SIM GLOBAL EDUCATION

UNIVERSITY OF WOLLONGONG AUSTRALIA

School of Computing and Information Technology

Student to complete:

Family name	
Other names	
Student number	
Table number	

CSCI235
Database Systems

Final Examination Paper
Session 2 2022

Exam duration	3 hours and 15 minutes
Start time	10:00 am
End time	1:15 pm
Weighting	40% of the subject assessment
Marks available	40 marks
Items permitted by examiner	Text-book, Lecture slides, and Tutorial notes
Directions to students	4 questions to be answered. Marks for each question are shown beside the question. All answers must be hand-written with a **BLACK** or **DARK BLUE** pens on white pieces of paper in A4 or foolscap paper. No writing on iPad or other tablet is allowed.

This examination is a take-it-home examination to be done on-line on the date of examination.

Version 2.0

Instructions to students

The final examination is an '**individual-effort**' assessment; that is, the final examination is to be done individually without collaboration and/or help from others. The total mark for the final examination assessment is 40 marks. The procedures for conducting and for students to take the final examination are as follow:

1. There are four (4) questions each with multiple parts. The time needed to complete each question ranges from 40 minutes to 50 minutes. The final examination questions will be released to students at 10:00 am Singapore Time via Moodle. The duration of the final examination is 3 hours and 15 minutes of submission time is provided after the three (3) hours examination.
2. The solution to all the examination questions **MUST** be **hand-written** with a **BLACK** or **DARK BLUE** pens on white pieces of paper in A4 or foolscap paper. Any typed solution will **NOT** be evaluated. **NO iPad or other tablet is allowed**.
3. Please write your student number and name on the top-right-hand of each page and label all questions properly.
4. All submissions of the solution must be in **pdf, png, or jpg format**. In the event that students need to scan a hand-written answer, the students must cater to the time to do the scanning. NO extra time will be given for such a case.
5. All submitted files must be named using the format: UOWStudentNumber-CSCI235. For example, 6354789-CSCI235 is the filename submitted by a student with student number 6354789. If you need a mobile app that can take pictures of your solutions and save them as pdf, you may consider the Office Lens. **Note: Please make sure you check your scanned solutions properly before you submit them to Moodle. It is your responsibility to ensure that all your answers are scanned. Any answer that is left out is considered an un-answered question and zero mark will be awarded for that question.**
6. Throughout the examination time, ALL forms of communication are NOT allowed. For example, to answer or make a phone call, messaging, and use of WhatsApp, telegrams, emails, Skypes, Zoom, uploading of answers to the cloud, etc.
7. You are allowed to clarify if the specification of the question is correct. However, you are **NOT** allowed to ask to explain the meaning or how to do the questions.
8. Students can start working on the questions immediately after the final examination questions are made available. Five (5) minutes before the end of the examination, the submission link will be open for students who have completed the examination early to start submitting their solutions. The submission link will be closed fifteen (15) minutes after the allocated time. **No submission will be allowed thereafter, and no submission through email will be accepted.**

Question 1 – (Total 10 marks)
Functional Dependency and Normalization

a) Consider the relational schemas given below and the respective sets of functional dependencies valid in the schemas. For each of the relational schemas **identify its highest normal form**. Remember the identification of a normal form requires analysis of the valid functional dependencies and the minimal keys. **A solution with no comprehensive analysis of the valid functional dependencies and the minimal keys scores no marks**.

(i) `P = (S, N, C, A, K)`

```
S->N
S->A
C->N
```

(1.0 mark)

(ii) `P = (R, N, O, C, E)`

```
R->N
R->E
O->C
O->R
```

(1.0 mark)

(iii) `R = (A, B, C)`

`The attributes A, B, and C do not have any functional dependency among them.`

(1.0 mark)

(iv) In a product promotion fair, promoters are engaged to promote various products. A promoter may promote more than one product, and each product may be promoted by many promoters. A promoter is paid by commission, and the commission is computed based on a total sale for the product, for example, if the total sale for a product is below $1000, a promoter is paid 10% of the total sale for commission; if the total sale for a product is between $1000 and $5000, a promoter is paid 20%, etc. The information about the commission is stored in the following relational table.

COMMISSION (PromoterId, ProductId, TotalSale, CommissionPaid)

(2.0 marks)

b) Consider the following un-normalized relational table FOODORDER and its functional dependencies below:

FOODORDER (OrderNum, FoodNum, FoodName, FoodPrice, Quantity, TableNumber, WaiterID, WaiterName)

The attributes of FOODORDER table satisfy the following properties:

- Each order may consist of many food items,
- Each waiter is responsible for many tables,
- Each table is in-charge by only one waiter,
- For each food item in an order, customer may order more than one serving.

Normalize the relational table FOODORDER into a minimal number of relational tables in BCNF.

(5.0 marks)

Question 2 - (Total 10 marks)
Indexing

Consider a relational database that consists of the relational tables created by the following CREATE TABLE statements:

```
CREATE TABLE BOOK (
B_ISBN          NUMBER(15)        NOT NULL,   /* Unique ISBN of book */
B_TITLE         VARCHAR2(15)      NOT NULL,   /* Title of book */
B_PUBDATE       VARCHAR2(15)      NOT NULL,   /* Publication date of book */
B_SUBJECT       VARCHAR2(30)      NOT NULL,  /* Subject of book */
B_COST          NUMBER(6,2)       NOT NULL,  /* Cost of book */
  CONSTRAINT BOOK_PKEY PRIMARY KEY (B_ISBN) );
```

In addition to the primary key index, the administrator has created an additional index `bookIdx(B_PUBDATE, B_SUBJECT, B_COST)` over the relational table BOOK.

a) Find **SELECT** statements that will use the index in the ways specified in the question (i) to (v) below. The values used to create the query such that the queries can meet the specified criteria is up to you.

(i) Execution of the first `SELECT` statement must traverse the index **vertically** and it **must not** access the relational table BOOK.

(1.0 mark)

(ii) Execution of the second `SELECT` statement must traverse the index **vertically** and later on **horizontally** and it **must not** access the relational table BOOK.

(1.0 mark)

(iii) Execution of the third `SELECT` statement must traverse the leaf level of the index **horizontally** and it **must not** access the relational table BOOK.

(1.0 mark)

(iv) Execution of the fourth `SELECT` statement must traverse the index **vertically** and it **must** access the relational table ITEM.

(1.0 mark)

(v) Execution of the fifth `SELECT` statement must traverse the index **vertically** and later on **horizontally** and it **must** access the relational table BOOK.

(1.0 mark)

b) Consider the following SELECT statement:

```
SELECT      B_TITLE, COUNT(*)
FROM        BOOK
WHERE       B_SUBJECT = 'Database Design'
AND         B_COST = 70.0
GROUP BY B_TITLE
HAVING COUNT(*) > 2
ORDER BY B_TITLE;
```

(i) Find the best index based on a single column (i.e., an index that consist of only one attribute) to speed up the processing of the query given above (Q2b). Write an SQL 'create index' statement to create the index. Write a brief explanation explaining how the index on single column is used when the query is processed.

(1.5 marks)

(ii) Find the best composite index based on two columns (i.e., an index that consists of two compounded attributes) to speed up the processing of the query given above. Write an SQL 'create index' statement to create the index. Write a brief explanation explaining how the index on two columns is used when the query is processed.

(1.5 marks)

(iii) Find the best composite index based on three columns (i.e., an index that consists of three compounded attributes) to speed up the processing of the query given above. Write an SQL 'create index' statement to create the index. Write a brief explanation explaining how the composite index is used when the query is processed.

(2.0 marks)

Question 3 refers to the relational database with a schema described below:

The hospital database contains information about the treatments patients get from doctors. The information on the prescriptions ordered by the doctors is also stored in the database. The schemas of relational tables, meanings of their attributes and specifications of primary, candidate, and foreign keys are given below.

HOSPITAL	
HospitalCd	Hospital Code
Name	Name of the hospital
Address	Address of the hospital
Estate	The estate where the hospital is located
PostalCode	The postal code of the address
EstablishedDate	The date the hospital was established
Primary key = HospitalCd	

DOCTOR	
DoctorId	The identification number of the doctor
Name	Name of the doctor
Citizenship	The citizenship of the doctor
WorkFor	The hospital the doctor is working for
Primary key = DoctorId	
Foreign key = WorkFor references HOSPITAL (HospitalCd)	

PATIENT	
NRIC	The NRIC of the patient
Name	The name of the patient
DateOfBirth	The birth date of the patient
Sex	The gender of the patient
Address	The residential address of the patient
Estate	The estate where the patient lives
PostalCode	The postal code of the address
Primary key = NRIC	

TREATMENT	
DoctorId	The doctor identification number
NRIC	The NRIC of the patient
TreatmentDate	The treatment date
Description	Description of the treatment
Primary key = (DoctorId, NRIC, TreatmentDate)	
Foreign key = (DoctorId) references DOCTOR(DoctorId)	
Foreign key = (NRIC) references PATIENT(NRIC)	

PRESCRIPTION	
DoctorId	The doctor identification number
NRIC	The NRIC of the patient
TreatmentDate	The treatment date
ItemNum	The item number of the prescription
Drug	The drug prescribed by the doctor
Dosage	The dosage prescribed
Primary key = (DoctorId, NRIC, TreatmentDate, ItemNum)	
Foreign key = (DoctorId, NRIC, TreatmentDate) references TREATMENT(DoctorId, NRIC, TreatmentDate)	

Question 3 - (Total 10 marks)
PL/SQL

a) Implement a **stored PL/SQL function** PATIENT(DoctorId) that finds the NRIC of all patients who received three or more treatments from the same doctor (DoctorId). The function should return a string of NRIC separated with one or more spaces. Assuming the identifier of the doctor (DoctorId) treating the patients is passed to the function as a value of parameter.

Hint: Consider creating a cursor of the form:

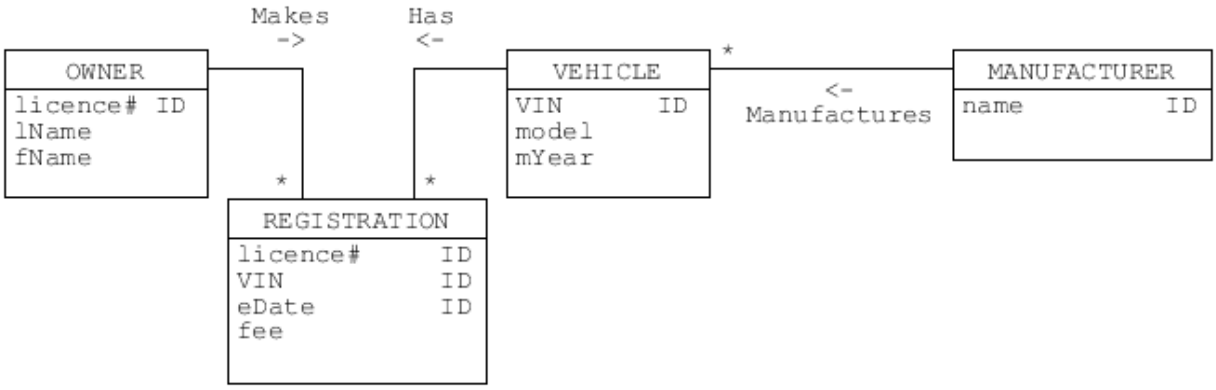
```
SELECT   an attribute
FROM     some table
WHERE    some condition
GROUP BY an attribute
HAVING count(*) > 2;
```

(5.0 marks)

b) Implement a **stored PL/SQL procedure** that adds to the database information about a new doctor. The information about the new doctor should be passed into the procedure through the values of parameters. The procedure must check the validity of hospital code before the verification of a referential integrity constraint. A hospital code is a sequence of characters that starts from a letter 'H' and is followed by two digits. **(5.0 marks)**

Question 4 - (Total 10 marks)
BSON document

a) Consider the following conceptual schema of a sample database that contains information about a vehicle registration done by an owner of a vehicle.



Write a sample JSON document that has a structure consistent with the conceptual schema given above. Your document must contain information about at least one manufacturer, three vehicles, two owners, and two registrations. One of the owners has made two registrations and another one has made one registration. The values for the attributes are up to you but must be sensible.

(5.0 marks)

b) Consider a sample BSON document given below. Assume that all documents in a collection `empeProject` has the same structure as the document shown below:

```
db.empeProject.insert({
"_id":"empe001",
"EMPLOYEE": {"empeId": "empe001",
          "fName":"James",
          "lName":"Bond",
          "email":"jamesbond@hotmail.com",
          "experience": ["Database Design",
                         "SQL",
                         "Java"],
          "Workon": {"EMPLOYEEPROJECT": [
                                    {"projId":"proj001",
                                     "hoursWorked": 4},
                                    {"projId":"proj003",
                                     "hoursWorked": 5}
                                    ]}
         }
});

db.empeProject.insert({
"_id":"empe002",
"EMPLOYEE": {"empeId":"empe002",
          "fName":"Harry",
          "lName":"Potter",
          "experience": [ "Data Warehouse",
                          "SQL",
                          "Spark Scala",
                          "Java Scripts"],
          "Workon": {"EMPLOYEEPROJECT": [
                                    {"projId":"proj002",
                                     "hoursWorked": 6}
                                    ]}
         }
});

db.empeProject.insert({ "_id":"proj001",
                      "projectTitle": "Install MongoDB" });
db.empeProject.insert({ "_id":"proj002",
                      "projectTitle": "Install Oracle" });
db.empeProject.insert({ "_id":"proj003",
                      "projectTitle": "Install Spark" });
```

Use either a method *find*() or a method *aggregate*() available in MongoDB to write the implementations of the following queries. Implementation of each query is worth 1 mark.

(i) Find the first name (fName) and last name (lName) of all employee who have experience in Database Design. Do not show the object identifier (_id).

(1.0 mark)

(ii) Find the employee id (empeId) and hours worked in project (hoursWorked) of all employee who worked in project "proj003".

(1.0 mark)

(iii) Find all employees who possess 4 experiences. Show only the employee's information.

(1.0 mark)

Use the method *update*() to write the implementations of the following data manipulation operations. Implementation of each data manipulation operation is worth 1 mark.

(iv) Add a new experience "HIVE" to the employee whose empeId is 'empe001'.

(1.0 mark)

(v) Change the email account for employee empe001 to "jamesbond$hotmail.com".

(1.0 mark)

END OF EXAMINATION