quick recap FDs

- **Functional Dependency**: The value of one set of attributes (the **determinant**) uniquely determines the value of another set of attributes (the **dependents**)
- A **superkey** is as a set of attributes of a relation schema upon which all attributes of the schema are functionally dependent.
- A **candidate key (CK)** is an attribute or non-redundant combination of attributes that uniquely identifies a row in a relation
- **2NF**: no **Partial FD**: A FD in which one or more nonkey attributes are functionally dependent on part (but not all) of the PK
- **3NF**: no **Transitive dependency**: An FD between two (or more) nonkey attributes



Boyce-Codd Normal Form (BCNF)

- Boyce-Codd normal form (BCNF)
  - A relation is in BCNF, if and only if, every determinant is a candidate key.

- The difference between 3NF and BCNF is that for a functional dependency A→B,
  - 3NF allows this dependency in a relation if B is a primary-key attribute and A is not a candidate key,
  - whereas BCNF insists that for this dependency to remain in a relation, A must be a candidate key.

3NF to BCNF

STUDENT ADVISOR

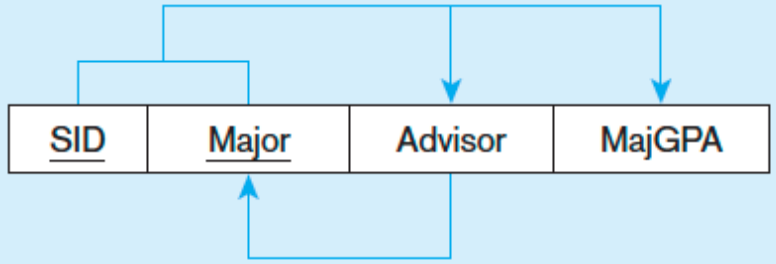
SID	Major	Advisor	MajGPA
123	Physics	Hawking	4.0
123	Music	Mahler	3.3
456	Literature	Michener	3.2
789	Music	Bach	3.7
678	Physics	Hawking	3.5

Source: Hoffer, Ramesh, Topi, Modern database management, 10th ed, Appendix B, 2010.

3NF to BCNF

STUDENT ADVISOR

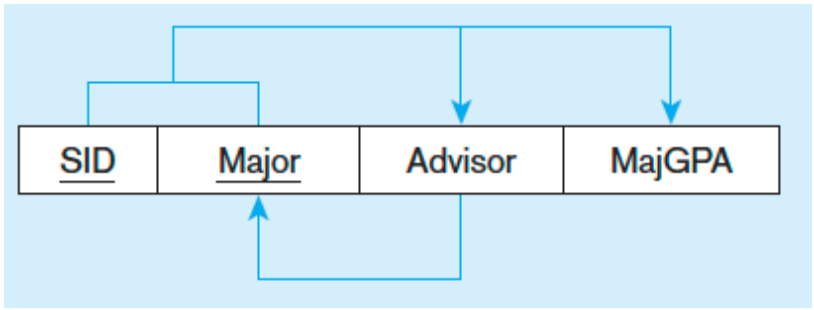
SID	Major	Advisor	MajGPA
123	Physics	Hawking	4.0
123	Music	Mahler	3.3
456	Literature	Michener	3.2
789	Music	Bach	3.7
678	Physics	Hawking	3.5

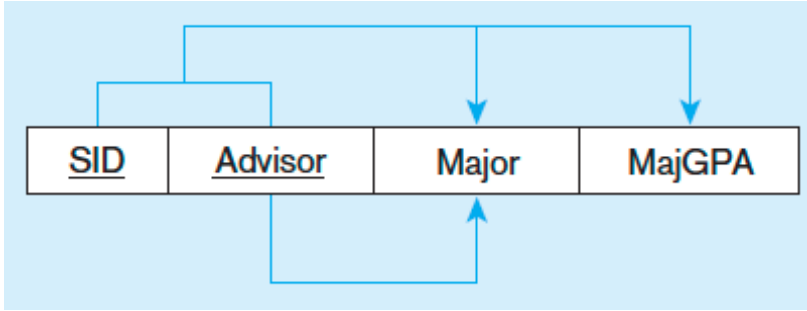


3NF to BCNF

STUDENT ADVISOR

SID	Major	Advisor	MajGPA
123	Physics	Hawking	4.0
123	Music	Mahler	3.3
456	Literature	Michener	3.2
789	Music	Bach	3.7
678	Physics	Hawking	3.5



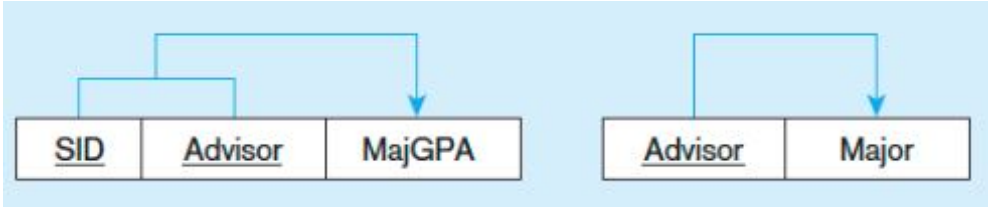


STUDENT

SID	Advisor	MajGPA
123	Hawking	4.0
123	Mahler	3.3
456	Michener	3.2
789	Bach	3.7
678	Hawking	3.5

ADVISOR

Advisor	Major
Hawking	Physics
Mahler	Music
Michener	Literature
Bach	Music



Source: Hoffer, Ramesh, Topi, Modern database management, 10th ed, Appendix B, 2010.

BCNF vs 3NF

- **BCNF**: For every functional dependency X->Y in a set *F* of functional dependencies over relation *R*, either:
  - X is a superkey of R
  - (or Y is a subset of X)

- **3NF**: For every functional dependency X->Y in a set F of functional dependencies over relation R, either:
  - X is a superkey of R, or
  - Y is a subset of K for some key K of R
    - *N.b.,* no subset of a key is a key
  - (or Y is a subset of X)

Back to Conceptual Design

- Now that we know how to find FDs, it's a straight-forward process:
  - Search for "bad" FDs
  - If there are any, then keep decomposing the table into sub-tables until no more bad FDs
  - When done, the database schema is normalized

Recall: there are several normal forms…

Boyce-Codd Normal Form (BCNF)

- Main idea is that we define “good” and “bad” FDs as follows:
  - X → A is a “good FD” if X is a (super)key
    - In other words, if A is the set of all attributes
  - X → A is a “bad FD” otherwise
- We will try to eliminate the “bad” FDs!

Boyce-Codd Normal Form (BCNF)

- Why does this definition of “good” and “bad” FDs make sense?

- If X is not a (super)key, it functionally determines some of the attributes; therefore, those other attributes can be duplicated

  - Recall: this means there is redundancy
  - And redundancy like this can lead to data anomalies!

EmpID	Name	Phone	Position
E0045	Smith	1234	Clerk
E3542	Mike	9876	Salesrep
E1111	Smith	9876	Salesrep
E9999	Mary	1234	Lawyer

Boyce-Codd Normal Form

BCNF is a simple condition for removing anomalies from relations:

A relation R is **in BCNF** if:

if **$\{A_1, ..., A_n\} \rightarrow B$** is a *non-trivial* FD in R

then **$\{A_1, ..., A_n\}$ is a superkey** for R

Equivalently: ∀ sets of attributes X, either ($X^+ = X$) or (X^+ = all attributes)

In other words: there are no "bad" FDs

Example

Name	SSN	PhoneNumber	City
Fred	123-45-6789	206-555-1234	Seattle
Fred	123-45-6789	206-555-6543	Seattle
Joe	987-65-4321	908-555-2121	Westfield
Joe	987-65-4321	908-555-1234	Westfield

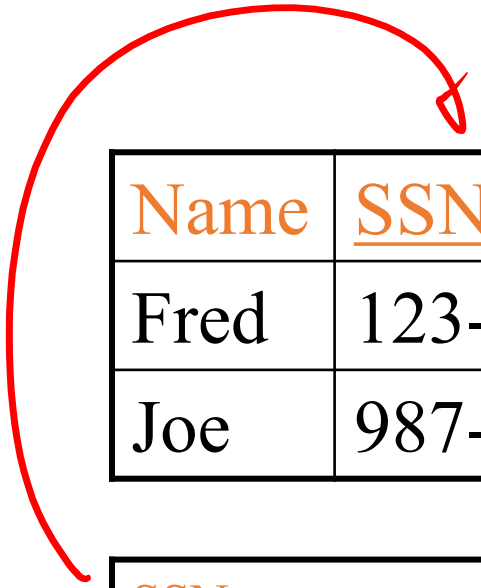
`{SSN} → {Name,City}`

This FD is *bad* because it is **not** a superkey

⟹ **Not** in BCNF

What is the key?
{SSN, PhoneNumber}

Example



Name	SSN	City
Fred	123-45-6789	Seattle
Joe	987-65-4321	Madison

SSN	PhoneNumber
123-45-6789	206-555-1234
123-45-6789	206-555-6543
987-65-4321	908-555-2121
987-65-4321	908-555-1234

{SSN} → {Name,City}

This FD is now *good* because it is the key

Let's check anomalies:

- Redundancy ?
- Update ?
- Delete ?

Now in BCNF!

BCNF Decomposition Algorithm

BCNFDecomp(R):

BCNF Decomposition Algorithm

BCNFDecomp(R):

Find *a set of attributes* X s.t.: $X^+ \neq X$ and $X^+ \neq$ [all attributes]

Find a set of attributes X which has non-trivial "bad" FDs, i.e. is not a superkey, using closures

BCNF Decomposition Algorithm

BCNFDecomp(R):

Find a *set of attributes* X s.t.: $X^+ \neq X$ and $X^+ \neq$ [all attributes]

if (not found) **then Return** R

If no "bad" FDs found, in BCNF!

BCNF Decomposition Algorithm

BCNFDecomp(R):

Find a *set of attributes* X s.t.: $X^+ \neq X$ and $X^+ \neq$ [all attributes]

if (not found) **then Return** R

let $Y = X^+ - X$, $Z = (X^+)^C$

Let Y be the attributes that ***X functionally determines*** (+ that are not in X)

And let Z be **the *complement*, the other attributes that it *doesn't***

BCNF Decomposition Algorithm

BCNFDecomp(R):

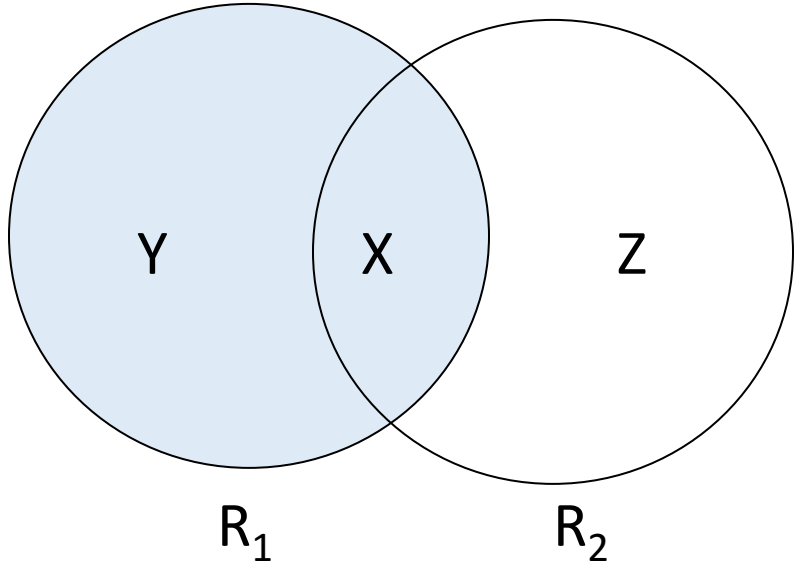
Find a *set of attributes* X s.t.: $X^+ \neq X$ and $X^+ \neq$ [all attributes]

if (not found) then Return R

let $Y = X^+ - X$, $Z = (X^+)^C$

decompose R into $\mathbf{R_1(X \cup Y)}$ and $\mathbf{R_2(X \cup Z)}$

Split into one relation (table) with X plus the attributes that X determines (Y)…



BCNF Decomposition Algorithm

BCNFDecomp(R):

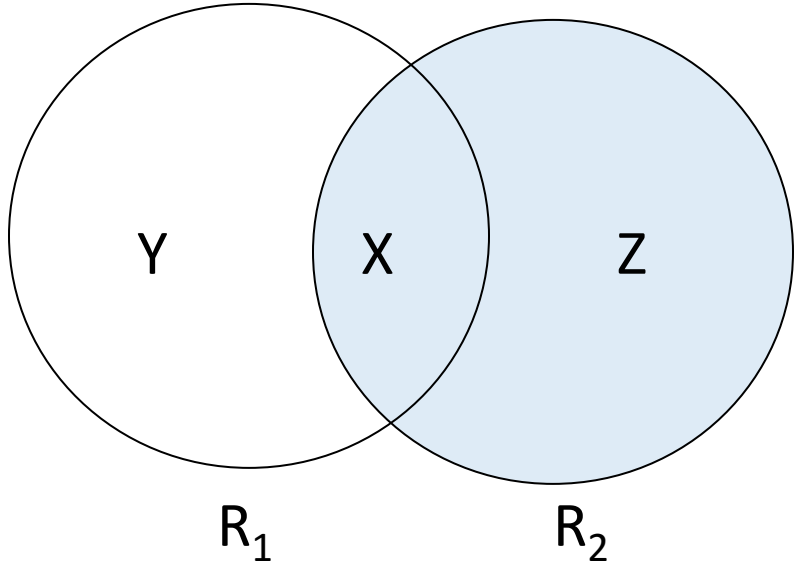
Find a *set of attributes* X s.t.: $X^+ \neq X$ and $X^+ \neq$ [all attributes]

if (not found) **then Return** R

let $Y = X^+ - X$, $Z = (X^+)^C$

decompose R into $\mathbf{R_1(X \cup Y)}$ and $\mathbf{R_2(X \cup Z)}$

And one relation with X plus the attributes it *does not* determine (Z)



BCNF Decomposition Algorithm

BCNFDecomp(R):

Find a *set of attributes* X s.t.: $X^+ \neq X$ and $X^+ \neq$ [all attributes]

if (not found) **then Return** R

let $Y = X^+ - X$, $Z = (X^+)^C$

decompose R into $\mathbf{R_1(X \cup Y)}$ and $\mathbf{R_2(X \cup Z)}$

Return BCNFDecomp(R_1), BCNFDecomp(R_2)

Proceed recursively until no more "bad" FDs!

Example

BCNFDecomp(R):

Find a *set of attributes* X s.t.: $X^+ \neq X$ and $X^+ \neq$ [all attributes]

if (not found) **then Return** R

let $Y = X^+ - X$, $Z = (X^+)^C$

decompose R into $\mathbf{R_1(X \cup Y)}$ and $\mathbf{R_2(X \cup Z)}$

Return BCNFDecomp(R_1), BCNFDecomp(R_2)

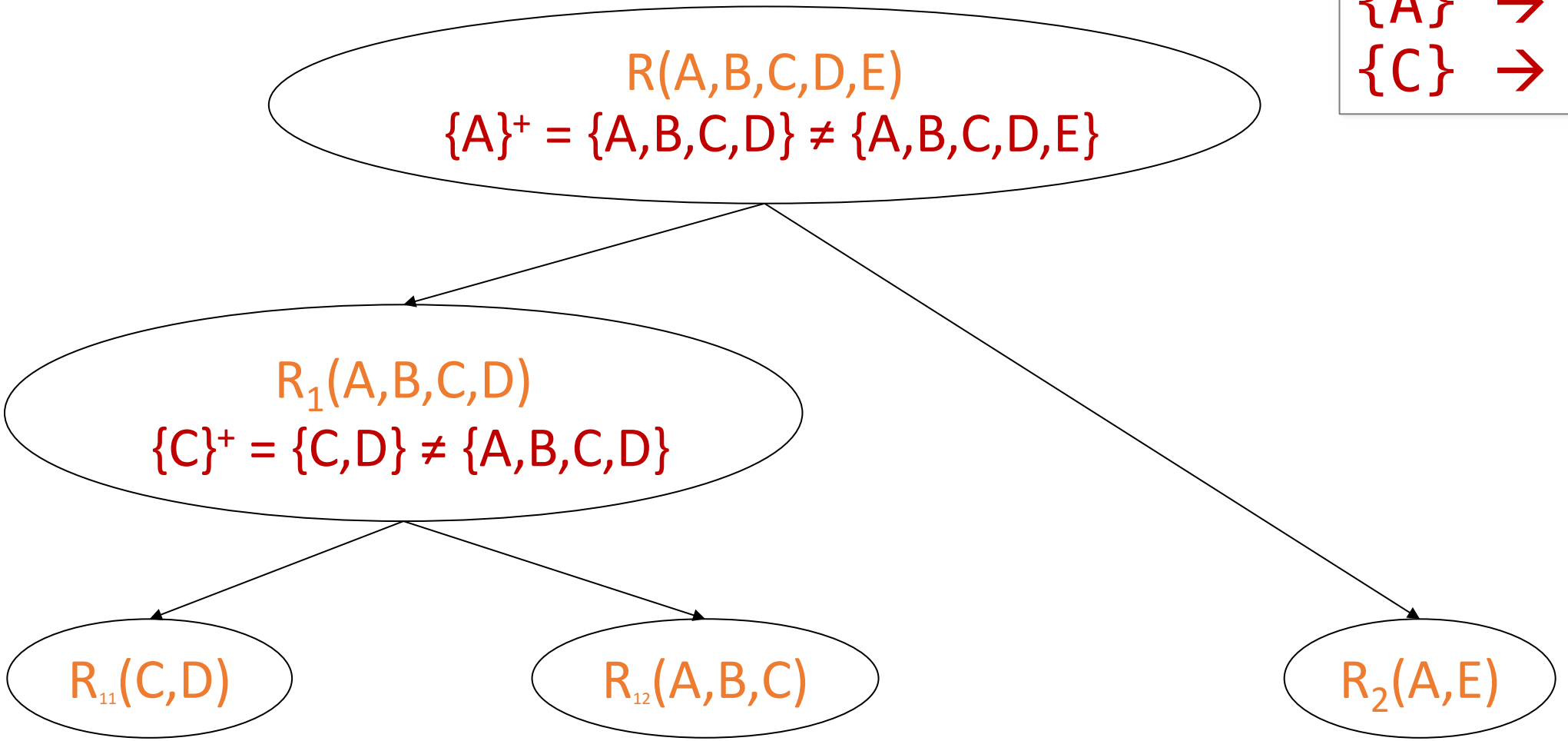
R(A,B,C,D,E)

{A} → {B,C}
{C} → {D}

Example

R(A,B,C,D,E)

{A} → {B,C}
{C} → {D}



Practice

- Activity-22.ipynb