Database Systems

Lecture #14

Sang-Wook Kim
Hanyang University

Objectives

- To learn relational database design algorithms
  - Properties of relational decompositions
  - Relational database design algorithms

Outline

- Relation Decomposition
- Dependency Preservation
- Relational Decomposition into 3NF with Dependency Preservation
- Lossless Join Property
- Relational Decomposition into BCNF with Lossless Join Property
- Dependency Preservation and Lossless Join Property

Relation Decomposition

- Relational database design by decomposition

1. Start from a *universal relation schema*
   - A relation schema $R = \{A_1, A_2, \ldots, A_n\}$ that includes all the attributes of the database
   - Every attribute name is unique

Relation Decomposition

2. Decompose R into a set of relation schemas $D = \{R_1, R_2, \ldots, R_m\}$

- Each relation R_i contains a subset of attributes from R
- Each attribute in R will appear in at least one relation schema R_i
- Each relation R_i is in BCNF or 3NF
- D is called a *decomposition* of R

Insufficiency of Normal Forms

- Any relation schema with *only two attributes* is automatically in BCNF
- A database can be designed in this way satisfying BCNF
  - Problem: spurious tuples

Insufficiency of Normal Forms

- Example: spurious tuples

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

EMP_PROJ1

Ssn	Pnumber	Hours	Pname	Plocation
123456789	1	32.5	ProductX	Bellaire
123456789	2	7.5	ProductY	Sugarland
666884444	3	40.0	ProductZ	Houston
453453453	1	20.0	ProductX	Bellaire
453453453	2	20.0	ProductY	Sugarland
333445555	2	10.0	ProductY	Sugarland
333445555	3	10.0	ProductZ	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

Insufficiency of Normal Forms

- Example: spurious tuples
  - Join EMP_LOCS and EMP_PROJ1 with Plocation

	Ssn	Pnumber	Hours	Pname	Plocation	Ename
	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.

* * *

Insufficiency of Normal Forms

- Additional properties for good relational databases
  - Lossless join property
  - Dependency preservation property

Dependency Preservation

- *F*: set of functional dependencies specified on a relational schema *R*
- Condition of dependency preservation
  - Decomposition *D* must preserve dependencies
  - Union of the dependencies F_i that hold on the individual relations R_i in *D* be *equivalent* to *F*

Dependency Preservation

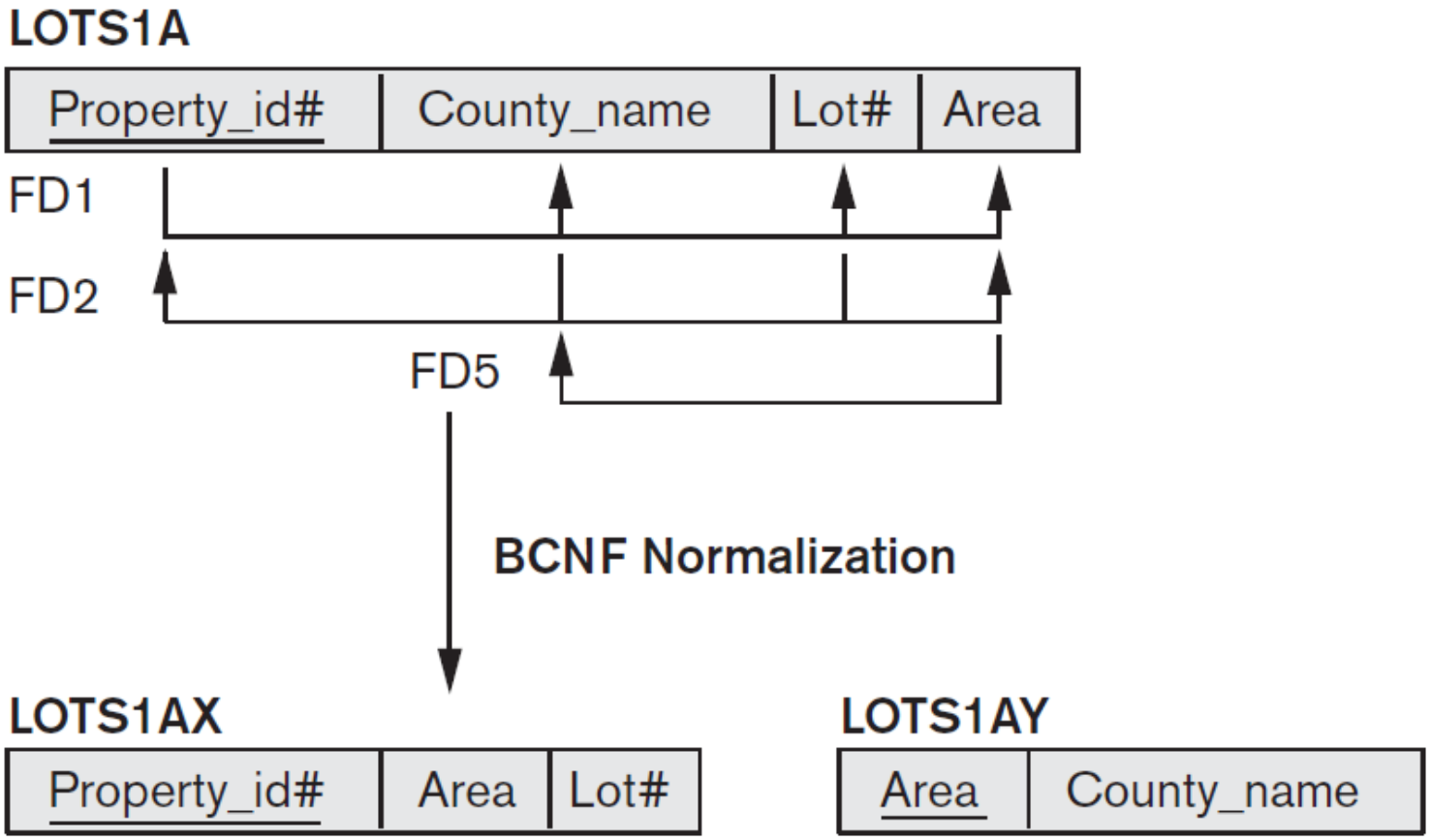
- Formal definition
  - Projection of F on R_i $\pi_{Ri}(F)$
    - Set of functional dependencies $X \rightarrow Y$ in F^+, such that the attributes in $X \cup Y$ are all contained in R_i
  - Decomposition D is *dependency-preserving*
    - Means $((\pi_{R1}(F)) \cup \ldots \cup (\pi_{Rm}(F)))^+ = F^+$

Dependency Preservation

- Dependency preservation is a constraint on *D*
  - Must be guaranteed
- If a decomposition is not dependency-preserving
  - Must take JOIN of two or more relations to check a dependency holds
  - *Not practical!*

Dependency Preservation

- Example



Dependency Preservation

- Example
  - FD2 is not dependency-preserving
    - Need JOIN to check if FD2 holds

Relational Decomposition into 3NF with Dependency Preservation

- *Minimal cover* G of F
  - A set of functional dependencies that is equivalent to F
  - Removing any FD in G makes G not equivalent to F

Relational Decomposition into 3NF with Dependency Preservation

- Finding the minimal cover *G* for *F*
  - Make each FD in *F* to be in a *canonical form*
    - Only one attribute on right-hand side
  - Remove redundant attributes (say, A) on left-hand side of each FD
    - Check whether its left-hand side could functionally determine its right-hand side without A (under the help of other FDs)
  - Remove redundant FDs from F
    - Check whether its left-hand side could functionally determine the its right-hand side without the FD (under the help of other FDs)

Relational Decomposition into 3NF with Dependency Preservation

- Algorithm for relational decomposition into 3NF with dependency preservation

  1. Find a minimal cover G for F
  2. For each left-hand-side X of a functional dependency that appears in G,

     Create a relation schema in D
     with attributes $\{X \cup \{A_1\} \cup \{A_2\} \ldots \cup \{A_k\}\}$,

     where $X \rightarrow A_1$, $X \rightarrow A_2$, ..., $X \rightarrow A_k$ are the only dependencies in G with X as left-hand-side (X is the key of this relation);

Relational Decomposition into 3NF with Dependency Preservation

- Algorithm for relational decomposition into 3NF with dependency preservation

  3. Place any remaining attributes (that have not been placed in any relation) in a single relation schema to ensure the attribute preservation property

Lossless Join Property

- Also called *non-additive* join property
- Ensures that *no spurious tuples are generated*
  - when a JOIN operation is applied to the relations resulting from the decomposition

Lossless Join Property

- Formal definition
  - Decomposition $D = \{R_1, R_2, ..., R_m\}$ has the *lossless join property* if following holds:
    - For every r, $*(\pi_{R1}(r), ..., \pi_{Rm}(r)) = r$
    - r: a relation state of R that satisfies F

Lossless Join Property

- Algorithm for testing lossless join property
  - Not covered here

Relational Decomposition into BCNF with Lossless Join Property

- Algorithm

1. Set $D := \{R\}$;
2. While there is a relation schema Q in D that is not in BCNF

```
do {
    choose a relation schema Q in D that is not in BCNF;
    find a functional dependency X → Y in Q that violates BCNF;
    replace Q in D by two relation schemas (Q - Y) and (X ∪ Y);
};
```

Relational Decomposition into BCNF with Lossless Join Property

- Two characteristics of lossless join property that the algorithm is based on
  - Decomposition $D = \{R1, R2\}$ of R has the lossless join property with respect to a set of FDs F on R
  - If and only if either:
    - The FD $((R_1 \cap R_2) \rightarrow (R_1 - R_2))$ is in F^+, or
    - The FD $((R_1 \cap R_2) \rightarrow (R_2 - R_1))$ is in F^+
  - Informally, *key and foreign-key pair*

Relational Decomposition into BCNF with Lossless Join Property

- Two characteristics of lossless join property that the algorithm is based on
  - If a decomposition $D = \{R_1, R_2, ..., R_m\}$ of R has the lossless join property with respect to a set of FDs F on R,
  - And if a decomposition $D_i = \{Q_1, Q_2, ..., Q_k\}$ of R_i has the lossless join property with respect to the projection of F on R_i,
  - Then, the decomposition $D_2 = \{R_1, R_2, ..., R_{i-1}, Q_1, Q_2, ..., Q_k, R_{i+1}, ..., R_m\}$ of R has the lossless join property with respect to F

Relational Decomposition into BCNF with Lossless Join Property

- Algorithm

1. Set $D := \{R\}$;
2. While there is a relation schema Q in D that is not in BCNF

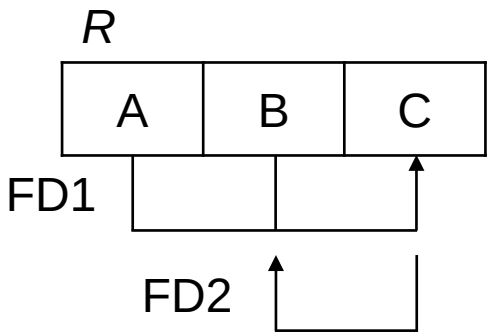
```
do {
    choose a relation schema Q in D that is not in BCNF;
    find a functional dependency X → Y in Q that violates BCNF;
    replace Q in D by two relation schemas (Q - Y) and (X ∪ Y);
};
```

Dependency Preservation and Lossless Join Property

- Algorithm for relational decomposition into BCNF with dependency preservation and lossless join property
  - Not possible to have all three properties:
    - BCNF
    - Dependency-preserving
    - Lossless join

Dependency Preservation and Lossless Join Property

- Algorithm for relational decomposition into BCNF with dependency preservation and lossless join property
  - Counter example



Dependency Preservation and Lossless Join Property

- Algorithm for relational decomposition into 3NF with dependency preservation and lossless join property

  1. Find a minimal cover G for F
  2. For each left-hand-side X of a functional dependency that appears in G,

     Create a relation schema in D with attributes $\{X \cup \{A1\} \cup \{A2\} \ldots \cup \{Ak\}\}$,

     where $X \rightarrow A1$, $X \rightarrow A2$, ..., $X \rightarrow Ak$ are the only dependencies in G with X as left-hand-side (X is the key of this relation)

Dependency Preservation and Lossless Join Property

- Algorithm for relational decomposition into 3NF with dependency preservation and lossless join property

  3. If none of the relation schemas in D contains a key of R, then create one more relation schema in D that contains attributes that form a key of R

Dependency Preservation and Lossless Join Property

- Algorithm for finding key

1. Set $K := R$;
2. For each attribute A in K {
   Compute $(K - A)^+$ with respect to F;
   If $(K - A)^+$ contains all the attributes in R,
   then set $K := K - \{A\}$;
   }

Dependency Preservation and Lossless Join Property

- Most of tables in 3NF are also in BCNF

Summary

- Relation decomposition
  - Dependency preservation
  - Lossless join property
- Algorithms
  - 3NF decomposition with dependency preservation
  - BCNF decomposition with lossless join property
  - 3NF decomposition with dependency preservation and lossless join property

References

1. Maier, David. *The theory of relational databases*. Vol. 11. Rockville: Computer science press, 1983.
2. Atzeni, Paolo, and Valeria De Antonellis. *Relational database theory*. Benjamin-Cummings Publishing Co., Inc., 1993.
3. Bernstein, Philip A. "Synthesizing third normal form relations from functional dependencies." *ACM Transactions on Database Systems (TODS)* 1.4 (1976): 277-298.
4. Biskup, Joachim, Umeshwar Dayal, and Philip A. Bernstein. "Synthesizing independent database schemas." *Proceedings of the 1979 ACM SIGMOD international conference on Management of data*. ACM, 1979.

References

5. Tsou, Don-Min, and Patrick C. Fischer. "Decomposition of a relation scheme into Boyce-Codd normal form." *ACM SIGACT News* 14.3 (1982): 23-29.
6. Ullman J. *Principles of Database and Knowledge-Base Systems*, Vol. 1, Computer Science Press, 1988.
7. Aho, Alfred V., Catriel Beeri, and Jeffrey D. Ullman. "The theory of joins in relational databases." *ACM Transactions on Database Systems (TODS)* 4.3 (1979): 297-314.
8. Osborn, Sylvia L. "Normal Forms for Relational Databases," Ph.D. dissertation, University of Waterloo, 1977.

References

9. Osborn, Sylvia L. "Towards a universal relation interface." *Very Large Data Bases, 1979. Fifth International Conference on*. IEEE, 1979.

10. Wang, Ke. "Polynomial time designs toward both BCNF and efficient data manipulation." *ACM SIGMOD Record*. Vol. 19. No. 2. ACM, 1990.

11. Hernández, Héctor J., and Edward PF Chan. "Constant-time-maintainable BCNF database schemes." *ACM Transactions on Database Systems (TODS)*16.4 (1991): 571-599.

Have a nice day!