THE UNIVERSITY OF AUCKLAND

Semester One, 2020
Campus: City

FINAL ASSESSMENT

SOFTWARE ENGINEERING

Fundamentals of Database Systems

(Time Allowed: 24 Hours)

NOTE:

- This is an assessment which you have 24 hours to complete and submit on Canvas.
- This assessment is due online by 12:59pm on July 4, 2020 NZST. Remember not to submit at precisely 1:00pm.
- It is your responsibility to ensure your answer is successfully submitted on time. Ensure it is downloadable and clearly readable by the assessors.
- By submitting this assessment, you confirm that:
  - You have completed this assessment with integrity and honesty and not committed any academic misconduct (i.e. cheating, plagiarism, assisting others to cheat or copy, engaging the use of third-party assistance including from your fellow classmates, families or friends).
  - Your submission represents your individual effort and does not contain plagiarised material.
  - You are responsible to ensure that no one copies your assessment.
  - You are responsible to fully comply with the University's Regulations, Statues and Guidelines as stated and available at http://www.auckland.ac.nz/uoa/home/about/teaching-learning/honesty/tl-uni-regs-statutes-guidelines

Question	Mark	Out Of
1		10
2		25
3		25
4		20
5		15
6		5
Total		100

By completing this assessment, I agree to the following declaration:

I understand the University expects all students to complete coursework with integrity and honesty. I promise to complete all online assessment with the same academic integrity standards and values. Any identified form of poor academic practice or academic misconduct will be followed up and may result in disciplinary action.

As a member of the University's student body, I will complete this assessment in a fair, honest, responsible and trustworthy manner. This means that:

- *I declare that this assessment is my own work, except where acknowledged appropriately (e.g., use of referencing).*
- *I will not seek out any unauthorised help in completing this assessment.*
- *I declare that this work has not been submitted for academic credit in another University of Auckland course, or elsewhere.*
- *I am aware the University of Auckland may use Turnitin or any other plagiarism detecting methods to check my content.*
- *I will not discuss the content of the assessment with anyone else in any form, including, Canvas, Piazza, Facebook, Twitter or any other social media within the assessment period.*
- *I will not reproduce the content of this assessment anywhere in any form.*

In agreeing the above declaration, please go to Quizzes tab of Canvas and then complete the Academic Honesty Declaration.

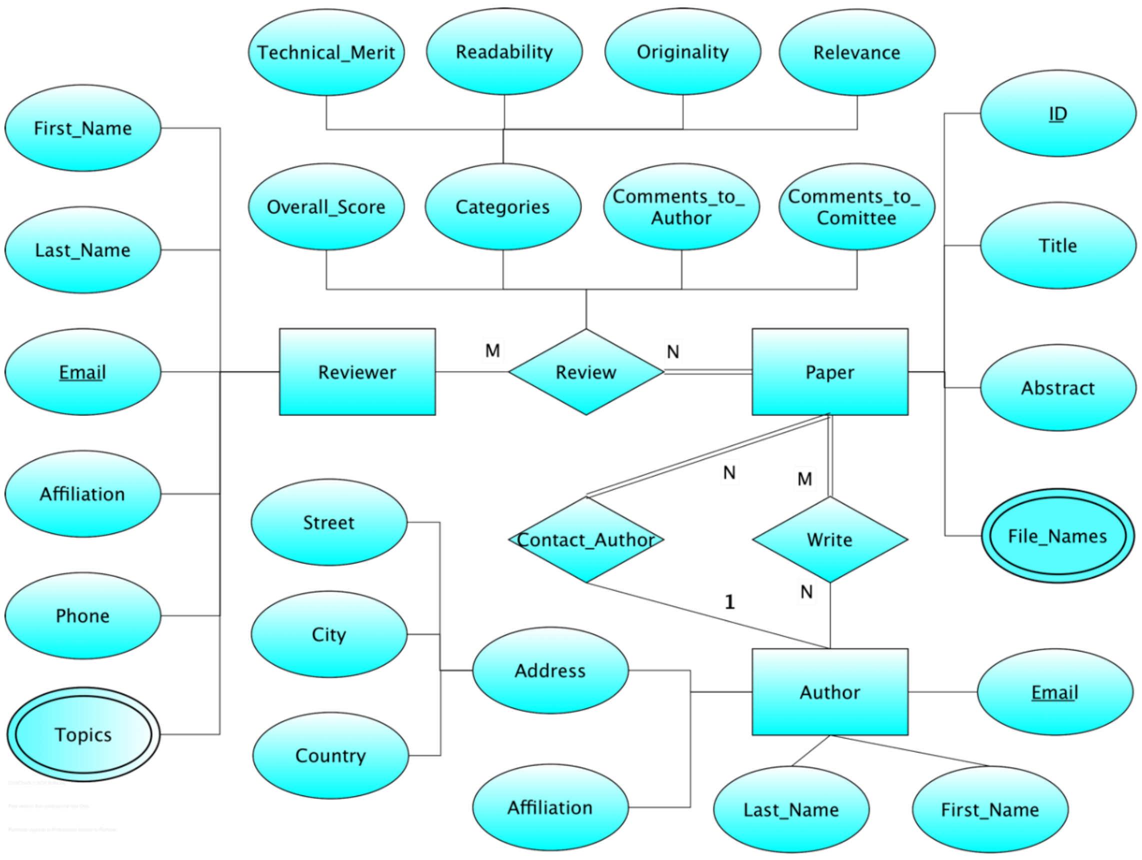
Student support instructions:

If any corrections are made during the 24 hours, you will be notified by a Canvas Announcement. Please ensure your notifications are turned on during this period.

If you wish to raise concerns during the Final Assessment, please call the Contact Centre for advice:
Auckland: 09 373 7513, Outside Auckland: 0800 61 62 63, International: +64 9 373 7513.
It is your responsibility to ensure your assessment is successfully submitted on time. Please don't leave it to the last minute to submit your assessment.
For any Canvas issues, please use 24/7 help on Canvas by chat or phone.

Question 1 (10 marks)

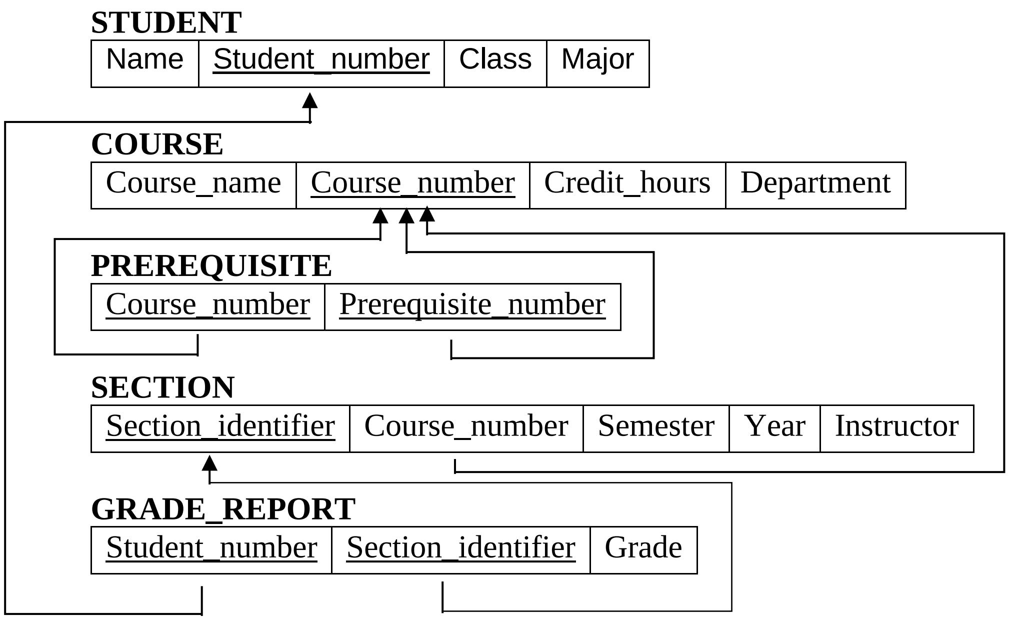
Using the algorithm taught in the course, map the following `Conference-Review` ER diagram into a relational database schema. Specify all primary keys and foreign keys.



[10 marks]

Question 2 (25 marks)

The following questions use the UNIVERSITY database schema and the sample database state shown below.



STUDENT

Name	Student_number	Class	Major
Smith	17	1	CS
Brown	8	2	CS

COURSE

Course_name	Course_number	Credit_hours	Department
Intro to Computer Science	CS1310	4	CS
Data Structures	CS3320	4	CS
Discrete Mathematics	MATH2410	3	MATH
Database	CS3380	3	CS

SECTION

Section_identifier	Course_number	Semester	Year	Instructor
85	MATH2410	Fall	07	King
92	CS1310	Fall	07	Anderson
102	CS3320	Spring	08	Knuth
112	MATH2410	Fall	08	Chang
119	CS1310	Fall	08	Anderson
135	CS3380	Fall	08	Stone

GRADE_REPORT

Student_number	Section_identifier	Grade
17	112	B
17	119	C
8	85	A
8	92	A
8	102	B
8	135	A

PREREQUISITE

Course_number	Prerequisite_number
CS3380	CS3320
CS3380	MATH2410
CS3320	CS1310

a) Define the following SQL statements:

i. Set the default value of the attribute 'Department' in the COURSE table to 'CS'.

[3 marks]

ii. Add a new attribute 'Birth_Date' to the STUDENT table with the 'DATE' type as its domain.

[3 marks]

iii. Change the grade value of the student number '17' with the section number '119' in the 'GRADE_REPORT' table to 'B' (assume that the domains of the Student_number and Section_identifier attributes are both CHAR).

[4 marks]

iv. Define a query to retrieve the transcript records of the student number '8'. Transcript includes the course name, course number, credit hours, semester, year, and grade for each course completed by the student.

[5 marks]

b) Define the following queries using Relational Algebra expressions:

i. For each section taught by Professor 'Anderson', retrieve the course name, course number, semester, year, and number of students who took the section.

[5 marks]

ii. Retrieve the names and major departments of the students who did not study all the courses that were offered in 2008.

[5 marks]

Question 3 (25 marks)

A B$^+$-tree on an attribute X can be alternatively constructed in a batched manner:

1. Sequentially store the relation by sorting the records in a non-descending order of X;
2. Build the first-level index (the smallest key of each block in the index can be safely removed), then the second- level index, $\cdots$, until the top level has only one block.

Consider 19 records inserted sequentially (not in ascending order) to a relation. The key values (in the order of the insertion) are 38,12,9,6,31,42,33,8,2,15,5,7,11,23,22,34, 27, 49, 50 respectively. Each block can hold up to 3 data records. Each block can hold up to 4 pointers together with 4 integers. Consider the two methods in building a B$^+$-tree:

a) Start with an initially empty B$^+$-tree, insert the records sequentially. Draw the snapshots of the B$^+$-tree after the insertion of 6, 7 and 50, respectively.

[12 marks]

b) Construct the B$^+$-tree in a batched manner:

i. Show the process of sorting the records (use a box to denote a block) and indicate the I/O cost of the sorting process (for simplicity, each record needs to be read once and written once in a pass), given that the memory can hold up to 3 blocks of data.

[9 marks]

ii. Draw the structure of the B$^+$-tree after the construction.

[4 marks]

Question 4 (20 marks)

Consider three relations x(A, B), y(B,C) and z(A,C). x has a primary index on the key of combined attributes (A, B); y has a primary index on the key of combined attributes (B, C); z has a primary index on the key of combined attributes (A, C). Both x(A, B) and y(B, C) have clustering index on B. z(A,C) has a clustering index on A and a secondary index on C. The parameters ($|x|$ denotes the number of tuples in relation x) are:

- x(A,B): $|x| = 500, V(A,x) = 30, V(B,x) = 50$,
- y(B,C): $|y| = 400, V(B,y) = 30, V(C,y) = 100$,
- z(A,C): $|z| = 300, V(C,z) = 40, V(A,z) = 50$,
- The memory can hold up to m = 102 pages,
- Each block holds 2 tuples for all relations including intermediate results.

Find the best "index-based" plan for $x \bowtie y \bowtie z$, i.e., the plan only considers index-based join algorithms introduced in the course without sorting any relation. Explain your answer by analyzing the I/O cost (under the default uniform distribution assumption, only count the I/Os for retrieving the underlying tuples). Assume that tuples of each attribute value, if the relation is stored sequentially, is aligned with the block boundaries. This problem does not consider pipeline, but the final join result does not need to be outputted.

[20 marks]

Question 5 (15 marks)

Consider the four transactions below:

- $T_1 : r_1(A); w_1(C); w_1(B);$
- $T_2 : r_2(B); w_2(D); w_2(A);$
- $T_3 : r_3(C); w_3(B);$
- $T_4 : w_4(D);$

a) Insert locks using conservative (static) 2PL: A transaction T requests all the shared and exclusive locks needed by T immediately before the first operation of T while a lock of T is released immediately after its final use by T. Explain why conservative (static) 2PL can prevent deadlock, i.e., eventually all transactions can complete.

[4 marks]

b) Draw the precedence graph of the schedule

$$S : r_1(A); r_2(B); w_1(C); w_2(D); r_3(C); w_3(B); w_1(B); w_4(D); w_2(A);$$

and then indicate if the schedule is conflict-serializable.

[3 marks]

c) Consider schedule S under a lock scheduler: each transaction requests the (shared / exclusive, depending on the operation) lock of the corresponding data item immediately before an operation and release all the locks immediately after the last operation of the transaction.

i. Find the first deadlock of the schedule by drawing the corresponding wait-for graph.

[3 marks]

ii. Explain why the wait-die protocol can prevent this deadlock. Assume that the transactions arrive to the system in the order of T_1,T_2,T_3,T_4.

[5 marks]

Question 6 (5 marks)

Consider a system that uses the immediate update protocol (steal/no-force) with checkpointing. Describe the recovery process from a system crash for the following transactions:

1. T_1 has committed before the checkpoint,
2. T_2 started before the checkpoint but has committed after the checkpoint,
3. T_3 started before the checkpoint but has not committed before the crash,
4. T_4 started after the checkpoint but has committed before the crash,
5. T_5 started after the checkpoint but has not committed before the crash.

Specifically, for each transaction, indicate the action (e.g., redo and undo), the application scope (i.e., operations before the checkpoint or after the checkpoint) and the order (i.e., new to old, or old to new) of operations on which the action should be applied. An operation (data access operation) w is newer than another operation w' if w is logged later than w'.

[5 marks]