Database slides on normalization

Chapter 11 Relational Database Design Algorithms and Further Dependencies Chapter Outline ? ? ? ? ? ? ? 0. Designing a Set of Relations 1. Properties of Relational Decompositions 2. Algorithms for Relational Database Schema 3. Multivalued Dependencies and Fourth Normal Form 4. Join Dependencies and Fifth Normal Form 5. Inclusion Dependencies 6. Other Dependencies and Normal Forms DESIGNING A SET OF RELATIONS ? Goals: ? Lossless join property (a must) ? Algorithm 11. 1 tests for general losslessness. Algorithm 11. decomposes a relation into BCNF components by sacrificing the dependency preservation. 4NF (based on multi-valued dependencies) 5NF (based on join dependencies) ? Dependency preservation property ? ? Additional normal forms ? ? 1. Properties of Relational Decompositions ? Relation Decomposition and Insufficiency of Normal Forms: ? Universal Relation Schema: ? A relation schema R = {A1, A2, …, An} that includes all the attributes of the database. Every attribute name is unique. ? Universal relation assumption: ? (Cont) ? Decomposition: ? ? Attribute preservation condition: ?

The process of decomposing the universal relation schema R into a set of relation schemas D = {R1, R2, …, Rm} that will become the relational database schema by using the functional dependencies. Each attribute in R will appear in at least one relation schema Ri in the decomposition so that no attributes are “ lost”. (Cont) ? ? Another goal of decomposition is to have each individual relation Ri in the decomposition D be in BCNF or 3NF. Additional properties of decomposition are needed to prevent from generating spurious tuples (Cont) ? Dependency Preservation Property of a Decomposition: ? Definition: Given a set of dependencies F on R, the projection of F on Ri, denoted by pRi(F) where Ri is a subset of R, is the set of dependencies X > Y in F+ such that the attributes in X U Y are all contained in Ri. Hence, the projection of F on each relation schema Ri in the decomposition D is the set of functional dependencies in F+, the closure of F, such that all their left- and right-hand-side attributes are in Ri. (Cont.) ? Dependency Preservation Property of a Decomposition (cont.): ? Dependency Preservation Property: ? ? A decomposition D = {R1, R2, ... Rm} of R is dependency-preserving with respect to F if the union of the projections of F on each Ri in D is equivalent to F; that is ((? R1(F)) U . . . U (? Rm(F)))+ = F+ (See examples in Fig 10. 12a and Fig 10. 11) ? Claim 1: ? It is always possible to find a dependency-preserving decomposition D with respect to F such that each relation Ri in D is in 3NF. Projection of F on Ri Given a set of dependencies F on R, the projection of F on Ri, denoted by ? Ri(F) where Ri is a subset of R, is the set of dependencies X > Y in F+ such that the attributes in X ?

Y are all contained in Ri. Dependency Preservation Condition Given R(A, B, C, D) and F = { A > B, B > C, C > D}   Let D1={R1(A, B), R2(B, C), R3(C, D)} ? R1(F)={A > B} ? R2(F)={B > C} ? R3(F)={C > D} FDs are preserved. (Cont.) ? Lossless (Non-additive) Join Property of a Decomposition: ? Definition: Lossless join property: a decomposition D = {R1, R2, ... , Rm} of R has the lossless (nonadditive) join property with respect to the set of dependencies F on R if, for every relation state r of R that satisfies F, the following holds, where * is the natural join of all the relations in D: (? R1(r), ... , ? Rm(r)) = r ? Note: The word loss in lossless refers to loss of information, not to loss of tuples. In fact, for “ loss of information” a better term is “

addition of spurious information” Example S s1 s2 s3 P p1 p2 p1 D d1 d2 d3 = S s1 s2 s3 P p1 p2 p1 * P p1 p2 p1 D d1 d2 d3 Lossless Join Decomposition ?? NO (Cont.) Lossless (Non-additive) Join Property of a Decomposition (cont.): Algorithm 11. 1: Testing for Lossless Join Property Input: A universal relation R, a decomposition D = {R1, R2, ... , Rm} of R, and a set F of functional dependencies. 1.

Create an initial matrix S with one row i for each relation Ri in D, and one column j for each attribute Aj in R. 2. Set S(i, j):= bij for all matrix entries. (/* each bij is a distinct symbol associated with indices (i, j) */). 3. For each row i representing relation schema Ri {for each column j representing attribute Aj {if (relation Ri includes attribute Aj) then set S(i, j):= aj;};}; ? (/* each aj is a distinct symbol associated with index (j) */) ? CONTINUED on NEXT SLIDE (Cont.) 4. Repeat the following loop until a complete loop execution results in no changes to S {for each functional dependency X >?

Y in F {for all rows in S which have the same symbols in the columns corresponding to attributes in X {make the symbols in each column that correspond to an attribute in Y be the same in all these rows as follows: If any of the rows has an “ a” symbol for the column, set the other rows to that same “ a” symbol in the column. If no “ a” symbol exists for the attribute in any of the rows, choose one of the “ b” symbols that appear in one of the rows for the attribute and set the other rows to that same “ b” symbol in the column ;}; }; }; 5.

If a row is made up entirely of “ a” symbols, then the decomposition has the lossless join property; otherwise it does not. (Cont.) Lossless (nonadditive) join test for n-ary decompositions. (a) Case 1: Decomposition of EMP_PROJ into EMP_PROJ1 and EMP_LOCS fails test. (b) A decomposition of EMP_PROJ that has the lossless join property. (Cont.) Lossless (nonadditive) join test for n-ary decompositions. (c) Case 2: Decomposition of EMP_PROJ into EMP, PROJECT, and WORKS_ON satisfies test. (Cont.) ? Testing Binary Decompositions for Lossless Join Property ? ?

Binary Decomposition: Decomposition of a relation R into two relations. PROPERTY LJ1 (lossless join test for binary decompositions): A decomposition D = {R1, R2} of R has the lossless join property with respect to a set of functional dependencies F on R if and only if either ? ? The FD ((R1 ? R2) >? (R1- R2)) is in F+, or The FD ((R1 ? R2) >? (R2 - R1)) is in F+. 2. Algorithms for Relational Database Schema Design Algorithm 11. 3: Relational Decomposition into BCNF with Lossless (non-additive) join property Input: A universal relation R and a set of functional dependencies F on the attributes of R. 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 U Y); }; Assumption: No null values are allowed for the join attributes. Algorithms for Relational Database Schema Design Algorithm 11. 4 Relational Synthesis into 3NF with Dependency Preservation and Lossless (Non-Additive) Join Property Input: A universal relation R and a set of functional dependencies F on the attributes of R. 1. Find a minimal cover G for F (Use Algorithm 10.). 2. For each left-hand-side X of a functional dependency that appears in G, create a relation schema in D with attributes {X U {A1} U {A2} ... U {Ak}}, where X >? A1, X >? A2, ... , X > Ak are the only dependencies in G with X as left-hand-side (X is the key of this relation). 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. (Use Algorithm 11. 4a to find the key of R) 4. Eliminate redundant relations from the result. A relation R is considered redundant if R is a projection of another relation S

Algorithms for Relational Database Schema Design Algorithm 11. 4a Finding a Key K for R Given a set F of Functional Dependencies Input: A universal relation R and a set of functional dependencies F on the attributes of R. 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}; } (Cont.) 3. Multivalued Dependencies and Fourth Normal Form (a) The EMP relation with two MVDs: ENAME —>> PNAME and ENAME —>> DNAME. (b) Decomposing the EMP relation into two 4NF relations EMP_PROJECTS and EMP_DEPENDENTS. (Cont.) c) The relation SUPPLY with no MVDs is in 4NF but not in 5NF if it has the JD(R1, R2, R3). (d) Decomposing the relation SUPPLY into the 5NF relations R1, R2, and R3. (Cont.) Definition: ? A multivalued dependency (MVD) X —>> Y specified on relation schema R, where X and Y are both subsets of R, specifies the following constraint on any relation state r of R: If two tuples t1 and t2 exist in r such that t1[X] = t2[X], then two tuples t3 and t4 should also exist in r with the following properties, where we use Z to denote (R -(X U Y)): ? t3[X] = t4[X] = t1[X] = t2[X]. t3[Y] = t1[Y] and t4[Y] = t2[Y]. t3[Z] = t2[Z] and t4[Z] = t1[Z].

An MVD X —>> Y in R is called a trivial MVD if (a) Y is a subset of X, or (b) X U Y = R. ? ? ? Multivalued Dependencies and Fourth Normal Form

Definition: ? A relation schema R is in 4NF with respect to a set of dependencies F (that includes functional dependencies and multivalued dependencies) if, for every nontrivial multivalued dependency X —>> Y in F+, X is a superkey for R. ? Informally, whenever 2 tuples that have different Y values but same X values, exists, then if these Y values get repeated in separate tuples with every distinct values of Z {Z = R – (X U Y)} that occurs with the same X value. Cont.) (Cont.) Lossless (Non-additive) Join Decomposition into 4NF Relations: ? PROPERTY LJ1’ ? The relation schemas R1 and R2 form a lossless (non-additive) join decomposition of R with respect to a set F of functional and multivalued dependencies if and only if ? (R1 ? R2) —>> (R1 - R2) (R1 ? R2) —>> (R2 - R1)). ? or ? (Cont.) Algorithm 11. 5: Relational decomposition into 4NF relations with non-additive join property ? Input: A universal relation R and a set of functional and multivalued dependencies F.

Set D := { R }; While there is a relation schema Q in D that is not in 4NF do { choose a relation schema Q in D that is not in 4NF; find a nontrivial MVD X —>> Y in Q that violates 4NF; replace Q in D by two relation schemas (Q - Y) and (X U Y); }; 1. 2. 4. Join Dependencies and Fifth Normal Form Definition: ? A join dependency (JD), denoted by JD(R1, R2, ... , Rn), specified on relation schema R, specifies a constraint on the states r of R. ? ? The constraint states that every legal state r of R should have a non-additive join decomposition into R1, R2, ... Rn; that is, for every such r we have * (? R1(r), ? R2(r), ... , ? Rn(r)) = r (Cont.) Definition: ? A relation schema R is in fifth normal form (5NF) (or Project-Join Normal Form (PJNF)) with respect to a set F of functional, multivalued, and join dependencies if, ? for every nontrivial join dependency JD(R1, R2, ... , Rn) in F+ (that is, implied by F), ? every Ri is a superkey of R. Recap ? ? ? ? ? Designing a Set of Relations Properties of Relational Decompositions Algorithms for Relational Database Schema Multivalued Dependencies and Fourth Normal Form Join Dependencies and Fifth Normal Form

Tutorial/Quiz 4 Q1) Consider a relation R with 5 attributes ABCDE, You are given the following dependencies: A > B, BC > E, ED > A a) List all the keys, b) Is R in 3 NF c) Is R in BCNF Q2) Consider the following decomposition for the relation schema R = {A, B, C, D, E, F, G, H, I, J} and the set of functional dependencies F = { {A, B} > {C}, {A} > {D, E}, {B} > {F}, {F} > {G, H}, {D} -> {I, J} }. Preserves Lossless Join and Dependencies? a) D1 = {R1, R2, R3, R4, R5}, R1={A, B, C} R2={A, D, E}, R3={B, F}, R4 = {F, G, H}, R5 = {D, I, J} b) D2 = {R1, R2, R3} R1 = {A, B, C, D, E} R2 = {B, F, G, H}, R3 = {D, I, J}