Student Id: _____

TE WHARE WĀNANGA O TE ŪPOKO O TE IKA A MĀUI

VICTORIA
UNIVERSITY OF WELLINGTON

EXAMINATIONS – 2015

TRIMESTER 2

SWEN304

Database System Engineering

Time allowed: THREE HOURS

CLOSED BOOK

Permitted materials: Only silent non-programmable calculators or silent programmable calculators with their memories cleared are permitted in this examination

Printed foreign language to English dictionaries are permitted

Instructions: Marks are shown for each question as a whole and also for their parts

Answer all questions

Make sure that your answers are clear and to the point.

Questions:

1.	**Relational Database Foundations**	**[15 marks]**
2.	**Database Updates and Queries**	**[40 marks]**
3.	**Query Processing and Optimisation**	**[30 marks]**
4.	**Functional Dependencies**	**[15 marks]**
5.	**Database Normalization**	**[30 marks]**
6.	**Entity Relationship Data Model**	**[15 marks]**
7.	**Mapping ER Diagrams into Relational Databases**	**[15 marks]**
8.	**Transaction Processing and Concurrency Control**	**[20 marks]**

Appendix: Simplified PostgreSQL Documentation

Student Id: _____

Question 1. Relational Database Foundations [15 marks]

a) [3 marks] What is a *relational database schema*?

ANSWER

b) [4 marks] What is a *relation schema*? What is a *relation* in the relational data model? How are these two related to each other?

ANSWER

c) [4 marks] What is a *relation schema key*, and what are the properties of a relation schema key?

ANSWER

d) [4 marks] What is the *foreign key*, and what is it used for?

ANSWER

Question 2. Database Updates and Queries [40 marks]

GreatProgrammers is a software development company with customers all over New Zealand. This company runs many software development projects.

Suppose the following relational database schema is used by *GreatProgrammers* to manage information on their software development business.

S = {
Staff ({StaffId, Name, JobTitle}, {StaffId}),
Project ({PId, PName, Customer, Location, PLeader}, {PId}),
AssignedTo ({StaffId, PId, DateFrom, DateTo}, {StaffId, PId}),
ContributesTo ({StaffId, PId, Activity, Hours}, {StaffId, PId, Activity}),
Payment ({JobTitle, Salary}, {JobTitle})
}

I = {
Staff [JobTitle] $\subseteq$ *Payment [JobTitle],*
Project [PLeader] $\subseteq$ *Staff [StaffId],*
AssignedTo [StaffId] $\subseteq$ *Staff [StaffId],*
AssignedTo [PId] $\subseteq$ *Project [PId],*
ContributesTo [(StaffId, PId)] $\subseteq$ *AssignedTo [(StaffId, PId)]*
}

Not Null Constraints = {
Null (Project, Location) = Not,
Null (Project, PLeader) = Not,
Null (Payment, Salary) = Not
}

Each project has a project ID, project name, customer name, location and project leader. Each staff member can be assigned to several projects. Each staff member can contribute various activities to projects they are assigned to.

Projects without customers are in-house projects. For them the attribute Customer will be Null.

Staff members have job titles such as 'Business Analyst' or 'Programmer' or 'DBA'. The job title of a staff member determines her/his salary.

a) [6 marks] Use SQL as a Data Definition Language to define the table *Project*. Assume *PId* and *PLeader* have integer number data type. The value of *PId* is limited to [0001, 9999]. *PName, Customer,* and *Location* have string data type with maximum length of 30. *PName* cannot be Null.

ANSWER

b) [2 marks] Use SQL as a Data Manipulation Language to insert a new tuple into the *Project* table. The new project is assigned a project ID of 1724 and project name is 'Courses'. The customer is 'VUW', location is 'Wellington', and project leader is 325.

ANSWER

c) [2 marks] Use SQL as a Data Manipulation Language to update the project leader of the inserted tuple in part **b)** to 318.

ANSWER

d) [3 marks] Use SQL as a Data Manipulation Language to delete all projects (from the Project table) whose project leader is 367 and location is 'Auckland'.

ANSWER

e) [2 marks] State ONE difference between the meanings of the SQL keywords DELETE and DROP.

ANSWER

f) [3 marks] Use SQL as a query language to retrieve all project names from the Project table. Ensure that there are no duplicates in the output.

ANSWER

g) [6 marks] Use SQL as a query language to retrieve the *PId* and *PName* of all the projects that a staff member with ID 456 is assigned some work from.

ANSWER

h) [12 marks] For each project, you want to know the total number of hours involved, which staff member contributed the most hours, and how many hours this staff member contributed. Use SQL as a query language to compute this information.

Hint: Use the step-wise approach (view) to query definition if you like.

ANSWER

i) [4 marks] Given the following two relations, R1 and R2:

R1

X	Y
a_1	ω
a_2	b_1

R2

Y	Z
b_1	c_1
b_2	c_1

Draw two tables showing the results of the following two SQL queries (one table for each query):

A. SELECT * FROM R1 JOIN R2 ON R1.Y = R2.Y;

B. SELECT * FROM R1 NATURAL JOIN R2

ANSWER

(Spare Page for Extra Working)

Question 3. Query Processing and Optimisation [30 marks]

a) [3 marks] Recall the ***GreatProgrammers*** database from **Question 2**. Translate the following relational algebra query into plain English:

$\pi_{\text{Customer}}(r(\text{Project}))$

ANSWER

b) [4 marks] Suppose you want to know the names of the projects that are located in Wellington and have a leader with the job title of 'Business Analyst'. To answer this question you can write the following SQL query:

```
SELECT PName
FROM Staff NATURAL JOIN Project
WHERE JobTitle = 'Business Analyst' AND Location = 'Wellington'
```

Translate the above SQL query into relational algebra.

ANSWER

c) [5 marks] Use relational algebra to retrieve the staff IDs of all staff who have never been assigned work from the Project with ID 101.

ANSWER

d) [6 marks] Translate the following relational algebra query into plain English.

π_{Salary} ($\sigma_{Location}$ = 'Wellington' ($\sigma_{DateFrom}$ >= 2012-01-01(r(Payment)* r(Staff)* r(Project)* r(AssignedTo))))

ANSWER

e) [5 marks] Draw a query tree for your relational algebra query in part **d)**.

ANSWER

f) **[7 marks]** Optimise the query tree in part **e)** using heuristic rules for algebraic query optimisation as discussed in lectures and tutorials. Explain which heuristic rules you can apply. Draw the optimised query tree.

ANSWER

(Spare Page for Extra Working)

Student Id: _____

Question 4. Functional Dependencies [15 marks]

a) [5 marks] Consider the database instance below. For which of the following functional dependencies can you conclude that they **do not hold** over the relation schema WorksFor?

WorksFor			
empId	depId	jobTitle	percentTime
301	12	Sales engineer	50
301	21	Programmer	50
302	12	Manager	100
311	21	Manager	100
320	12	Ad writer	100

i) empId → depId
ii) empId + depId → jobTitle
iii) empId → jobTitle

ANSWER

b) [10 marks] Consider the set of functional dependencies $F = \{A \rightarrow D, C \rightarrow D, AD \rightarrow C\}$. Compute a minimal cover of F. Justify your answer.

ANSWER

(Spare Page for extra working)

Question 5. Database Normalization [30 marks]

Consider the following set of attributes

$$U = \{\text{order\_id, product\_no, product\_name, price, order\_status, date}\}$$

and the following set of functional dependencies

$$F = \{\text{order\_id} \rightarrow \text{order\_status + date, order\_id + product\_no} \rightarrow \text{price,}$$
$$\text{product\_no} \rightarrow \text{product\_name}\}.$$

a) [5 marks] Determine the attribute closure of the set {order_id, product_no} with respect to U and F. Justify your answer.

ANSWER

b) [5 marks] Determine all candidate keys for the universal relation schema $N(U, F)$. Justify your answer.

ANSWER

c) [8 marks] Is $N(U, F)$ in Third Normal Form? Justify your answer. If not, determine a lossless Third Normal Form decomposition.

ANSWER

d) [8 marks] Is $N(U, F)$ in Boyce Codd Normal Form (BCNF)? Justify your answer. If not, determine a lossless BCNF decomposition.

ANSWER

e) **[4 marks]** Under what conditions is a decomposition called *dependency-preserving*. State whether the decomposition computed in part **d)** is dependency-preserving. Justify your answer.

ANSWER

(Spare page for extra working)

Question 6. Entity Relationship Data Model [15 marks]

Suppose Wellington's most popular DVD rental shop, *Rent'n'Watch*, hires you to design a new DVD rental database. The shop provides you with the following list of requirements:

- For each customer, it is required to store the customer number, first name, last name, and address.
- For each staff member, it is required to store the staff id, first name, last name, and contact phone number.
- *Rent'n'Watch* rents out DVDs containing movies. Each DVD contains exactly one movie. For each movie there may be many DVD copies.
- For each movie, it is required to store its title, production year, and length.
- Each DVD has its unique DVD number and some features.
- When a customer hires a DVD from a certain staff member, the hiring date and due date will be recorded.

Draw an EER diagram of the new database for *Rent'n'Watch.* Show the entity types and relationship types, but skip the attributes on the diagram. Show *structural constraints* explicitly on the diagram.

Give attributes separately in the form

$$Entity\_Type\_Name\ \{Attribute_1, \ldots, Attribute_n\},$$

with entity type keys underlined, or

$$Relationship\_Type\_Name\ (Entity\_Type_1, \ldots, Entity\_Type_n, \{Attribute_1, \ldots, Attribute_m\}),$$

where the set of a relationship type attributes can be empty.

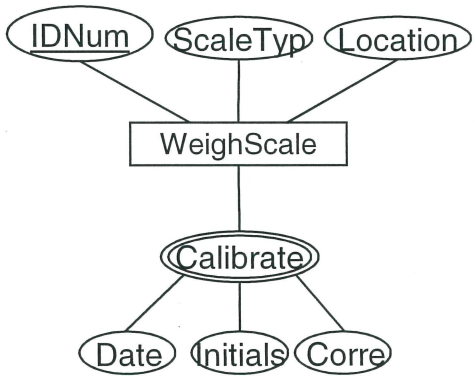
Note: Use the notation and conventions for drawing ER diagrams used in lectures. If you prefer to use another notation that differs from the one we used in lectures, clearly define all differences.

ANSWER

Question 7. Mapping ER Diagrams into Relational Databases [15 marks]

For each of the following ER diagrams, map the diagram to an equivalent relational database schema. Specify the attributes and keys of the relation schemas, any attribute constraints, and a set of non-redundant referential integrity constraints. You do not need to specify the domains of the attributes.

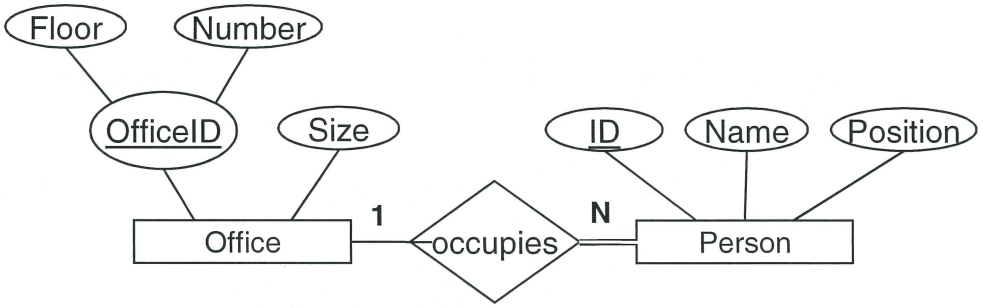
a) [5 marks] A database of all the measuring scales in a research facility and a record of their calibration history.



Note: No WeighScale is calibrated more than once a day.

ANSWER

b) [6 marks] A database of the allocation of staff to offices.



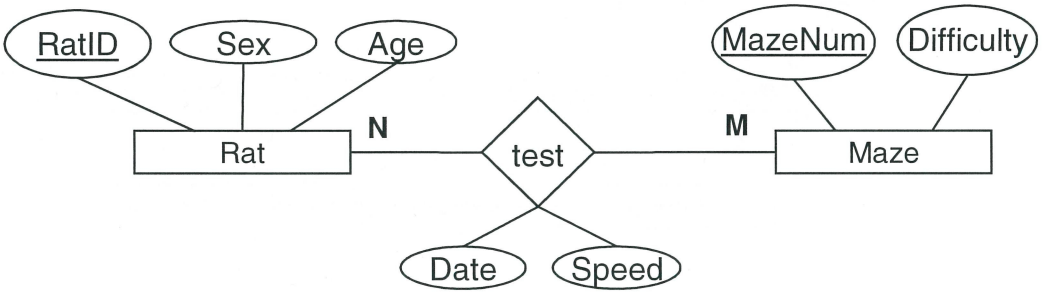
i) [4 marks] Give your relational database schema.

ANSWER

ii) [2 marks] Explain how the relationship with total participation is enforced in your relational database schema.

ANSWER

c) **[4 Marks]** A database of results of experiments involving rats and mazes. A rat may not be tested on the same maze more than once.



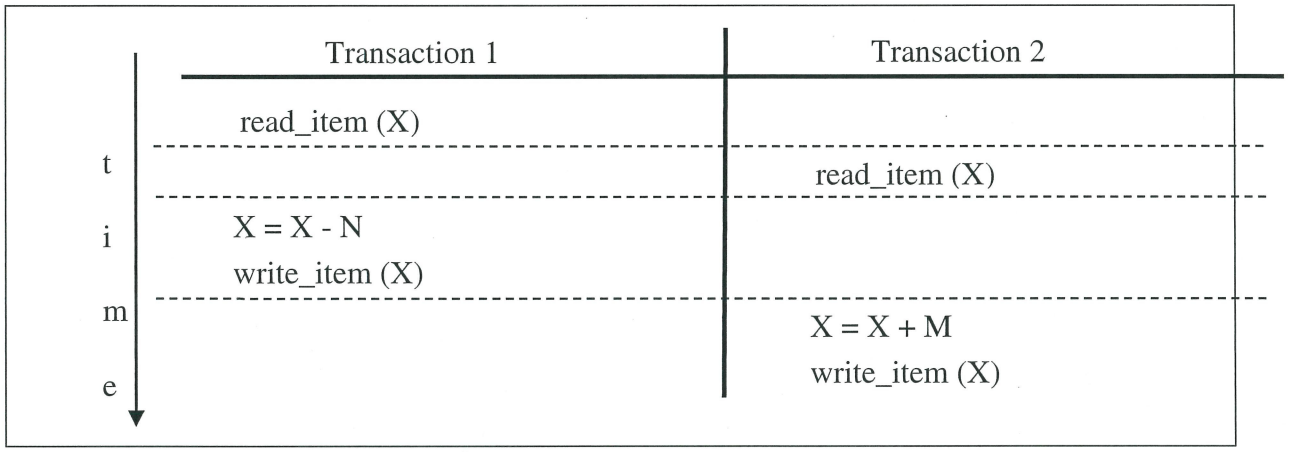
ANSWER

(Spare Page for extra working)

Student Id: _____

Question 8. Transaction Processing and Concurrency Control [20 marks]

a) **[12 marks]** Consider the concurrent execution of two transactions shown below. The dashed horizontal lines designate the moments when the CPU control changes from one transaction program to the other.



i. **[2 marks]** What is the name of the transaction anomaly that occurred?

ANSWER

ii. **[4 marks]** Why has the transaction anomaly happened, and what is its consequence?

ANSWER

iii. [6 marks] Redraw the Figure above. Use locks to avoid the transaction anomaly. Suppose a database item can be in one of the following three states: shared locked, exclusively locked, or unlocked. Retain the same order of transaction commands as above. Indicate waits and points of time when a lock is granted.

ANSWER

b) [8 marks] We discussed three variants of the two-phase locking protocol in lectures.

i. [4 marks] Briefly describe the *basic two-phase locking* protocol.

ANSWER

ii. [2 marks] Briefly describe the *strict two-phase locking* protocol as an extension of the basic two-phase locking protocol.

ANSWER

iii. [2 marks] Briefly describe the *conservative two-phase locking* protocol as an extension of the basic and strict two phase-locking protocols.

ANSWER

(Spare Page for extra working)

APPENDIX: 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*.

****