**DTA (M) Technical Report 2019** Wesley Scott 2460681S

**Task 1:**

*Tables*

STUDENT (SID, SNAME, HCID, HCNAME, TID, TNAME, JYEAR)

TOPIC(SID, ADVNAME, ADVID, TOPIC)

TEXTBOOK(COURSE, ADVID, TEXTBOOK)

**Task 1.1:**

*TABLE: STUDENT*

FD1: SID > {SNAME, HCID}

The unique student ID entry defines the student’s name and their home city.

FD2: HCID > HCNAME

The students home city ID defines home city name.

Therefore, SID > HCNAME transitive dependency.

FD3: TID > TNAME

The unique team ID (TID) defines the team name.

FD4: SID, TID > JYEAR

Join year is defined by a combination of the unique student ID and their team ID, as students can have joined multiple teams over the years.

*TABLE: TOPIC*

Each TOPIC a student has their own advisor; the same student can have multiple advisors depending on their topics, therefore…

FD1: SID, TOPIC > ADVID

ADVID is provided by a combination of SID and TOPIC. An advisor may only advise one TOPIC.

FD2: ADVID > ADVNAME

Each unique advisor has their own name.

*TABLE: TEXTBOOK*

Each course has a set number of TEXTBOOK, these textbook(s) must be accounted for every time an ADVIOR for a COURSE is referenced in TEXTBOOK.

To uniquely identify a tuple in this table currently we need all three attributes to act as a primary key.

FD1: COURSE, ADVID, TEXTBOOK > COURSE, ADVID, TEXTBOOK

**Task 1.2:**

In TABLE: STUDENT: We have FD4 which states that join year to a team is defined by both the student ID and the team ID. We encounter an issue when a player joins more than one team within the same year, as we would not know the student’s current team. \*\*\*

In TABLE: TOPIC: In topic we are told that an advisor can only cover one topic (FD2), so due to the database design, when inserting a new student with an advisor already present in the table, (see ADVID 1 (McReader)), we must duplicate information in this tables topic attribute. This means that if an Advisor were to change Topic, we would have to update many tuples.

In TABLE: TEXTBOOK: Due to the nature of the relationships of this table, where each Advisor will use all of the necessary textbooks within a course, every time we insert a new ADVID against a course, we will require multiple tuples for each of the textbooks applicable to that course with the new Advisor.

**Task 1.3:**

*TABLE:*

STUDENT (SID, SNAME, HCID, HCNAME, TID, TNAME, JYEAR)

2NF:

STUDENT(SID, SNAME, HCID(FK), TID(FK!), JYEAR)

SID and TID are the Primary Keys, HCID becomes the foreign key, referencing HOMECITY(HCID), due to STUDENT FD1 noted in task 1.1, SID > HCID > HCNAME

HOMECITY(HCID, HCNAME)

HCID becomes the primary key of a new relation HOMECITY.

TEAM(TID, TNAME)

Team ID becomes the primary key of a new relation TEAM and is referenced by STUDENT to give the value of JYEAR. TEAM is in BCNF.

3NF/BCNF:

STUDENT(SID, SNAME, HCID(FK))

STUDENT\_JOINYEAR(SID, TID(FK), JYEAR)

HOMECITY(HCID, HCNAME)

TEAM(TID, TNAME)

STUDENT and STUDENT\_JOINYEAR’s non-prime attributes now directly depend on every candidate key. This also removes the need for two FKs In relation STUDENT.

*TABLE: TOPIC*

TOPIC(SID, ADVNAME, ADVID, TOPIC)

ADVNAME is given by ADVID, and is transitively dependent on {SID, TOPIC}

2NF/3NF?:

Split Topic into two relations, Advisee and Advisor.

ADVISEE\_TOPIC(SID, TOPIC, ADVID(FK))

A student can have multiple Advisors, one for each topic, therefor SID and TOPIC are required to uniquely identify a tuple.

ADVISOR(ADVID, ADVNAME)

ADVID uniquely identifies ADVNAME, which would be partially dependant on primary key before split.

**Task 1.4:**

Due to the nature of the relationship in this table, where each Advisor will use all of the necessary textbooks within a course, every time we insert a new ADVID against a course, we will require multiple tuples for each of the textbooks applicable to that course.

**TEXTBOOK**  
  
TEXTBOOK(COURSE, ADVID, TEXTBOOK)

Therefore, I suggest a split of the table as follows, which can be further integrated and normalised with the other tables.

COURSE\_TEXTBOOK(COURSE, TEXTBOOK);

COURSE\_ADVISOR(ADVID, COURSE);