NATIONAL UNIVERSITY OF SINGAPORE

SCHOOL OF COMPUTING
MIDTERM ASSESSMENT FOR
Semester 1 AY2025/2026

IT5008 Database Design and Programming

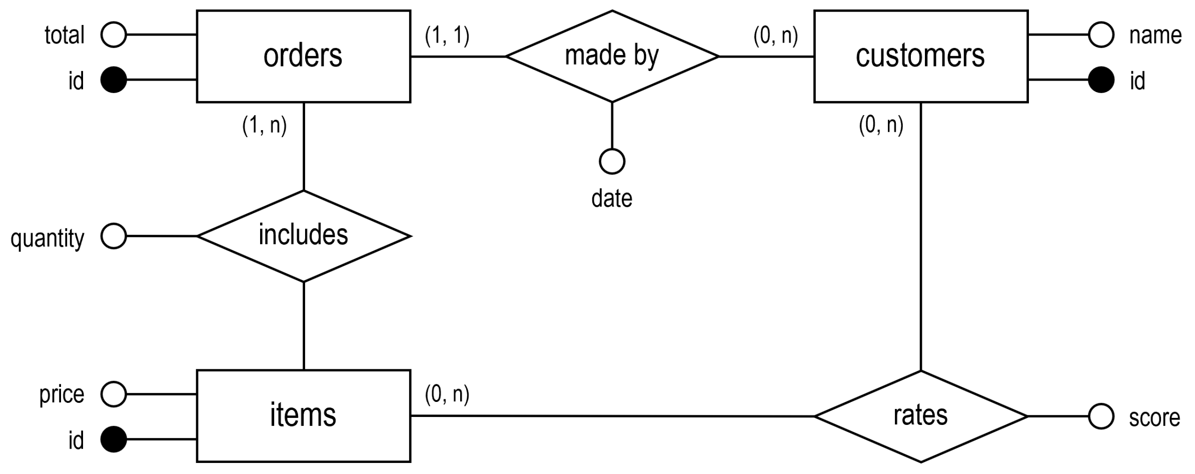
October 2025 Time Allowed 60 minutes

INSTRUCTIONS TO CANDIDATES

1. This assessment paper contains **10** questions and comprises **8** printed pages, including this page. The last **2** pages is left blank for your working.
2. The total marks for this assessment is **50**.
3. This is a **CLOSED-BOOK** assessment. You are only allowed to refer to one double-sided A4-size paper.
4. Shade and write your student number in the answer sheet. Do **NOT** write your name.
5. Submit only the answer sheet at the end of the assessment.
6. Write all your answers in the answer sheet provided.
7. Your answer must fit into the given space (*i.e.*, number of lines, box, *etc*).
8. Please read the additional instruction for multiple choice (MCQ) and multiple response (MRQ) questions. More specific instructions *may* be given at the beginning of each section, please read them carefully.
   - Any answer not in the space provided will not be graded.
   - For MCQ and MRQ, **shade your answer in the corresponding bubble on the answer sheet**.
   - For MCQ, select the most appropriate answer.
   - For MCQ, if multiple answers are equally appropriate, pick one and shade **ONLY** the chosen answer on the answer sheet. Do **NOT** shade more than one answer.
   - For MRQ, shade **ALL** correct answers. Partial marks *may* be given, but not guaranteed.
   - For MRQ, choose "*None of the above*" only if there is no other appropriate choice. If chosen, you should not choose any other choice.
9. All SQL query in this assessment are run on PostgreSQL 17.
10. We will use the monospace `NULL` to represent NULL-values. We will represent string with single-quote `'string'` .

GOOD LUCK

In this paper, we will use the following entity-relationship diagram.



Schema (13 points)

1. (3 points) **(MCQ)** Consider the following constraints.

   (i) An order is made by exactly one customer.

   (ii) A customer can only rate the item they have bought at least once.

   (iii) An order must include at least one item.

   Which constraints above are enforced by the entity-relationship diagram? Select the most appropriate option.

   ✗ only (i)

   ✔ only (i) and (iii)

   ✗ only (i) and (ii)

   ✗ only (ii) and (iii)

   ✗ (i), (ii), and (iii)

   Notes: For (i), it is enforced by cardinality of (1,1) between `orders` and `made by`. For (iii), it is enforced by cardinality of (1,n) between `orders` and `includes`.

   For (ii), this is not enforced by the entity-relationship diagram as `rates` connect only `customers` and `items`. Hence, it has no information on `orders`.

Consider a schema translation from entity-relationship diagram into `CREATE TABLE` statements following the convention in the lecture (i.e., 3 rules + 3 exceptions). Constraints should only be enforced by the schema using only " `PRIMARY KEY` ", " `FOREIGN KEY` ", " `UNIQUE` ", " `NOT NULL` ", and " `CHECK` ".

(A) `customers(id, name)` with the primary key of `id` .

(B) `orders(total, id, cid, date)` with the primary key of `id` .

(C) `items(id, price)` with the primary key of `id` .

(D) `includes(iid, oid, quantity)` with the primary key of `(iid, oid)` .

(E) `rates(iid, cid, score)` with the primary key of `(iid, cid)` .

2. (3 points) **(MCQ)** Consider the following constraints.

   (i) An order is made by exactly one customer.

   (ii) A customer can only rate the item they have bought at least once.

   (iii) An order must include at least one item.

   Which constraints above can be enforced by the schema? Select the most appropriate option.

   ✔ only (i)
   ✗ only (i) and (iii)
   ✗ only (i) and (ii)
   ✗ only (ii) and (iii)
   ✗ (i), (ii), and (iii)

   Notes: For (i), it is enforced because the primary key of `orders` is `id` , hence, it uniquely identifies `cid` .

   For (ii), we do not have information regarding the `orders.id` , so we cannot enforce this. For (iii), since the tables are different (i.e., `orders` and `items`), we cannot enforce that insertion to `orders` is followed by insertion to `items` .

3. (3 points) **(MCQ)** Consider the following constraints.

   (i) The total price (i.e., `total`) of an order must be the sum of the prices of all included items.

   (ii) The score rating can only be an integer 1, 2, 3, 4, or 5.

   (iii) The quantity must be greater than 0.

   Which constraints above can be enforced by the schema? Select the most appropriate option.

   ✗ only (i)
   ✗ only (i) and (iii)
   ✗ only (i) and (ii)
   ✔ only (ii) and (iii)
   ✗ (i), (ii), and (iii)

Notes: For (i), this cannot be enforced with a `CHECK` as we need to query multiple rows.

For (ii), this is a simple `CHECK (score >= 1 AND score <= 5)`.

For (iii), this is a simple `CHECK (quantity > 0)`.

4. (4 points) **(MRQ)** We use the symbol `(t1.attr1, t1.attr2)` $\rightsquigarrow$ `(t2.attr3, t2.attr4)` to be equivalent to the following constraint.

```
FOREIGN KEY (attr1, attr2) REFERENCES t2 (attr3, attr4)
```

Which of the following foreign keys are required by the schema translation? Select all options that apply.

- ✗ `(customers.id)` $\rightsquigarrow$ `(orders.cid)`
- ✓ `(orders.cid)` $\rightsquigarrow$ `(customers.id)`
- ✗ `(rates.cid)` $\rightsquigarrow$ `(orders.cid)`
- ✓ `(rates.cid)` $\rightsquigarrow$ `(customers.id)`
- ✗ *None of the above.*

Select "*None of the above*" only if there are no other options that apply. Do not select "*None of the above*" with any other options.

Notes: Should a straightforward foreign key to the primary key.

SQL Query **(25 points)**

Recap the tables obtained from the entity-relationship diagram using our schema translation technique.

(A) `customers(id, name)` with the primary key of `id`.

(B) `orders(total, id, cid, date)` with the primary key of `id`.

(C) `items(id, price)` with the primary key of `id`.

(D) `includes(iid, oid, quantity)` with the primary key of `(iid, oid)`.

(E) `rates(iid, cid, score)` with the primary key of `(iid, cid)`.

Assume that the correct foreign keys have been added regardless of your answer to Question 4.

5. (4 points) **(MRQ)** Consider the following problem.

Find the name of the different customers that made at least one order with a total price over 1000.

Which of the SQL queries below solve the problem above? Select all options that apply.

✗
```
SELECT c.name
FROM customers c INNER JOIN orders o
  ON o.cid = c.id AND o.total > 1000;
```

✗
```
SELECT name
FROM customers NATURAL JOIN orders
WHERE total > 1000;
```

- ✗
```
SELECT c.name
FROM customers c LEFT JOIN orders o
  ON o.cid = c.id
WHERE o.id IS NOT NULL AND o.total > 1000;
```
- ✗
```
SELECT name
FROM customers
EXCEPT
SELECT c.name
FROM customers c, orders o
WHERE o.cid = c.id AND o.total <= 1000;
```
- ✓ *None of the above.*

Select "*None of the above*" only if there are no other options that apply. Do not select "*None of the above*" with any other options.

Notes: `INNER JOIN` and `LEFT JOIN` may contain different customer and not just different customer names. This is because the same customer can have multiple orders. Hence, neither A nor C can an answer.

`NATURAL JOIN` cannot work here because the attributes do not match. Hence, B cannot be an answer.

`EXCEPT` cannot work here as it may remove duplicates. But we want the names of the different customers. We may have two different customers (i.e., different `id`) but the same name. So, we cannot remove duplicates. Hence, D cannot be an answer.

6. (4 points) **(MRQ)** Consider the following SQL query.

```
SELECT tmp.name
FROM (
  SELECT c1.id, c1.name
  FROM customers c1
  EXCEPT
  SELECT r2.cid, c2.name
  FROM customers c2, rates r2
  WHERE c2.id = r2.cid
) tmp;
```

Which of the following description is true about the query above? Select all options that apply.

- ✗ The query finds the **different** name of the customers that have not rated any items.
- ✓ The query finds the named of the **different** customers that have not rated any items.
- ✓ The query may produce duplicate rows.
- ✗ The query will not produce duplicate rows.
- ✗ *None of the above.*

Select "*None of the above*" only if there are no other options that apply. Do not select "*None of the above*" with any other options.

Notes: Here, we assume that the query is correct. The inner query find the different customers as the `customers.id` is part of the columns. So, the `id` is guaranteed to be distinct. Hence, we find the **different** students and not the different names. The names can be the same as we will only remove the `id` column later on. Because the names can be the same, we may have duplicate rows.

7. (5 points) **(Fill in the Blank)** Consider the following problem.

   Find all the items with the highest average score for its rating (i.e., in the table `rates`). Exclude items that has never been rated before. Output the item id and the average score.

   You came up with a solution but accidentally spilled coffee all over your query. After erasing the stain, you need to fill in the blank again. Here is what you can recover.

```
SELECT r.iid, _____1_____(r.score) AS average
FROM rates r
GROUP BY _____2_____
HAVING _____3_____ >= ALL (
  SELECT _____4_____
  FROM rates r
  GROUP BY _____5_____
)
```

   Your task is to fill in the blanks from _____1_____ to _____5_____ so that the query solves the problem above. Write your answer on the answer sheet.

Notes:

```
SELECT r.iid, AVG(r.score) AS average
FROM rates r
GROUP BY r.iid
HAVING average >= ALL (    -- or HAVING AVG(r.score)
  SELECT AVG(r.score)
  FROM rates r
  GROUP BY r.iid
)
```

For the following SQL queries, you are **NOT** allowed to use **Common Table Expressions** (i.e., CTE).

8. (6 points) **(SQL)** Write an SQL query to solve the following problem.

   Find the different items that are the most expensive. Output only the item id and the price. Sort your result in descending order of price followed by ascending order of item id.

   You are **NOT** allowed to use **aggregate queries**.

Notes: As noted by one of the student, we do not need to sort in descending order of `price` as the most expensive item can only be a single price!

```
SELECT i1.id, i1.price
FROM items i1
WHERE i1.price >= ALL (
  SELECT i2.price
  FROM items i2
)
ORDER BY i1.price DESC, i1.id ASC;
```

9. (6 points) **(SQL)** Write an SQL query to solve the following problem.

   For each order, find the total quantity of all items in the order. Output only the order id and the total quantity. Sort your result in descending order of quantity followed by ascending order of order id.

Notes:

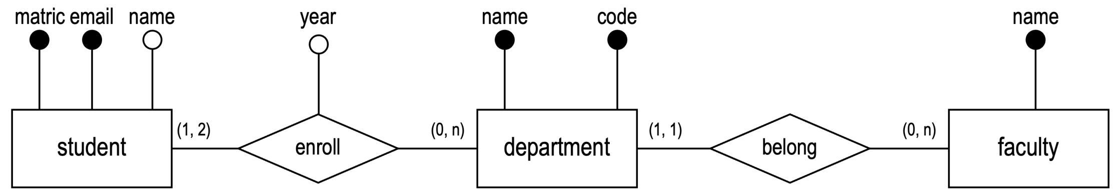
```
SELECT o.id, SUM(i.quantity) AS total
FROM orders o, includes i
WHERE o.id = i.oid
GROUP BY o.id
ORDER BY total DESC, o.id ASC;
```

Entity-Relationship Diagram (12 points)

10. (12 points) **(ERD)** NUS IT is hiring! In your first interview, you are asked to design the student database for the university. Your design should be drawn as entity-relationship diagram following the convention used in this semester. Additionally, it must satisfies the following constraints.

    1. A student is uniquely identified by their matric number.
    2. A student is uniquely identified by their email address.
    3. A student name must be recorded.
    4. A student must be enrolled to at least 1 department.
    5. A student may be enrolled to at most 2 department (e.g., double major program).
    6. A student year of enrolment to a department must be recorded. Note that the same student may be enrolled to two different departments at different year.
    7. A department is uniquely identified by the department name.
    8. A department is uniquely identified by the department code.
    9. A department belong to exactly one faculty.
    10. A department may have any number of students (including 0).
    11. A faculty only has a name.
    12. A faculty may have any number of department (including 0).

Notes:



END OF PAPER