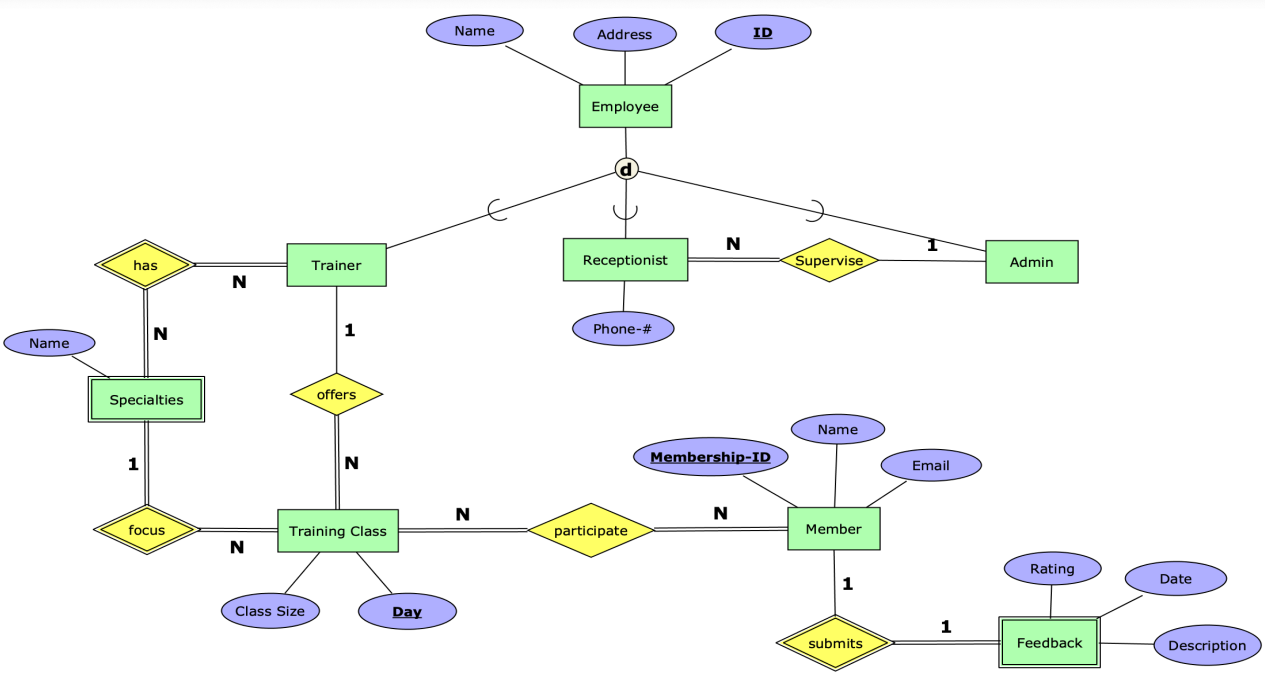
COMP2400 Assignment 2

u7205329

October 2020

1 EER Diagram



Assumptions

- Each employee can only take on one role.
- Some trainers may share the same specialty.
- There can be many classes that focus on the same specialty.
- At least one member needs to participate in the class for the training class to go on.
- Two members cannot together write a feedback.

Requirements that cannot be captured

- We cannot show the constraint on the Class Size attribute of the Training Class entity to be no more than 10. This requirement belongs to the Domain Constraint.
- We also cannot show that the Rating attribute of the Feedback entity can only range from 1 to 5. This requirement also belongs to the Domain Constraint.

2

2.1 Candidate Keys

Our goal is to find the set of the candidate keys of R. Observing the set $\sum$ of FDs, it appears that D never appears in the dependent of any FD. It can be concluded that D must be part of each candidate key we are looking for.

We can find the closure of D and the combination of other attributes with D to see if they form a candidate key. We have $D^+ = D$ so D alone is not a candidate key of R.

$$\begin{aligned}(AD)^+ &= ABD \quad (A \to B)\\ &= ABCD \quad (AB \to C)\\ &= ABCDE \quad (BC \to AE).\end{aligned}$$

As $(AD)^+ = R$, any other combination with AD as a proper subset is not a candidate key of R.

$$\begin{aligned}(DE)^+ &= DEAB \quad (DE \to AB)\\ &= DEABC \quad (AB \to C).\end{aligned}$$

As $(DE)^+ = R$, any other combination with DE as a proper subset is not a candidate key of R.
BD and CD are not candiate keys of R as $(BD)^+ = BD$ and $(CD)^+ = CD$. However,

$$(BCD)^+ = BCAED \quad (BC \to AE).$$

There are no other combinations of attributes with D that form a candidate key. Hence, the set of candidate keys for R is $\{AD, DE, BCD\}$.

2.2 Minimal Cover

We start off with $\sum_m = \sum = \{A \to B, AB \to C, BC \to AE, DE \to AB\}$. We first apply the dependent property on $\sum_m$ such that the right hand side of each FD should has only one attribute. We rewrite $\sum_m$ as F_2 as follows

$$F_2 = \{A \to B, AB \to C, BC \to A, BC \to E, DE \to A, DE \to B\}$$

Given that C is in the closure of A ($A^+ = ABCE$), we can replace $AB \to C$ to $A \to C$. Furthermore, if DE $\to$ A and A $\to$ B, then DE $\to$ B is true and can be implied from the other 2 FDs. Hence, the FD DE $\to$ B is redundant and can be removed. We get the final set of FDs as follows:

$$F_3 = \{A \to B, A \to C, BC \to A, BC \to E, DE \to A\}$$

Finally, the set of FDs F_3 is the minimal cover of R.

3 BCNF Decomposition

Let us rename the attributes of **Appointment** by C (Customer), B (Brance), D (Date), T (Time), S (Staff), and R (Room). The relation schema **Appointment** is not in BCNF since the functional dependency {S $\to$ B} violates the properties of BCNF since Staff is not a candidate key for **Appointment**. Only {CBDT $\to$ SR} and {BDTR $\to$ C} do not violate because CBDT and BDTR are among the superkeys of **Appointment**.

We decompose **Appointment** by using one of the violating FDs {S $\to$ B}. We replace **Appointment** by two seperate relations R_1 = {S,R} and R_2 = {C,B,D,T,S}. Both R_1 and R_2 are in BCNF. However, this BCNF decomposition is **not** dependency preserving as the only FD we have now is {S $\to$ B} while the other FDs in $\sum$ are lost.

4 Relational Algebra

4.1 Query Processing

4.1.1 Question a

The question is asking to List the CourseNo of courses without any tutors in 'S2 2020'. Firstly, we list all the courses that are offered in 'S2 2020'. Note that we denote the space in 'S2 2020' here as '-'.

$$R_1 = \sigma_{Semester='S2-2020'}(Course)$$

We then find the list of courses with tutors in "S2 2020".

$$R_2 = \sigma_{Semester='S2-2020'}(Tutor) \bowtie \sigma_{Semester='S2-2020'}(Course)$$

Finally, we take the difference $R_1 - R_2$, that is, we take the *CourseNo* of the list of courses that are offered in "S2 2020" but not in the list of courses with tutors in "S2 2020".

$$Result = \pi_{CourseNo}(R_1 - R_2)$$

4.1.2 Question b

The question is asking to List the names of students who have been enrolled in a same course for at least two times. Our initial goal is to find the *SID* of the students who have enrolled in the same course at least twice, whose information can be derived from the **Enrol** table.
We first need to create a copy of **Enrol** named R_{11} with the attribute *SID*, *CourseNo*, and *Semester* and the attribute *Semester* to be renamed by and *Sem*. We keep the name of SID and CourseNo the same for a natural join operation later. As this is directly from the **Enrol** table, we know each tuple in here representing a student enrolling in a course. In other words, every student ID exists in this table have enrolled in at least one course.

$$R_{11} = \rho_{R_{11}(SID,CourseNo,Sem)}(\pi_{SID,CourseNo,Semester}(Enrol))$$

A projection operation is then performed on the original copy of **Enrol** by keeping the three attributes SID, CourseNo, and Semester.

$$R_{12} = \pi_{SID,CourseNo,Semester}(Enrol)$$

We then perform natural join operation on R_{11} and R_{12}.

$$R_{13} = R_{11} \bowtie R_{12}$$

The table R_{13} has a format of 4 columns (SID, CourseNo, Sem, Semester), where the two table were joined by the SID and CourseNo attributes since they share the same name. We then select the tuples where Sem is not equal to Semester. By keeping only the SID attribute, we get the list of SIDs of students who have enrolled in the same course at least twice.

$$R_{Enrol} = \pi_{SID}(\sigma_{Sem \neq Semester}(R_{13}))$$

As we want the names of the student, we can use the foreign key reference between **Student** and **Enrol** where [SID] $\subseteq$ Student[SID]. Thus, we first select the columns of *SID* and *Name* in **Student**.

$$R_{Student} = \pi_{SID,Name}(Student)$$

Performing natural join on **Enrol** and **Student**, we now have at table of two columns: the SIDs and their corresponding names. Finally, the result can be achieved by keeping the column *Name*.

$$Result = \pi_{Name}(R_{Enrol} \bowtie R_{Student})$$

4.1.3 Question c

Our approach is to find the LIDs of lecturers who have taught at least two courses in 'S1 2020' and then the LIDs of those who have taught at least three. The difference of these two tables will give us the LIDs of those who taught exactly two. Since *Email* is not the primary key of **Lecturer**, we cannot uniquely identify the lecturers by *Email*. We can identify the corresponding emails from the LIDs instead.

Taught at least two
We will use the same approach from the question above by first selecting the tuples in **Lecturer** where Semester = 'S1 2020' and performing projection by keeping only the columns *LID* and *CourseNo*. We then create another copy of this.

$$R_{11} = \pi_{LID,CourseNo}(\sigma_{Semester='S1-2020'}(Lecturer))$$

$$R_{12} = \pi_{LID,CourseNo}(\sigma_{Semester='S1-2020'}(Lecturer))$$

We then change the *CourseNo* attribute name in one of the copies (R_{11}) to *CNo* for later comparison.

$$R_{11} = \rho_{(LID,CNo)}(R_{11})$$

Finally, by performing natural join on R_{11} and R_{12} and selecting the tuples where $CNo \neq CourseNo$, we successfully achieve a relation of 3 attributes (*LID*, CNo, *CourseNo*) where each *CNo* and *CourseNo* in each tuple are different. We perform projection by keeping only the *LID* for later unique identification.

$$R_{atleast2} = \pi_{LID}(\sigma_{CNo \neq CourseNo}(R_{11} \bowtie R_{12}))$$

Taught at least three
We perform the same steps as above, however, now with three copies.

$$R_{21} = \pi_{LID,CourseNo}(\sigma_{Semester='S1-2020'}(Lecturer))$$

$$R_{22} = \pi_{LID,CourseNo}(\sigma_{Semester='S1-2020'}(Lecturer))$$

$$R_{23} = \pi_{LID,CourseNo}(\sigma_{Semester='S1-2020'}(Lecturer))$$

We again rename the *CourseNo* attribute to *CNo* and *CNo2* in R_{21} and R_{23}, respectively.

$$R_{21} = \rho_{(LID,CNo)}(R_{21})$$

$$R_{23} = \rho_{(LID,CNo2)}(R_{21})$$

We then perform natural join on the three tables.

$$R_2 = R_{21} \bowtie R_{22} \bowtie R_{23}$$

At this point, the table R_2 will have the format of 4 columns (*LID*, *CNo*, *CourseNo*, *CNo2*). We then selet the tuples where the CNo, CourseNo, and CNo2 are completely different from each other. We again keep only the LID attribute.

$$R_{atleast3} = \pi_{LID}(\sigma_{CNo \neq CourseNo \wedge CNo \neq CNo2 \wedge CourseNo \neq CNo2}(R_2))$$

Exactly two
Finally, since we need the emails of the lectures at the end, we will identify the emails by the lecturers' LIDs.

$$R_{LectEmail} = \pi_{LID,Email}(Lecturer)$$

Since $R_{atleast3}$ is the list of LIDs of the lecturers who taught at least 3 courses and $R_{atleast2}$ is the list of LIDs who taught at least 2. By taking the difference, we get the list of LIDs of those who taught exactly 2. We then perform natural join on the resulting difference table and $R_{LectEmail}$, we get an intermediate result of 2 tables: LIDs and their corresponding emails. The final can be achieved by keeping the *Email* attribute.

$$Result = \pi_{Email}((R_{atleast3} - R_{atleast2}) \bowtie R_{LectEmail})$$

4.2 Query Optimisation

4.2.1 Question a

The question is asking to list the Lecture ID and Student ID of the Students and Lecturers who share the same name and the Student belongs to the CESC college. The given relation algebra query perform Cartesian product on two full tables which is not efficient. We approach this problem by performing selection and projection first instead. Firstly, for the **Lecturer** table, we keep only the *LID* (for result) and *Name* (for comparing with Student) attributes.

$$R_1 = \pi_{LID,Name}(Lecturer)$$

For the **Student** table, we first select the tuples with College = 'CESC' which will reduce the number of rows immensely. Then we perform projection by keeping only the *SID* (for result) and *Name* attributes.

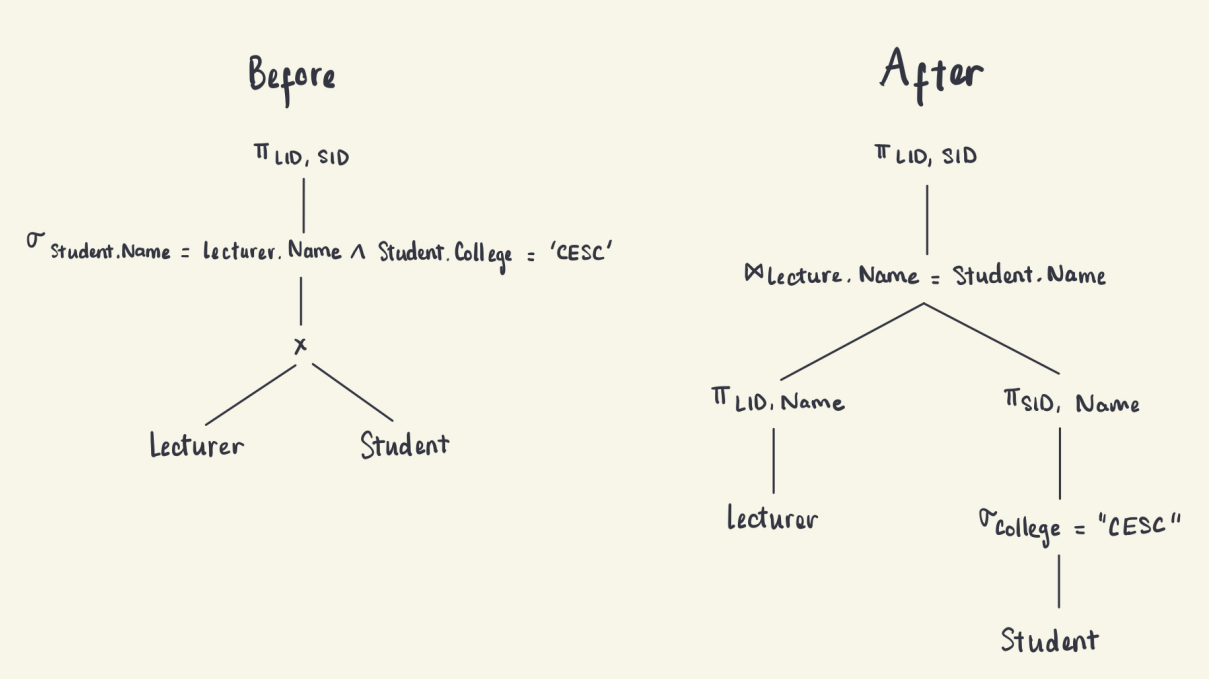
$$R_2 = \pi_{SID,Name}(\sigma_{College='CESC'}(Student))$$

We can then join the tables by performing the Join operation.

$$R_3 = R_1 \bowtie_{Lecture.Name=Student.Name} R_2$$

At this point, the intermediate table R_3 has four attributes LID, SID, R1.Name and R2.Name where each tuple satisfies the condition that the Lecture's Name is the same as the Student's Name. Finally, the result can be achieved by keeping only the LID and SID attributes.

$$Result = \pi_{LID,SID}(R_3)$$



4.2.2 Question b

The question is asking to list the SID of students whose College is CESC or have withdrawn from a course. The given relation algebra performing join on three full tables which is an inefficient practice as the intermediate table will have at most m x n x p tuples.

Our approach is to push down selection and projection wherever we can first and join the table later when it is appropriate.

Given that **Enrol** and **Course** have the reference relationship based on two attributes CourseNo and Semester, we will process them first. Firstly, we select all tuples with the Status attribute equal to 'Withdrawn' in **Enrol**, which will hugely reduce the number of tuples. We then project the attributes SID, CourseNo, Semester to remove irrelevant attributes and prepare for the joining operation with **Course**.

$$R_{11} = \pi_{SID,CourseNo,Semester}(\sigma_{Status='withdrawn'}(Enrol))$$

With **Course**, we project the attributes *CourseNo* and *Semester* also in preparation for the joining operation.

$$R_{12} = \pi_{CourseNo,Semester}(Course)$$

At this time, it is appropriate to perform natural join on R_{11} and R_{12}. The intermediate result is the table of 3 attributes SID, CourseNo, and Semester, each tuple represents an enrolment of a Student whose status is 'withdrawn'. Hence, we can then select the SID attribute only.

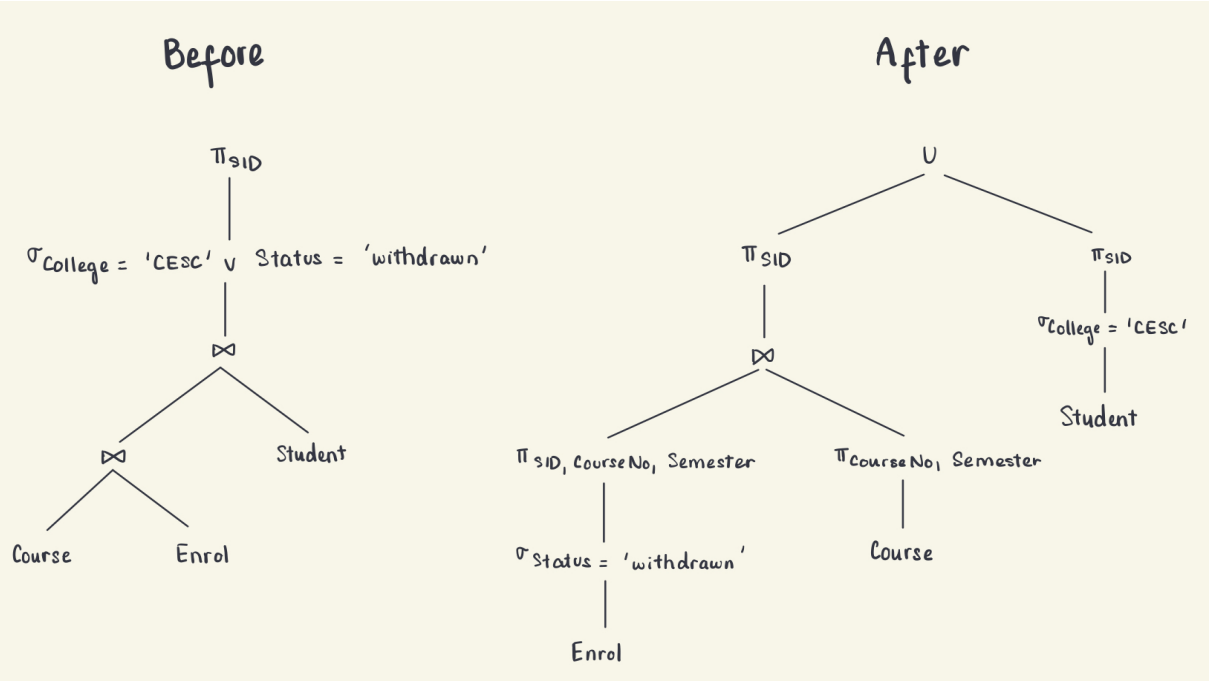
$$R_1 = \pi_{SID}(R_{11} \bowtie R_{12})$$

With **Student**, we select the tuples with *College* = 'CESC' and then project on the SID attribute only.

$$R_2 = \pi_{SID}(\sigma_{College='CESC'}(Student))$$

Finally, the result can be achieved by performing union operation on the two relations R_1 and R_2.

$$Result = R_1 \cup R_2$$


Before
π_{SID}
$\sigma_{College = 'CESC' \vee Status = 'withdrawn'}$
$\bowtie$
$\bowtie$
Student
Course
Enrol
After
$\cup$
π_{SID}
π_{SID}
$\bowtie$
$\sigma_{College = 'CESC'}$
Student
$\pi_{SID, CourseNo, Semester}$
$\pi_{CourseNo, Semester}$
$\sigma_{Status = 'withdrawn'}$
Course
Enrol