University of Glasgow | School of Computing Science

Assessed Coursework

Course Name	**Database Theory & Applications DTA(M)**			
Coursework Number	**1/1**			
Deadline	**Time:**	**16h00**	**Date:**	**10/12/2019**
% Contribution to final course mark	**20%**			
Solo or Group ✓	**Solo**	✓	**Group**	
Anticipated Hours	**Average 20 hours**			
Submission Instructions	**Submission of only <u>one</u> PDF document; please read in the description.**			
Please Note: This Coursework cannot be Re-Assessed				

Code of Assessment Rules for Coursework Submission

Deadlines for the submission of coursework which is to be formally assessed will be published in course documentation, and work which is submitted later than the deadline will be subject to penalty as set out below.

The primary grade and secondary band awarded for coursework which is submitted after the published deadline will be calculated as follows:

(i) in respect of work submitted not more than five working days after the deadline
   a. the work will be assessed in the usual way;
   b. the primary grade and secondary band so determined will then be reduced by two secondary bands for each working day (or part of a working day) the work was submitted late.

(ii) work submitted more than five working days after the deadline will be awarded Grade H.

Penalties for late submission of coursework will not be imposed if good cause is established for the late submission. You should submit documents supporting good cause via MyCampus.

Penalty for non-adherence to Submission Instructions is: 2 bands

Technical Report 2019

Task 1: Relational Schema Modelling [Marks: 30]

Consider the following tables/relations storing information about students in a University.

The table: **STUDENT** stores information about the unique ID of a student (attribute: SID), their name (attribute: SNAME), and the unique ID of the student's home city and home city name (attributes: HCID and HCNAME, respectively). Moreover, in the same table, we store information about the particular team where a student belongs to a particular year (unique Team ID and Team Name in the attributes: TID and TNAME, respectively) and the year (attribute: JYEAR) when the student joins a team. It is possible that a student can join to zero or one team in a specific year only, as you can see from the STUDENT instance.

For instance: Chris is a student (SID=1) whose home city is Bradford (HCID=BD) and has joined to Team 1 (TID=1) and to Team 2 (TID=2) in 2019 and 2020, respectively. Philip's home city is Munich and has joined to Team 2 in 2019.

Table 1: STUDENT

SID	SNAME	HCID	HCNAME	TID	TNAME	JYEAR
1	Chris	BD	Bradford	1	Team1	2019
2	Stella	RM	Rome	1	Team1	2019
3	Philip	MU	Munich	2	Team2	2019
4	John	GL	Glasgow	2	Team2	2018
1	Chris	BD	Bradford	2	Team2	2020
4	John	GL	Glasgow	1	Team1	2017
2	Stella	RM	Rome	2	Team2	2020

The table: **TOPIC** stores information about students' advisers and their associated topics. For each topic (attribute: Topic), a student, who is also referred to as 'advisee' (attribute: SID) has a specific advisor (attribute: ADVID uniquely identifies an adviser and ADVNAME is the adviser's name). Moreover, each advisor is responsible for a single specific topic. In the instance below, the student with SID=1 has the adviser McReader with ADVID = 1 on the topic Database.

Table 2:TOPIC

SID	ADVNAME	ADVID	TOPIC
1	McReader	1	Database
1	Burn	2	Java
2	Edward	3	Database
3	McReader	1	Database

The table: **TEXTBOOK** stores information about the courses (attribute: Course), the corresponding adviser (attribute: ADVID) and the associated textbook(s). For any course, there is a fixed set of advisers, e.g., for the 'Databases' course, there are two advisers: 1 and 2, i.e., McReader and Burn (see table Topic for the names of the advisers). Moreover, for any course, there is a fixed set of textbooks, which is independent of the corresponding adviser. For instance, for the course 'Advanced DB', there are three textbooks: 'Intro to DB', 'Java & DB', and 'Oracle DB', while for the course 'Databases' there are two textbooks: 'Intro to DB' and 'DB Management'.

Table 3:TEXTBOOK

COURSE	ADVID	TEXTBOOK
Databases	1	Intro to DB
Databases	1	DB Management
Databases	2	Intro to DB
Databases	2	DB Management
Advanced DB	1	Intro to DB
Advanced DB	1	Java & DB
Advanced DB	1	Oracle DB

Task 1.1: In this task, you are about to determine the possible Functional Dependencies (FDs) of the relations/tables: **STUDENT**, **TOPIC**, and **TEXTBOOK** derived from the provided instances. For each identified FD, provide a short explanation.

[Marks 5]: Show the possible FDs; for each FD, give a short explanation on the involved attributes.

Task 1.2: Based on the FDs identified in Task 1.1, for each table, provide only one example of a possible modification anomaly (choose either insertion, or deletion, or update) and provide an explanation about this anomaly.

[Marks 5]: For each table/relation, describe an example case where an anomaly can occur, e.g., when you delete a tuple, or update an attribute value, or insert a new tuple.

Task 1.3: Based on the FDs identified in Task 1.1, normalize the Tables: **STUDENT** and **TOPIC** only in the highest possible Normal Form (NF). For each step on normalization, provide a short explanation on the FDs you base your normalization process.

[Marks 10]: Normalize the STUDENT and TOPIC relations only, starting from their current NF; at each step of normalization, check whether a relation violates a NF given its FDs.

Task 1.4: Propose how we could split the relation/table **TEXTBOOK** in 'smaller' relations such that you can eliminate or reduce possible modification anomalies (choose a split e.g., to reduce or eliminate anomalies derived from insertion, or deletion, or update). Provide the rationale of your proposed split including an example anomaly in the original TEXTBOOK relation and how your proposed new schema after the split eliminates or reduces this anomaly. In your new split relations, clearly define the primary keys, if any.

[Marks 10]: First, find out a modification anomaly based on the current TEXTBOOK relation (you can use the instance provided). This can be, e.g., deleting a tuple, updating an attribute value or inserting a new tuple. Based on this anomaly, try to split the TEXTBOOK relation into two, or more than two,

relations and, then, check again if the same modification has now been eliminated or its impact is significantly reduced.

Task 2: SQL Statements [Marks 40]

Task 2.1: Based on your proposed SQL normalized schema from Task 1 of the Tables: **STUDENT** and **TOPIC** only, provide the **SQL CREATE TABLE** statements of your schema and **enhance** the CREATE statements with Referential Integrity Constraints in terms of UPDATE and DELETE CASCADE options. Explain the constraints you declared.

[Marks 5]: Provide a SQL CREATE TABLE SCHEMA of your normalized STUDENT and TOPIC relations only. Include certain constraints in your schema especially focusing on triggering and cascading actions like ON UPDATE CASCADE, ON DELETE SET NULL, etc.

Task 2.2: Provide the SQL SELECT statements of the following queries, based on your normalized schema from the Task 2.1.

- **SQL 1:** Show the number of students per home city. **[Marks 5]**
- **SQL 2:** For the year 2001, show the number of students per team. **[Marks 10]**
- **SQL 3:** Show the names of the advisers with more than 10 advisees. **[Marks 5]**
- **SQL 4:** Which topic(s) has the most advisers? **[Marks 10]**
- **SQL5:** Which student(s) has joined the most teams since 2001 (incl. 2001)? **[Marks 10]**

Note: Your answered statements are marked against correct SQL syntax and rationale, i.e., that the provided SQL query indeed reflects its description. You can also provide instances/examples of the involved tables in each SQL query to provide evidence that your SQL works correctly.

Task 3: Data Cleaning & Analytics Query [Marks 30]

Consider the relation **EMPLOYEE(SSN, FName, Salary).** The database designer did not declare in the schema that the SSN attribute (Social Security Number) is a Primary Key (they might have forgotten it…). Hence, there may be duplications of employees…Your task is to 'clean' the EMPLOYEE relation by finding duplicate entries.

Task 3.1: Provide a SQL query which returns the duplicate employee entries. An example is provided:

EMPLOYEE

SSN	FName	Salary
1	Philip	30000
1	Philip	30000
2	Stella	40000
2	Stella	40000
3	Iona	50000
1	Philip	30000

Your SQL query then should show only the duplicate tuples (Philip and Stella in this example):

SSN	FName	Salary
1	Philip	30000
2	Stella	40000

[Marks 10]: Provide a SQL query with the above-mentioned functionality and 'test' it over the example EMPLOYEE relation to check whether you obtain a 'clean' relation after its potential execution (as the example provided).

Task 3.2: Provide a SQL query which counts the duplicate employee entries. Considering the example in Task 3.1, the SQL should return the value of 3, since there are 3 extra employee entries (2 extra for Philip and one extra for Stella).

[Marks 10]: Provide a SQL query with the above-mentioned functionality and 'test' it over the example EMPLOYEE relation to obtain the total number of duplicates.

Task 3.3: Assume now that the SSN attribute is declared as the Primary Key of the relation EMPLOYEE, thus, there are **no** duplicates. In this case, your task is to provide a SQL query which shows the top-3 highest salary values **without** using the SQL LIMIT operator. **Any provided SQL query using the SQL LIMIT operator is not going to get any grade.**

[Marks 10]: First, assume that there are no duplicates, since SSN is now a PK of the relation EMPLOYEE. Then, provide a SQL query with the above-mentioned functionality and 'test' it over an instance of the EMPLOYEE relation without duplicates to obtain only those tuples corresponding to the top-3 highest salaries. Do not use the LIMIT operator in your answer (otherwise, this will be graded with 0 marks). Note: it might be the case that more than one employee has the same salary.

Notes & Submission

Note 1: The Assessed Coursework is graded out of **100 Marks**: Task 1 [30 marks], Task 2 [40 marks], and Task 3 [30 marks].
Note 2: Answer to ALL tasks.
Note 3: Submit a document including your details and **all** answers in **one** PDF document file.