

Normalization

What it's all about

- Given a relation, R , and a set of functional dependencies, F , on R .
- Assume that R is not in a desirable form for enforcing F .
- Decompose relation R into relations, R_1, \dots, R_k , with associated functional dependencies, F_1, \dots, F_k , such that R_1, \dots, R_k are in a more desirable form, 3NF or BCNF.
- While decomposing R , make sure to preserve the dependencies, and make sure not to lose information.

Contents

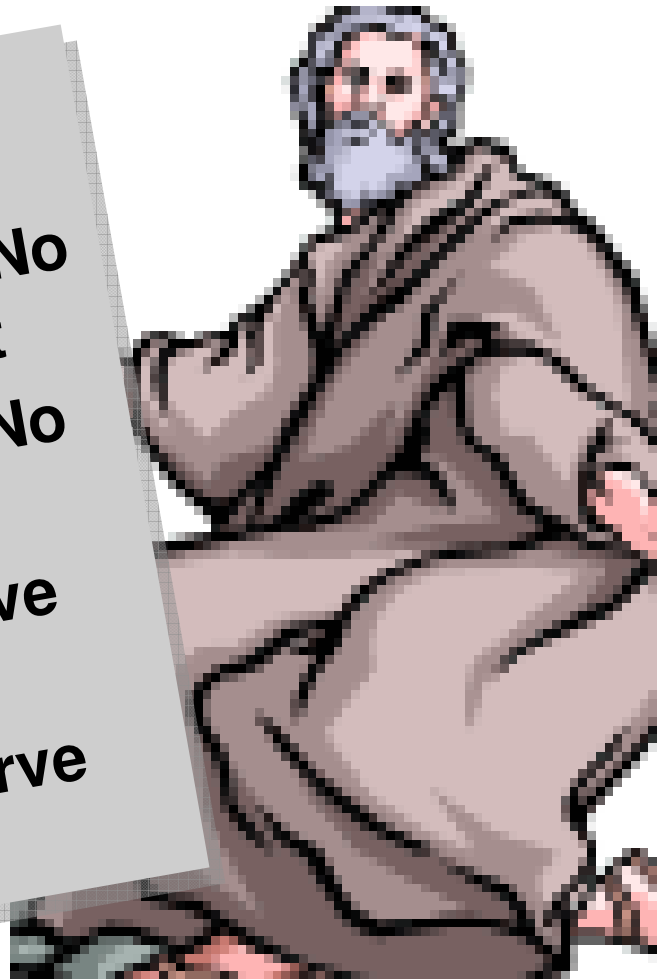
- The Good and the Bad
- Bad database design
 - redundancy of fact
 - fact clutter
 - information loss
 - dependency loss
- Good database design
- How to compute with meaning
 - functional dependencies - FDs
 - Armstrong's inference rules
 - the meaning of a set of FDs
 - minimal cover of a set of FDs
- Normal Forms - overview
- 1NF, 2NF, 3NF, BCNF

The Good

The Four

Commandments:

- Thou Shalt Commit No Redundancy of Fact
- Thou Shalt Clutter No Facts
- Thou Shalt Preserve Information
- Thou Shalt Preserve Functional Dependencies



Primitive Domains

FLT-SCHEDULE

flt#	weekday	airline	dtime	from	atime	to
DL242	MO WE FR	DELTA	10:40	ATL	12:30	BOS
SK912	SA SU	SAS	12:00	CPH	15:30	JFK
AA242	MO FR	AA	08:00	CHI	10:10	ATL

Attributes must be defined over domains with atomic values

FLT-SCHEDULE

flt#	weekday	airline	dtime	from	atime	to
DL242	MO	DELTA	10:40	ATL	12:30	BOS
DL242	WE	DELTA	10:40	ATL	12:30	BOS
DL242	FR	DELTA	10:40	ATL	12:30	BOS
SK912	SA	SAS	12:00	CPH	15:30	JFK
SK912	SU	SAS	12:00	CPH	15:30	JFK
AA242	MO	AA	08:00	CHI	10:10	ATL
AA242	FR	AA	08:00	CHI	10:10	ATL

Bad Database Design

- redundancy of fact

FLIGHTS

flt#	date	airline	plane#
DL242	10/23/00	Delta	k-yo-33297
DL242	10/24/00	Delta	t-up-73356
DL242	10/25/00	Delta	o-ge-98722
AA121	10/24/00	American	p-rw-84663
AA121	10/25/00	American	q-yg-98237
AA411	10/22/00	American	h-fe-65748

- **redundancy:** airline name repeated for same flight
- **inconsistency:** when airline name for a flight changes, it must be changed many places

Bad Database Design - fact clutter

FLIGHTS

flt#	date	airline	plane#
DL242	10/23/00	Delta	k-yo-33297
DL242	10/24/00	Delta	t-up-73356
DL242	10/25/00	Delta	o-ge-98722
AA121	10/24/00	American	p-rw-84663
AA121	10/25/00	American	q-yg-98237
AA411	10/22/00	American	h-fe-65748

- **insertion anomalies:** how do we represent that SK912 is flown by Scandinavian without there being a date and a plane assigned?
- **deletion anomalies:** cancelling AA411 on 10/22/00 makes us lose that it is flown by American.
- **update anomalies:** if DL242 is flown by Sabena, we must change it everywhere.

Bad Database Design - information loss

FLIGHTS

flt#	date	airline	plane#
DL242	10/23/00	Delta	k-yo-33297
DL242	10/24/00	Delta	t-up-73356
DL242	10/25/00	Delta	o-ge-98722
AA121	10/24/00	American	p-rw-84663
AA121	10/25/00	American	q-yg-98237
AA411	10/22/00	American	h-fe-65748

FLIGHTS-AIRLINE

flt#	airline
DL242	Delta
AA121	American
AA411	American

DATE-AIRLINE-PLANE

date	airline	plane#
10/23/00	Delta	k-yo-33297
10/24/00	Delta	t-up-73356
10/25/00	Delta	o-ge-98722
10/24/00	American	p-rw-84663
10/25/00	American	q-yg-98237
10/22/00	American	h-fe-65748

Bad Database Design

- information loss

FLIGHTS-AIRLINE

flt#	airline
DL242	Delta
AA121	American
AA411	American

DATE-AIRLINE-PLANE

date	airline	plane#
10/23/00	Delta	k-yo-33297
10/24/00	Delta	t-up-73356
10/25/00	Delta	o-ge-98722
10/24/00	American	p-rw-84663
10/25/00	American	q-yg-98237
10/22/00	American	h-fe-65748

FLIGHTS

flt#	date	airline	plane#
DL242	10/23/00	Delta	k-yo-33297
DL242	10/24/00	Delta	t-up-73356
DL242	10/25/00	Delta	o-ge-98722
AA121	10/24/00	American	p-rw-84663
AA121	10/25/00	American	q-yg-98237
<i>AA211</i>	<i>10/22/00</i>	<i>American</i>	<i>h-fe-65748</i>
<i>AA411</i>	<i>10/24/00</i>	<i>American</i>	<i>p-rw-84663</i>
<i>AA411</i>	<i>10/25/00</i>	<i>American</i>	<i>q-yg-98237</i>
<i>AA411</i>	<i>10/22/00</i>	<i>American</i>	<i>h-fe-65748</i>

- **information loss:** we polluted the database with false facts; we can't find the true facts.

Bad Database Design - dependency loss

FLIGHTS-AIRLINE

flt#	airline
DL242	Delta
AA121	American
AA411	American

DATE-AIRLINE-PLANE

date	airline	plane#
10/23/00	Delta	k-yo-33297
10/24/00	Delta	t-up-73356
10/25/00	Delta	o-ge-98722
10/24/00	American	p-rw-84663
10/25/00	American	q-yg-98237
10/22/00	American	h-fe-65748

- **dependency loss:** we lost the fact that (flt#, date) → plane#

Good Database Design

FLIGHTS-AIRLINE

flt#	airline
DL242	Delta
AA121	American
AA411	American

FLIGHTS-DATE-PLANE

flt#	date	plane#
DL242	10/23/00	k-yo-33297
DL242	10/24/00	t-up-73356
DL242	10/25/00	o-ge-98722
AA121	10/24/00	p-rw-84663
AA121	10/25/00	q-yg-98237
AA411	10/22/00	h-fe-65748

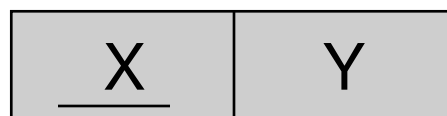
- no redundancy of ***FACT (!)***
- no inconsistency
- no insertion, deletion or update anomalies
- no information loss
- no dependency loss

Functional Dependencies and Keys

Let X and Y be sets of attributes in R

- Y is **functionally dependent** on X in R **iff** for each $x \in R.X$ there is precisely one $y \in R.Y$
- Y is **fully functional dependent** on X in R if Y is functional dependent on X and Y is not functional dependent on any proper subset of X
- We use **keys** to enforce functional dependencies in relations:

$$X \rightarrow Y$$



Functional Dependencies and Keys

FLIGHTS

flt#	date	airline	plane#
------	------	---------	--------

the FLIGHT relation will **not** allow the FDs to be enforced by keys

FLIGHTS

<u>flt#</u>	date	airline	plane#
-------------	------	---------	--------

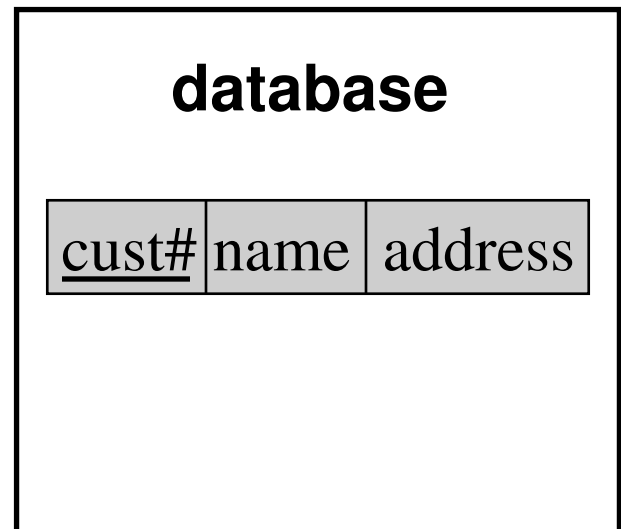
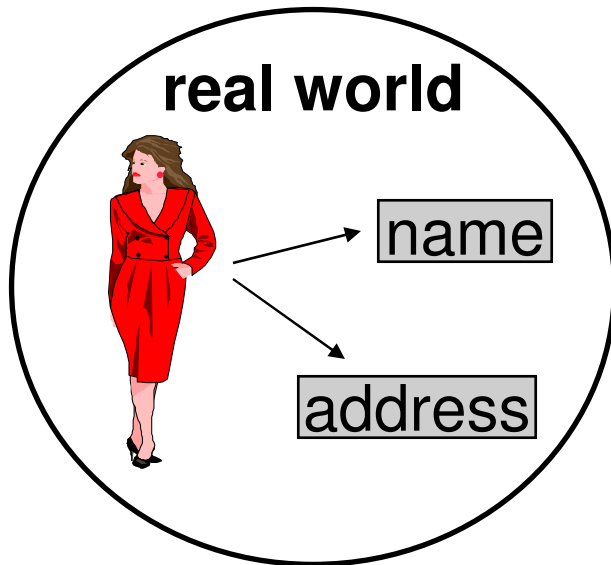
plane# is **not** determined by flt# alone

FLIGHTS

<u>flt#</u>	<u>date</u>	airline	plane#
-------------	-------------	---------	--------

airline is **not** determined by flt# and date

Functional Dependencies and Keys



Consider the meaning

<u>cust#</u>	name	address
--------------	------	---------

cust#	name	<u>address</u>
-------	------	----------------

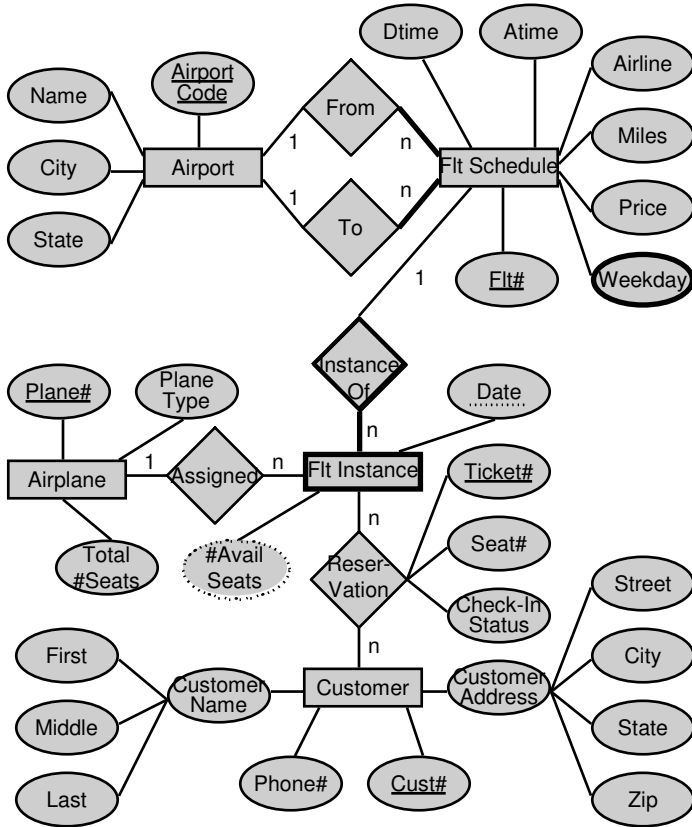
<u>cust#</u>	name	<u>address</u>
--------------	------	----------------

separate

<u>cust#</u>	name	<u>address</u>
--------------	------	----------------

combined

Functional Dependencies



Functional Dependencies in the ER-Diagram

- AIRPORT ↔ Airportcode
- FLT-SCHEDULE ↔ Flt#
- FLT-INSTANCE ↔ (Flt#, Date)
- AIRPLANE ↔ Plane#
- CUSTOMER ↔ Cust#
- RESERVATION ↔ (Cust#, Flt#, Date)
- RESERVATION ↔ Ticket#

- Airportcode → name, City, State
- Flt# → Airline, Dtime, Atime, Miles, Price, (from) Airportcode, (τ) Airportcode
- (Flt#, Date) → Flt#, Date, Plane#
- (Cust#, Flt#, Date) → Cust#, Flt#, Date, Ticket#, Seat#, CheckInStatus,
- Ticket# → Cust#, Flt#, Date
- Cust# → CustomerName, CustomerAddress, Phone#

AIRPORT

<u>airportcode</u>	name	city	state
--------------------	------	------	-------

FLT-SCHEDULE

<u>flt#</u>	airline	dtime	from-airportcode	atime	to-airportcode	miles	price
-------------	---------	-------	------------------	-------	----------------	-------	-------

FLT-WEEKDAY

<u>flt#</u>	<u>weekday</u>
-------------	----------------

FLT-INSTANCE

<u>flt#</u>	<u>date</u>	plane#	#avail-seats
-------------	-------------	--------	--------------

AIRPLANE

<u>plane#</u>	plane-type	total-seats
---------------	------------	-------------

CUSTOMER

<u>cust#</u>	first	middle	last	phone#	street	city	state	zip
--------------	-------	--------	------	--------	--------	------	-------	-----

RESERVATION

<u>flt#</u>	<u>date</u>	<u>cust#</u>	seat#	check-in-status	<u>ticket#</u>
-------------	-------------	--------------	-------	-----------------	----------------

How to Compute Meaning

- Armstrong's inference rules

Rules of the computation:

- reflexivity: if $Y \subseteq X$, then $X \rightarrow Y$
- Augmentation: if $X \rightarrow Y$, then $WX \rightarrow WY$
- Transitivity: if $X \rightarrow Y$ and $Y \rightarrow Z$, then $X \rightarrow Z$

Derived rules:

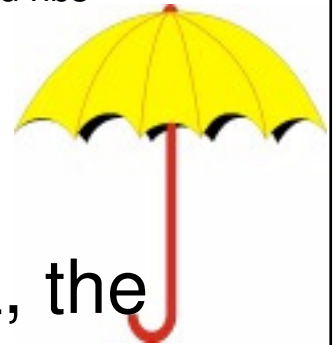
- Union: if $X \rightarrow Y$ and $X \rightarrow Z$, then $X \rightarrow YZ$
- Decomposition: if $X \rightarrow YZ$, then $X \rightarrow Y$ and $X \rightarrow Z$
- Pseudotransitivity: if $X \rightarrow Y$ and $WY \rightarrow Z$, then $XW \rightarrow Z$

Armstrong's Axioms:

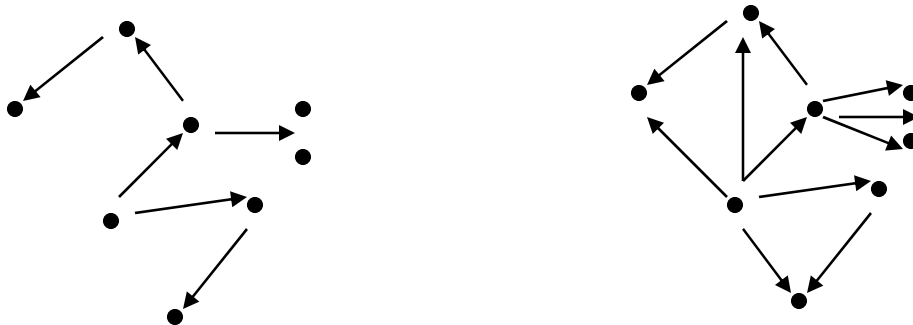
- sound
- complete

How to Compute Meaning -the meaning of a set of FDs, F^+

umbrella: a collapsible shade consisting of fabric stretched over hinged ribs radiating from a central pole



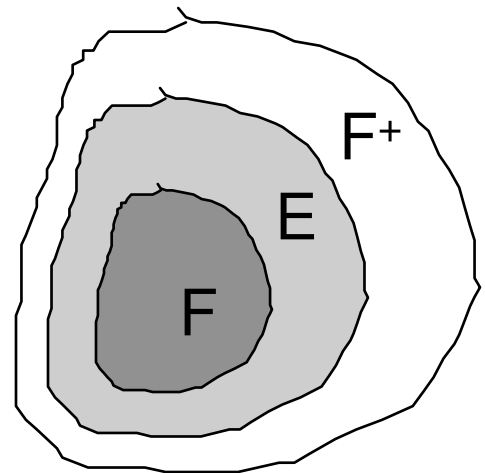
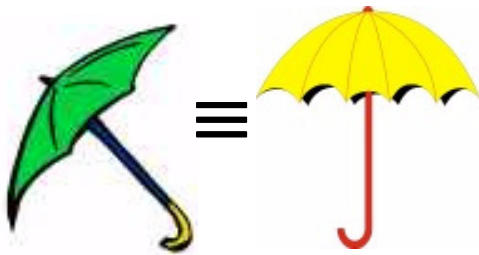
- Given the ribs of an umbrella, the FDs, what does the whole umbrella, F^+ , look like?



- Determine each set of attributes, X , that appears on a left-hand side of a FD. Determine the set, X^+ , the closure of X under F .

How to Compute Meaning when do sets of FDs mean the same?

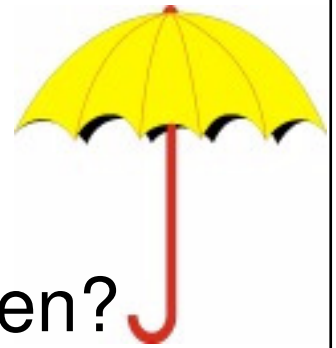
- F **covers** E if every FD in E is also in F^+



- F and E are **equivalent** if F covers E and E covers F .
- We can determine whether F covers E by calculating X^+ with respect to F for each FD, $X \rightarrow Y$ in E , and then checking whether this X^+ includes the attributes in Y . If this is the case for every FD in E , then F covers E .

How to Compute Meaning

- minimal cover of a set of FDs



- Is there a minimal set of ribs that will hold the umbrella open?

F is **minimal** if:

- every dependency in **F** has a single attribute as right-hand side
- we can't replace any dependency $X \rightarrow A$ in **F** with a dependency $Y \rightarrow A$ where $Y \subset X$ and still have a set of dependencies equivalent with **F**
- we can't remove any dependency from **F** and still have a set of dependencies equivalent with **F**

How to guarantee lossless joins

$$R_1 \bowtie R_2 = R$$

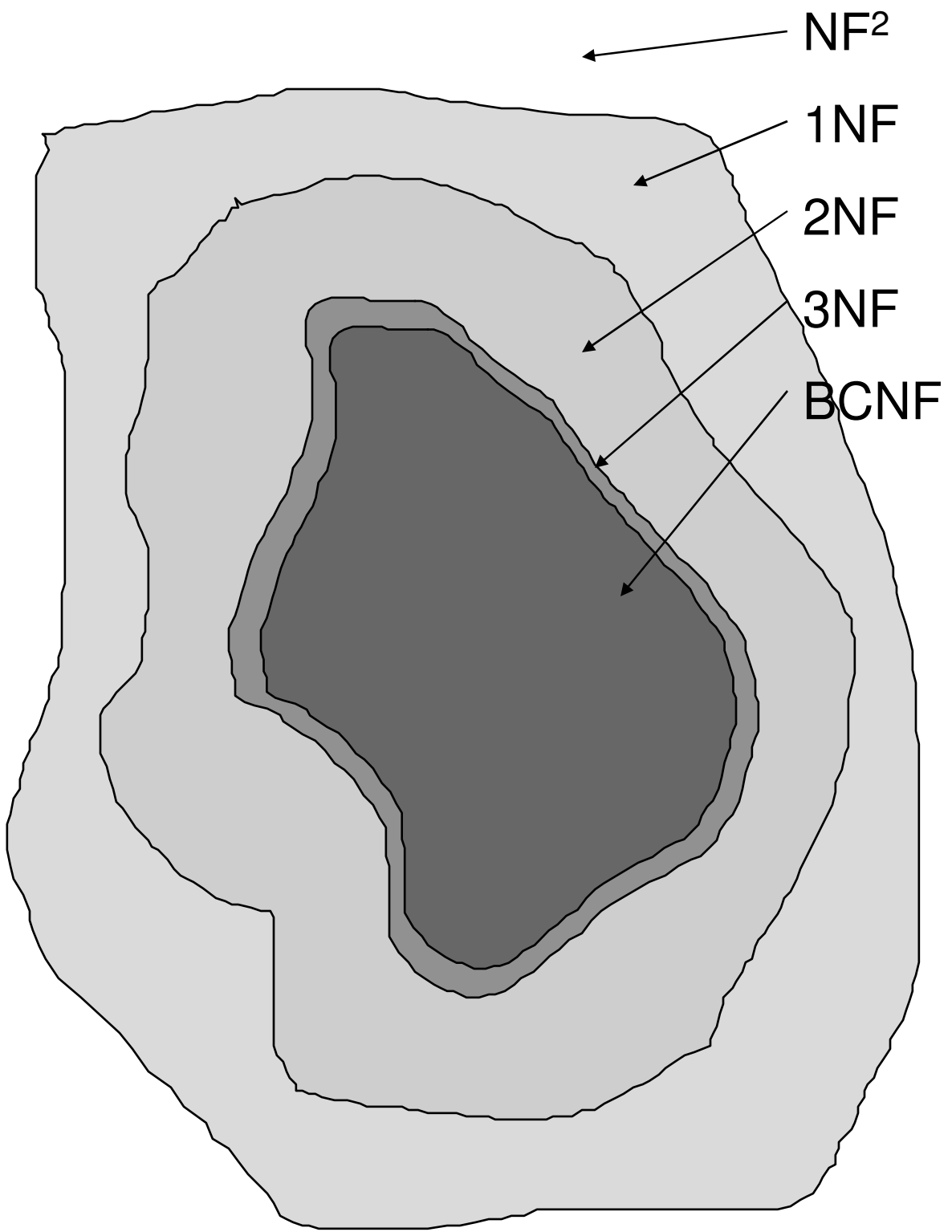
- Decompose relation, R , with functional dependencies, F , into relations, R_1 and R_2 , with attributes, A_1 and A_2 , and associated functional dependencies, F_1 and F_2 .
- The decomposition is **lossless iff**:
 - $A_1 \cap A_2 \rightarrow A_1 \setminus A_2$ is in F^+ , or
 - $A_1 \cap A_2 \rightarrow A_2 \setminus A_1$ is in F^+

How to guarantee preservation of FDs

$$F^+ = (F_1 \cup \dots \cup F_k)^+$$

- Decompose relation, R , with functional dependencies, F , into relations, R_1, \dots, R_k , with associated functional dependencies, F_1, \dots, F_k .
- The decomposition is **dependency preserving iff**:
- $F^+ = (F_1 \cup \dots \cup F_k)^+$

Overview of NFs



Normal Forms

- definitions

- **NF²**: non-first normal form
- **1NF**: R is in 1NF. **iff** all domain values are atomic²
- **2NF**: R is in 2. NF. **iff** R is in 1NF and every nonkey attribute is fully dependent on the key
- **3NF**: R is in 3NF **iff** R is 2NF and every nonkey attribute is non-transitively dependent on the key
- **BCNF**: R is in BCNF **iff** every determinant is a candidate key

- **Determinant**: an attribute on which some other attribute is fully functionally dependent.

Example of Normalization

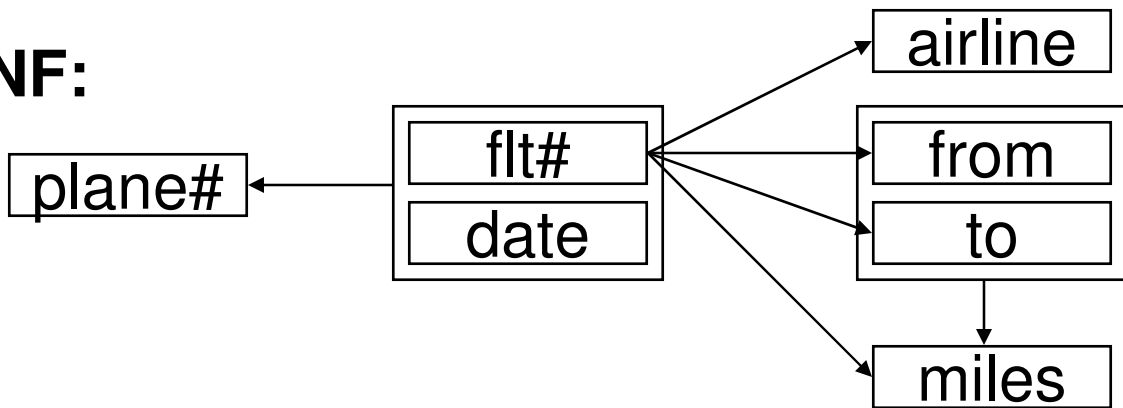
FLT-INSTANCE

flt#	date	plane#	airline	from	to	miles
------	------	--------	---------	------	----	-------

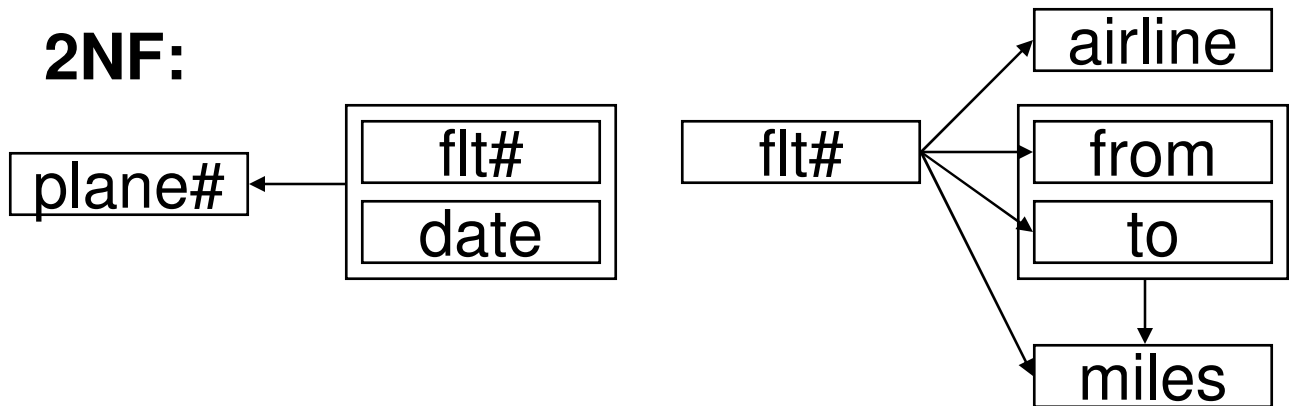


Example of Normalization

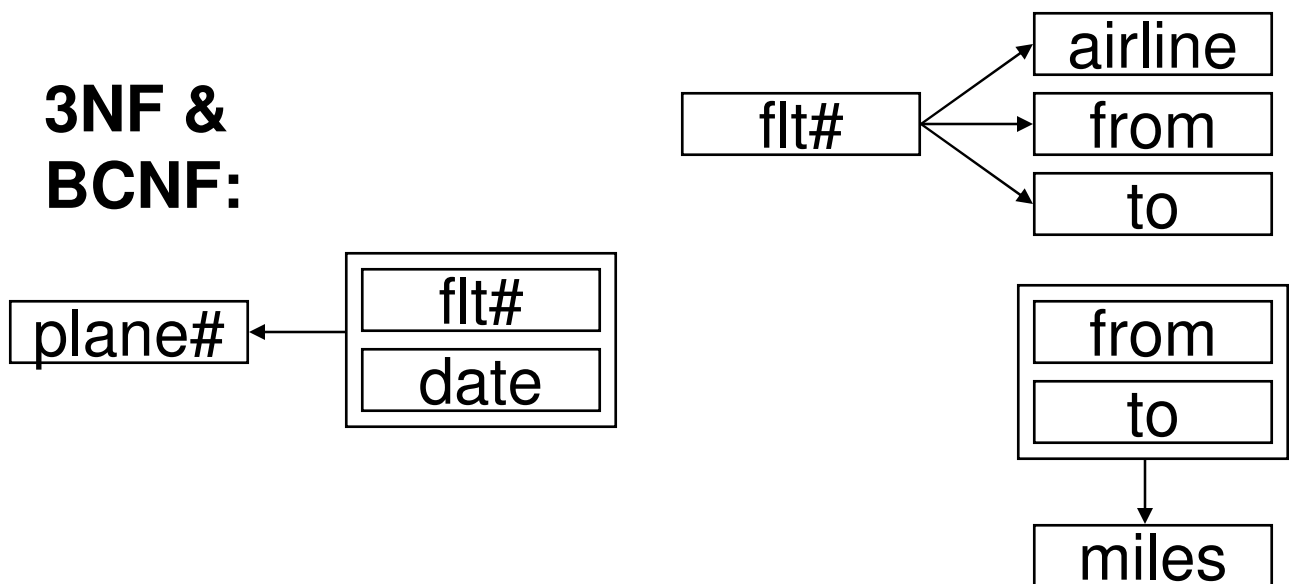
1NF:



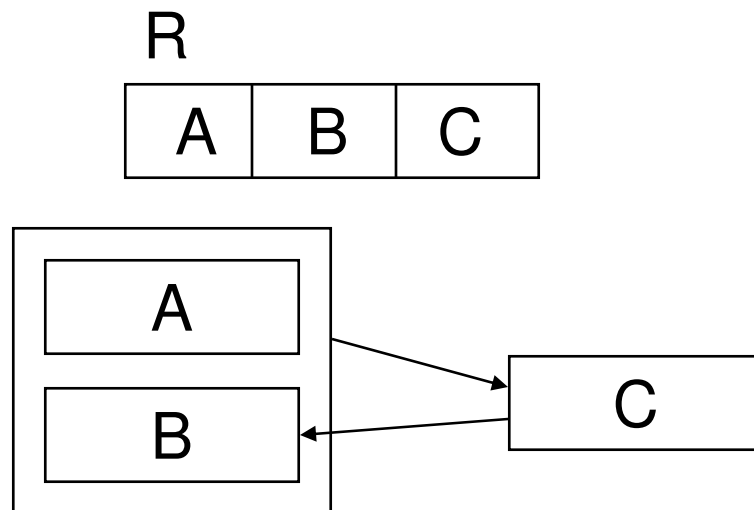
2NF:



3NF & BCNF:



3NF that is not BCNF



Candidate keys: $\{A, B\}$ and $\{A, C\}$

Determinants: $\{A, B\}$ and $\{C\}$

A decomposition:



Lossless, but **not** dependency preserving!

Major Results in Normalization Theory

Theorem:

- There is an algorithm for testing a decomposition for lossless join wrt. a set of FDs

Theorem:

- There is an algorithm for testing a decomposition for dependency preservation

Theorem:

- There is an algorithm for lossless join decomposition into BCNF

Theorem:

- There is an algorithm for dependency preserving decomposition into 3NF