THE UNIVERSITY OF SYDNEY

Room Number	S458
Seat Number	4
Student Number	49005148

ANONYMOUSLY MARKED
(Please do not write your name on this exam paper)

CONFIDENTIAL EXAM PAPER

This paper is not to be removed from the exam venue

Computer Science

EXAMINATION

Semester 1 - Final, 2025

COMP9120 Database Management Systems

EXAM WRITING TIME: 2 hours

READING TIME: 10 minutes

EXAM CONDITIONS:
This is a CLOSED book exam - no material permitted.

MATERIALS PERMITTED IN THE EXAM VENUE:
(No electronic aids are permitted e.g. laptops, phones)

Calculator – non-programmable

MATERIALS TO BE SUPPLIED TO STUDENTS:
Multiple Choice and Short Answer Questions paper

INSTRUCTIONS TO STUDENTS:

This exam consists of **4 Multiple Choice Questions** worth a total of 8 marks, and **8 Short Answer Questions** worth a total of 42 marks.

Answer all questions in the spaces provided on this paper. You may use pencil or ink. Marks may not be given where there is insufficient evidence of the working required to obtain the solution. If you need additional writing space, please use the extra pages provided at the end of this exam booklet. Only pages in this exam booklet will be marked.

Please tick the box to confirm that your examination paper is complete. [✓]

Question 1: (2 marks)

With regard to database design, tick the correct statement(s).

0...X →

[ ] A cardinality constraint is the same as a key constraint.

[✓] A cardinality constraint refers to how often an entity participates in a relationship.

[✓] By default, the primary key of a relationship is the combination of the primary keys of those entity types participating in that relationship.

[ ] The primary key of a weak entity is its discriminator. X

composite PK

Question 2: (2 marks)

Consider the following relation:
Employee(empid, name, salary)
with 5000 tuples and a BEFORE UPDATE trigger on salary.
If the company gives the employees a pay raise using the following statement:
UPDATE Employee SET salary=salary*1.025;

How often is a statement-level trigger executed? Tick the right answer.

[ ] 5000
[✓] 1
[ ] 0
[ ] 500

Question 3: (2 marks)

Consider a relation schema:
R(A, B, C, D, E, F, G)

With the following set of **functional dependencies**:

AB → C

AC → D

D → E

E → F

F → G

B → D

AB . ABC
AC ABCD
D ABCDE
ABCDEF
ABCDEFG

What is the closure of **AB** (i.e., **$(AB)^+$**)?

[ ] {AB}
[ ] {ABCD}
[ ] {CD}
[✓] {ABCDEFG}

Question 4: (2 marks) Unknown OR True

In three-valued logic, the result of (5 + NULL) OR (TRUE OR UNKNOWN) is:

[✓] True
[ ] Unknown
[ ] False

Question 5: (6 marks)

Consider the airline database consisting of the following tables:

Employee (EmployeeID, Name, Address, Salary) -- Represents airline staff handling reservations.

Passenger (PassengerID, Name, PhoneNumber) -- Represents customers who book flights.

Booking (BookingID, EmployeeID, PassengerID, BookingDate) -- Represents flight reservations made by passengers with the help of employees.

FlightTicket (TicketID, BookingID, FlightNumber, TicketCost) -- Represents tickets associated with each booking.

Produce the SQL statement for each of the following questions

a. Find all the passenger's names who have made the highest number of bookings. **(3 marks)**
b. Find the total number of bookings where the total ticket cost in each booking exceeds $200. **(3 marks)**

>

(a)
```
Select P.Name
From Passenger P
Join Booking B ON P.PassengerID = B.PassengerID
Group By P.Name
Having Count(B.BookingID) >= All ( Select Count(B2.BookingID)
                                   From Passenger P2
                                   Join Booking B2 on P2.PassengerID
                                   =B2.PassengerID
                                   Group By P.Name );
```

(b)
```
Select Count(*)
From ( select Distinct B.BookingID
       From Booking B
       Join FlightTicket F on B.BookingID = F.BookingID
       Group By B.BookingID
       Having Sum( F.TicketCost) >200) ;
```

Question 6: (6 marks)

This question is based on the following E-R diagram shown in Figure 1 which describes the information kept by a real estate agent.

a) Translate the E-R diagram into a relational model using the following **textual notation**. Each relation should be written in the form: (**3 marks)**

Name (attribute, attribute, …)
PK= (attribute list), FK = (attribute list->parent relation)

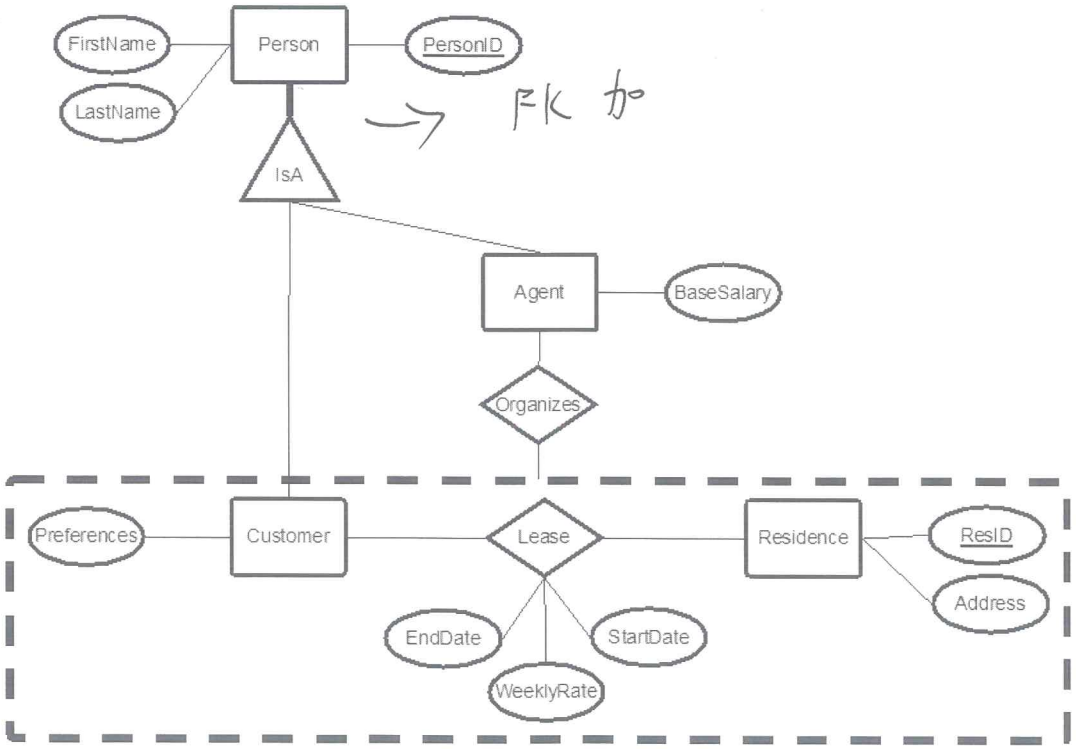
An 'attribute list' is one or more, comma-separated attribute names. Each relation must have a primary key (PK) defined. A relation can have zero or more FKs specified.

For example: Enrolment (studentId, courseId, mark)

PK = (studentId, courseId), FK=(studentId->Student, courseId->Course)

Do not write this relational model in SQL DDL syntax, use the above convention to describe your model.

b) Write SQL CREATE TABLE statements for the Customer and Residence relations from your relational model, as well as any associated relationship relations between these two relations. You should show all relevant attributes, types and key constraints for these relations. You should include foreign key integrity constraints, like ON DELETE clauses where applicable, as well as any applicable NOT NULL or UNIQUE constraints. (**3 marks)**



(a) Person (FirstName, LastName, PersonID)
PK = (PersonID)
Agent (BaseSalary, PersonID)
PK = (PersonID) FK = (PersonID → Person)
Customer (Preference, PersonID)
PK = (PersonID) FK = (PersonID → Person)
Residence = (ResID, Address)
PK = (ResID)
Lease (EndDate, WeeklyRate, StartDate, CID, RID)
PK = (CID, RID) FK = (CID → Customer, RID → Residence)
Organizes (AId, CId, RId)
PK = (AId, Cid, Rid) FK = (AId → Agent, (cid, Rid) → Lease)

(b)
```
Create Table Person(
    FirstName Varchar(50) not null,
    LastName Varchar(50) not null,
    PersonID Int,
    Primary key (PersonID)
);
Create Table Customer(
    Preference Varchar(255),
    PersonId Int,
    Primary key (PersonId)
    Foreign key (PersonId) References Person(PersonID) on delete Cascade
                                                       on update Cascade
);
Create Table Residence(
    ResID Int,
    Address Varchar(255) Not null,
    Primary key (ResID)
);
Create Table lease(
    EndDate Date, StartDate Date, WeeklyRate Float, CId Int,
    RIdInt, Primary key (CId, RId),
        Foreign key (CId) References customer(PersonId),
        Foreign key (RId) References Residence(ResId) );
```

on delete Cascade
on update Cascade

On delete Cascade
on update Cascade

Question 7: (3 marks)

Given a hospital database with the following relations:

Department (department_id, department_name, total_salary_budget)

Position (position_id, department_id, position_title, position_number)

Staff (staff_id, name, salary, position_id)

```
CREATE ASSERTION check_department_salary_budget
CHECK (NOT EXISTS
   (SELECT department_id
    FROM Department
    WHERE
       (SELECT SUM(Staff.salary)
        FROM Staff
        JOIN Position ON Staff.position_id = Position.position_id
        WHERE
              Position.department_id = Department.department_id) >
Department.total_salary_budget));
```

State the above assertion in one clear sentence.

~~Make sure "The sum of all staffs' salary, who belong to a certain department, cannot exceed the total salary budget of that department.~~

There is no department's "total salary budget" less than the total salary this department need to pay to its staffs.

Question 8: (6 marks) A (P,Q,R,S)

Suppose you are given a relation A with four attributes P, Q, R, S and the following sets of FDs:

$R \rightarrow S$

$R \rightarrow P$

$Q \rightarrow R$

1. Identify the candidate key for A. Show how you obtained the key. **(2 marks)**
2. Identify the highest normal form that A satisfies (1NF, 2NF, 3NF, or BCNF). Show how you arrived at your answer. **(2 marks)**
3. If A is not in BCNF, decompose it into a set of BCNF relations that preserve the dependencies. **(2 mark)** 忘记解释为什么A不是BCNF所以扣了一分

1. only Q in LHS, so it be part of key since it's minimum(only one element) so it should be a candidate key.
- also check $(R)^+$, $(S)^+$, $(P)^+$, $(RS)^+$, $(RP)^+$, $(SP)^+$, $(RS,P)^+$ but all of them are not even a superkey

2. The highest is 2NF, As you can see for "R->S", both of them are non-key, so violate 3FN.

3. $R_1(P,Q,R,S)$

R_2 (Q,R) R_3 (P,R,S)

$R_1(P,Q,R,S)$ split in left way.

- R_2 is BCNF as Q is the Super key so Q->R satisfy BCNF
- R_3 is BCNF as R is the superkey so R->S is BCNF R->P
- It's Preserve, all FD can be found in either R_2 or R_3

Question 9: [5 marks]

Consider the database consisting of the following tables

Hotel (hotel_id, hotel_name, address, fare) 1 2 3 4 5 6

Booking (hotel_id, guest_id) 1 2 5 6 1

Guest (guest_id, guest_name, guest_address)

Write down the Relation Algebra (RA) expressions for the following queries

a) Find the name and address of all guests who has made a booking in 'Crown' hotel. **(2.5 marks)**

b) Find the name of the hotel that has been booked by at least 1 guest. **(2.5 marks)**

(a) $\pi_{guest\_name,\ guest\_address}(\sigma_{hotel\_name='Crown'}(Hotel \bowtie Booking \bowtie Guest))$

(b) $\pi_{hotel\_name}(Hotel \bowtie_{Hotel.hotel\_id = Booking.hotel\_id} Booking)$

Question 10: (6 marks)

Determine whether the following schedule

R2(x), R1(y), W2(y), R3(z), W3(x), R1(z), R1(x)

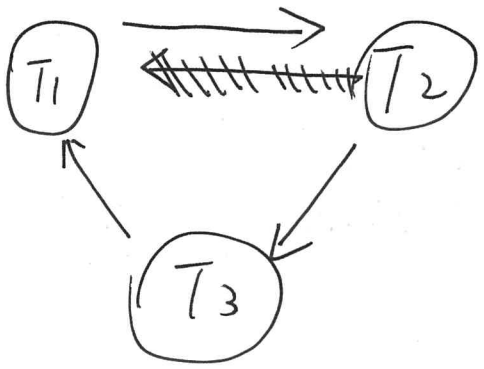
is conflict serializable; justify your answer by drawing the precedence graph. If it is conflict serializable, please provide a conflict equivalent serial schedule.

Conflict pairs are: R2(x) W3(x)

~~R2(x) R1(x)~~

R1(y) W2(y)

W3(x) R1(x)



it's not Conflict serializable as a cycle between T1, T2, T3

Question 11: (6 marks)

You have been brought in as a consultant to improve the performance of a system dealing with the tracking of Shipment at the dock. You focus on database operations involving the relation

Shipment(shipmentId, invoice, weight, container, date, company_id)

Each row in this relation takes up 16 bytes, including the 4-byte primary key shipmentId. There are 200,000 rows in the table, and the table has been defined with a target load factor of 75%, so there is about 25% free space in each table page. The database uses 2048-byte pages for storing both data and indexes, with the first 196 bytes of each page reserved for header data. Records and index entries cannot span pages.

a) Estimate the space, in bytes, required to store just the relation data, excluding any indexes. State any assumptions you make. **(2 marks)**

Your analysis of the time taken for various transactions shows that many of the problematic ones are doing searches over *container*, such as the following:

```
SELECT COUNT(*)
        FROM Shipment
        WHERE container= 'C-101' AND weight < 20;
```

b) Estimate the number of I/Os needed to process the above query, ignoring any effects of buffering. **(2 marks)**

c) What changes would you recommend fixing this issue, and what effect would you expect it to have on the performance of such queries? **(2 mark)**

(a) ~~Usable~~ size for a page without head = 2048 − 196 = 1852

record/page = $\lfloor \frac{1852}{16} \times 0.75 \rfloor = 86$

# of page = $\lceil \frac{200{,}000}{86} \rceil = 2326$

Final space

2326 × 2048

476 3648 bytes

(b) The data is unsort, and no B+ index tree, so we can only do linear scan. The condition is range query, so we need to scan all pages, from (a) we know we need scan 2326 pages So 2326 I/Os needed

(c) we should build a b+ index tree ~~with~~ using "container" and "weight" (~~only~~ effective for the query provided). The performance is much better, It usually take the height of B+ tree + retrive page number I/Os to finish query when we use the corresponding B+ tree key for the query "where" condition

	Tuple	Page
R	55,000	55,000/50=1100
S	85,000	

Question 12: (4 marks)

Assume we have two relations R(A,B,C) and S(C,D.E). The number of tuples in R is 55,000. S has 85,000 tuples. One block would hold exactly 50 tuples of R or 170 tuples of S. One page contains exactly 1 block.

Estimate the cost of the natural join of R and S:

a. Nested-loop join. Which order of the join would yield a minimal cost? **(2 marks)**

b. Block nest-loop join. Which order of the join would yield a minimal cost? **(2 marks)**

	Tuple	Page
R	55,000	55000/50=1100
S	85,000	85000/170=500

(a) # of outer page + # of outer tuple × # of inner page

R is outer: 1100 + 55,000 × 500 = 27501100 ✓

S is outer: 500 + 85,000 × 1100 = 93500500

so use R as outer, S as inner is prefered

(b) # of outer page + # of outer page × # of inner page

R is outer: 1100 + 1100 × 500 = 551100

S is outer: 500 + 500 × 1100 = 550500 ✓

So use S as outer, R as inner is prefered

THIS PAGE LEFT INTENTIONALLY BLANK.

Use this page for additional writing space if you run out of space in an answer area. If you want any writing on this page to be marked, you MUST write "see additional page" in the relevant answer area and clearly indicate the question number you are answering on this page. Any writing outside of this booklet will not be examined.

THIS PAGE LEFT INTENTIONALLY BLANK.

Use this page for additional writing space if you run out of space in an answer area. If you want any writing on this page to be marked, you MUST write "see additional page" in the relevant answer area and clearly indicate the question number you are answering on this page. Any writing outside of this booklet will not be examined.

END OF EXAMINATION