Functional Dependencies

1

Roadmap

- We extracted relations/tables for our design.
- How good is our relational design?
- Can we reason about it?
- Is there any methodology that aids in obtaining a good design or correcting possible problems with our current design?
  - E.g., redundancy

2

Facts

- Commonly we know certain facts about dependencies and/or constraints existing in our application scenario:
  - You cannot have two classes on the same room and at the same time/day in the campus!
- We will formalize such dependencies / constraints and utilize them to get a 'good' design.

3

Functional Dependencies (FD's)

$X \rightarrow A$ = assertion about a relation R that whenever two tuples agree on all the attributes of X, then they must also agree on attribute A.

4

Example

Movies(*title*, *year*, *length*, *film-type*, *studio-Name*, *star-Name*)

title	*year*	*length*	*film-type*	*studio-Name*	*star-Name*
Star Wars	1977	124	color	Fox	Carrie Fisher
Star Wars	1977	124	color	Fox	Mark Hamill
Star Wars	2002	142	color	Fox	Natalie Portman
Wayne's	1992	104	color	Paramount	Dana Carvey
Wayne's	1992	104	color	Paramount	Mike Meyers

Reasonable FD's to assert:

- *title*, *year* → *length*
- *title*, *year* → *film-type*
- *title*, *year* → *studio-Name*

What about the FD: *title*, *year* → *star-Name* !?!??

5

Functional Dependencies (cont.)

- Shorthand: combine FD's with common left side by concatenating their right sides.
  Example:
  Given *title*, *year* → *length* and *title*, *year* → *film-type*
  we can have *title*, *year* → *length, film-type*
- Sometimes, several attributes jointly determine another attribute, although neither does by itself.
  Example:
  We know that *title*, *year* → *length* holds…
  It is easy to see that *title* → *length* does not hold
- Rules (combining/splitting rules):
  - A1A2 -> B1, A1A2 -> B2, A1A2 -> B3 <=> A1A2 -> B1 B2 B3

6

Functional Depend. (cont')

- Assume A1A2..An -> B1B2..Bm
  - Trivial if the B's are a subset of the A's
  - Nontrivial if at least one of the B's is not among the A's.
  - Completely non trivial if none of the B's is also one of the A's.
- You can remove from the right side all attributes that appear on the left:
  - Trivial dependency rule

7

Keys of Relations

Define key *K* for relation *R* w.r.t. FD's. Set *K* of attrs is key if:

1. $K \rightarrow$ all attributes of *R*. **(Uniqueness)**
2. For no proper subset of *K* is (1) true. **(Minimality)**
   If *K* at least satisfies (1), then *K* is a *superkey*.
   AB -> CDE
   A -> BCDE

A few words about notation…

- Pick one key; underline key attributes in the relation schema.
- *A, B, C* etc., represent single attributes; *X*, *Y, Z* etc., represent sets of attributes
- sometimes we use *ABC* instead of {*A*, *B*, *C*}

8

Example

Movies(*title*, *year*, *length*, *film-type*, *studio-Name*, *star-Name*)

- {*title*, *year*, *star-Name*} $\rightarrow$ all attributes. Thus, it 's a superkey.
- *title* $\rightarrow$ *year* is false, so *title* is not a superkey.
- *star-Name* $\rightarrow$ *title* also false, so *star-Name* not a superkey.
- Thus, {*title*, *year*, *star-Name*} is a key.
- No other keys in this example.

9

Who Determines Keys/FD's?

- We could assert a key *K*.
  - Then the only FD's asserted are that $K \rightarrow A$ for every attribute *A*.
  - No surprise: *K* is then the only key for those FD's, according to the formal definition of "key."
- Or, we could assert some FD's and *deduce* one or more keys by the formal definition.
  - E/R diagram implies FD's by key declarations and many-one relationship declarations.
- Rule of thumb: FD's either come from keys, many-1 relationships, or from known facts.
  - *E.g.*, "no two courses can meet in the same room at the same time" yields *rooms, times* $\rightarrow$ *courses.*

10

Functional Dependencies and Many-One Relationships

- Consider $R(A_1,\ldots, A_n)$ and *X* is a key
  Then $X \rightarrow Y$ for any attributes *Y* in $A_1,\ldots, A_n$
  even if they overlap with X
- Suppose *R* is used to represent a many $\rightarrow$ one relationship:
  *E*1 entity set $\rightarrow$ *E*2 entity set
  where *X* key for *E*1, *Y* key for *E*2,
  Then, $X \rightarrow Y$ holds,
  And $Y \rightarrow X$ does not hold unless the relationship is one-one.

11

Inferring FD's

When we talk about improving relational designs, we often need to ask:

"does this FD hold in this relation?"

More formally, given FD's

$$X_1 \rightarrow A_1, X_2 \rightarrow A_2,\ldots, X_n \rightarrow A_n,$$

does FD $Y \rightarrow B$ necessarily hold in the same relation?
Try the following: Start by assuming two tuples agree in *Y*.
Use given FD's to infer other attributes on which they must agree. If *B* is among them, then yes, else no.

12

Example

Consider $R(A, B, C)$ and FD's $A \rightarrow B$ and $B \rightarrow C$
What about: $A \rightarrow C$?

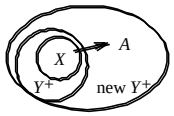
Consider two tuples (a, b_1, c_1), (a, b_2, c_2)
Then, due to $A \rightarrow B$, we have (a, b, c_1), (a, b, c_2)
Since $B \rightarrow C$ we have (a, b, c), (a, b, c)
Thus, $A \rightarrow C$

13

Algorithm

The *closure* Y^+ of attribute set Y is the set of attributes functionally determined by Y

- Basis: $Y^+ := Y$.
- Induction: If $X \subseteq Y^+$, and $X \rightarrow A$ is a given FD, then add A to Y^+.



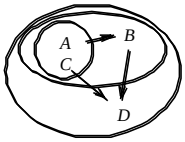
- Terminate: when Y^+ cannot be changed.

14

Example

Initial set of FDs: $A \rightarrow B$, $BC \rightarrow D$.

- $A^+ = AB$.
- $C^+ = C$.
- $(AC)^+ = ABCD$.



15

Another Example

- Set of FD's in *ABCGHI*:

  $A \rightarrow B$
  $A \rightarrow C$
  $CG \rightarrow H$
  $CG \rightarrow I$
  $B \rightarrow H$

- Compute $(CG)^+$, $(BG)^+$, $(AG)^+$
  - $(CG)^+ = CGHI$
  - $(BG)^+ = BGH$
  - $(AG)^+ = ABCGHI$
- How can I use this algorithm to determine whether a set of attributes forms a key? What is the complexity?

16

Armstrong's axioms

- Armstrong's Axioms:
  - *Reflexivity:* If $Y \subseteq X$, then $X \rightarrow Y$ (trivial FD's)
  - *Augmentation:* If $X \rightarrow Y$, then $XZ \rightarrow YZ$ for any Z
  - *Transitivity:* If $X \rightarrow Y$ and $Y \rightarrow Z$, then $X \rightarrow Z$
- These are *sound* and *complete* inference rules for FDs!

Example:
Initial set of FDs: $A \rightarrow B$, $BC \rightarrow D$. Show that $AC \rightarrow D$
Solution:
$A \rightarrow B$, then $AC \rightarrow BC$ (Augmentation).
$AC \rightarrow BC$ and $BC \rightarrow D$, then $AC \rightarrow D$ (Transitivity)

17

Given Versus Implied FD's

Typically, we state a few FD's that are known to hold for a relation R.

- Other FD's may follow logically from the given FD's; these are *implied* FD's.
- We are free to choose any *basis* for the FD's of R – a set of FD's that imply all the FD's that hold for R.
- Minimal basis (no subset a basis)

18

Projecting FD's

Motivation: Suppose we have a relation *ABCD* with some FD's *F*. If we decide to decompose *ABCD* into *ABC* and *AD*, what are the FD's for *ABC*, *AD*?

Closure of a set of functional dependencies F.

- Example: $F = AB \rightarrow C$, $C \rightarrow D$, $D \rightarrow A$. It looks like just $AB \rightarrow C$ holds in *ABC*, but in fact $C \rightarrow A$ follows from *F* and applies to relation *ABC*.
- Get closure for each subset of ABC wrt F
- Problem is exponential in worst case.

19

Algorithm

Assume relation *R,* set *F* of FD's and set $S \subseteq$ R

- Compute X^+ for each $X \subseteq S$
  - But skip the cases $X = \emptyset$, X = all attributes.
  - Add $X \rightarrow A$ for each A in $X^+ - X$.
- If X^+ is all attributes, then there is no point in computing closure of supersets of *X*.
- Finally, project the FD's by selecting only those FD's that involve only attributes of *S*

20

Example

$F = AB \rightarrow C$, $C \rightarrow D$, $D \rightarrow A$. What FD's follow from F?

- $A^+ = A$; $B^+ = B$ (nothing).
- $C^+ = ACD$ (add $C \rightarrow A$).
- $D^+ = AD$ (nothing new).
- $(AB)^+ = ABCD$ (add $AB \rightarrow D$; skip all supersets of *AB*).
- $(BC)^+ = ABCD$ (nothing new; skip all supersets of *BC*).
- $(BD)^+ = ABCD$ (add $BD \rightarrow C$; skip all supersets of *BD*).
- $(AC)^+ = ACD$; $(AD)^+ = AD$; $(CD)^+ = ACD$ (nothing new).
- $(ACD)^+ = ACD$ (nothing new).
- All other sets contain *AB*, *BC*, or *BD*, so skip.
- Thus, the only interesting FD's that follow from *F* are: $C \rightarrow A$, $AB \rightarrow D$, $BD \rightarrow C$.

21

Another Example

In *ABC* with FD's $A \rightarrow B$, $B \rightarrow C$, project onto *AC*

- $A^+ = ABC$; yields $A \rightarrow B$, $A \rightarrow C$.
- $B^+ = BC$; yields $B \rightarrow C$.
- $C^+ = C$ and $BC^+ = BC$; adds nothing.
- Resulting FD's: $A \rightarrow B$, $A \rightarrow C$, $B \rightarrow C$.
- Projection onto *AC*: $A \rightarrow C$.

22

Design Anomalies and Normalization

23

Design Anomalies

Movies(*title*, *year*, *length*, *film-type*, *studio-Name*, *star-Name*)

title	*year*	*length*	*film-type*	*studio-Name*	*star-Name*
Star Wars	1977	124	color	???	Carrie Fisher
Star Wars	1977	???	color	Fox	Mark Hamill
Star Wars	1977	124	???	Fox	Harrison Ford
Wayne's	1992	104	color	Paramount	Mike Meyers

- *title*, *year* → *length*
- *title*, *year* → *film-type*
- *title*, *year* → *studio-Name*

- ???'s are redundant, since we can figure them out from the FD's.
- Update anomalies: If Star Wars is actually 125 minutes long, will we change *length* in each of its tuples?
- Deletion anomalies: If we delete Mike Meyers, we lose the info for Wayne's World.

24

Eliminate Anomalies

- Decomposition
  - Intuitively break the schema into two or more pieces.
  - Premise
    - anomalies do not exist anymore
    - We can recover the original information by manipulating the pieces.
      - Example: R(A,B,C,D) = {(a1 b1 c1 d1),(a1 b1 c2 d2) (a1 b2 c1 d3)} decomposed to R1(A,B) and R2(C,D)
      - R1(A,B) = {(a1 b1),(a1,b2)} R2(C,D) = {(c1,d1),(c2,d2),(c1,d3)}

25

Decomposition of a Relation Scheme

- Suppose that relation R contains attributes $A_1 \ldots A_n$
  A *decomposition* of R consists of replacing R by two (or more) relations such that:
  - Each new relation scheme contains a subset of the attributes of R (and no attributes that do not appear in R), and
  - Every attribute of R appears as an attribute in at least one of the new relations.
- Intuitively, decomposing R means we will store instances of the relation schemes produced by the decomposition, instead of instances of R.

26

Decomposition cont.

- **DEFINITION:** Given a set of attributes A and a set FD of functional dependencies, a **decomposition** of (A, FD) is a set $(A_1,FD_1), \ldots, (A_n,FD_n)$ where $A_i \subseteq A$, $A = \cup A_i$ and Fd_i is a set of functional dependencies over attributes A_i
- **DEFINITION:** A decomposition of A and FD is **lossless** if for every relation R over A that satisfies all functional dependencies in FD : - each $\pi_{Ai}(R)$ satisfies Fd_i
  - $R = \pi_{A1}(R) \bowtie \ldots \bowtie \pi_{An}(R)$.
- DEFINITION: A decomposition of a and FD is **dependency preserving** if $F^* = \bigcup_i F_i$

27

Normal Forms

- Given relation R, the first question to ask is whether any refinement/decomposition is needed!
- If a relation is in a certain *normal form* (BCNF, 3NF etc.), it is known that certain kinds of problems are avoided/minimized. This can be used to help us decide whether decomposing the relation will help.
- Role of FD's in detecting redundancy:
  - Consider a relation R with 3 attributes, ABC.
    - No FD's hold: There is no redundancy here.
    - Given A $\rightarrow$ B: Several tuples could have the same A value, and if so, they'll all have the same B value!

28

Boyce-Codd Normal Form (BCNF)

Formally, *R* is in BCNF if for every nontrivial FD for *R*, say $X \rightarrow A$, then *X* is a superkey.

Advantages of BCNF:
1. Guarantees no redundancy due to FD's.
2. Guarantees no update anomalies
3. Guarantees no deletion anomalies

29

Example

Movies(*title*, *year*, *length*, *film-type*, *studio-Name*, *star-Name*)

title	*year*	*length*	*film-type*	*studio-Name*	*star-Name*
Star Wars	1977	124	color	Fox	Carrie Fisher
Star Wars	1977	124	color	Fox	Mark Hamill
Star Wars	1977	124	color	Fox	Harrison Ford
Wayne's	1992	104	color	Paramount	Mike Meyers

Functional dependencies:
- *title*, *year* $\rightarrow$ *length*
- *title*, *year* $\rightarrow$ *film-type*
- *title*, *year* $\rightarrow$ *studio-Name*

Each of the given FD's is a BCNF violation… Why?
Remember that the key is {*title*, *year*, *star-Name*}

30

Decomposition to Reach BCNF

The setting: relation R, set F of FD's.

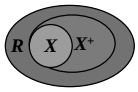
Suppose relation R has BCNF violation $X \rightarrow B$

- We need only look among FD's of F for a BCNF violation, not those that follow from F.
- Proof: If $Y \rightarrow A$ is a BCNF violation and follows from F, then the computation of Y^+ used at least one FD $X \rightarrow B$ from F.
  - X must be a subset of Y.
  - Thus, if Y is not a superkey, X cannot be a superkey either, and $X \rightarrow B$ is also a BCNF violation.

31

Decomposition to Reach BCNF (cont.)

1. Compute X^+.
   - Cannot be all attributes – why?
2. Decompose R into X^+ and $(R–X^+) \cup X$.



3. Find the FD's for the decomposed relations. Project the FD's from F calculate all consequents of F that involve only attributes from X^+ or only from $(R–X^+) \cup X$.

32

Example

R = *Movies*(*title*, *year*, *length*, *film-type*, *studio-Name*, *star-Name*)
F =

- *title*, *year* → *length*
- *title*, *year* → *film-type*
- *title*, *year* → *studio-Name*

Pick BCNF violation, e.g.,
title, *year* → *length, film-type, studio-Name*

- Find the closure
  (*title*, *year*)$^+$… = {*title*, *year* , *length, film-type, studio-Name}*
- Create decomposed relations:
  Movies$_1$(*title*, *year*, *length, film-type, studio-Name)*
  Movies$_2$(*title, year*, *star-Name*)

33

Another Example

R = Movies(*title*, *year*, *studio-Name*, *president, presAddr*)
F =

- *title, year* → *studio-Name*
- *studio-Name* → *president*
- *president* → *presAddr*

Pick BCNF violation, e.g., *studio-Name* → *president*

- Find the closure (*studio-Name*)$^+$… = {*studio-Name, president, presAddr}*
- Create decomposed relations:
  Movies$_1$(*title*, *year*, *studio-Name)*
  Movies$_2$(*studio-Name*, *president, presAddr*)
- Projected FD's (skipping a lot of work that leads nowhere interesting):
  - For *Movies*$_1$: *title*, *year* → *studio-Name*
  - For *Movies*$_2$: *studio-Name* → *president* and *president* → *presAddr*

34

Another Example (cont.)

(Continuing…)

- Decomposed relations:
  Movies$_1$(*title*, *year*, *studio-Name)*
  Movies$_2$(*studio-Name*, *president, presAddr*)
  Projected FD's:
  - For *Movies*$_1$: *title*, *year* → *studio-Name*
  - For *Movies*$_2$: *studio-Name* → *president* and *president* → *presAddr*

- BCNF violations?
  - For *Movies*$_1$, {*title, year*} is key and all left sides of FD's are superkeys.
  - For *Movies*$_2$, {*studio-Name*} is the key… but *president* → *presAddr* violates BCNF.

35

Decomposing *Movies*$_{2…}$

- First set of decomposed relations:
  Movies$_1$(*title*, *year*, *studio-Name)*
  Movies$_2$(*studio-Name*, *president, presAddr*)
- Closure *president* $^+$… = {*president*, *presAddr*}
- Decompose *Movies*$_2$ into:
  Movies$_{2\text{-}1}$(*studio-Name*, *president*)
  Movies$_{2\text{-}2}$(*president*, *presAddr*)
- Resulting relations are all in BCNF:
  Movies$_1$(*title*, *year*, *studio-Name)*
  Movies$_{2\text{-}1}$(*studio-Name*, *president*)
  Movies$_{2\text{-}2}$(*president*, *presAddr*)

36

Issues with BCNF

BCNF: no anomalies, can always recover information.
BCNF is not, in general, dependency preserving

- If we decompose, you can't check all the FD's in the decomposed relations only.
- If we don't decompose, we violate BCNF.

Abstractly: $R(A, B, C)$, FD: $AB \rightarrow C$ and $C \rightarrow B$.

- Example 1: *title*, *city* $\rightarrow$ *theater* and *theater* $\rightarrow$ *city*
- Example 2: *street*, *city* $\rightarrow$ *zip* and *zip* $\rightarrow$ *city*

Keys: {*A*, *B*} and {*A*, *C*}, but $C \rightarrow B$ has a left side that is not a superkey. Suggests decomposition into *BC* and *AC*.

But you can't check the FD $AB \rightarrow C$ in only these relations.

37

Example

A = *title*, *B* = *city*, *C* = *theater*

theater	*city*
Guild	Menlo Park
Park	Menlo Park

theater	*title*
Guild	The Net
Park	The Net

theater $\rightarrow$ *city*

Only trivial FD's hold

Join:

city	*title*	*theater*
Menlo Park	The Net	Guild
Menlo Park	The Net	Park

city, title $\rightarrow$ *theater* does not hold

38

Dependency Preserving Decomposition

- Consider *CSJDPQV*, *C* is key, *JP* $\rightarrow$ *C* and *SD* $\rightarrow$ *P*
  - BCNF decomposition: *CSJDQV* and *SDP*
- Dependency preserving decomposition (Intuitive):If *R* is decomposed into *X*, *Y* and *Z*, and we enforce the FD's that hold on *X*, on *Y* and on *Z*, then all FD's that were given to hold on *R* must also hold.
- The projection of a set of FD's *F*: If *R* is decomposed into *X*, ... projection of *F* onto *X* (denoted F_X) is the set of FD's $U \rightarrow V$ in F^+ (closure of *F*) such that *U*, *V* are in *X*.

39

Dependency Preserving Decomposition (Contd.)

- Decomposition of *R* into *X* and *Y* is dependency preserving if $(F_X \cup F_Y)^+ = F^+$
  - i.e., if we consider only dependencies in the closure F^+ that can be checked in X without considering Y, and in Y without considering X, these imply all dependencies in F^+.
- Important to consider F^+, not *F*, in this definition:
  - ABC, A $\rightarrow$ B, B $\rightarrow$ C, C $\rightarrow$ A, decomposed into AB and BC.
  - Is this dependency preserving? Is C $\rightarrow$ A preserved !?!
- For BCNF we can check if decomposition is dependency preserving (how?)
- Faster algorithm exists (not part of this course).

40

"Elegant" Workaround

Define the problem away.

- A relation *R* is in 3NF iff (if and only if) for every nontrivial FD $X \rightarrow A$, either:
  1. *X* is a superkey, or
  2. *A* is *prime* = member of at least one key.
- Thus, the canonical problem goes away: you don't have to decompose because all attributes are prime.

41

What 3NF Gives You

There are two important properties of a decomposition:

- We should be able to recover from the decomposed relations the data of the original.
  - Recovery involves projection and join, which we shall defer until we've discussed relational algebra.
- We should be able to check that the FD's for the original relation are satisfied by checking the projections of those FD's in the decomposed relations.
- Without proof, we assert that it is always possible to decompose into BCNF and satisfy (1).
- Also without proof, we can decompose into 3NF and satisfy both (1) and (2).
- But it is not always possible to decompose into BNCF and get both (1) and (2).
  - title-city-theater is an example of this point.

42

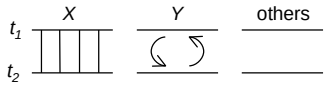
3NF Decomposition Algorithm

- Let F be a minimal cover
  i:=0
  for each FD A -->B in F
  { if none of the schemas Rj, 1<= j <= i, contains A -->B
  { increment i;
  Ri := (A, B); }}
  if no schema Rj, 1<= j <= i contains a candidate key for R
  { increment i;
  Ri := any candidate key for R}
  return (R1, ..., Ri)

43

Multivalued Dependencies

The *multivalued dependency* $X \rightarrow\rightarrow Y$ holds in a relation R if whenever we have two tuples of R that agree in all the attributes of X, then we can swap their Y components and get two new tuples that are also in R.



44

Example

Consider *Stars-in*(*name*, *street*, *city*, *title*, *year*) with MVD *name* $\rightarrow\rightarrow$ *street*, *city*. If *Stars-in* has the two tuples:

name	*street*	*city*	*title*	*year*
Fisher	123 Maple St.	Toronto	Episode IV	1977
Fisher	5 Laurier St.	Ottawa	Episode V	1980

it must also have the same tuples with *street*, *city* swapped:

name	*street*	*city*	*title*	*year*
Fisher	5 Laurier St.	Ottawa	Episode IV	1977
Fisher	123 Maple St.	Toronto	Episode V	1980

Note 1: we must check this condition for *all* pairs of tuples that agree on name, not just one pair

Note 2: *Stars-in* is in BCNF !!!!!!!!!!

45

(New) MVD Rules

1. Every FD is an MVD.
   - Because if $X \rightarrow Y$, then swapping Y's between tuples that agree on X doesn't create new tuples.
   - Example, in *Stars-in*: *name, street, city, title, year* $\rightarrow\rightarrow$ *year*.
2. *Complementation*
   Given relation $R(X, Y, Z, W)$
   if $X \rightarrow\rightarrow Y$, then $X \rightarrow\rightarrow ZW$,
   - Example in *Stars-in*: since *name* $\rightarrow\rightarrow$ *street, city* holds, so does *name* $\rightarrow\rightarrow$ *title, year*.

46

Splitting Doesn't Hold

Sometimes you need to have several attributes on the right side of an MVD. For example:

name	*street*	*city*	*title*	*year*
Fisher	123 Maple St.	Toronto	Episode IV	1977
Fisher	5 Laurier St.	Ottawa	Episode V	1980
Fisher	5 Laurier St.	Ottawa	Episode IV	1977
Fisher	123 Maple St.	Toronto	Episode V	1980

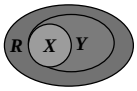
We know that *name* $\rightarrow\rightarrow$ *street, city* holds…
… but neither *name* $\rightarrow\rightarrow$ *street* nor *name* $\rightarrow\rightarrow$ *city* do.

47

4NF

Eliminate redundancy due to multiplicative effect of MVD's.

- Roughly: treat MVD's as FD's for decomposition… … but not for key discovery.
- Formally: R is in 4NF if whenever MVD $X \rightarrow\rightarrow Y$ is *nontrivial* (Y is not a subset of X, and X *union* Y is not all attributes), then X is a superkey.
  - Remember, $X \rightarrow Y$ implies $X \rightarrow\rightarrow Y$, so 4NF is more stringent than BCNF.
- Decompose R, using 4NF violation $X \rightarrow\rightarrow Y$, into XY and X *union* (R—Y).



48

4NF Decomposition Algorithm

- result :={R} ;
  done := false;
  compute MVD^+;
  while (not done) do
  if (there is a schema R_i in result that is not in 4NF)
  then begin
  let a x -->-> y be a nontrivial multivalued dependency that holds on R_i such that x --> R_i is not in D^+, and $x \cap y = \emptyset$;
  result = (result - R_i) U (R_i - y) U (x, y)
  end
  else done = true;

49

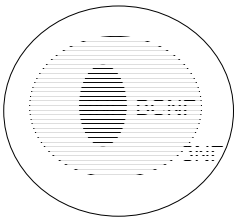
Example

Consider *Stars-in*(*name*, *street*, *city*, *title*, *year*)

- Nontrivial MVD's: *name* $\rightarrow\rightarrow$ *street, city* and *name* $\rightarrow\rightarrow$ *title, year*.
- Only key: {*name*, *street, city, title*, *year*}
- Both dependencies above violate 4NF.
- Successive decomposition yields 4NF relations:
  Stars-in$_1$(*name*, *street, city*)
  Stars-in$_2$(*name*, *title, year*)

50

Relationships Among Normal Forms



51