CS301 Database Management Systems
Spring 2024
Assignment #4

Relational Algebra and Normalization

Due: March 22 at 11:59 pm

Relational Algebra (10 points each) : Specify the following queries using relational algebra.

1. Find the SSN of employees who work for the Headquarters Department.
   π ssn (σ dname='Headquarters' (Department ⋈ Employee));

2. Find the Pnumber and Pname of each project controlled by the department with
   Dname='Research'.
   π pnumber, pname (σ dname='Research' (Department ⋈ Project));

3. For each department, find the dnumber, dname and the number of employees who work for
   that department.
   π dnumber, dname, count(ssn) (Department ⋈ Employee);

4. Find the number of employees who have a supervisor role.
   count(ssn) (σ superssn IS NOT NULL (Employee));

5. For each employee who has worked on the 'Newbenefits' project for more than 10 hours, list
   Fname, Lname, and Ssn.
   π fname, lname, ssn (σ pname='Newbenefits' AND hours>10 (Employee ⋈ Works_on ⋈
   Project));

6. List the ssn, lname of each employee who works on both of the projects ProductX and
   ProductY.
   π ssn, lname (Employee ⋈ Works_on ⋈ Project) ⋈ π ssn (σ pname='ProductX' (Project)) ⋈
   π ssn (σ pname='ProductY' (Project));

7. Get Ssn and Salary for each employee who has worked on all the projects.
   π ssn, salary (Employee) - π ssn, salary (Employee - (π ssn (Employee ⋈ Works_on ⋈
   Project) - π ssn (Project)));

8. Find the project number and name for the projects that involve all the employees in the
   Department of 'Administration.'
   π pnumber, pname (Project) - π pnumber, pname (Project - (π pnumber (Employee ⋈
   Works_on ⋈ Project) - π pnumber (σ dname='Administration' (Department ⋈ Employee))));

9. **Normalization (20 points) :**For each relation and its related function dependencies below, indicate which normal form (1NF, 2NF or 3NF) it is in, and if it is not in 3NF, decompose it into 3NF relations. The primary key is in bold face.

a). R(**B**, **C**, D, E, F, G): BC → D, BC → E, B → F, B → G

This relation is in 2NF but not in 3NF. It is decomposed into R1(B, F, G) and R2(B, C, D, E).

b). R(**B**, **C**, D, E, F, G): BC → D, BC → E, D → F, E → G

This relation is in 3NF because the left-hand side of each functional dependency is a superkey.