CSC 370
Database Systems

Assignment 2

Yasaman Zarrinkia

Spring 2020

Part A

1. ***A* → *B*, *BC* → *D*, *A* → *C***

   Clearly, $\{A\}^+ = \{A, B, C, D\}$, and A is the only key. Since BC is not a super-key, $BC \rightarrow D$ is a BCNF violation.

   - $R_1 = \{B, C\}^+ = \{B, C, D\}$
   - $R_2 = R - \left(\{B, C\}^+ \setminus \{B, C\}\right) = \{A, B, C\}$

   FD's for the first relation: $BC \rightarrow D$. Since $\{B, C\}$ is the only key for the first relation, it is in BCNF.

   FD's for the second relation: $A \rightarrow B,\ A \rightarrow C$. Since $\{A\}$ is the only key for the second relation, it is in BCNF.

   Suppose tuple $t = abcd$ is the join of tuples projected onto R_1 and R_2. Tableau for the decomposition of R onto $\{A, B, C\}$ and $\{B, C, D\}$ is shown in Table 1.

A	B	C	D
a_1	b	c	d
a	b	c	$\cancel{d_2}\, d$

   Table 1: Tableau for the decomposition of R onto $\{A, B, C\}$ and $\{B, C, D\}$

   First row corresponds to set attributes B , C and D, and the second row corresponds to set attributes A , B and C. Both of these rows are the tuples of the original relation R, and they have the same values for B and C attributes. Using FD $BC \rightarrow D$ will deduce that d_1 and d are the same. So we can replace d_1 by d. Therefore, the first row becomes equal to t. As one of them got equal to the original tuple, so the decomposition produces a lossless join.

2. ***AB* → *C*, *AB* → *D*, *C* → *A*, *D* → *B***

   $\{A\}^+ = \{A\}$

   $\{B\}^+ = \{B\}$

   $\{C\}^+ = \{A, C\}$

   $\{D\}^+ = \{B, D\}$

   $\{A, B\}^+ = \{A, B, C, D\}$

   $\{B, C\}^+ = \{A, B, C, D\}$

   $\{C, D\}^+ = \{A, B, C, D\}$

   $\{A, D\}^+ = \{A, B, C, D\}$

   $\{A, D\}^+ = \{A, B, D\}$

$\{A, C\}^+ = \{A, C\}$

So, the keys are $\{A, B\}, \{B, C\}, \{C, D\}, \{A, D\}$. Therefore, $C \to A$ and $D \to B$ violate BCNF.

Consider $C \to A$.

- $R_1 = \{C\}^+ = \{A, C\}$.
- $R_2 = R - \left(\{C\}^+ \setminus \{C\}\right) = \{B, C, D\}$.

FD's correspond to R_1: $C \to A$. It is in BCNF, because C is a key and therefore it is a super-key.

FD's correspond to R_2: $D \to B$. Since D is not a super-key, it is not in BCNF and we need to decompose it.

- $R_3 = \{D\}^+ = \{B, D\}$.
- $R_4 = R_2 - \left(\{D\}^+ \setminus \{D\}\right) = \{C, D\}$.

FD's correspond to R_3: $D \to B$. It is in BCNF, because D is a key and therefore it is a super-key. Since there is no FD that correspond to R_4, it is in BCNF.

Therefore, our decomposition consists of $R_1 = \{A, C\}$, $R_3 = \{B, D\}$ and $R_4 = \{C, D\}$. Suppose tuple $t = abcd$ is the join of tuples projected onto R_1, R_3 and R_4. Tableau for the decomposition of R onto $\{A, C\}$, $\{B, D\}$ and $\{C, D\}$, respectively, is showed in Table 2.

A	B	C	D
a	b_1	c	d_1
a_2	b	c_2	d
$\cancel{a_3}\, a$	$\cancel{b_3}\, b$	c	d

Table 2: Tableau for the decomposition of R onto $\{A, C\}$, $\{B, D\}$ and $\{C, D\}$

We deduced the last row using $C \to A$ and $D \to B$. As the last row got equal to the original tuple t, the resulting decomposition produces a lossless join.

Part B

1. **Keys for Stocks:**

   - $\{B\}^+ = \{B, O\}^+ = \{B, O\}$
   - $\{O\}^+ = \{O\}$
   - $\{C\}^+ = \{B, C\}^+ = \{O, C\}^+ = \{B, C, O\}$
   - $\{S\}^+ = \{S, R\}^+ = \{S, R\}$
   - $\{Q\}^+ = \{Q\}$
   - $\{R\}^+ = \{R\}$
   - $\{B, S\}^+ = \{B, O, S, R\}$
   - $\{B, Q\}^+ = \{B, O, Q\}$
   - $\{B, R\}^+ = \{B, O, R\}$
   - $\{O, S\}^+ = \{O, S, R\}$
   - $\{O, Q\}^+ = \{O, Q\}$
   - $\{O, R\}^+ = \{O, R\}$
   - $\{\boldsymbol{C}, \boldsymbol{S}\}^+ = \{\boldsymbol{B}, \boldsymbol{O}, \boldsymbol{C}, \boldsymbol{S}, \boldsymbol{Q}, \boldsymbol{R}\}$

   As C and S does not appear on the right hand side of any rule, so any key(super-key) must contain them. Therefore, the only key is $\{C, S\}$, and we do not need to continue our computation for founding a key.

2. They are in their minimal basis. First of all, all of them have singleton right sides. Now we need to check if we remove any of them, the result is no longer a basis.

   - $\boldsymbol{S \to R}$. Without this rule, $\{C, S\}$ is not a key anymore. $\{C, S\}^+ = \{B, O, C, S, Q\} \neq \{B, O, C, S, Q, R\}$.
   - $\boldsymbol{C \to B}$. Without this rule, $\{C, S\}$ is not a key anymore. $\{C, S\}^+ = \{C, S, Q\} \neq \{B, O, C, S, Q, R\}$.
   - $\boldsymbol{CS \to Q}$. Without this rule, $\{C, S\}$ is not a key anymore. $\{C, S\}^+ = \{B, O, C, S, R\} \neq \{B, O, C, S, Q, R\}$. Moreover, if we remove C from the left hand side, then $\{S\}^+ = \{S, R, Q\} \neq \{S, R\}$. Similarly, if we remove S from the left hand side, then $\{C\}^+ = \{B, C, O, Q\} \neq \{B, C, O\}$.
   - $\boldsymbol{B \to O}$. Without this rule, $\{C, S\}$ is not a key anymore. $\{C, S\}^+ = \{B, C, S, Q\} \neq \{B, O, C, S, Q, R\}$.

3. Using each FD, we define their corresponding relations. We construct one schema for each FD in the minimal basis. Schema is the union of the left and right hand sides of each FD.

- $\boldsymbol{R_1} = \{\boldsymbol{S}, \boldsymbol{R}\}$. The correspondence FD for this relation is $S \to R$. Since S is a key of this relation, R_1 is in 3NF.
- $\boldsymbol{R_2} = \{\boldsymbol{C}, \boldsymbol{B}\}$. The correspondence FD for this relation is $C \to B$. Since C is a key of this relation, R_2 is in 3NF.
- $\boldsymbol{R_3} = \{\boldsymbol{C}, \boldsymbol{S}, \boldsymbol{Q}\}$. The correspondence FD for this relation is $CS \to Q$. Since $\{C, S\}$ is the key of this relation, R_3 is in 3NF.
- $\boldsymbol{R_4} = \{\boldsymbol{B}, \boldsymbol{O}\}$. The correspondence FD for this relation is $B \to O$. Since B is a key of this relation, R_4 is in 3NF.

Since the key $\{C, S\}$ is contained in a FD, then there is no need to add one relationship whose schema is some key, because we already have relation R_3 whose schema is a super-key.
As all the relations R_1, R_2, R_3 and R_4 are in 3NF, we can get both lossless-join and dependency-preservation.

4. As all of the relations R_1, R_2, R_3 and R_4 have a key of their own relation on the left-hand side of their FD's, then they are all in BCNF.

Part C

Our design has following entities and their associated attributes.

- **AcademicUnit**(<u>UnitCode</u>, FullName, MainOffice)
- **GradStudent**(<u>SIN</u>, Name, StartYear, Program)
- **Project**(<u>ID</u>, FundingAgency, StartDate, FundingMonths, Budget)
- **Professor**(<u>SIN</u>, Name, StartYear, Rank, HomeUnit)

There are some relations between entities that are described below.

- A *GradStudent* has exactly one **HomeUnit** among *AcademicUnits*, but an *AcademicUnit* can be a **HomeUnit** for many *GradStudents*. So the relation **HomeUnit** is a many-to-one relation from *GradStudent* to *AcademicUnits*.
- Each *GradStudent* which **WorksOn** a *Project* has a unique *Professor* as a supervisor. A *GradStudent* can **WorkOn** one or many *Projects* and a *Professor* can **WorkOn** multiple *Projects*. So the relation **WorksOn** is many-to-many between *GradStudent* and *Project*, but it is many-to-one from *Project* and *GradStudent* to *Professor*.
- An *AcademicUnit* **Leads** by exactly one *Professor*, and each *Professor* **Leads** at most one *AcademicUnit*. So **Leads** is a one-to-one relation.
- A *Professor* **WorksAt** one or two *AcademicUnits* with a *Percentage* of their dedication for each *AcademicUnit* that they are working at, and each *AcademicUnit* can have multiple *Professors* which are **WorkAt** that *AcademicUnit*. So **WorksAt** is a many-to-many relation with an attribute *Percentage*.
- Exactly one *Professor* works **AsPI** on one or multiple *Projects*. So **AsPI** is a many-to-one relation from *Project* to *Professor*.
- A *Project* has one or more *Professors* **AsCI**, and each *Professor* can be on multiple *Projects* **AsCI**. So **AsCI** is a many-to-many relation between *Professor* and *Project*.

Figure 1 shows the ER diagram for the university with the information of the problem statement. Noted that, we have used degree constraints for some of our relationships. For example, we have used $\geq$ **1** for **AsCI** on the professor side; which means a project has at least one *Professor* **AsCI**.

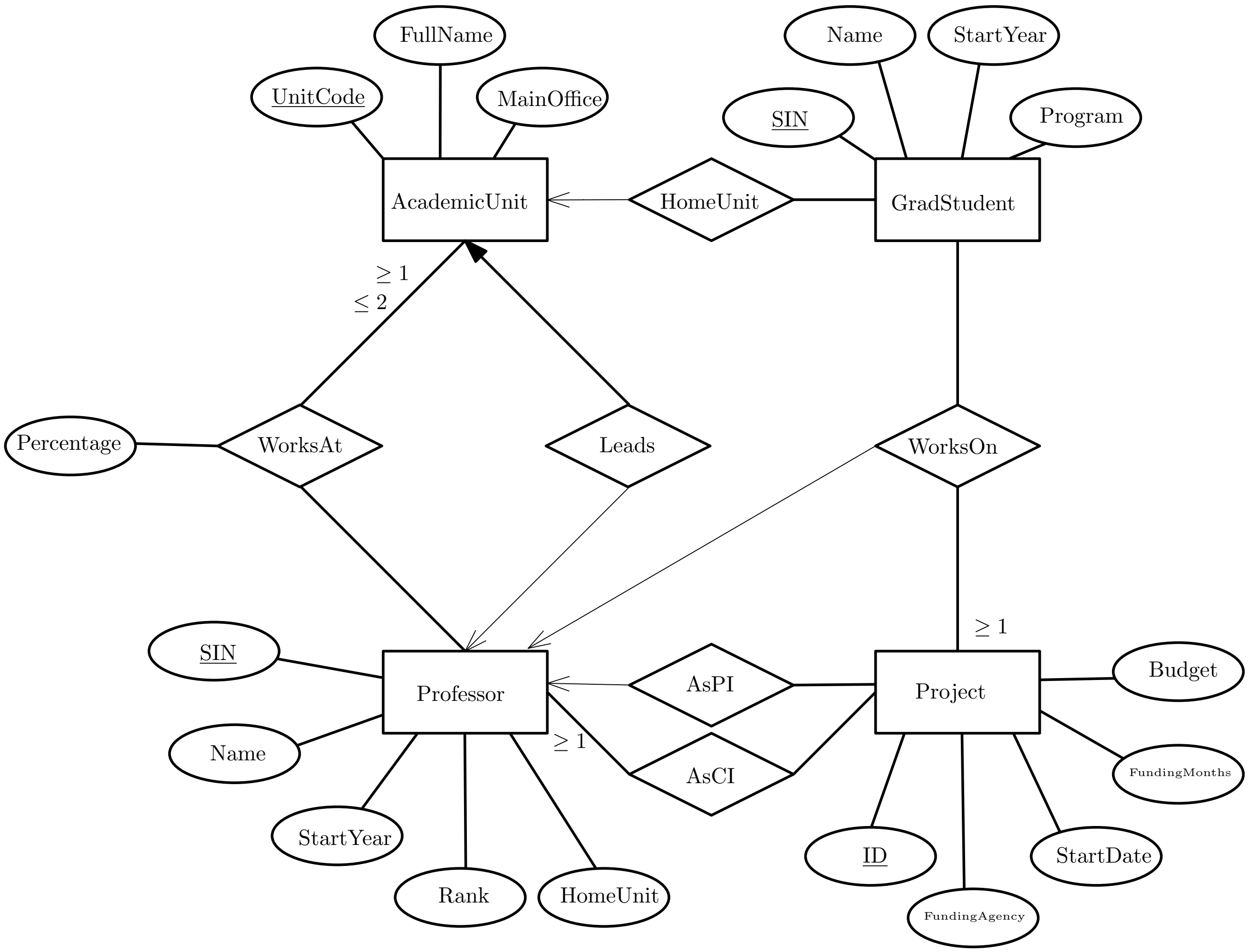

Figure 1: ER diagram of the University

Part D

First we define the relations and their attributes. An entity becomes a relation, and entity set attributes become the attributes associated to that relation. So the relations are as follows.

- *Flights*(aircraft,day, number)
- *Customers*(SIN , name, addr , phone)
- *Bookings*(row, seats, day, number, SIN)

Since *Bookings* entity set is a weak entity set, its corresponding relation must include attributes for its complete key, including those belong to supporting entity sets. In this case, the supporting entity sets' keys are **day** and **number** for flights, and **SIN** for customers. Also, it must include its own non-key attributes, i.e. **row** and **seats**.