**Fall 2022**

# COMP2411 Tutorial No. 4 (Week 8)

1. A database is to be set up to maintain the pool of lecture theatres and to assist in their allocation to courses. Consider the following relation and the set of functional dependencies F defined on its attributes:

```
CourseRmAlloc(CourseId, CourseName, Year, Lecturer, Enrollment,
              RoomId, RoomCapacity, Day, Time)

F = {CourseId -> CourseName,
     CourseName -> CourseId,
     CourseId, Year -> Lecturer,
     CourseId, Year -> Enrollment,
     CourseId, Year, Day, Time -> RoomId,
     RoomId -> RoomCapacity,
     RoomId, Year, Day, Time -> CourseId}
```

After decomposition of the tables, we arrive at a set of tables:

```
Course(CourseId, CourseName)
CourseEnroll(CourseId, Year, Lecturer, Enrollment)
RoomAlloc(RoomId, Year, Day, Time, CourseId)
Room(RoomId, RoomCapacity)
```

Please verify that these are all in BCNF, and translate the following relational algebra expressions into plain language in English:

*a)* $\prod_{RoomID}(Room) - \prod_{RoomID}\left(\sigma_{Day\ =\ '15-Sep-2022'}\ (RoomAlloc)\right)$

*b)* $\prod_{Lecturer,RoomID,Year,Day,Time}(CourseEnroll * RoomAlloc)$

**Solution:**

For the four tables, `Course` table in 3NF since there no "partial dependence" nor "transitive dependence" cases; also, since it has no overlapping candidate keys, it follows that it is also in BCNF.

For `CourseEnroll` table, again it does not have any "partial dependence" nor "transitive dependence" cases, so it's in at least 3NF; furthermore, since {`CourseId, Year`} is the only candidate key, it follows that 3NF and BCNF are equivalent for this case!

For `RoomAlloc` table, the determinants (i.e., the left-hand-side) of the two functional dependencies are candidate keys, hence it satisfies the BCNF definition therefore in BCNF.

For `Room` table, the only candidate key is `RoomId`, so it is clear that it is in BCNF.

The meanings of the two algebra expressions:

a) Display all the rooms which are not booked on the day 15-Sep-2022.

b) Display the info of "what room is allocated to which lecturer on which time slot".

2. Consider the relations ***r1(A,B,C)***, ***r2(C,D,E)***, and ***r3(E,F)***, with primary keys **A**, **C**, and **E**, respectively. Assume that ***r1*** has 1000 tuples, ***r2*** has 1500 tuples, and ***r3*** has 750 tuples. Estimate the size of ***r1 * r2 * r3*** (*where "*" denotes natural join*), and solve the following two problems:

a) We have two ways to do the natural joins:
   i. ***r1*** with ***r2*** first and then with ***r3*** or
   ii. ***r2*** with ***r3*** first and then with ***r1***.

   Which one is more efficient in terms of comparisons?

b) Assume that every primary key has a dense index built already. Give a most efficient strategy for computing the join.

**Solution:**

a) For (*i*) in the worst case we need *1000*1500+1000*750* comparisons. This is because ***C*** is the primary key in ***r2***, we know that at most one tuple in r2 will match to a specific tuple in ***r1***. Therefore, in the result of ***r1 * r2***, there are at most 1000 tuples.

   For (*ii*) in the worst case we need *1500*750+1500*1000* comparisons. Therefore, (*i*) is more efficient in terms of comparisons.

b) For any tuple from ***r1***, we want to find the matching tuples from ***r2*** and ***r3***. By using the dense index, we can find the matching tuples in the following way.
   1. Take out one tuple from ***r1*** and assume that this tuple has value ***c*** for attribute ***C***.
   2. Use the index on ***C*** in table ***r2*** and find the tuple in ***r2*** whose value for ***C*** is ***c*** (*suppose this tuple has value **e** for **E***).
   3. Use the index on ***E*** in table ***r3*** and find the tuple in ***r3*** whose value for ***E*** is ***e***.

   These three tuples' combination will be one tuple in the result ***r1 * r2 * r3***.

   Because finding tuples by using dense index will be very fast, *Step 2* and *Step 3* consume little time