Sample/practice exam 4 June, questions

Databases (Monash University)

Office Use Only

Monash University

Semester Two Examination Period 2012

Faculty Of Information Technology

EXAM CODES: **FIT9019**

TITLE OF PAPER: **DATABASE TECHNOLOGY - PAPER 1**

EXAM DURATION: 2 hours writing time

READING TIME: 10 minutes

THIS PAPER IS FOR STUDENTS STUDYING AT:(tick where applicable)

☐ Berwick ☐ Clayton ☐ Malaysia ☐ Off Campus Learning ☐ Open Learning
☐ Caulfield ☐ Gippsland ☐ Peninsula ☐ Enhancement Studies ☐ Sth Africa
☐ Pharmacy ☐ Other (specify)

During an exam, you must not have in your possession, a book, notes, paper, electronic device/s, calculator, pencil case, mobile phone or other material/item which has not been authorised for the exam or specifically permitted as noted below. Any material or item on your desk, chair or person will be deemed to be in your possession. You are reminded that possession of unauthorised materials in an exam is a discipline offence under Monash Statute 4.1.

No examination papers are to be removed from the room.

AUTHORISED MATERIALS

CALCULATORS	**NO**
OPEN BOOK	**NO**
SPECIFICALLY PERMITTED ITEMS	**NO**
if yes, items permitted are:	

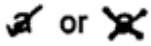
Candidates must complete this section if required to write answers within this paper

STUDENT ID __ __ __ __ __ __ __ __ DESK NUMBER __ __ __ __

OFFICE USE ONLY

PART A (20)	PART B Q1(25)	PART B Q2(15)	PART B Q3(10)	PART B Q4(20)	PART B Q5(10)

PART A. Multiple Choice Questions. (10 x 2 marks = 20 marks).

Mark your selection by placing a √ or a X through your selected answer
for example  **or**
If you change your selection during the review of your paper, prior to the end of the Examination, make sure that the alteration is clear.

Note: There is only ONE correct answer for each question. There is no penalty for incorrect answer.

1. Which one of the following properties is **NOT** exhibited by a file management system?

    a. difficult to update
    b. **data independence**
    c. data redundancy
    d. program dependence

2. Which SQL clause is required to implement the PROJECT(π) operation?

    a. GROUP BY clause
    b. FROM clause
    c. **SELECT clause**
    d. ORDER BY clause

3. Which statement is correct in relation to RELATIONAL MODEL properties?

    a. **The order of the attribute is immaterial.**
    b. Duplicate rows are possible.
    c. Attribute may have multiple values.
    d. Tuples should be addressed by their position in the relation.

4. Which statement is correct in relation to VIEW?

    a. The data in a view can always be updated.
    b. **The data in a view is generated only when the view is used.**
    c. The data in view can always be inserted using INSERT statement.
    d. Derived column is not allowed in view.

5. A table that is in 2NF and contains no transitive dependencies is said to be in ____.

    a. 1NF
    b. 2NF
    c. **3NF**
    d. 4NF

6. The following SQL is executed in ORACLE and produces an error message. What is the most likely cause of the error message?

```
SELECT student_id, student_name
FROM student
WHERE course='MAIT'
UNION
SELECT student_id, student_name, student_dob
FROM student
WHERE course='MIT'
```

   a. The union operation is not supported for two tables.
   b. The attribute student_name does not have the correct data type.
   c. The union operation should not be used for retrieving data from a single table.
   d. **The two select statements are not union compatible.**

7. ANSI SPARC recommends view integration into database design. The recommendation suggests three levels of design. What are the levels?

   a. External, conceptual, database
   b. Conceptual, logical, physical
   c. External, conceptual, logical
   d. **External, conceptual, internal.**

8. An ER diagram needs to be developed to represent the fact that an employee may supervise another employee. Both the supervisor and the supervisee are employees. What would be the best way for you to model this fact?

   a. Supervisor is a strong entity and the supervisee is a weak entity.
   b. **A recursive relationship called supervises on an entity called employee.**
   c. A multi-varied attribute called supervisor in an entity called employee.
   d. A many to many relationship between an entity called supervisor and an entity called supervisee.

9. A trigger to perform update cascade needs to be implemented to ensure data integrity between EMPLOYEE and DEPARTMENT on the deptno. An update of deptno in the DEPARTMENT should cause all the associated tuples in the EMPLOYEE table be updated. Which statement is correct in relation to this trigger?

   a. The trigger should be created for the EMPLOYEE table.
   b. It is not possible to implement this trigger because it will involved mutating table.
   c. **The trigger can only be implemented as row level trigger.**
   d. The trigger has to contain an IF.. THEN.. ELSE..statement.

10. Which of the following is an example of a soft crash in database?

    a. Disk crash.
    b. **Power Failure.**
    c. Software bugs.
    d. Concurrency.

END OF PART A

PART B. Answer all questions on the space provided. Total: 80 marks

1. Define the following terms that are related to relational model. Provide an example in your answer. (**5 x 5 marks = 25 marks**)

   a. Candidate Key

   A candidate key is a minimal super key (a subset of attributes which uniquely identify a tuple)

   EMPLOYEE (empno, empname. …… taxfileno)

   empno and taxfileno are both candidate keys (minimal superkeys) since either attribute uniquely identifies the EMPLOYEE.

(5 marks)

   b. Primary Key

   A primary key is the name given to the candidate key which is selected to identify the tuples in the relation - in the example above the primary key would be empno (taxfileno has too many embedded semantics)

(5 marks)

c. Foreign Key

A foreign key is an attribute (or set of attributes) which exists in one relation and points to a primary key of a relation. In this manner the relational model, using pairs of primary and foreign keys creates logical parent-child relationships.

CUSTOMER (custno (PK), custname, custaddress, …)

ORDER (orderno (PK), orderdate, custno (FK))

Here the relationship between CUSTOMER and ORDER is established by placing the PK of CUSTOMER (custno) in ORDER as a FK.

(5 marks)

d. Entity Integrity

Entity integrity requires that no primary key of a relation may be null or have a null component. The primary key value held by a tuple, according to the uniqueness requirement of the relational model (no duplicate tuples), must uniquely identify the tuple.

CUSTOMER (custno (PK), custname, custaddress, …)

In the CUSTOMER relation above, custno must have a value, it may not be null. The value must be unique for each tuple.

(5 marks)

e. Referential Integrity

Referential integrity requires that for each not null FK value there must exist an equal value of a primary key.

If the foreign key is a composite there must exist an equal value of a primary key based on the same components (ie. the entire FK must point to one entire PK)

CUSTOMER (custno (PK), custname, custaddress, …)

ORDER (orderno (PK), orderdate, custno (FK))

here each value of the custno attribute in ORDER must exactly match a value of custno in CUSTOMER

(5 marks)

2. Draw a *relational model* using *crowsfoot notations* that depicts the situation described below. (**15 marks**)

The local under-sixteen football league needs a database to help track teams, children that sign up to play in the league, the parents of these children and the coaches for each team.

The league wishes to record the details for each parent of a particular player (the parents last name, first name, phone contact number and address).

For each player the system needs to record the player's last name, first name, blood type and their date of birth. Any allergies that the player has also need to be recorded. A player may only play in one team with this league.

Each team is given a unique id, the system needs to record this id, the teams name and the city their home ground is located in (a city may have several under-sixteen teams based in it). A team's colours are also to be recorded, teams may choose to use a single or multiple team colours. A team may have several coaches – one of the coaches is designated as the head coach. A coach is only permitted to coach one team in the under-sixteen league. All communications from the league to the team are via the head coach. To be registered to play in this league a team must have a head coach and at least one player. The database needs to track a coach's first and last name, phone and address contact details and the team that they are coaching.

- Parents – Player 3 marks
- Allergy – Player 3 marks
- Coach - Team 2 marks
- Player – Team 2 marks
- Team – Colour 3 marks
- Team – city 2 marks

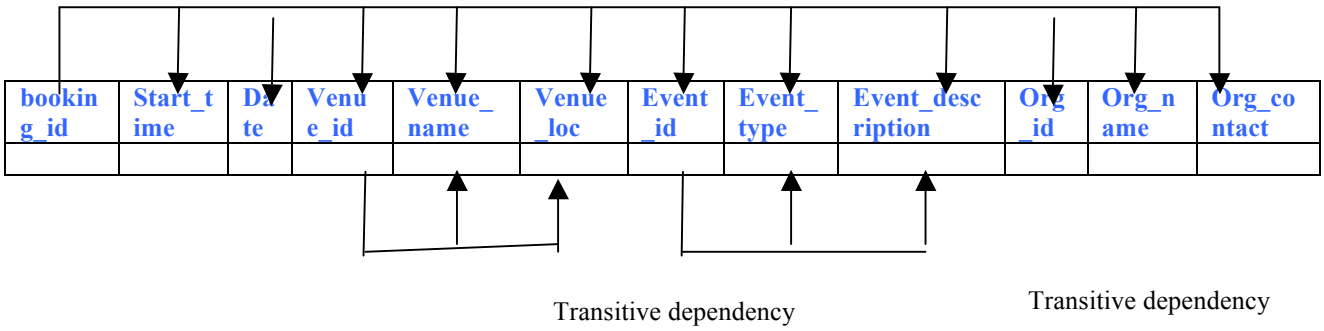
3. Normalisation (**10 marks**)

Monash University owns several performance halls that are used by organizations within and outside Monash University. The following table shows the booking information for several performances across different venues in Monash University. Multiple performances can be organized at the same time across multiple venues.

Start Time	Date	Venue	Venue Location	Event Type	Event Description	Organiser	Organiser's contact
8 PM	12-Jan-2012	Robert Blackwood Hall	Clayton	Music concert	Melbourne Symphony	MSO	(03) 99021212
8 PM	12-Jan-2012	K3.24	Caulfield	Comedy	Adam Hill	Melbourne Comedy Festival	(03) 99031456
2 PM	14-Jan-2012	Robert Blackwood Hall	Clayton	Musical	Cats	Monash Student Association	(03) 99012233
8 PM	14-Jan-2012	Alexander Theatre	Clayton	Comedy	Dave Hughes	Melbourne Comedy Festival	(03) 99031456
8 PM	16-Jan-2012	Robert Blackwood Hall	Clayton	Music concert	Hoodoo Guru	Mushroom Promoter	(02) 90021002

a) Draw a dependency diagram for this table.
b) Convert the table shown above to Third Normal Form (3NF), showing each stage of the process. Clearly state any assumptions that you make. You can add new attribute(s) if you think the attributes in the tables are not suitable for primary key(s).

Dependency Diagram



2 marks

UNF

BOOKING(booking_id, time, date, venue_id, venue_name, venue_location, event_id, event_type, event_desc, organiser_id, organizer, organiser's contact)

Primary Key: booking_id

2 marks

1NF

BOOKING(booking_id, time, date, venue_id, venue_name, venue_location, event_id, event_type, event_desc, organiser_id, organizer, organiser's contact)

2 marks

2NF

There is no partial dependency. 2NF = UNF

2 marks

3NF

BOOKING(booking_id, time,date, venue_id, event_id, organizer_id)

VENUE(venue_id, venue, venue_location)

ORGANIZER(organizer_id, oraganiser_name. Organizer_contact)

EVENT(event_id, event_type, event_description)

2 marks

4. Appendix A shows a logical diagram and database schema for PAYROLL database. Use this diagram and write SQL for each of the following queries:

a) Display all the names of MANAGERs in the company. **(3 marks)**

```
SELECT empname
FROM employee
WHERE empjob = 'MANAGER';
```

b) Display the employees' name and monthly salary for SALESREP who have not earned any commission yet. **(3 marks)**

```
SELECT empname, empmsal
FROM employee
WHERE empjob = 'SALESREP'
AND
empcomm = 0;
```

c) Display the employee name, job, department name, location and monthly salary of employees that work in Dallas. The result should be ordered by job. **(3 marks)**

```
SELECT e.empname, e.empjob, d.deptname, d.deptlocation,e.empmsal
FROM payroll.employee e, payroll.department d
WHERE e.deptno = d.deptno
AND d.deptlocation = 'DALLAS'
ORDER BY e.empjob;
```

d) Display the names and jobs of employees who do **NOT** work in the TRAINING or the SALES department. **(3 marks)**

```
SELECT e.empname, e.empjob, d.deptname, d.deptlocation
FROM payroll.employee e, payroll.department d
WHERE e.deptno = d.deptno
AND d.deptname NOT IN ('TRAINING', 'SALES')
ORDER BY e.empname;
```

e) Display employee name and the amount of the annual salary for the highest paid employee in the company. **(3 marks)**

```
SELECT empname, empmsal*12+NVL(empcomm,0)
FROM employee
WHERE empmsal*12+NVL(empcomm,0) =
        (SELECT MAX(empmsal*12+NVL(empcomm,0)) FROM employee)
ORDER BY empname;
```

f) Which department has the greatest average monthly salary? **(5 marks)**

```
select deptno,avg(empmsal)
from employee
group by deptno
having avg(empmsal) = (select max(avg(empmsal))
from employee
group by deptno);
```

5. a. What does it mean by ACID properties in database transaction?

A = atomicity
A transaction should be processed entirely, transaction can be either committed or rollback.

C = consistency
A transaction should be bring a database from one consistent state to another consistent state.

I = Isolation
A result of a transaction should not be seen by another transaction until it has been commited.

D = Durability
Once a transaction is committed, the result of this transaction should never be lost.

(4 marks)

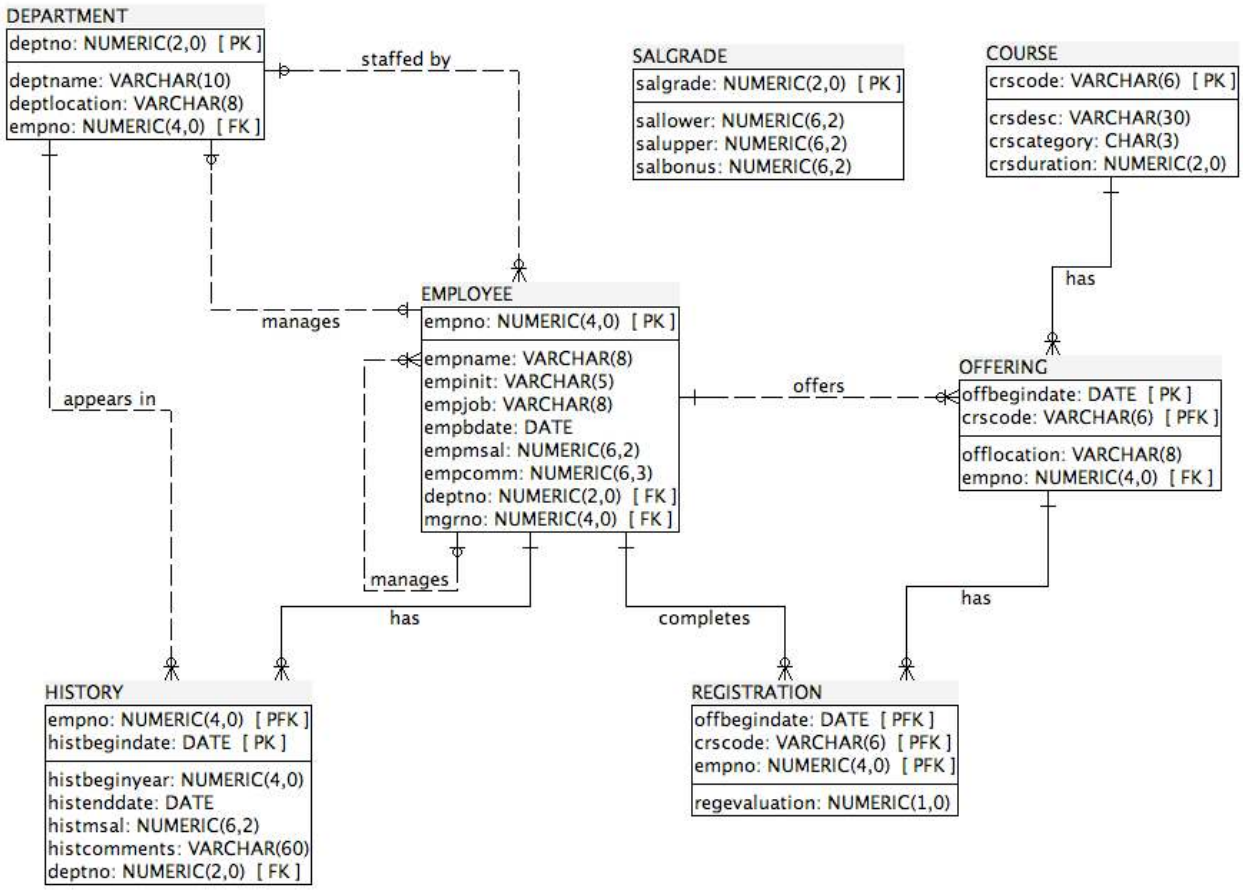
b. Give an example how adherence to ACID properties can prevent lost update problem.

A lost update occurs when two transactions is reading the same data and want to update the data. For example, two transactions are reading Quantity on Hand (QOH=30) of item no 50. Transaction 1 wants to add 5 to QOH, Transaction 2 wants to reduce QOH by 2. Transaction 1 and transaction 2 read the QOH at the same time. Transaction 1 update QOH first to 30+5 = 35 and commits. The second transaction performs 30-2=28 and commits. The final value for QOH=28, which is wrong. It should be 30+5-2= 33. The update made by transaction 1 is lost since transaction 2 read QOH at the same time with transaction 1.

If a protocol follows an ACID principle is devised, it will ensure that transaction 2 can only read QOH once transaction 1 has committed the change from 30 to 35. The lost update can be avoided. The atomicity and isolation will ensure that transaction 2 will only see QOH when transaction 1 commits the changes made to QOH. Once a transaction is committed, the result of the transaction should never be lost according to the durability. Hence, even in a situation of power failure after transaction 1 commit, the restart process will ensure that transaction 2 will read QOH=35. If a power failure occurs before transaction 1 made the commit, the consistency principle will ensure the restart process roll back the QOH to 30 and restart transaction 1.

2 marks for lost update explanation
4 marks for the explanation of the protocol based on ACID.

APPENDIX A




```
CREATE TABLE SALGRADE (
  salgrade NUMBER(2)   NOT NULL ,
  sallower NUMBER(6,2)   NOT NULL ,
  salupper NUMBER(6,2)   NOT NULL ,
  salbonus NUMBER(6,2)   NOT NULL   ,
CONSTRAINT salgrade_pk PRIMARY KEY (salgrade),
CONSTRAINT salgrade_chk1 CHECK (sallower >= 0),
CONSTRAINT salgrade_chk2 CHECK (sallower <= salupper));

--
-- Create Table    : 'COURSE'
-- crscode         : Course Code
-- crsdesc         : Course Description
-- crscategory     : Course Category
-- crsduration     : Course Duration
--
CREATE TABLE course (
  crscode VARCHAR(6)   NOT NULL ,
  crsdesc VARCHAR(30)   NOT NULL ,
  crscategory CHAR(3)   NOT NULL ,
  crsduration NUMBER(2)   NOT NULL   ,
CONSTRAINT course_pk PRIMARY KEY (crscode),
CONSTRAINT course_chk1 CHECK (crscode = upper(crscode)),
CONSTRAINT course_chk2 CHECK (crscategory in ('GEN','BLD','DSG')));

--
-- Create Table    : 'DEPARTMENT'
```

```
-- deptno          : Department Number
-- deptname        : Department Name
-- deptlocation    : Location of department
-- empno           : Employee who manages department (references EMPLOYEE.empno)
--                   FK constraint cannot be added until employee table is
created
--
CREATE TABLE DEPARTMENT (
  deptno NUMBER(2)   NOT NULL ,
  deptname VARCHAR(10)   NOT NULL ,
  deptlocation VARCHAR(8)   NOT NULL ,
  empno NUMBER(4) ,
CONSTRAINT department_pk PRIMARY KEY (deptno),
CONSTRAINT department_un UNIQUE (deptname),
CONSTRAINT department_chk1 CHECK (deptname = upper(deptname)),
CONSTRAINT department_chk2 CHECK (deptlocation = upper(deptlocation)));

--
-- Create Table    : 'EMPLOYEE'
-- empno           : Employee number
-- empname         : Employee name
-- empinit         : Employee initials
-- empjob          : Employee job
-- empbdate        : Employee birthdate
-- empmsal         : Employee monthly salary
-- empcomm         : Employee commission
-- deptno          : Department Number (references DEPARTMENT.deptno)
-- mgrno           : Employees manager (empno of manager) (references
EMPLOYEE.empno)
--
CREATE TABLE EMPLOYEE (
  empno NUMBER(4)   NOT NULL ,
  empname VARCHAR(8)   NOT NULL ,
  empinit VARCHAR(5)   NOT NULL ,
  empjob VARCHAR(8)    ,
  empbdate DATE   NOT NULL ,
  empmsal NUMBER(6,2)   NOT NULL ,
  empcomm NUMBER(6,2) ,
  deptno NUMBER(2) ,
  mgrno NUMBER(4) ,
CONSTRAINT employee_pk PRIMARY KEY (empno),
CONSTRAINT employee_fk1  FOREIGN KEY (mgrno)
    REFERENCES EMPLOYEE (empno),
CONSTRAINT employee_fk2 FOREIGN KEY (deptno)
    REFERENCES DEPARTMENT (deptno));

-- Alter Table    : 'DEPARTMENT'
-- empno           : Employee who manages department (references EMPLOYEE.empno)
-- Add constraint for Department table now employee exists:
ALTER TABLE DEPARTMENT
ADD (CONSTRAINT department_fk FOREIGN KEY (empno)
            REFERENCES employee (empno));

--
-- Create Table    : 'HISTORY'
-- deptno          : Department Number (references DEPARTMENT.deptno)
-- histbegindate   : Date history record begins
-- histbeginyear   : Year history record begins
-- histenddate     : Date history record ends
-- histmsal        : Monthly Salary for this history record
-- histcomments    : Comments for this history record
-- empno           : Employee number (references EMPLOYEE.empno)
--
CREATE TABLE HISTORY (
```

```
  empno NUMBER(4)   NOT NULL ,
  histbegindate DATE   NOT NULL ,
  histbeginyear NUMBER(4)   NOT NULL ,
  histenddate DATE    ,
  histmsal NUMBER(6,2)   NOT NULL ,
  histcomments VARCHAR(60)    ,
  deptno NUMBER(2)   NOT NULL   ,
CONSTRAINT history_pk PRIMARY KEY (empno, histbegindate),
CONSTRAINT history_chk CHECK (histbegindate < histenddate),
CONSTRAINT history_fk1 FOREIGN KEY (empno)
    REFERENCES EMPLOYEE (empno)
      ON DELETE CASCADE,
CONSTRAINT history_fk2 FOREIGN KEY (deptno)
    REFERENCES DEPARTMENT (deptno));

--
-- Create Table    : 'OFFERING'
-- offbegindate    : Begin date for offering
-- crscode         : Course Code (references COURSE.crscode)
-- offlocation     : Location for offering
-- empno           : Employee number for employee running offering (references
EMPLOYEE.empno)
--
CREATE TABLE OFFERING (
  offbegindate DATE   NOT NULL ,
  crscode VARCHAR(6)   NOT NULL ,
  offlocation VARCHAR(8) ,
  empno NUMBER(4) ,
CONSTRAINT offering_pk PRIMARY KEY (offbegindate, crscode),
CONSTRAINT offering_fk1 FOREIGN KEY (crscode)
    REFERENCES course(crscode),
CONSTRAINT offering_fk2 FOREIGN KEY (empno)
    REFERENCES EMPLOYEE (empno));

--
-- Create Table    : 'REGISTRATION'
-- offbegindate    : Begin date for offering (references OFFERING.offbegindate)
-- crscode         : Course Code (references OFFERING.crscode)
-- regevaluation   : Grade for course completed
-- empno           : Employee number of employee completing course (references
EMPLOYEE.empno)
--
CREATE TABLE REGISTRATION (
  offbegindate DATE   NOT NULL ,
  crscode VARCHAR(6)   NOT NULL ,
  empno NUMBER(4) NOT NULL,
  regevaluation NUMBER(1)      ,
CONSTRAINT registration_pk PRIMARY KEY (offbegindate, crscode, empno),
CONSTRAINT resgitration_chk CHECK (regevaluation in (1,2,3,4,5)),
CONSTRAINT registration_fk1 FOREIGN KEY (empno)
    REFERENCES EMPLOYEE (empno),
CONSTRAINT registration_fk2 FOREIGN KEY (offbegindate, crscode)
    REFERENCES OFFERING (offbegindate, crscode));
```