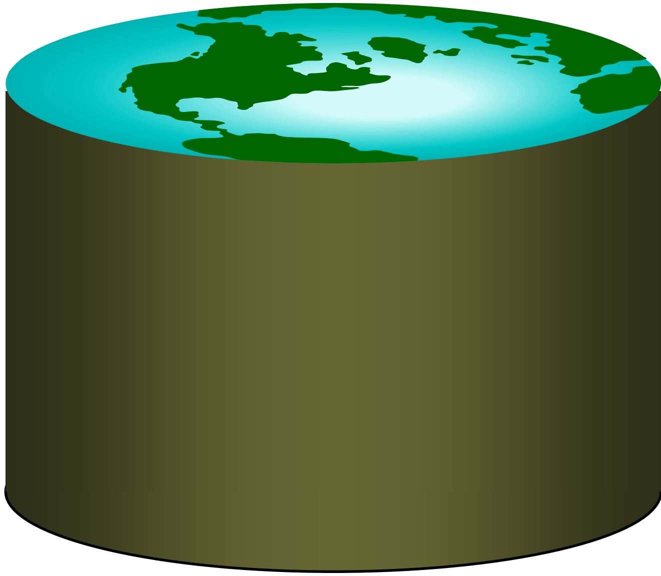
Functional Dependencies

R&G Chapter 19

Science is the knowledge of consequences, and dependence of one fact upon another.

Thomas Hobbes (1588-1679)

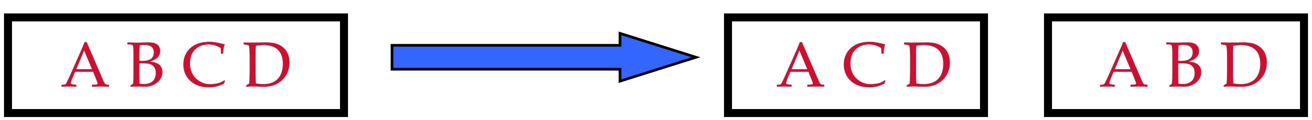


Review: Database Design

- **Requirements Analysis**
  - user needs; what must database do?
- **Conceptual Design**
  - high level descr (often done w/ER model)
- **Logical Design**
  - translate ER into DBMS data model
- **Schema Refinement**
  - consistency,normalization
- **Physical Design - indexes, disk layout**
- **Security Design - who accesses what**

The Evils of Redundancy

- ***Redundancy:*** **root of several problems with relational schemas:**
  - redundant storage, *insert/delete/update anomalies*
- ***Functional dependencies:***
  - *integrity constraints* that can identify redundancy and suggest refinements.
- **Main refinement technique:** ***decomposition***
  - replacing ABCD with, say, AB and BCD, or ACD and ABD.



- **Decomposition should be used judiciously:**
  - Is there reason to decompose a relation?
  - What problems (if any) does the decomposition cause?

Functional Dependencies (FDs)

- **A <u>functional dependency</u> X $\rightarrow$ Y holds over relation schema R if, for every allowable instance *r* of R:**

$$t1 \in r,\ \ t2 \in r,\ \ \pi_X(t1) = \pi_X(t2)$$

$$\text{implies} \quad \pi_Y(t1) = \pi_Y(t2)$$

(where t1 and t2 are tuples;X and Y are sets of attributes)

- **Explanation:**
  - X $\rightarrow$ Y means:

    If for 2 tuples X is the same, then Y must also be the same.

CAUTION: The opposite is not true.

- **Read "$\rightarrow$" as "determines"**

FD's Continued

- **An FD is a statement about *all* allowable relations.**
  - Identified based on semantics, NOT instances
  - Given an instance of R, we can disprove a FD, but we cannot verify the validity of a FD.

- **Question: Are FDs related to keys?**
- **if "K $\rightarrow$ all attributes of R" then K is a *superkey* for R**

  (does not require K to be *minimal*.)
- **FDs are a generalization of keys.**

Example: Constraints on Entity Set

- **Consider relation obtained from** Hourly_Emps:

  Hourly_Emps (<u>*ssn*</u>, *name*, *lot*, *rating*, *wage_per_hr*, *hrs_per_wk*)

- We sometimes denote a relation schema by listing the attributes: e.g., SNLRWH

- This is really the *set* of attributes {S,N,L,R,W,H}.

What are some FDs on Hourly_Emps?

ssn is the key: $S \rightarrow SNLRWH$

rating determines *wage_per_hr*: $R \rightarrow W$

lot determines *lot*: $L \rightarrow L$ ("trivial" dependency)

Problems Due to R → W

Hourly_Emps

S	N	L	R	W	H
123-22-3666	Attishoo	48	8	10	40
231-31-5368	Smiley	22	8	10	30
131-24-3650	Smethurst	35	5	7	30
434-26-3751	Guldu	35	5	7	32
612-67-4134	Madayan	35	8	10	40

R=6, W=?

- ***Update anomaly***: Should we be allowed to modify W in only the 1st tuple of SNLRWH?
- ***Insertion anomaly***: What if we want to insert an employee and don't know the hourly wage for his or her rating? (or we get it wrong?)
- ***Deletion anomaly***: If we delete all employees with rating 5, we lose the information about the wage for rating 5!

Detecting Reduncancy

Hourly_Emps

S	N	L	R	W	H
123-22-3666	Attishoo	48	8	10	40
231-31-5368	Smiley	22	8	10	30
131-24-3650	Smethurst	35	5	7	30
434-26-3751	Guldu	35	5	7	32
612-67-4134	Madayan	35	8	10	40

Q: Why was R $\rightarrow$ W problematic, but S$\rightarrow$W not?

Decomposing a Relation

- **Redundancy can be removed by "chopping" the relation into pieces (vertically!)**
- **FD's are used to drive this process.**

  R $\rightarrow$ W is causing the problems, so decompose SNLRWH into what relations?

S	N	L	R	H
123-22-3666	Attishoo	48	8	40
231-31-5368	Smiley	22	8	30
131-24-3650	Smethurst	35	5	30
434-26-3751	Guldu	35	5	32
612-67-4134	Madayan	35	8	40

Hourly_Emps2

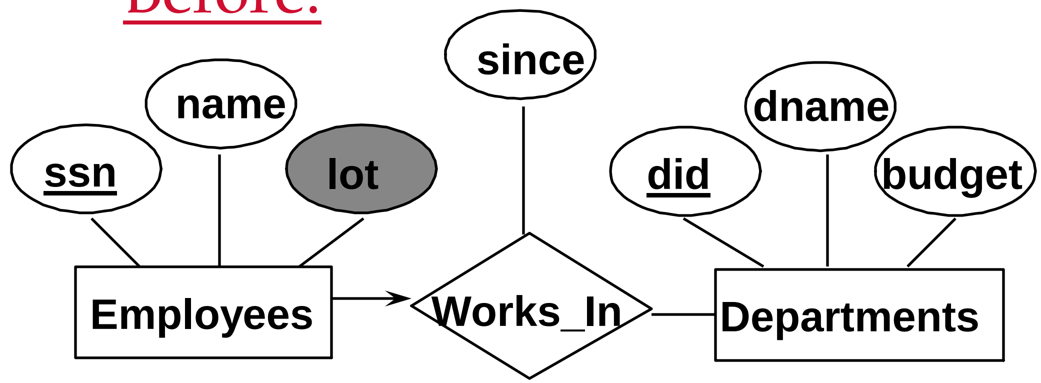
R	W
8	10
5	7

Wages

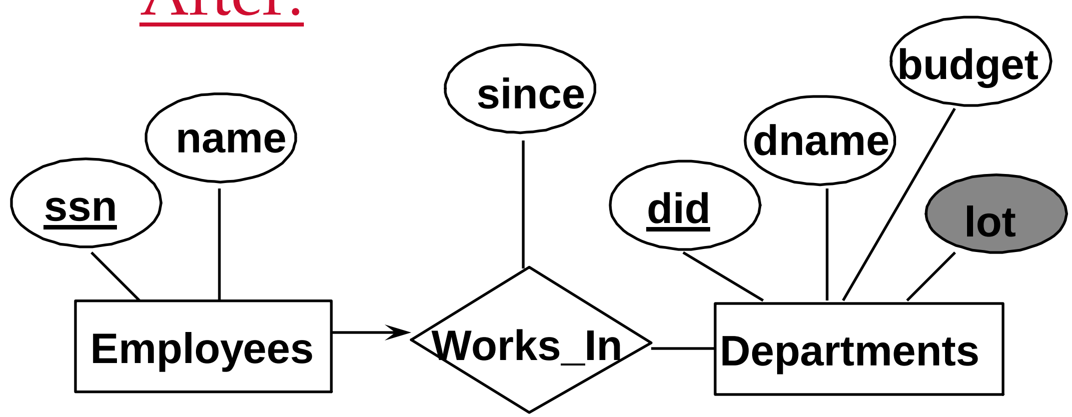
Refining an ER Diagram

- **1st diagram becomes:**
  Workers(S,N,L,D,Si)
  Departments(D,M,B)
  - Lots associated with workers.
- **Suppose all workers in a dept are assigned the same lot: D $\rightarrow$ L**
- **Redundancy; fixed by:**
  Workers2(S,N,D,Si)
  Dept_Lots(D,L)
  Departments(D,M,B)
- **Can fine-tune this:**
  Workers2(S,N,D,Si)
  Departments(D,M,B,L)

Before:



After:



Reasoning About FDs

- **Given some FDs, we can usually infer additional FDs:**

  title $\rightarrow$ *studio, star* implies *title* $\rightarrow$ *studio* and *title* $\rightarrow$ *star*

  title $\rightarrow$ *studio* and *title* $\rightarrow$ *star* implies *title* $\rightarrow$ *studio, star*

  title $\rightarrow$ *studio*, *studio* $\rightarrow$ *star* implies *title* $\rightarrow$ *star*

  But,

  title, star $\rightarrow$ *studio* does NOT necessarily imply that *title* $\rightarrow$ *studio* or that *star* $\rightarrow$ *studio*

- **An FD *f* is *implied by* a set of FDs *F* if *f* holds whenever all FDs in *F* hold.**

- **F^+ = *closure of F* is the set of all FDs that are implied by *F*. (includes "trivial dependencies")**

Rules of Inference

- **Armstrong's Axioms (X, Y, Z are <u>sets</u> of attributes):**
  - *<u>Reflexivity</u>*: If $X \supseteq Y$, then $X \rightarrow Y$
  - *<u>Augmentation</u>*: If $X \rightarrow Y$, then $XZ \rightarrow YZ$ for any Z
  - *<u>Transitivity</u>*: If $X \rightarrow Y$ and $Y \rightarrow Z$, then $X \rightarrow Z$

- **These are *sound* and *complete* inference rules for FDs!**
  - i.e., using AA you can compute all the FDs in F+ and only these FDs.

- **Some additional rules (that follow from AA):**
  - *Union*: If $X \rightarrow Y$ and $X \rightarrow Z$, then $X \rightarrow YZ$
  - *Decomposition*: If $X \rightarrow YZ$, then $X \rightarrow Y$ and $X \rightarrow Z$

Example

- **Contracts(*<u>cid</u>,sid,jid,did,pid,qty,value*), and:**
  - C is the key: C $\rightarrow$ CSJDPQV
  - Proj purchases each part using single contract: JP $\rightarrow$ C
  - Dept purchases at most 1 part from a supplier: SD $\rightarrow$ P
- **Problem: Prove that SDJ is a key for Contracts**
- **JP $\rightarrow$ C, C $\rightarrow$ CSJDPQV imply JP $\rightarrow$ CSJDPQV**
  (by transitivity) (shows that JP is a key)
- **SD $\rightarrow$ P implies SDJ $\rightarrow$ JP** (by augmentation)
- **SDJ $\rightarrow$ JP, JP $\rightarrow$ CSJDPQV imply SDJ $\rightarrow$ CSJDPQV**
  (by transitivity) thus SDJ is a key.

Q: can you now infer that SD $\rightarrow$ CSDPQV (i.e., drop J on both sides)?

No! FD inference is not like arithmetic multiplication.

Attribute Closure

- **Computing the closure of a set of FDs can be expensive. (Size of closure is exponential in # attrs!)**
- **Typically, we just want to check if a given FD *X* $\rightarrow$ *Y* is in the closure of a set of FDs *F*. An efficient check:**
  - Compute *attribute closure* of X (denoted X^+) wrt *F.*
    X^+ = Set of all attributes A such that X $\rightarrow$ A is in F^+
    - $X^+ := X$
    - Repeat until no change: if there is an FD U $\rightarrow$ V in *F* such that U is in X^+, then add V to X^+
  - Check if Y is in X^+
  - Approach can also be used to find the keys of a relation.
    - If all attributes of R are in the closure of X then X is a superkey for R.
    - Q: How to check if X is a "candidate key"?

Attribute Closure (example)

- $R = \{A, B, C, D, E\}$
- $F = \{ B \rightarrow CD, D \rightarrow E, B \rightarrow A, E \rightarrow C, AD \rightarrow B \}$
- **Is $B \rightarrow E$ in F^+ ?**

  $B^+ = B$

  $B^+ = BCD$

  $B^+ = BCDA$

  $B^+ = BCDAE$ ... Yes!
  and B is a key for R too!

- **Is D a key for R?**

  $D^+ = D$

  $D^+ = DE$

  $D^+ = DEC$

  ... Nope!

- **Is AD a key for R?**

  $AD^+ = AD$

  $AD^+ = ABD$ and B is a key, so Yes!

- **Is AD a *candidate* key for R?**

  $A^+ = A$, $D+ = DEC$

  ... A,D not keys, so Yes!

- **Is ADE a *candidate* key for R?**

  ... No! AD is a key, so ADE is a superkey, but not a cand. key

Next Class…

- **Normal forms and normalization**
- **Table decompositions**