VICTORIA UNIVERSITY OF WELLINGTON
Te Whare Wananga o te Upoko o te Ika a Maui

EXAMINATIONS — 2006
MID-YEAR

COMP 302
Database Systems

Time allowed: 3 Hours

Instructions: Answer all questions.
Make sure that your answers are clear and to the point.
Calculators and printed foreign language dictionaries are allowed.
No reference material is allowed.
There are 180 marks on the exam.
Marks are shown for each question as a whole and also for their parts.

CONTENTS:

Question 1. Database Basics [25 marks]

a) [5 marks] What is a database and what are its main features?

ANSWER

b) [7 marks] What are the tasks of a Database Management System?

ANSWER

c) [4 marks] What is a relational database schema (the schema of a whole relational database)?

ANSWER

d) [4 marks] What is a relation schema key? List the properties of a relation schema key.

ANSWER

e) [5 marks] What is a foreign key? List the properties of a foreign key.

ANSWER

Question 2. SQL and Relational Algebra [26 marks]

Consider the Musicians database schema given below.

```
create table Musician (
MusicianId smallint primary key,
Mus_Name varchar (15) not null,
Telephone varchar (15),
Type varchar (15) not null
);

create table Instrument (
InstrumentId smallint primary key,
Inst_Name varchar (15) not null,
Musical_Key varchar (10) not null
);

create table Played_By (
MusicianId smallint not null references Musician,
InstrumentId smallint not null references Instrument,
primary key (MusicianId, InstrumentId)
);

create table Album (
AlbumId smallint primary key,
Alb_Title varchar (15) not null,
Format varchar (5),
Producer smallint not null references Musician(MusicianId)
);

create table Song (
AlbumId smallint not null references Album,
Song_No smallint not null,
Song_Title varchar (15) not null,
primary key (AlbumId, Song_No)
);

create table Performed_By(
MusicianId smallint not null references Musician,
AlbumId smallint not null,
Song_No smallint not null,
primary key (MusicianId, AlbumId, Song_No),
foreign key (AlbumId, Song_No) references Song
);
```

a) [23 marks] Write the following SQL queries on the Musicians database given above:

i. [3 marks] Retrieve MusicianId and Mus_Name of musicians whose Type is 'singer'.

ANSWER

ii. [4 marks] Retrieve MusicianId and Mus_Name of musicians who produced at least one Album with a Format 'CD'.

ANSWER

iii. [5 marks] Retrieve MusicianId and Mus_Name of all musicians who did not perform any song.

ANSWER

iv. **[11 marks]** Retrieve MusicianId and Mus_Name of all musicians who performed the largest number of songs.

Note: In SQL, if a nested query returns a single attribute and a single tuple, the query result will be treated as a single scalar value. Unlike SQL, Relational Algebra expressions always produce sets of tuples at outputs.

Hint: You may find it useful to do this query in a stepwise way.

ANSWER

b) [3 marks] Use relational algebra to write the following query:

Retrieve MusicianId and Mus_Name of musicians whose Type is 'singer'.

ANSWER

(Spare Page for extra working)

Question 3. Enhanced Entity Relationship Data Model [30 marks]

In this question you will be asked to draw EER diagrams of two external views of the Musicians database defined in Question 2 of this examination. Recall that external views express views of different users on possibly overlapping parts of the same mini world (universe of discourse - UOD).

Use the EER notation we introduced in lectures. If you use a different notation, define it clearly in your answer.

a) [18 marks] Map the Musicians relational database schema to the entity relationship data model by drawing an EER diagram. State any assumptions you make.

Note: There is a copy of the Musicians database schema in Appendix B.

ANSWER

b) **[12 marks]** Draw the EER diagram of another external view of the musicians universe of discourse, described as follows:

- Musicians can be classified as:
  Singers,
  Instrument players,
  Conductors, or
  Producers.
- Each musician belongs to at least one of these classes, but can belong to more than one.
- A producer has an association with albums:
  Each album is produced by exactly one producer.
  A producer may be responsible for the production of many albums, but there may exist producers without any album.
- An instrument player has an association with instruments:
  Each instrument player plays at least one instrument.
  Each instrument may be played by many instrument players, but there may be instruments that are not played by any instrument player.

In your answer, show entity and relationship types, relationship type constraints and the classification relationship only. Avoid showing any attributes.

ANSWER

Question 4. Mapping EER to Relational Data Model [33 marks]

In this question you will be asked to map a number of EER diagrams into relational database schemas (S, I), where S is a set of relation schemas, and I is a set of referential integrity constraints. The EER diagrams will contain only symbols for entity and relationship types, and relationship constraints. Attributes will be given separately in the form

$$Entity\_Type\_Name\ (Attribute_1, \ldots, Attribute_n),$$

with entity type keys underlined, or

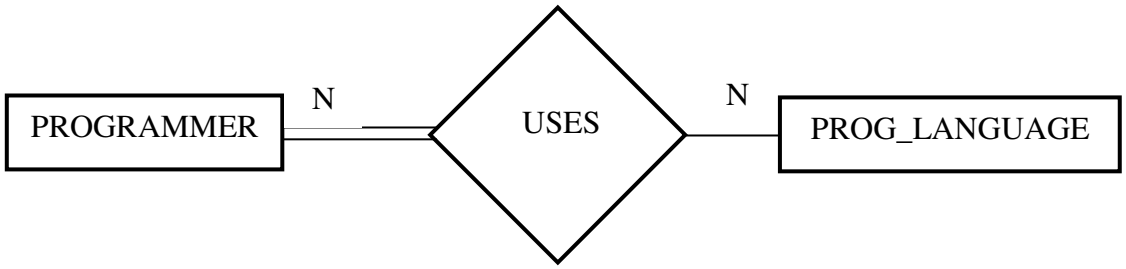
$$Relationship\_Type\_Name\ (Attribute_1, \ldots, Attribute_n).$$

Represent your relation schema in the form

$$N(R, K),$$

where R is the set of attributes and K is the set of keys of the relation schema with the name N. Indicate if any foreign key has to be not null.

a) [7.5 marks] Map the following ER diagram into a relational database schema.



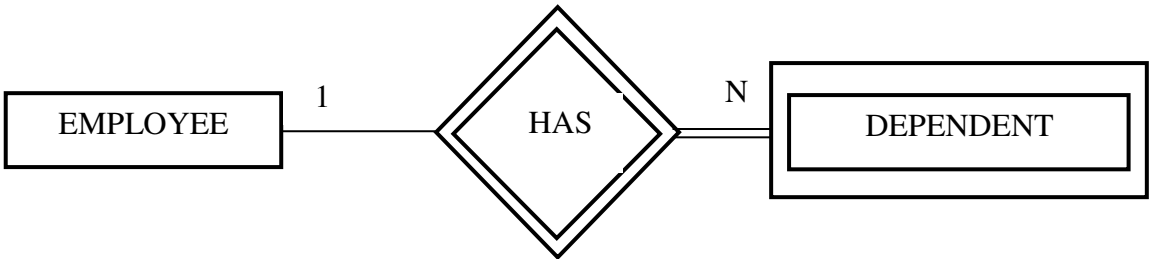
PROGRAMMER(<u>ProgrammerId</u>, Prog_Name)

USES(Skill_Level)

PROG_LANGUAGE(<u>LanguageId</u>, Lang_Name)

ANSWER

b) [4.5 marks] Map the following ER diagram into a relational database schema.



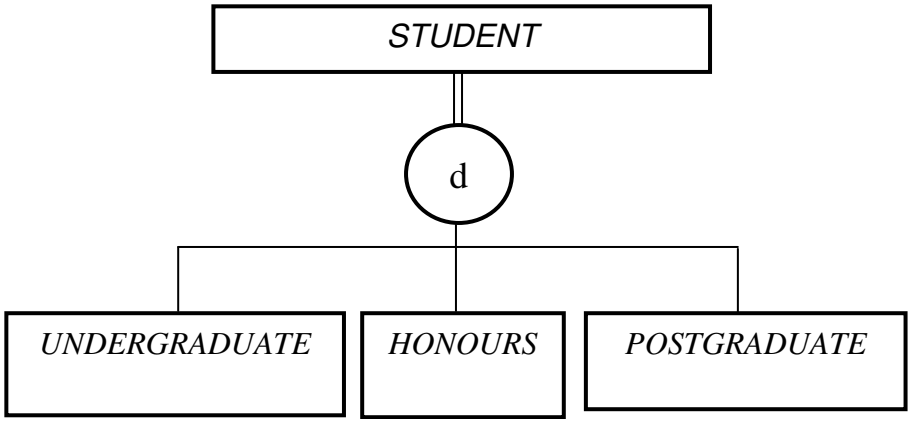
EMPLOYEE(EmployeeId, Emp_Name)

HAS( )

DEPENDENT(DepName, Age)

ANSWER

c) [9 marks] Map the following ER diagram into a relational database schema. Include all constraints you think should be satisfied.



STUDENT(StudentId, Stud_Name)

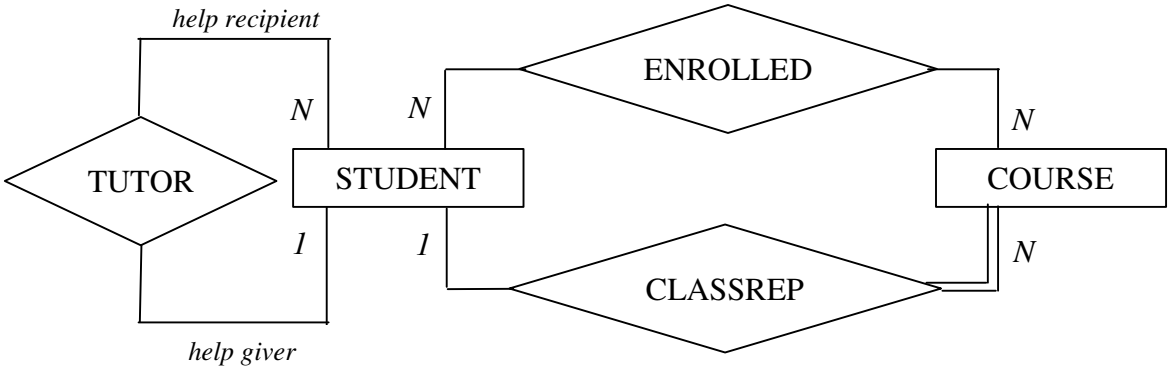
UNDERGRADUATE(NoOfPts)

HONOURS(Class)

POSTGRADUATE(Program)

ANSWER

d) [12 marks] Map the following ER diagram into a relational database schema. Include all constraints you think should be satisfied.



STUDENT(<u>StudentId</u>, Stud_Name, NoOfPts)

COURSE(<u>CourseId</u>, Cour_Name)

ENROLLED(InTermPts)

CLASSREP( )

TUTOR( )

ANSWER

Question 5. Functional Dependencies and Normalization [31 marks]

a) [8 marks] Express the following real world facts using functional dependencies:

i. [2 marks] A musical album has exactly one producer (but the same producer may produce many albums).

ANSWER

ii. [3 marks] Each song in an album is performed by at most one singer.

ANSWER

iii. [3 marks] A singer sings at most one song in an album.

ANSWER

b) [5 marks] Let a set of functional dependencies

$$F_2 = \{AB \rightarrow C, DEG \rightarrow H, A \rightarrow B, DG \rightarrow H\}$$

be given. Transform the set F_2 into a new set of fd's G_2, where each fd is left reduced (ie has no redundant attribute on its left hand side).

ANSWER

c) [8 marks] Let a set of left reduced functional dependencies

$$F_3 = \{AB \rightarrow C, C \rightarrow D, C \rightarrow C, AB \rightarrow D, CE \rightarrow G, DE \rightarrow G\}$$

be given. Transform the set F_3 into a new, non redundant set of fd's G_3.

ANSWER

d) **[3 marks]** Let a set of left reduced and non redundant functional dependencies

$$F_4 = \{A \rightarrow B, A \rightarrow C, A \rightarrow D, B \rightarrow E, B \rightarrow G, GH \rightarrow I\}$$

be given. Use the set F_4 to produce a set of relation schemes that are at least in the third normal form. (In your answer, use notation $N(R, K)$, where N is the relation schema name, R is the set of attributes, and K is the set of keys.)

ANSWER

e) **[7 marks]** Let a universal relation schema (U, F) be given, where

$$U = \{A, B, C, D, E, G\} \text{ and } F = \{AG \rightarrow C, C \rightarrow D, E \rightarrow D\}.$$

Suppose that starting from (U, F) the following set of BCNF relation schemas is produced:

$$S = \{N_1(\{A, G, C\}, \{AG\}), N_2(\{C, D\}, \{C\}), N_3(\{D, E\}, \{E\})\}$$

If you consider that S is a lossless join decomposition of (U, F) explain why it is. If you consider that S is not a lossless join decomposition of (U, F), explain why it is not, and transform it into a new set of relation schemas S' that is a BCNF and lossless join decomposition of (U, F).

ANSWER

(Spare Page for extra working)

Question 6. Query Optimisation [13 marks]

Consider the Musicians database schema given in Question 2 and Appendix B. Draw the heuristic optimization tree that corresponds to the SQL query:

```
SELECT MusicianId, Mus_Name, Song_Title
FROM (Musician NATURAL JOIN Performed_By NATURAL JOIN Song)
WHERE Type = 'singer';
```

ANSWER

Question 7. Transaction Processing [22 marks]

a) [4 marks] List the names of four transaction anomalies we discussed in lectures.

ANSWER

b) [4 marks] Define the two-phase locking protocol.

ANSWER

c) [4 marks] What is the isolation level Read Committed and which transaction anomaly does it prevent?

ANSWER

d) [5 marks] Consider a concurrent (interleaved) execution of two transactions in Figure 7.1 below. Show in the space provided for the answer how two-phase locking protocol will prevent this transaction anomaly.

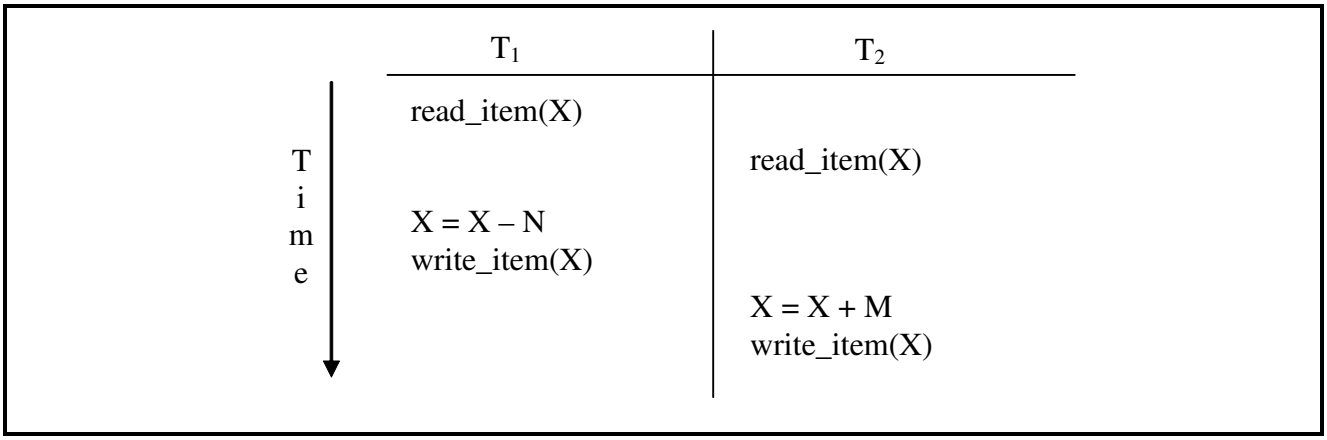


Figure 7.1

ANSWER

e) [5 marks] Figure 7.2 shows two transactions in a dead lock state. Describe how the conservative two-phase locking protocol will prevent dead lock, and show a corresponding diagram.

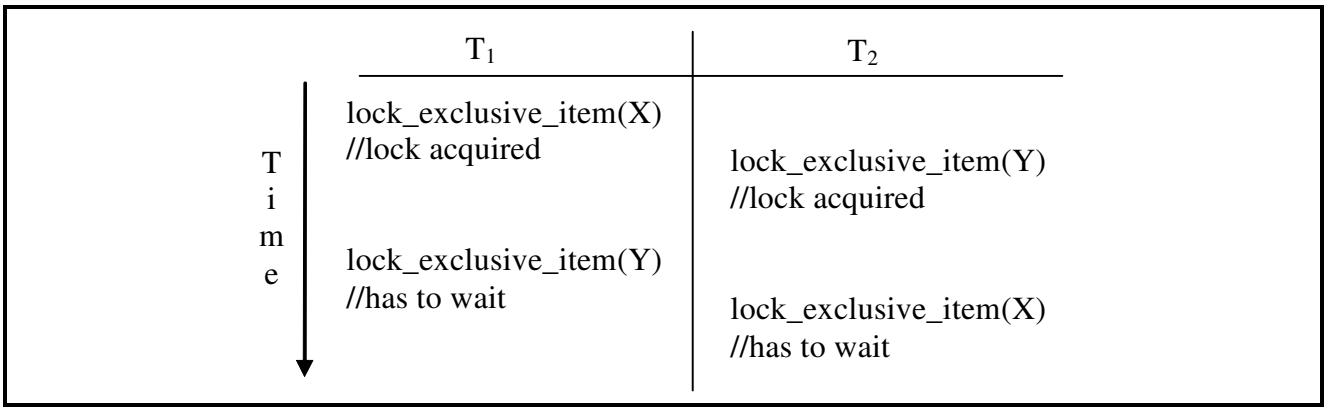


Figure 7.2

ANSWER

****

APPENDIX A

Simplified PostgreSQL documentation:

CREATE TABLE

CREATE TABLE *table_name* (
{ *column_name data_type* [DEFAULT *default_expr*] [*column_constraint* [, ...]]
| *table_constraint* } [, ...]
)

where *column_constraint* is:

[CONSTRAINT *constraint_name*]
{ NOT NULL | NULL | UNIQUE | PRIMARY KEY | CHECK (*expression*) |
REFERENCES *reftable* [(*refcolumn*)] [ON DELETE *action*] [ON UPDATE *action*] }

table_constraint is:

[CONSTRAINT *constraint_name*]
{ UNIQUE (*column_name* [, ...]) |
PRIMARY KEY (*column_name* [, ...]) |
CHECK (*expression*) |
FOREIGN KEY (*column_name* [, ...]) REFERENCES *reftable* [(*refcolumn* [, ...])]
[ON DELETE *action*] [ON UPDATE *action*] }

and *action* is one of RESTRICT, CASCADE, SET NULL, or SET DEFAULT

SELECT

SELECT [ALL | DISTINCT]
* | *expression* [AS *output_name*] [, ...]
[FROM *from_item* [, ...]]
[WHERE *condition*]
[GROUP BY *expression* [, ...]]
[HAVING *condition* [, ...]]
[{ UNION | INTERSECT | EXCEPT } [ALL] *select*]
[ORDER BY *expression* [ASC | DESC | USING *operator*] [, ...]]
[FOR UPDATE [OF *tablename* [, ...]]]

where *from_item* can be:

[ONLY] *table_name* [*] [[AS] *alias* [(*column_alias_list*)]] |
(*select*) [AS] *alias* [(*column_alias_list*)] |
from_item [NATURAL] [*join_type*] JOIN *from_item* [ON *join_condition* | USING (*join_column_list*)]

and *join_type* can be:

INNER |
LEFT [OUTER] |
RIGHT [OUTER] |
FULL [OUTER] |
CROSS

For INNER (the default) and OUTER join types, exactly one of NATURAL, ON *join_condition*, or USING (*join_column_list*) must appear. For CROSS JOIN, none of these items may appear.

CREATE VIEW

CREATE VIEW *view* [(*column name list*)] AS SELECT *query*

Some Data Types

integer, int, smallint

character[*n*], char[*n*], character varying[*n*], varchar[*n*], varchar

numeric, numeric[*precision*], numeric[*precision, scale*], real, double

boolean, date,

Note: [*xxx*] means *xxx* is optional, { *xxx* | *yyy* } means *xxx* or *yyy*.

APPENDIX B

The Musicians Database Schema

```
create table Musician (
MusicianId smallint primary key,
Mus_Name varchar (15) not null,
Telephone varchar (15),
Type varchar (15) not null
);

create table Instrument (
InstrumentId smallint primary key,
Inst_Name varchar (15) not null,
Musical_Key varchar (10) not null
);

create table Played_By (
MusicianId smallint not null references Musician,
InstrumentId smallint not null references Instrument,
primary key (MusicianId, InstrumentId)
);

create table Album (
AlbumId smallint primary key,
Alb_Title varchar (15) not null,
Format varchar (5),
Producer smallint not null references Musician(MusicianId)
);

create table Song (
AlbumId smallint not null references Album,
Song_No smallint not null,
Song_Title varchar (15) not null,
primary key (AlbumId, Song_No)
);

create table Performed_By(
MusicianId smallint not null references Musician,
AlbumId smallint not null,
Song_No smallint not null,
primary key (MusicianId, AlbumId, Song_No),
foreign key (AlbumId, Song_No) references Song
);
```