Database Systems 236363

Normal Forms

Decomposition

- Suppose we have a relation $R[U]$ with a schema $U=\{A_1,\ldots,A_n\}$
  - A decomposition of U is a set of schemas $U_1,\ldots,U_t$ for which $U_1 \cup \ldots \cup U_t=U$
- A decomposition is ***information preserving*** if it satisfies $(\pi_{U_1}R)\bowtie(\pi_{U_2}R) \bowtie\ldots \bowtie(\pi_{U_t}R)=R$
- If we have an information preserving decomposition, we can maintain tables for $\pi_{U_1}R,\ldots,\pi_{U_t}R$ instead of one big table for R
- Our goal is to find information preserving decompositions for each possible content of R based on the functional dependencies identified from the system's analysis
  - Hence, we need to identify conditions for information preserving that follow from functional dependencies

A Decomposition Example

- Consider the following relation

T=

Size	Color	Animal
Large	Black	Horse
Large	Black	Cat
Small	Black	Cat

- The following decomposition is not information preserving

T_1=

Color	Animal
Black	Horse
Black	Cat

T_2=

Size	Color
Large	Black
Small	Black

- $T_1 \bowtie T_2$ includes (Horse,Black,Small) which is not included in T

- Notice that when applying a join on decomposed tables, new records that did not appear in the original relation (table) can be added, but existing records cannot be removed (why?)

Keys - Reminder

- Given a relation R with a schema U and a set of dependencies F defined over U, a set of attributes $X \subseteq U$ is a ***super-key*** if the values of the attributes in X uniquely identify a record in R
  - In other words, X is a super-key if $X_F^+ = U$

- The set X is a ***candidate-key***, or simply a ***key***, if it is a minimal super-key
  - That is, X is a super-key and no proper subset of X is a super-key
    - Notice that minimal key does ***not*** imply smallest key

Theorem

- If $R[U]$ is a relation over the schema U preserving a dependency set F, then the decomposition $\{U_1,U_2\}$ of U is information preserving (for any content of R) if and only if $U_1 \cap U_2$ is a super-key for $\pi_{U_1}R$ or for $\pi_{U_2}R$ (or both)

Proof – First Direction

- Suppose that $U_1 \cap U_2$ is a super-key for $\pi_{U_2} R$
- Hence, the expression $F \vdash U_1 \cap U_2 \rightarrow U_2$ holds

- $R \subseteq \pi_{U_1} R \bowtie \pi_{U_2} R$ always holds (regardless of $U_1 \cap U_2$)
- Hence, we must prove that any record in $\pi_{U_1} R \bowtie \pi_{U_2} R$ also appears in R

- Let t be a record in $\pi_{U_1} R \bowtie \pi_{U_2} R$
- Hence, there is a record $t_1 \in R$ for which $\pi_{U_1}(t_1) = \pi_{U_1}(t)$

- For these, $\pi_{U_1 \cap U_2} t_1 = \pi_{U_2}(\pi_{U_1} t_1) = \pi_{U_2}(\pi_{U_1} t) = \pi_{U_1 \cap U_2} t$ holds
- Consequently, from $U_1 \cap U_2 \rightarrow U_2$ we deduce that $\pi_{U_2}(t_1) = \pi_{U_2}(t)$ holds
- Thus, $t = t_1$ – that is, $t \in R$ as needed

Proof – Second Direction

- Assume by way of contradiction that neither $F \vdash U_1 \cap U_2 \rightarrow U_1$ nor $F \vdash U_1 \cap U_2 \rightarrow U_2$ hold
  - We will construct a possible content for $R[U]$ that satisfies F but is not information preserving

- Denote V the set $(U_1 \cap U_2)_F^+$ that includes every attribute A for which $F \vdash U_1 \cap U_2 \rightarrow A$
  - By the assumptions, neither $U_1 \setminus V$ nor $U_2 \setminus V$ are empty
  - Further, $U_1 \cap U_2 \subseteq V$

- Since V is a closure, $V_F^+ = V$ holds
  - In particular, V does not include any dependency of the form $X \rightarrow Y$ for which X is included in V while Y includes an attribute that is not in V

Proof – Second Direction (continued)

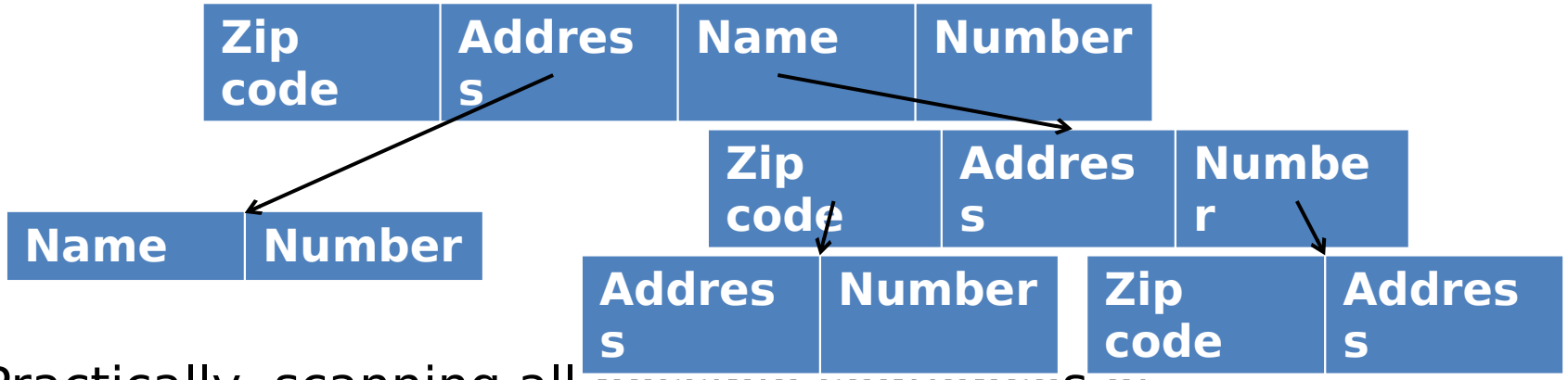
- Consider the set of the following two records for R:
  - t_1 will have the value “0” in all attributes of $U_1 \cup V$ and the value “1” on all attributes of $U_2 \setminus V$
  - t_2 will have the value “0” in all attributes of $U_2 \cup V$ and the value “1” on all attributes of $U_1 \setminus V$

- This set satisfies F since for each dependency $X \rightarrow Y$ that is violated by this set it must be that X is in V while Y includes attributes that are not in V

- On the other hand, $\pi_{U_1} R \bowtie \pi_{U_2} R$ includes the record that contains all “0”s, which is not in R
- Q.E.D.

Decomposition into Multiple Schemas

- A decomposition $U_1,\ldots,U_k$ for a relation $R[U]$ is information preserving if and only if it can be presented as a sequence of information preserving decompositions, each of which decomposes a schema into two



- Practically, scanning all possible sequences of decompositions for $U_1,\ldots,U_k$ may take exponential time
  - In the recitation, a more efficient algorithm for verifying a decomposition into multiple relations will be presented

Functional Dependencies and Redundancy

- A functional dependency $X \rightarrow Y$ for a relation $R[U]$ may indicate redundancy since the values of Y are duplicated in all records that agree on X
- Example:
  - Recall the (ill-designed) single big schema of students registered to courses
  - In that schema, there is a functional dependency between the student number and his address
  - Yet, there is no functional dependency between the student number and the course number
  - Indeed, we have seen that there is redundancy in the student address

- Hence, we shall define a “goodness” criteria for a schema U in the sense that no further decomposition is needed based on the inexistence of certain functional dependencies

Boyce & Codd Normal Form

- **Definition:** A relation $R[U]$ is in ***BCNF*** for a set of dependencies F (derived from the system's requirements) if and only if for each dependency $X \rightarrow Y$ implied by F at least one of the following holds:
  - $Y \subseteq X$ (i.e., the dependency is trivial)
  - X is a superkey of R (in this case there is no redundancy since no two distinct rows agree on X)

- Ideally, we would like to design our database such that each schema is in BCNF
- For U itself, it is enough to check F, however, to verify for any $V \subseteq U$ if it is in BCNF we need to check F^+

BCNF and Information Preserving Decompositions

- Theorem:
  - If $R[U]$ is not in BCNF with respect to a functional dependency set F, then there is an information preserving decomposition for R
- Proof:
  - Assume that $F \vdash X \rightarrow Y$ holds where Y is not a subset of X and X is not a superkey for R
  - Consider the decomposition $U_1 = X_{F^+}$, $U_2 = U \setminus (X_{F^+} \setminus X)$
  - This is a valid decomposition (neither U_1 nor U_2 are equal to U but their unification is)
  - Further, $U_1 \cap U_2 = X$ and X is a superkey for $\pi_{U_1} R$ (by the definition of U_1)
  - Hence, by the theorem about decomposition into two relations this is an information preserving decomposition
- Conclusion:
  - Each relation has an information preserving decomposition into relations in BCNF

Dependency Preserving

- Another possible quality criteria for a decomposition is whether it preserves functional dependencies without requiring reconstruction of the original relation (using joins)

- We prefer a decomposition that enables verifying dependencies by only checking the resulting schemas

- A decomposition $U_1,\ldots,U_t$ is called ***dependency preserving*** if $(F_1 \cup \ldots \cup F_t)^+ = F^+$ where F_i is the set of dependencies in F^+ that includes only attributes of U_i

- Verifying dependency preserving by calculating the closure of the dependency set is inefficient
  - In the recitation, we will see an algorithm for verifying dependency preserving by computing closures of sets (similar to the algorithm we saw for comparing dependency sets closures)

BCNF and Dependency Preserving Decompositions

- It is possible that some R[U] is not in BCNF with respect to F, yet R[U] does not have a dependency preserving decomposition
  - BCNF is not a good criterion for dependency preserving decompositions
- For example, consider the relation Addr[Street, City, Zip] and the dependency set F={Zip→City,{Street,City}→Zip}
  - This relation is not in BCNF since Zip is not a superkey for it despite the dependency Zip→City
- However, any valid decomposition of R would not preserve F
  - Notice that Zip→City is the only non-trivial dependency in F^+ that involves only two attributes
  - Hence, in any decomposition of Addr, the only dependency that can be included in any U_i is Zip→City, but the dependency {Street,City}→Zip cannot be deduced from it

3NF – 3rd Normal Form

- We would like to have a criterion that is less strict than BCNF, but would be dependency preserving
  - I.e., we are willing to "pay" a little bit in redundancy in order to preserve dependencies as well

- **Definition:** A relation R is in ***3NF*** for a dependency set F if for each dependency $X \rightarrow A$ in F^+ where A is a single attribute, at least one of the following holds:
  - $A \in X$ (i.e., the dependency is trivial)
  - X is a superkey of R (as in the definition of BCNF)
  - A belongs to (at least) one of the candidate keys of R

- For example, the relation Addr from the previous slide is in 3NF despite not being in BCNF

3NF and Information and Dependency Preserving Decomposition

- **Theorem:**
  - For each relation R[U] satisfying a dependency set F, there is an information and dependency preserving decomposition into a set of relations, each of which is in 3NF
- Proof:
  - [illegible]

Let G be a minimal covering of F

For every $X \rightarrow A$ in G do

Add the set $U_{X,A} = X \cup \{A\}$ to the list

If no set $U_{X,A}$ contains a key for R

Add a candidate key U_{key} to the list

  - The algorithm shown in the recitation is an improvement

Proof – Validity of the Decomposition

- We will show that the union of all sets is U
  - That is, every $B \in U$ is included in at least one of the sets
- If G includes a dependency $Y \rightarrow B$ then we have $B \in U_{Y,B}$ and we are done
- Otherwise, it can be verified that for each Y for which $B \notin Y$ it is not possible that $B \in Y^+$ and therefore every superkey of the relation includes B
  - The algorithm ensures that at least one of the sets will be a superkey and therefore include B

- **Comment:** It is possible that during the decomposition we will get one set that is a subset of another – in this case, the included set will be eliminated

Proof – the Decomposition is in 3NF

- In the set of attributes U_{key} there are no non-trivial dependencies
  - Otherwise, if $Y \rightarrow Z$ is a non-trivial dependency that follows from F and is included in U_{key} then the set $U_{key} \setminus (Z \setminus Y)$ would have been a superkey in contradiction to the assumption that U_{key} is a candidate key

- Similarly, for each dependency $X \rightarrow A$ in G, over the set $U_{X,A}$ there is no non-trivial dependency $Y \rightarrow Z$ for which $Y \subseteq X$ except for $X \rightarrow A$ itself
  - Otherwise, it would contradict the assumption that G is a minimal cover

- There is another possibility for $U_{X,A}$, in which it includes a non-trivial dependency $Y \rightarrow B$ for which $A \in Y$
  - However, in this case B is part of a candidate key in $U_{X,A}$, since X is such a key from the minimality of G (it is not possible that A=B since then the dependency is trivial)

Proof – Information and Dependency Preserving

- Dependency preserving
  - Since we assumed that G covers F, the dependency preservation follows from the fact that for each dependency in G we define a set of attributes that includes it
- Information preserving
  - The algorithm ensures that at least one of the sets will be a superkey for R
  - Next, we use the following general claim
- Claim
  - A dependency preserving decomposition for which one of the sets is a superkey for the entire scheme is information preserving
  - The proof is left as an exercise

Multivalued Dependencies

- **Definition:** For a relation $R[U]$ and subsets X,Y of the attribute set U, we say that R satisfies the multivalued dependency $X \twoheadrightarrow Y$ if for each possible content of R we have
  - If t_1 and t_2 are records in R for which $\pi_X(t_2)=\pi_X(t_1)$ holds, then there exists a record t_3 in R for which $\pi_{X \cup Y}(t_3)=\pi_{X \cup Y}(t_1)$ and $\pi_{U \setminus Y}(t_3)=\pi_{U \setminus Y}(t_2)$

- The basic idea is that for each record t in R, we examine all records in R that agree with t on all attributes except for $Y \setminus X$
  - The projection of this set on Y depends only on the values of t on X and not on the values of t on $U \setminus (X \cup Y)$

Multivalued Dependencies – Example

- In the following relation, there is a multivalued dependency from “Animal” to “Behavior”, but not from “Color” to “B

Behavior	Color	Animal
Growl	Black	Cat
Meow	Black	Cat
Growl	White	Cat
Meow	White	Cat
Neigh	White	Horse
Gallop	White	Horse

What does the fo

Dependencies Inference

- Given a set of dependencies (both functional and multivalued) F for a relation R[U], we say that a dependency X↠Y follows from F if every possible content of R that satisfies F also satisfies X↠Y
  - We define a functional dependency X→Y that follows from F similarly (it might follow from the multivalued dependencies and not just the functional dependencies of F)

- There is a set of axioms corresponding to Armstrong's axioms such that a dependency follows from F if and only if it can be inferred from F using a finite number of applications of these axioms
  - This is beyond the scope of this course

4NF – the 4th Normal Form

- **Definition:** A relation $R[u]$ is in ***4NF*** for a dependency set F (both functional and multivalued) if every for dependency $X \twoheadrightarrow Y$ or $X \rightarrow Y$ that follows from F, either $Y \subseteq X$ (i.e., the dependency is trivial) or X is a superkey for R

- This condition is even stronger than BCNF
  - Therefore there can be relations that are not in 4NF for which there does not exist a dependency preserving decomposition

- However, each relation has an information preserving decomposition into relations in 4NF
  - In fact, the definition of the dependency $X \twoheadrightarrow Y$ is equivalent to saying that "the content of the relation always satisfies $R = (\pi_{X \cup Y} R) \bowtie (\pi_{U \setminus (Y \setminus X)} R)$"

Other Types of Dependencies

- ***Embedded dependencies***
  - Certain types of dependencies (except for functional dependencies) might be satisfied only w.r.t. a certain projection of the relation, but not the entire relation

- Example:
  - A bank customer might open multiple accounts and use a different address in each of them
  - In the relation (id,name,address,branch,account_no) there is no dependency between id and address
  - However, in the projection to the attributes (id,name,address) there is such a multivalued dependency

Additional Types of Dependencies

- Inclusion dependencies
  - Dependencies that involve multiple relations such as "$\pi_W(R) \subseteq \pi_W(S)$" where W is a set of attributes that exist in both R and S
  - In ERD, these dependencies are expressed naturally through the connections between the relationship set and the entity sets connected to it

- Such dependencies are out of the scope of this course