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 $\sigma_{((Dno = 5) \&)}$ s, π, $\bowtie$, 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 → 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 $F = \{ \{A, B\}\to\{C\}, \{A\}\to\{D, E\}, \{B\}\to\{F\}, \{F\}\to\{G, H\}, \{D\}\to\{I, J\} \}$. 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**
R11 = {A,B,C} → F11 = {A,B} -> {C}
R12 = {A,D,E} → F12 = {A} -> {D, E}
R13 = {B,F} → F13 = {B} -> {F}
R21 = {F, G, H} → F21 = {F} -> {G,H}
R22 = {D, I, J} → F22 = {D} -> {I,J}

3NF:
R11 = {A,B,C} → F11 = {A,B} -> {C}
R12 = {A,D,E} → F12 = {A} -> {D}, {A} -> {E}
R13 = {B,F} → F13 = {B} -> {F}
R21 = {F, G, H} → F21 = {F} -> {G}, {F} ->{H}
R22 = {D, I, J} → F22 = {D} -> {I}, {D} -> {J}

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

15.25
2NF:
R1 = {A,B,C} → F1 = {A, B} -> {C}
R2 = {B,D,E,F} → F2 = {B,D} -> {E,F}
R3 = {A,D,G,H} → F3 = {A,D} -> {G,H}
R4 = {A,I} → F4 = {A} -> {I}
R5 = {H,J} → F5 = {H} -> {J}

3NF: **need to add primary key relation**, did not exist before
R1 = {A,B,C} → F1 = {A, B} -> {C}
R2 = {B,D,E,F} → F2 = {B,D} -> {E}, {B,D} -> {F}
R3 = {A,D,G,H} → F3 = {A,D} -> {G}, {A,D} -> {H}
R4 = {A,I} → F4 = {A} -> {I}
R5 = {H,J} → F5 = {H} -> {J}
R6 = {A, B, D} → F6 =

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 → 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} → F1 = {Book_title} -> {Publisher, Book_type}
R2: {Book_type, List_price} → F2 = {Book_type} -> {List_price}
R3: {Author_name} → F3 = {Author_name} -> {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} → F1 = {Book_title} -> {Publisher}, {Book_title} -> {Book_type}
R2: {Book_type, List_price} → F2 = {Book_type} -> {List_price}
R3: {Author_name} → F3 = {Author_name} -> {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 → not in BCNF since it's stricter.

c) $R_{1,}$ & R_2 = M | M → (R_1 - R_2) or **M →(R_2 - R_1)** ? M→{MP, C} is in the closure of F which can be derived from the transitive relation M → MP → 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

$$\begin{tikzcd} 1 \arrow[r] & 3 \end{tikzcd}$$

Y

$$\begin{tikzcd} 3 \arrow[r] & 2 \end{tikzcd}$$

Z

$$\begin{tikzcd} 1 \arrow[r] & 2 \end{tikzcd}$$

S_2

X

$$\begin{tikzcd} 1 & 3 \arrow[l] \end{tikzcd}$$

Y

$$\begin{tikzcd} 2 \arrow[r, shift left] & 3 \arrow[l, shift left] \end{tikzcd}$$

Z

$$\begin{tikzcd} 1 & 2 \arrow[l] \end{tikzcd}$$

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