EBU5602 Past paper 2 sample solution

Question 1

a) Explain 3 advantages of using a database system for storing information compared with storing information in a number of files on the computer.

	Do not write in this column
Control of data redundancy, Data consistency, More information from the same amount of data, Sharing of data, Improved data integrity, Improved security, Enforcement of standards, Economy of scale, Balance conflicting requirements, Improved data accessibility and responsiveness, Increased productivity, Improved maintenance through data independence, Increased concurrency, Improved backup and recovery services	
Note: the above advantages need to be explained.	
	6 marks

b) The following schema describes a database application for storing student data:

```
Student (studentId, studentName, email, phone)

Module (moduleId, moduleName, moduleLeader, semester)

Module_registration(moduleId, studentId, status)

Lecturer(lecturerId, lecturerName, officeNo)
```

where `Module_registration.status` gives the status of the student with respect to that module, e.g. passed, failed, resitting, etc. `moduleLeader` in relation `Module` is a foreign key referencing `lecturerId` in relation `Lecturer`.

Express the following queries in Relational Algebra:

i) List the names and module leaders' ID for all modules with a module code higher than 100.

ii) List all modules along with the details of their module leader.

iii) List all modules led by the lecturer whose name is "John White".

iv) List the IDs of all students currently taking the "Computer Science" module.

i)	Do not write in this column
$\Pi_{\text{moduleName, moduleLeader}} (\sigma_{\text{moduleId} > 100} (\text{Module}))$	
ii)	
Module ⋈ Lecturer	
iii)	
$\Pi_{\text{moduleId, moduleName}} (\sigma_{\text{lecturerName = "John White"}} (\text{Module} \bowtie \text{Lecturer}))$	

iV)	
$\Pi_{\text{studentId}}$ ($\sigma_{\text{moduleName = “Computer Science”}}$ (Module ⋈ Module_registration)	
	10 marks

c) Explain the referential integrity rule of the Relational model and give an example of referential integrity.

Referential Integrity	**Do not write in this column**
If foreign key exists in a relation, either foreign key value must match a candidate key value of some tuple in its home relation or foreign key value must be null.	
e.g. in relation student(studentId, name, gender, programmeId) and relation Programme(programmeId, programmeName, modules), the programmeId in student relation is a foreign key referencing Programme relation, and must match the values of programmeId in Programme relation, or foreign key value must be null	
	4 marks

Question 2

a) Explain, and give an example of each of the following terms:

v) Candidate key

vi) Functional dependency

vii) Transitive dependency

viii) Multi-valued dependency

i) Candidate key	**Do not write in this column**
A set of attributes that uniquely identifies a tuple within a relation.	
Example: Student (studentId, studentName, email), studentId is a candidate key.	
ii) Functional dependency	
Functional dependency describes relationship between attributes.	
A and B are attributes of relation R, if each value of A in R is associated with exactly one value of B in R, then B is functionally dependent on A (denoted A → B).	
Example: studentId → studentName, email	

iii) Transitive dependency	
describes a condition where A, B, and C are attributes of a relation such that if A → B and B → C, then C is transitively dependent on A via B (provided that A is not functionally dependent on B or C).	
Example: Student (studentId, moduleId, moduleName)	
StudentId → moduleId, moduleId → moduleName, then moduleName is transitively dependent on studentId via moduleId.	
iv) multi-valued dependency	
Dependency between attributes (for example, A, B, and C) in a relation, such that for each value of A there is a set of values for B and a set of values for C. However, the set of values for B and C are independent of each other. Denoted A −>> B, A −>> C	
Example: student (studentId, subjects, hobbies)	
studentId −>> subjects, studentId −>> hobbies	
	11 marks

b) Examine the table shown below. This table represents the hours worked per week for temporary staff at each branch of a company. We assume that each staff can only have one position, but can work in different branch for different hours per week.

staffNo	branchNo	branchAddress	name	position	hoursPerWeek
S4555	B002	City Center Plaza, Seattle, WA 98122	Ellen Layman	Assistant	16
S4555	B004	16 – 14th Avenue, Seattle, WA 98128	Ellen Layman	Assistant	9
S4612	B002	City Center Plaza, Seattle, WA 98122	Dave Sinclair	Assistant	14
S4612	B004	16 – 14th Avenue, Seattle, WA 98128	Dave Sinclair	Assistant	10
S4721	B003	36 Main street, Seattle, WA 98103	Peter Hoffman	Assistant	9

i) The table shown above is susceptible to update anomalies. Provide examples of how insertion, deletion, and modification anomalies could occur on this table.

Insertion anomaly:	Do not write in this column
It is impossible to insert a new branch with no staff – staffNo is part of primary key	
Deletion anomaly:	
If we delete Peter Hoffman from the relation, then all information related to branch B003 will be deleted.	
Modification anomaly:	
To change the address of branch B004, multiple rows of information need to be changed.	**9 marks**

ii) Identify the functional dependencies represented by the data shown in the table and primary key. State any assumptions you make about the data (if necessary).

Fd1: branchNo →branchAddress	Do not write in this column
Fd2: staffNo →name, position	
Fd3: staffNo, branchNo →hoursPerWeek	
Primary key: staffNo, branchNo	
	4 marks

Is the relation in the table in Third Normal Form (3NF)? If not, decompose the table into 3NF relations. Identify the primary and foreign keys in your 3NF relations.

The relation is not in 3NF.	Do not write in this column
After removing partial dependencies fd1 and fd2, the 3NF relations are:	
R1(branchNo, branchAddress)	
R2(staffNo, name, position)	
R3(staffNo, branchNo, hoursPerWeek) foreign key: staffNo, branchNo	**6 marks**

Question 3

a) You have just been hired as a consultant for a big airplane manufacturer. Impressed by your background in databases, they want you to completely redesign their database system. Talking with the people in the company, you get the following information:

The database contains information about employees, factories and parts.
Each employee has a social security number (SSN), name and salary. An employee is uniquely identified by his or her SSN.
Each factory has an id, name and a budget. The id uniquely identifies a factory.
Each part has an id and a name. The id uniquely identifies a part.
Each employee reports to at most one other employee.
Each employee works in at least one factory.
Each part is manufactured in exactly one factory.
Each part is a component of zero or more other parts.

A partial Entity-Relationship diagram for the above application is shown in Figure 1 below.

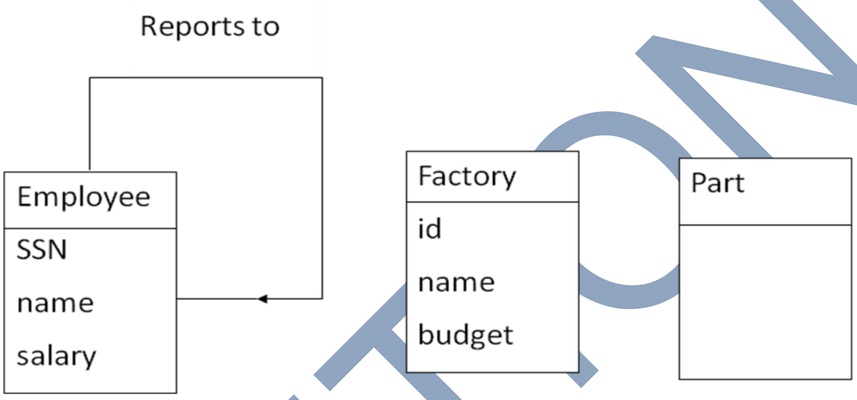


Figure 1

You are requested to do the following:

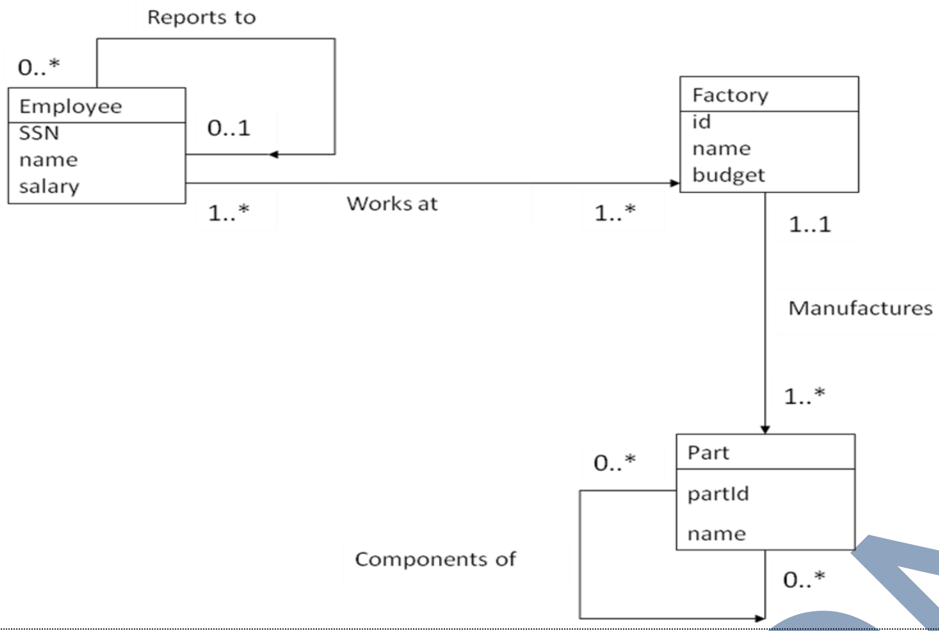
i) Add attributes for Part

ii) Add missing relationships

iii) Identify the primary keys

iv) Give the cardinality and participation constraints on relationships

i) Part (partId, name)	Do not write in this column
ii)	

 Reports to 0..* Employee SSN name salary 0..1 1..* Works at 1..* Factory id name budget 1..1 Manufactures 1..* 0..* Part partId name Components of 0..*	
Missing relationships: Works at, manufactures, components of	
iii)	
Primary keys:	
Employee: SSN	
Factory: id	
Part: partId	
iv)	
The cardinality and participation of relationships are as above figure in ii).	
	16 marks

b) Consider the following schema (primary keys are underlined)

```
Student (sname, sid, grade, level, deptno)
Course (cno, cname, deptno, units)
Dept (dname, deptno)
Takes (sid, cno)
```

Answer the following:

i) Formulate the following query using SQL:

EBU5602 Past paper 2 sample solution

1) List the IDs of students in the department of “EECS”.	Do not write in this column
SELECT sid FROM Student s, Dept d	
WHERE s.deptno =d.deptno AND d.dname = ‘EECS’;	
2) List the names, grades of students who are doing “Software Engineering” course.	
SELECT sname, grade	
FROM Student s, Course c, Takes t	
WHERE s.sid = t.sid AND c.cno = t.cno AND c.cname = ‘Software Engineering’;	
3) Calculate the total number of courses in each department	
SELECT deptno, COUNT(cno)	
FROM Course	
GROUP BY deptno;	
4) For each department with more than 20 courses, find number of courses and sum of their units.	
SELECT deptno, COUNT(cno), SUM(units)	
FROM Course	
GROUP BY deptno	
HAVING COUNT(cno) > 20;	**10 marks**

ii) Which of the following queries returns the id of the student with the highest grade? **More than one choice may be correct**.

A)
```
SELECT S.sid
FROM Students S
WHERE S.grade = MAX(S.grade);
```

B)
```
 SELECT S.sid, MAX(S.grade)
FROM Students S
GROUP BY S.grade;
```

C)
```
SELECT S.sid
FROM Student S
WHERE S.grade > ALL (SELECT S.grade FROM Student S);
```

D) SELECT S.sid
FROM Student S
WHERE S.grade = (SELECT MAX(S.grade) FROM Student S);

E) None of the above

B, D	Do not write in this column
	4 marks

Question 4

a) An ACID transaction would permit the following situations (**mark "True" or "False" for each**):

i) After a failure, an uncommitted transaction is rolled back and all of its effects are erased.

ii) After a failure, a committed transaction is rolled back and all of its effects are erased.

iii) A transaction updates a tuple, but has not committed the changes. Another transaction reads the uncommitted data.

iv) A transaction reads the same tuple twice without writing it in between and sees two different values.

i) true ii) false	Do not write in this column
iii) false	
iv) false	
	4 marks

b) For the following schedule of two transactions T1, T2:

time	T1	T2
t1		Begin_transaction
t2	Begin_transaction	Read(X)
t3	Read(X)	X:=X+200
t4	X:=X-50	Write(X)
t5	Write(X)	Commit
t6	Commit	

Figure 2

i) Discuss the problem of updating X in the two transactions T1 and T2.

This is lost update problem in concurrency. An apparently completed update by one user can be overridden by another user. While T1 reads the value of X, the value of X is changed by T2. This can lead to situations where one of the changes (updates) of one transaction are disregarded (lost).

Do not write in this column

3 marks

ii) Explain the principles of two-phase locking (2PL). Rewrite T1 and T2 in Figure 2 using 2PL.

Principles of 2PL: Every transaction must lock an item (read or write) before accessing it.

Once a lock has been released, no new items can be locked

Do not write in this column

time	T1	T2
t1		Begin_transaction
t2	Begin_transaction	Write_lock(X)
t3	Write_lock(X)	Read(X)
t4	wait	X:=X+200
t5	wait	Write(X)
t6	wait	Commit/unlock(X)
t7	Read(X)	
t8	X:=X-50	
t9	Write(X)	
t10	Commit/unlock(X)	

7 marks

c) 'A data warehouse is a subject-oriented, integrated, time-variant and non-volatile collection of data in support of management's decision-making process'. Pick any three of the major characteristics of the data held in a data warehouse and describe them.

Subject-oriented: as the warehouse is organized around the major subjects of the enterprise (such as customers, products, and sales) rather than the major application areas (such as customer invoicing, stock control, and product sales). This is reflected in the need to store decision-support data rather than application-oriented data.	**Do not write in this column**
Integrated: because of the coming together of source data from different enterprise-wide applications systems. The source data is often inconsistent using, for example, different formats. The integrated data source must be made consistent to present a unified view of the data to the users.	
Time-variant: because data in the warehouse is only accurate and valid at some point in time or over some time interval. The time-variance of the data warehouse is also shown in the extended time that the data is held, the implicit or explicit association of time with all data, and the fact that the data represents a series of snapshots.	
Non-volatile: as the data is not updated in real time but is refreshed from operational systems on a regular basis. New data is always added as a supplement to the database, rather than a replacement. The database continually absorbs this new data, incrementally integrating it with the previous data.	
	6 marks