Theory of Relational Database Design

1 Informal Design Guidelines for Relational Databases (1)

- What is relational database design?

  The grouping of attributes to form "good" relation schemas
- Design is concerned mainly with base relations
- What are the criteria for "good" base relations?

Informal Design Guidelines for Relational Databases (2)

- We first discuss informal guidelines for good relational design
- Then we discuss formal concepts of functional dependencies and normal forms
  - 1NF (First Normal Form)
  - 2NF (Second Normal Form)
  - 3NF (Third Normal Form)
  - BCNF (Boyce-Codd Normal Form)
- Additional types of dependencies, further normal forms, relational design algorithms by synthesis will be discussed later

1.1 Semantics of the Relation Attributes

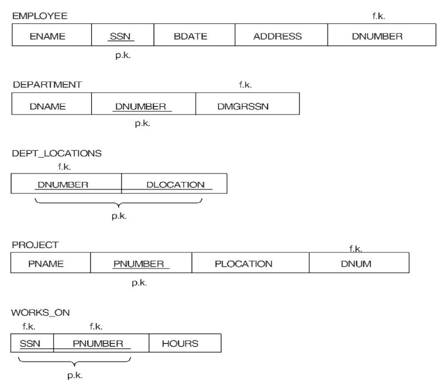
GUIDELINE 1: Informally, each tuple in a relation should represent one entity or relationship instance. (Applies to individual relations and their attributes).

- Attributes of different entities (EMPLOYEEs, DEPARTMENTs, PROJECTs) should not be mixed in the same relation
- Only foreign keys should be used to refer to other entities

Bottom Line: Design a schema that can be explained easily relation by relation. The semantics of attributes should be easy to interpret.

A simplified COMPANY relational database schema

Figure 14.1 Simplified version of the COMPANY relational database schema.



1.2 Redundant Information in Tuples and Update Anomalies

- Mixing attributes of multiple entities may cause problems
- Information is stored redundantly wasting storage
- Problems with update anomalies
  - Insertion anomalies
  - Deletion anomalies
  - Modification anomalies

EXAMPLE OF AN UPDATE ANOMALY (1)

Consider the relation:

EMP_PROJ (<u>Emp#, Proj#,</u> Ename, Pname, No_hours)

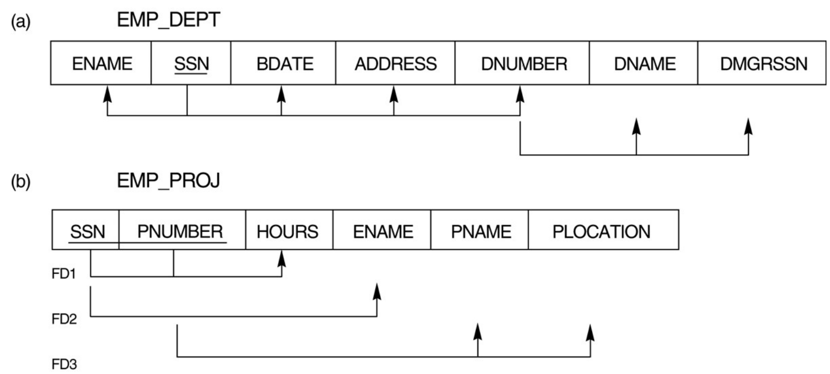
- **Update Anomaly:** Changing the name of project number P1 from "Billing" to "Customer-Accounting" may cause this update to be made for all 100 employees working on project P1.

EXAMPLE OF AN UPDATE ANOMALY (2)

- **Insert Anomaly:** Cannot insert a project unless an employee is assigned to .

  Inversely - Cannot insert an employee unless he/she is assigned to a project.
- **Delete Anomaly:** When a project is deleted, it will result in deleting all the employees who work on that project. Alternately, if an employee is the sole employee on a project, deleting that employee would result in deleting the corresponding project.

Two relation schemas suffering from update anomalies.



Example States for EMP_DEPT and EMP_PROJ

Figure 14.4 Example relations for the schemas in Figure 14.3 that result from applying NATURAL JOIN to the relations in Figure 14.2. These may be stored as base relations for performance reasons.

EMP_DEPT

ENAME	SSN	BDATE	ADDRESS	DNUMBER	DNAME	DMGRSSN
Smith,John B.	123456789	1965-01-09	731 Fondren,Houston,TX	5	Research	333445555
Wong,Franklin T.	333445555	1955-12-08	638 Voss,Houston,TX	5	Research	333445555
Zelaya, Alicia J.	999887777	1968-07-19	3321 Castle,Spring,TX	4	Administration	987654321
Wallace,Jennifer S.	987654321	1941-06-20	291 Berry,Bellaire,TX	4	Administration	987654321
Narayan,Ramesh K.	666884444	1962-09-15	975 FireOak,Humble,TX	5	Research	333445555
English,Joyce A.	453453453	1972-07-31	5631 Rice,Houston,TX	5	Research	333445555
Jabbar,Ahmad V.	987987987	1969-03-29	980 Dallas,Houston,TX	4	Administration	987654321
Borg,James E.	888665555	1937-11-10	450 Stone,Houston,TX	1	Headquarters	888665555

EMP_PROJ

SSN	PNUMBER	HOURS	ENAME	PNAME	PLOCATION
123456789	1	32.5	Smith,John B.	ProductX	Bellaire
123456789	2	7.5	Smith,John B.	ProductY	Sugarland
666884444	3	40.0	Narayan,Ramesh K.	ProductZ	Houston
453453453	1	20.0	English,Joyce A.	ProductX	Bellaire
453453453	2	20.0	English,Joyce A.	ProductY	Sugarland
333445555	2	10.0	Wong,Franklin T.	ProductY	Sugarland
333445555	3	10.0	Wong,Franklin T.	ProductZ	Houston
333445555	10	10.0	Wong,Franklin T.	Computerization	Stafford
333445555	20	10.0	Wong,Franklin T.	Reorganization	Houston
999887777	30	30.0	Zelaya,Alicia J.	Newbenefits	Stafford
999887777	10	10.0	Zelaya,Alicia J.	Computerization	Stafford
987987987	10	35.0	Jabbar,Ahmad V.	Computerization	Stafford
987987987	30	5.0	Jabbar,Ahmad V.	Newbenefits	Stafford
987654321	30	20.0	Wallace,Jennifer S.	Newbenefits	Stafford
987654321	20	15.0	Wallace,Jennifer S.	Reorganization	Houston
888665555	20	null	Borg,James E.	Reorganization	Houston

Guideline to Redundant Information in Tuples and Update Anomalies

- **GUIDELINE 2:** Design a schema that does not suffer from the insertion, deletion and update anomalies. If there are any present, then note them so that applications can be made to take them into account

1.3 Null Values in Tuples

GUIDELINE 3: Relations should be designed such that their tuples will have as few NULL values as possible

- Attributes that are NULL frequently could be placed in separate relations (with the primary key)
- Reasons for nulls:
  - attribute not applicable or invalid
  - attribute value unknown (may exist)
  - value known to exist, but unavailable

1.4 Spurious Tuples

- Bad designs for a relational database may result in erroneous results for certain JOIN operations
- The "lossless join" property is used to guarantee meaningful results for join operations

GUIDELINE 4: The relations should be designed to satisfy the lossless join condition. No spurious tuples should be generated by doing a natural-join of any relations.

Design relation schemas so that they can be joined with equality conditions on attributes that are either primary keys or foreign keys.

Avoid relations that contain matching attributes that are not (foreign key, primary key) combinations.

FIGURE 10.5
Particularly poor design for the EMP_PROJ relation of Figure 10.3b.
(a) The two relation schemas EMP_LOCS and EMP_PROJ1. (b) The result of projecting the extension of EMP_PROJ from Figure 10.4 onto the relations EMP_LOCS and EMP_PROJ1.

(a)

EMP_LOCS

ENAME	PLOCATION

p.k.

EMP_PROJ1

SSN	PNUMBER	HOURS	PNAME	PLOCATION

p.k.

(b)

EMP_LOCS

ENAME	PLOCATION
Smith, John B.	Bellaire
Smith, John B.	Sugarland
Narayan, Ramesh K.	Houston
English, Joyce A.	Bellaire
English, Joyce A.	Sugarland
Wong, Franklin T.	Sugarland
Wong, Franklin T.	Houston
Wong, Franklin T.	Stafford
Zelaya, Alicia J.	Stafford
Jabbar, Ahmad V.	Stafford
Wallace, Jennifer S.	Stafford
Wallace, Jennifer S.	Houston
Borg,James E.	Houston

FIGURE 10.5 (continued)
Particularly poor design for the EMP_PROJ relation of Figure 10.3b. (a) The two relation schemas EMP_LOCS and EMP_PROJ1. (b) The result of projecting the extension of EMP_PROJ from Figure 10.4 onto the relations EMP_LOCS and EMP_PROJ1.

EMP_PROJ1

SSN	PNUMBER	HOURS	PNAME	PLOCATION
123456789	1	32.5	Product X	Bellaire
123456789	2	7.5	Product Y	Sugarland
666884444	3	40.0	Product Z	Houston
453453453	1	20.0	Product X	Bellaire
453453453	2	20.0	Product Y	Sugarland
333445555	2	10.0	Product Y	Sugarland
333445555	3	10.0	Product Z	Houston
333445555	10	10.0	Computerization	Stafford
333445555	20	10.0	Reorganization	Houston
999887777	30	30.0	Newbenefits	Stafford
999887777	10	10.0	Computerization	Stafford
987987987	10	35.0	Computerization	Stafford
987987987	30	5.0	Newbenefits	Stafford
987654321	30	20.0	Newbenefits	Stafford
987654321	20	15.0	Reorganization	Houston
888665555	20	null	Reorganization	Houston

Result of applying NATURAL JOIN to the tuples above the dotted lines in EMP_PROJ1 and EMP_LOCS

SSN	PNUMBER	HOURS	PNAME	PLOCATION	
123456789	1	32.5	ProductX	Bellaire	Smith,John B.
123456789	1	32.5	ProductX	Bellaire	English,Joyce A.
123456789	2	7.5	ProductY	Sugarland	Smith,John B.
123456789	2	7.5	ProductY	Sugarland	English,Joyce A.
123456789	2	7.5	ProductY	Sugarland	Wong,Franklin T.
666884444	3	40.0	ProductZ	Houston	Narayan,Ramesh K.
666884444	3	40.0	ProductZ	Houston	Wong,Franklin T.
453453453	1	20.0	ProductX	Bellaire	Smith,John B.
453453453	1	20.0	ProductX	Bellaire	English,Joyce A.
453453453	2	20.0	ProductY	Sugarland	Smith,John B.
453453453	2	20.0	ProductY	Sugarland	English,Joyce A.
453453453	2	20.0	ProductY	Sugarland	Wong,Franklin T.
333445555	2	10.0	ProductY	Sugarland	Smith,John B.
333445555	2	10.0	ProductY	Sugarland	English,Joyce A.
333445555	2	10.0	ProductY	Sugarland	Wong,Franklin T.
333445555	3	10.0	ProductZ	Houston	Narayan,Ramesh K.
333445555	3	10.0	ProductZ	Houston	Wong,Franklin T.
333445555	10	10.0	Computerization	Stafford	Wong,Franklin T.
333445555	20	10.0	Reorganization	Houston	Narayan,Ramesh K.
333445555	20	10.0	Reorganization	Houston	Wong,Franklin T.

⋮

Spurious Tuples (2)

There are two important properties of decompositions:

(a) non-additive or losslessness of the corresponding join

(b) preservation of the functional dependencies.

Note that property (a) is extremely important and *cannot* be sacrificed. Property (b) is less stringent and may be sacrificed.

Functional Dependencies

- Functional dependencies (FDs) are used to specify *formal measures* of the "goodness" of relational designs
- FDs and keys are used to define **normal forms** for relations
- FDs are **constraints** that are derived from the *meaning* and *interrelationships* of the data attributes
- A set of attributes X *functionally determines* a set of attributes Y if the value of X determines a unique value for Y

Functional Dependencies (2)

- X → Y holds if whenever two tuples have the same value for X, they *must have* the same value for Y
- For any two tuples t1 and t2 in any relation instance r(R): *If* t1[X]=t2[X], *then* t1[Y]=t2[Y]
- X → Y in R specifies a *constraint* on all relation instances r(R)
- Written as X → Y; can be displayed graphically on a relation schema as in Figures. (denoted by the arrow:).
- FDs are derived from the real-world constraints on the attributes

Examples of FD constraints (1)

- social security number determines employee name
  SSN → ENAME
- project number determines project name and location
  PNUMBER → {PNAME, PLOCATION}
- employee ssn and project number determines the hours per week that the employee works on the project
  {SSN, PNUMBER} → HOURS

Examples of FD constraints (2)

- An FD is a property of the attributes in the schema R
- The constraint must hold on *every relation instance* r(R)
- If K is a key of R, then K functionally determines all attributes in R (since we never have two distinct tuples with t1[K]=t2[K])

Inference Rules for FDs

- Given a set of FDs F, we can *infer* additional FDs that hold whenever the FDs in F hold

Armstrong's inference rules:

IR1. (**Reflexive**) If $X \supseteq Y$, then $X \rightarrow Y$

IR2. (**Augmentation**) If $X \rightarrow Y$, then $XZ \rightarrow YZ$

$\{X \rightarrow Y\} \models \{XZ \rightarrow YZ\}$

(XZ stands for X U Z)

IR3. (**Transitive**) If $X \rightarrow Y$ and $Y \rightarrow Z$, then $X \rightarrow Z$

$\{X \rightarrow Y, Y \rightarrow Z\} \models X \rightarrow Z$

Proofs of Armstrong's Axioms

IR1: Suppose $X \supseteq Y$ and that two tuples t1 and t2 exists in some relation instance r of R s. t. t1[X]=t2[X].
Then t1[Y]=t2[Y] because $X \supseteq Y$; hence $X \rightarrow Y$ must hold in R.

IR2: Assume that $X \rightarrow Y$ holds in a relation instance r of R , but $XZ \rightarrow YZ$ does not hold.
Then there must exist two tuples t1 and t2 in r s.t.
(1) t1[X] = t2[X]　　　(2) t1[Y] = t2[Y]
(3) t1[XZ] = t2[XZ]　　　(4) t1[YZ] $\neq$ t2[YZ]
This is not possible because from (1) and (3) we deduce
(5) t1[Z] = t2[Z] and from (2) and (5) we deduce
(6)t1[YZ] = t2[YZ], contradicting (4).

Proofs of Armstrong's Axioms

IR3: Assume that **(1)** $X \rightarrow Y$ and **(2)** $Y \rightarrow Z$ both hold in a relation r. Then for any two tuples t1 and t2 in r such that t1[X] = t2[X], we must have
(3) t1[Y] = t2[Y] (from assumption 1). Hence we must also have
(4) t1[Z] = t2[Z] (from 3 and assumption (2))
Hence $X \rightarrow Z$ must hold in r.

Additional Inference Rules

<u>Some **additional inference rules** that are useful:</u>

(**Decomposition**) If X -> YZ, then X -> Y and X -> Z

(**Union**) If X -> Y and X -> Z, then X -> YZ

(**Psuedotransitivity**) If X -> Y and WY -> Z, then WX -> Z

- The last three inference rules, as well as any other inference rules, can be deduced from IR1, IR2, and IR3

Proofs of Additional Inference Rules

IR4: {X → YZ} ⊧ X → Y

Proof:

1. X → YZ (given)
2. YZ → Y (using IR1 and knowing that YZ ⊇ Y)
3. X → Y (using IR3 on 1 and 2)

IR5: {X → Y, X → Z} ⊧ X → YZ

Proof:

- X → Y (given)
- X → Z (given)
- X → XY (using IR2 on 1 by augmenting with X)
- XY → YZ (using IR2 on 2 by augmenting with Y)
- X → YZ (using IR3 on 3 and 4)

Proofs of Additional Inference Rules

IR6: {X→ Y, WY → Z} ⊨ WX → Z

Proof:

1. X → Y (given)
2. WY → Z (given)
3. WX → WY (using IR2 on 1 and augmenting with W)
4. WX → Z (using IR3 on 3 and 2)

Armstrong's axioms are SOUND & COMPLETE

Soundness:

Given a set of FDs F specified on a relational schema R, any dependency that we can infer from F by using IR1 through IR3 holds in every relation state r of R that satisfies the dependencies in F.

Completeness:

Using IR1 through IR3 repeatedly to infer dependencies until no more dependencies can be inferred results in the complete set of all possible dependencies that can be inferred from F

Closure of a set of FDs *F*

- **Closure** of a set F of FDs is the set F^+ of all FDs that can be inferred from F
- **Closure** of a set of attributes X with respect to F is the set X^+ of all attributes that are functionally determined by X
- X^+ can be calculated by repeatedly applying IR1, IR2, IR3 using the FDs in F

Closure of a set of FDs *F*

Algorithm: Determining X^+, the closure of X under F

$X^+ \leftarrow X$;
repeat
  old$X^+ \leftarrow X^+$;
  for each FD $Y \rightarrow Z$ in F do
    if $X^+ \supseteq Y$ then $X^+ \leftarrow X^+ \cup Z$;
until (X^+ = oldX^+);

Closure of a set of FDs *F*

Example: Given the relation
EMP_PROJ (SSN, PNUMBER, HOURS, ENAME, PNAME, PLOCATION)
and a set of FDs F on it, as follows:

F = {SSN → ENAME,
PNUMBER → {PNAME, PLOCATION},
{SSN, PNUMBER → HOURS}

Find F^+ the closure of F.

$\{SSN\}^+ = \{SSN, ENAME\}$

$\{PNUMBER\}^+ = \{PNUMBER, PNAME, PLOCATION\}$

$\{SSN, PNUMBER\}^+ = \{SSN, PNUMBER, ENAME, PNAME, PLOCATION, HOURS\}$

Equivalence of Sets of FDs

- Two sets of FDs F and G are **equivalent** if:
  - every FD in F can be inferred from G, *and*
  - every FD in G can be inferred from F
- Hence, F and G are equivalent if $F^+ = G^+$

Definition: F **covers** G if every FD in G can be inferred from F (i.e., if G^+ *subset-of* F^+)

- F and G are equivalent if F covers G and G covers F

Determining whether *F covers G*

- Calculate X+ with respect to F for each FD, X → Y in G
- Check whether this X+ includes the attributes in Y
- If this is the case for every FD in G, then ***F covers G***

- We can determine **whether F and G are equivalent** by checking whether ***F covers G and G covers F***

Minimal Sets of FDs

- A set of FDs is **minimal** if it satisfies the following conditions:

(1) Every dependency in F has a single attribute for its RHS.

(2) We cannot remove any dependency from F and still have a set of dependencies that is equivalent to F.

(3) We cannot replace any dependency X → A in F with a dependency Y → A, where Y is a proper subset of X and still have a set of dependencies that is equivalent to F.

Minimal Sets of FDs (2)

- A minimal set of dependencies is a set of dependencies in a standard canonical form and with no redundancies
- Condition 1 represents every dependency in a canonical form with a single attribute on the RHS
- Condition 2 and 3 ensure that there is no redundancy either by having a
  - Redundant dependency that can be inferred from the remaining FDs in F
  - Redundant attributes on the LHS of a dependency

Minimal Cover

- A Minimal cover of a set of FDs E is a minimal set of dependencies F that is equivalent to E
- There can be several minimal covers for a set of FDs
- Additional criteria for minimality
  - Minimal set with the smallest no. of dependencies
  - Minimal set with the smallest total length
    - Total Length is obtained by concatenating all the dependencies and treating them as one long character string

Algorithm: Finding a Minimal Cover F for a set of FDs E

1. Set F $\leftarrow$ E;
2. Replace each FD X $\rightarrow$ {A1, A2, ……,An} in F by the n FDs X $\rightarrow$ A1, X $\rightarrow$ A2,……., X$\rightarrow$ An.
3. For each FD X $\rightarrow$ A in F

   for each attribute B that is an element of X

   if {{F – {X $\rightarrow$ A}} U {(X - {B}) $\rightarrow$ A}} is equivalent to F, then replace X $\rightarrow$ A with (X – {B}) $\rightarrow$ A in F.

4. For each remaining FD X $\rightarrow$ A in F

   if { F - {X $\rightarrow$ A}} is equivalent to F,

   then remove X $\rightarrow$ A from F.

Example

Consider the relation schema
EMP_DEPT (ENAME, SSN, BDATE, ADDRESS, DNUMBER, DNAME, DMGRSSN) and the following set G of functional dependencies on EMP_DEPT:
G = {SSN ->{ENAME, BDATE, ADDRESS, DNUMBER} , DNUMBER ->{DNAME, DMGRSSN} }
Is the set of functional dependencies G minimal? If not, try to find a minimal set of functional dependencies that is equivalent to G. Prove that your set is equivalent to G.

ANSWER:
The set G of functional dependencies is not minimal, because it violates rule 1 of minimality (every FD has a single attribute for its right hand side). The set F is an equivalent minimal set:
F= {SSN ->{ENAME},SSN ->{BDATE},
SSN-> {ADDRESS}, SSN ->{DNUMBER} , DNUMBER ->{DNAME}, DNUMBER->{DMGRSSN}}
To show equivalence, we prove that G is covered by F and F is covered by G.
Proof that G is covered by F:
{SSN} + = {SSN, ENAME, BDATE, ADDRESS, DNUMBER, DNAME, DMGRSSN} (with respect to F), which covers SSN ->{ENAME, BDATE, ADDRESS, DNUMBER} in G
{DNUMBER} + ={DNUMBER, DNAME, DMGRSSN} (with respect to F), which covers DNUMBER ->{DNAME, DMGRSSN} in G
Proof that F is covered by G:
{SSN} + = {SSN, ENAME, BDATE, ADDRESS, DNUMBER, DNAME, DMGRSSN} (with respect to G), which covers SSN ->{ENAME}, SSN ->{BDATE}, SSN ->{ADDRESS}, and SSN ->{DNUMBER} in F
{DNUMBER} + = {DNUMBER, DNAME, DMGRSSN} (with respect to G), which covers DNUMBER ->{DNAME} and DNUMBER->{DMGRSSN} in F

Acknowledgement

Reference for this lecture is

- Ramez Elmasri and Shamkant B. Navathe, *Fundamentals of Database Systems,* Pearson Education.

The authors and the publishers are gratefully acknowledged.