Department of Computer Engineering
University of Peradeniya
CO226 – Database Systems

Tutorial 02 – ANSWERS

Question 1

a) Database normalization is the process of organizing the fields and tables of a relational database to minimize redundancy. Normalization usually involves dividing large tables into smaller (and less redundant) tables and defining relationships between them.

b) If many of the attributes do not apply to all tuples in the relation, we end up with many NULLs in those tuples. This can waste space at the storage level and may also lead to problems with understanding the meaning of the attributes and with specifying JOIN operations at the logical level.
Another problem with NULLs is how to account for them when aggregate operations such as COUNT or SUM are applied. SELECT and JOIN operations involve comparisons; if NULL values are present, the results may become unpredictable.

c)

EMPLOYEE (EmpID, FName, LastName, DeptID, DeptName,DeptAddrs)

i. Insert, Delete, Update

ii. To insert an employee, department details should be added.
EMPID $\rightarrow \{$FNAME,LNAME,DEPTID$\}$
DEPTID $\rightarrow \{$DEPTNAME,DEPTADDRS$\}$

iii. 2NF.
Only one attribute for the primary key. So that all the non prime attributes must be full functionally depend on the primary key. If not it is violates 2NF.

iv.
EMPLOYEE(EMPID, FNAME, LNAME, DEPTID)
DEPARTMENT (DEPTID, DEPTNAME, DEPTADDRS$\}$

Question 2

a) A table is not in 1NF if domain of any attribute is non atomic or if there are any nested relations.

b) A functional dependency X →Y is said to be a full functional dependency if removal of any attribute in X will cause the functional dependency to not hold anymore. A relation will be in 2NF if it is in 1NF and each non-prime attribute in the relation is fully functional dependent on the primary key.

The general definition of 2nd normal form states that for a relation to be in 2NF every non-prime attribute in the relation should not partially depend on **any key** of the relation.

c) X→ Y will be called a transitive dependency if there is an attribute Z in the relation where relations both X → Z and Z→ Y holds. Attribute Z must not be a candidate key or a subset of any key in the relation. A relation is in 3NF if it is in 2NF and each non-prime attribute is not transitive dependent on primary key.

The general definition of 3NF says that for a relation to be in 3NF when a dependency X →A holds then either X is a super key or A is a prime attribute of the relation.

d) BCNF is simpler than 3NF but it is stricter than 3NF. Every relation in BCNF is always in 3NF. The definition of BCNF is that for a relation to be in BCNF when a dependency X → A holds then X must be a super key. In 3NF, A is allowed to be a prime attribute but it is not allowed in BCNF.
BCNF form helps to further minimize redundancies in relations than what is feasible in 3NF.

e) Consider the following relation in Figure 01.

A	B	C	D	E	F	G
1	small	silver	new	8	John	foot
2	small	gold	old	8	John	inch
1	medium	silver	new	8	John	foot
1	medium	silver	old	8	John	foot
2	medium	gold	old	7	John	inch

Figure 01

i. F →A is not a functional dependency. For same value ‘John’ in F there are 2 values.

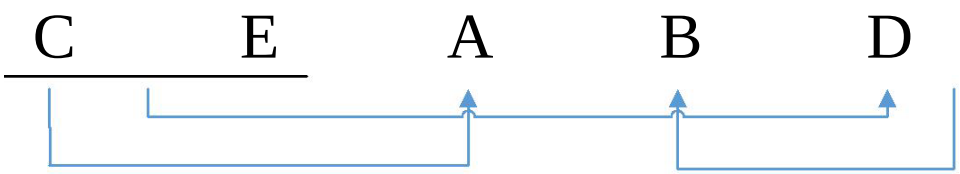
ii. A → F is a functional dependency. when A is ‘1’ F is always John and also when A is ‘2’ F is always ‘John’.

iii. { A, B } → F is a functional dependency because for any combination of A,B F is always ‘john’. So when for any tuple t1 and t2 if t1[{A,B}]=t2[{A,B}] then t1[F]=t2[F]

iv. C →G is a functional dependency. For ‘silver’ in C it is always ‘foot’ in G. For ‘gold’ in C it is always ‘inch’ in G.{A, B, G}→ {C, E, F, G}

v. { A, B, G } → {C, E, F, G} is a functional dependency.

f)

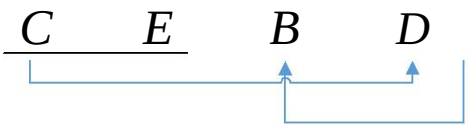


Since this is defined as a relation we can conclude that then from the definition itself, it has atomic values always. So this **is in 1NF**.

Here C which is a part of the primary key has a functional dependency to A which is a non-key attribute. So this is not in 2NF. So this is **not in BCNF** as well.

After decomposing to satisfy 2NF we have the following

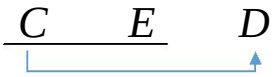
R



S

But still we have a transitive dependency. C → B can be obtained using C → D and D → B. So further decomposing we have it in 3NF as following.

R



S

T

There are no nontrivial dependencies as well. So this is in BCNF now.

Question 3

a) Answer following questions.

i. Selection, Projection, Rename, Join, Division...

ii. The two relations must have the same number of attributes and each attribute must be from the same domain.
Because attribute of the two relations must be compare in order to do the set operation which we need.

b)

STUDENT (StuID, StuName, Bdate, Address, AdvisorID, DeptID)
FOLLOWS (StuID, CourseID, Grade, Points)
COURSE (CourseID, CourseName, Credits, Semester, DeptID)
TEACHERS (LecID, CourseID)
LECTURER (LecID, LecName, Salary, DeptID)
DEPARTMENT (DeptID, DeptName, HeadID)
PREREQUISITE (CourseID, PrerequisiteID)

i. Π_{STUID} (STUDENT $\bowtie_{DEPTID=DEPTID}$ DEPARTMENT)

or

Π_{STUID} (STUDENT * DEPARTMENT)

ii. $\Pi_{LECID,LECNAME}$ (LECTURER $\bowtie_{LECID = HEADNAME}$ DEPARTMENT)

iii. STU_SEM5 ← Π_{STUID} ($\sigma_{SEMESTER=5}$ (FOLLOWS*COURSE))
ALL_STU ← Π_{STUID} (STUDENT)
RESULT ← ALL_STU – STU_SEM5

iv. PREDB ← $\Pi_{PREREQUISITEID}$ ($\sigma_{COURSEID = 'CO226'}$ (COURSE * PREREQUISITITE))
STU_COURSEID ← $\Pi_{STUID,COURSEID}$ ($\sigma_{POINTS>1.3}$ (FOLLOWS))
RESULT ← STU_COURSEID ÷ PREDB

v. COURSE_CO ← $\sigma_{DEPARMENT = 'COMPUTER'}$ (COURSE * DEPARMENT)
STU_COURSE_CO ← $\Pi_{STUID,COURSEID}$ (FOLLOWS* COURSE_CO)
RESULT ← $\Pi_{STUID,DEPTID,COURSEID}$ (STUDENT $\bowtie_{STUID=STUID}$ STU_COURSE_CO)