CSC 355 Database Systems
Lecture 13

Eric J. Schwabe

School of Computing, DePaul University

Spring 2020

Today:

- Relational Database Design
  - Lossless join property
  - Third Normal Form (3NF)
  - Minimal basis
  - Algorithm for 3NF decomposition

Relational Database Design

- Starting with $R(A_1, A_2, \ldots, A_n)$, the decomposition $D = \{R_1, R_2, \ldots, R_m\}$ should satisfy the following conditions:
  1. The union of the R_i's is R
  2. Each of the R_i's is in BCNF
  3. D has the dependency preservation property
  4. D has the lossless join property

Lossless Join Property

4. For every relation state r of R, the natural join of the restrictions of r to the relations R_1, R_2, …, R_m in the decomposition is r itself

- If we decompose and then join back the resulting relations, we get the original relation state – no *spurious tuples* are added
  - Not all decompositions have this property!

Binary Lossless Join Test

- $D = \{R_1, R_2\}$ has the lossless join property if and only if one or both of the following hold:
  1. $(R_1 \cap R_2) \rightarrow (R_1 - R_2)$ can be derived from F
  2. $(R_1 \cap R_2) \rightarrow (R_2 - R_1)$ can be derived from F
- That is, if and only if the intersection between the two sets of attributes is a superkey in one of the relations…

General Test for Lossless Join

1. Create a matrix S with a row i for each R_i and a column for j for each A_j
2. Set each S(i,j) = “b_{ij}”
3. For each entry (i,j)

   If relation R_i includes A_j, then set S(i,j) = “a_j”
4. Repeat the following loop until there are no changes to S:

   For each X→Y in F

   For all rows in S that have the same symbols in all columns in X,

   set all of the columns in Y in those rows to agree as follows:

   If any row has a_j for the columns, set all rows to that same a_j

   If not, choose one of the b_{ij}s and set all rows to that same b_{ij}
5. If any row has all a_j’s, return true (D has lossless join); if not, return false D does not have lossless join).

BCNF

- A relation R is in Boyce-Codd Normal Form (BCNF) if for every non-trivial functional dependency $X \rightarrow Y$ in R, X is a superkey
  - "Every determinant must contain a candidate key"
  - A relation in BCNF will not have any redundancy, since every functional dependency in the relation will have a superkey as its determinant

3NF

- A relation R is in Third Normal Form (3NF) if for every non-trivial functional dependency $X \rightarrow \{A\}$ in R, <u>either X is a superkey or A is a prime attribute</u>
  - This is a weaker condition than BCNF, since X doesn't have to be a superkey if A is prime
  - 3NF may allow some redundancy if there is more than one candidate key

BCNF vs. 3NF

- Every relation in BCNF is in 3NF
- Not every relation in 3NF is in BCNF
  - In a 3NF relation, a prime attribute may be determined by something that is not a superkey, but BCNF will not allow that
  - 3NF and BCNF conditions are equivalent if there is only one candidate key in the relation

Example

- R(A, B, C, D)
- F = { A,C $\rightarrow$ B ; A,C $\rightarrow$ D ; D $\rightarrow$ C }
  - Identify candidate keys, prime attributes
  - Is R in BCNF? Is it in 3NF?
  - Construct a BCNF decomposition…
  - …but how do we construct 3NF decompositions?

Minimal Basis

- A *minimal basis* of F is a set G that is equivalent to F and is "as small as possible":
  1. The right side of every dependency is a single attribute
  2. No $X \rightarrow A$ can be replaced with $Y \rightarrow A$, where Y is a proper subset of X, and still be equivalent to F
  3. No $X \rightarrow A$ can be removed and still be equivalent to F

Constructing a Minimal Basis

- Start with a set of functional dependencies…
  1. Split each X→Z into an equivalent set with only one attribute on the right side of each f.d.
  2. For each X→{a}, replace it with Y→{a} (where Y is a proper subset of X) as long as the resulting set is equivalent
  3. Remove every X→{a} that can be removed as long as the resulting set is equivalent

3NF Decomposition Algorithm

1. Find a minimal basis G of F
2. For each set X that appears as the determinant of some functional dependency in G:

   Find all k dependencies of the form X → A_i in G, and create a relation in D with the attributes in X and $A_1,\ldots,A_k$.
3. If none of the relations in D contains a candidate key of R, find a candidate key K of R and create a relation in D whose attributes are the attributes of K.
4. Remove any redundant relations. (A relation Q_1 in D is redundant if all of its attributes are included in another relation Q_2 in D.)

Comparison of Algorithms

- Decomposition into BCNF relations:
  - No redundancy left in relations
  - Dependency preservation is not guaranteed
  - Lossless join is guaranteed
- Decomposition into 3NF relations:
  - Some redundancy may remain in relations that have multiple candidate keys
  - Dependency preservation is guaranteed
  - Lossless join is guaranteed

Next Time:

- Finish Relational Database Design
- Introduction to Triggers