

Kaden Chan

ID: 2021716308

CS 6360.001 Database Design Assignment 3

**Chapter 4:**

Specify Queries of 6.16(a, b, d, e, f) in SQL, also 4.12, 4.13 (a,b,c);

Operators  $\sigma_{((Dno = 5) \& )}$ ,  $\pi$ ,  $\bowtie$ , GROUP

a)

```
SELECT FName, Minit, LName FROM (SELECT SSN FROM WORKS_ON W, PROJECT P
WHERE W.Pno = P.Pnumber AND P.Pname = "ProductX" AND W.Hours > 10), EMPLOYEE E
WHERE Essn = E.ssn AND Dno = 5
```

b)

```
SELECT E.Fname, E.Minit, E.Lname FROM EMPLOYEE E, DEPENDENT D WHERE E.Essn =
D.Ssn AND E.Fname = D.Dependent_name
```

d)

```
SELECT Pname, SUM(W.Hours) FROM WORKS_ON W, PROJECT P WHERE W.Pno =
P.Number GROUP BY P.name
```

e)

```
SELECT E.Fname, E.Minit, E.Lname FROM EMPLOYEE E, WORKS_ON W WHERE E.Essn =
W.Essn GROUP BY E.Essn HAVING (SELECT COUNT(*) FROM PROJECT) =
COUNT(DISTINCT W.Pno)
```

f)

```
SELECT E.Fname, E.Minit, E.Lname FROM EMPLOYEE, (SELECT E.Essn FROM
EMPLOYEE E EXCEPT (SELECT E.Essn FROM WORKS_ON)) NW WHERE E.ssn =
NW.Essn
```

4.12

a) SELECT S.Name FROM STUDENT S WHERE S.Class = 4 AND S.Major = 4

b) SELECT DISTINCT C.Course\_name FROM COURSE C, SECTION S WHERE S.Professor = "King" AND C.Course\_number = S.Course\_number AND (S.Year = 2008 OR S.Year = 2007)

c) SELECT S.Section\_identifier S.Course\_number, S.Semester, S.Year, COUNT(\*) FROM SECTION S JOIN GRADE\_REPORT G ON Section\_identifier WHERE S.Professor = "King" GROUP BY Section\_identifier

d) SELECT ST.Name, C.Course\_number, C.Credit\_hours, SE.Semester, SE.Year, G.Grade FROM STUDENT ST, COURSE C, SECTION SE, GRADE\_REPORT G WHERE ST.Student\_number = G.Student\_number AND C.Course\_number = SE.Course\_number AND SE.Section\_identifier = G.Section\_identifier AND ST.Class = 4

4.13

a) INSERT INTO STUDENT (Name, Student\_number, Class, Major) Values ('Johnson', 25, 1, 'Math')

b) UPDATE STUDENT SET Class = 2 WHERE Name = 'Smith'

c) INSERT INTO COURSE (Course\_name, Course\_number, Credit\_hours, Department) Values ('Knowledge Engineering', 'CS4390', 3, 'CS')

## **Chapter 5:**

5.6 SELECT S.Name, S.Major FROM STUDENT S WHERE NOT EXISTS ( SELECT G.Student\_number FROM GRADE\_REPORT G WHERE S.Student\_number = G.Student\_number AND G.Grade <> 'A')

5.8 SELECT S.Name, S.Major FROM STUDENT S WHERE NOT EXISTS ( SELECT G.Student\_number FROM GRADE\_REPORT G WHERE S.Student\_number = G.Student\_number AND G.Grade = 'A')

## Chapter 15

15.12 Boyce-Codd normal form eliminates redundancy, Functional dependencies can either be trivial or for the FD  $X \rightarrow A$ ,  $X$  is a superkey. BCNF is stronger than 3NF because every relation in BCNF is also in 3NF, but not every relation in 3NF is in BCNF, so it has a stricter set.

**15.24.** Consider the universal relation  $R = \{A, B, C, D, E, F, G, H, I, J\}$  and the set of functional dependencies  $F = \{ \{A, B\} \rightarrow \{C\}, \{A\} \rightarrow \{D, E\}, \{B\} \rightarrow \{F\}, \{F\} \rightarrow \{G, H\}, \{D\} \rightarrow \{I, J\} \}$ . What is the key for  $R$ ? Decompose  $R$  into 2NF and then 3NF relations.

15.24 The Key for  $R$  is  $\{A, B\}$  to access all attributes

2NF: **remove partial dependencies**

$R_{11} = \{A, B, C\} \rightarrow F_{11} = \{A, B\} \rightarrow \{C\}$

$R_{12} = \{A, D, E\} \rightarrow F_{12} = \{A\} \rightarrow \{D, E\}$

$R_{13} = \{B, F\} \rightarrow F_{13} = \{B\} \rightarrow \{F\}$

$R_{21} = \{F, G, H\} \rightarrow F_{21} = \{F\} \rightarrow \{G, H\}$

$R_{22} = \{D, I, J\} \rightarrow F_{22} = \{D\} \rightarrow \{I, J\}$

3NF:

$R_{11} = \{A, B, C\} \rightarrow F_{11} = \{A, B\} \rightarrow \{C\}$

$R_{12} = \{A, D, E\} \rightarrow F_{12} = \{A\} \rightarrow \{D\}, \{A\} \rightarrow \{E\}$

$R_{13} = \{B, F\} \rightarrow F_{13} = \{B\} \rightarrow \{F\}$

$R_{21} = \{F, G, H\} \rightarrow F_{21} = \{F\} \rightarrow \{G\}, \{F\} \rightarrow \{H\}$

$R_{22} = \{D, I, J\} \rightarrow F_{22} = \{D\} \rightarrow \{I\}, \{D\} \rightarrow \{J\}$

**15.25.** Repeat Exercise 15.24 for the following different set of functional dependencies  $G = \{ \{A, B\} \rightarrow \{C\}, \{B, D\} \rightarrow \{E, F\}, \{A, D\} \rightarrow \{G, H\}, \{A\} \rightarrow \{I\}, \{H\} \rightarrow \{J\} \}$ .

15.25

2NF:

$R_1 = \{A, B, C\} \rightarrow F_1 = \{A, B\} \rightarrow \{C\}$

$R_2 = \{B, D, E, F\} \rightarrow F_2 = \{B, D\} \rightarrow \{E, F\}$

$R_3 = \{A, D, G, H\} \rightarrow F_3 = \{A, D\} \rightarrow \{G, H\}$

$R_4 = \{A, I\} \rightarrow F_4 = \{A\} \rightarrow \{I\}$

$R_5 = \{H, J\} \rightarrow F_5 = \{H\} \rightarrow \{J\}$

3NF: **need to add primary key relation**, did not exist before

$R_1 = \{A, B, C\} \rightarrow F_1 = \{A, B\} \rightarrow \{C\}$

$R_2 = \{B, D, E, F\} \rightarrow F_2 = \{B, D\} \rightarrow \{E\}, \{B, D\} \rightarrow \{F\}$

$R_3 = \{A, D, G, H\} \rightarrow F_3 = \{A, D\} \rightarrow \{G\}, \{A, D\} \rightarrow \{H\}$

$R_4 = \{A, I\} \rightarrow F_4 = \{A\} \rightarrow \{I\}$

$R_5 = \{H, J\} \rightarrow F_5 = \{H\} \rightarrow \{J\}$

$R_6 = \{A, B, D\} \rightarrow F_6 =$

15.27 No, AB is not a candidate key bc it only gets to {A,B,C}

ABD is a candidate key since it accesses C in F1, then it can access E through FD 2

15.28

primary key: {Course\_no, Sec\_no, Semester, Year} would produce all attributes in its closure  
Course\_no is a good key for a smaller relation only containing FD 1

I would decompose into 2NF to separate each FD into it's own relation to escape update, insertion, and deletion anomalies. I would keep it in 2NF because the information is correlated with each other and should be viewed that way in the table, since it's fully functional.

15.30

It's in 1NF since the FD Date\_sold  $\rightarrow$  Discount\_amt violates fully functional dependency.

15.31

a) This relation is in 1NF because the key is Book\_title, Author\_name. The non-key attributes don't fully functionally depend on the key

b) To normalize this composition, the relation can decompose into 2NF:

R1: {Book\_title, Publisher, Book\_type}  $\rightarrow$  F1 = {Book\_title}  $\rightarrow$  {Publisher, Book\_type}

R2: {Book\_type, List\_price}  $\rightarrow$  F2 = {Book\_type}  $\rightarrow$  {List\_price}

R3: {Author\_name}  $\rightarrow$  F3 = {Author\_name}  $\rightarrow$  {Author\_affil}

Where each FD is isolated to be fully functionally dependent in its relation. This way, we can escape insertion, deletion, and update anomalies that occur due to redundancy.

This decomposition is already in 3NF because each FD uses the super key as the LHS. The 3NF decomposition enforces that each non-prime attribute is nontransitively dependent on the whole key, which avoids inconsistencies when updating transitively dependent attributes.

But running the 3NF decomposition method:

R1: {Book\_title, Publisher, Book\_type}  $\rightarrow$  F1 = {Book\_title}  $\rightarrow$  {Publisher}, {Book\_title}  $\rightarrow$  {Book\_type}

R2: {Book\_type, List\_price}  $\rightarrow$  F2 = {Book\_type}  $\rightarrow$  {List\_price}

R3: {Author\_name}  $\rightarrow$  F3 = {Author\_name}  $\rightarrow$  {Author\_affil}

R4: {Book\_title, Author\_name}

It ensure the right hand side has only 1 attribute and creates a relation with the key of the original relation to get lossless and dependency preserving decomposition.

## Chapter 16

16.32

a)  $\{M\}$  and  $\{M, C\}$  are not candidate keys because they only yield  $\{M, MP, C\}$   
 $\{M, Y\}$  is a candidate key because it yields all attributes  $\{M, MP, C, Y, P\}$

b) REFRIG is not in 3NF or BCNF because FD 1 and 3 don't satisfy the requirements for FDs in 3NF form  $\rightarrow$  not in BCNF since it's stricter.

c)  $R_1, \& R_2 = M \mid M \rightarrow (R_1 - R_2)$  or  $M \rightarrow (R_2 - R_1)$ ?  $M \rightarrow \{MP, C\}$  is in the closure of F which can be derived from the transitive relation  $M \rightarrow MP \rightarrow C$ , so the decomposition is lossless.

## Chapter 21

21.9

A serial schedule has for every transaction T, all operations are executed consecutively in the schedule where one (whole) transaction is executed at a time, lacks concurrency.

A serializable schedule is a schedule of n transactions that is equivalent to some serial schedule of the same n transactions (n! possibilities)

A serial schedule is correct because every transaction is considered correct if executed on its own, and since the transaction is isolated from operations of other transactions, it's correct.

A serializable schedule is correct because it is equivalent to a serial schedule which is correct

21.22

a) a is not conflict serializable because a cycle was found in the precedence graph

b) same as a, it is not conflict serializable

c) c is conflict serializable because no cycle was found. Serial:  $T_2 T_3 T_1$

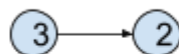
21.23

$S_1$

X



Y



Z

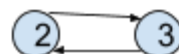


$S_2$

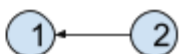
X



Y



Z



$S_1$  can be rewritten to:  $T_1 \rightarrow T_3 \rightarrow T_2$

$S_2$  is not conflict serializable because a cycle was found