CITY UNIVERSITY OF HONG KONG

Course code & title	:	CS3402 Database Systems
Session	:	Semester A 2023/24
Time allowed	:	2 Hours

This paper has 12 pages (including this cover page).

---

1. This paper consists of **FIVE** questions.
2. Write down your answer in the space provided.

---

*This is an **open-book** examination.*

Candidates are allowed to use the following materials/aids:

Printed lecture notes, personal notes, textbook and other course handout materials.

Materials/aids other than those stated above are not permitted.

No Electronic devices.

STUDENT ID		**VENUE**	
NAME		**SEAT NO**	

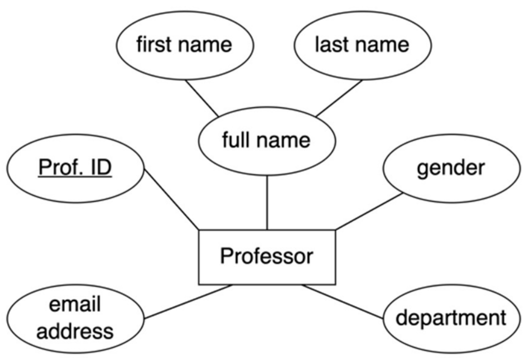
Q1 (20%)	Q2 (15%)	Q3 (20%)	Q4 (20%)	Q5 (25%)	Total (100%)

Problem One: ER Model [20 points]

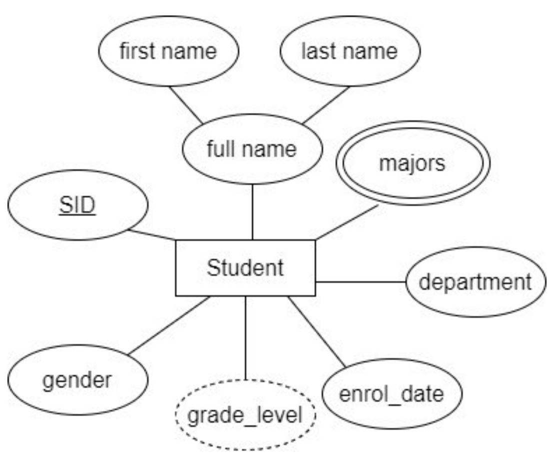
Consider a university undergraduate teaching management system database consisting of the following entity types: (a) **Professor**, which has a unique professor ID and other attributes such as a full name (composed of a first name and a last name), a gender, a department, and an email address. (b) **Student**, which has a unique student ID and other attributes such as a full name (composed of a first name and a last name), a gender, an enrollment date (date enrolled in the university), a grade level (year of study), a department, and major(s). (c) **Course**, which has a unique course ID and other attributes such as a course name, a credit value, a homepage and some prerequisite course(s). As a course is offered in many semesters, we also have a weak entity type (d) **Course offering**. Each instance of it describes course that is offered in a semester, as well as the professor who teaches it, the students who enroll in it, and the classroom in which it is held.

Based on the above description, please answer the following questions about ER diagram of this database.

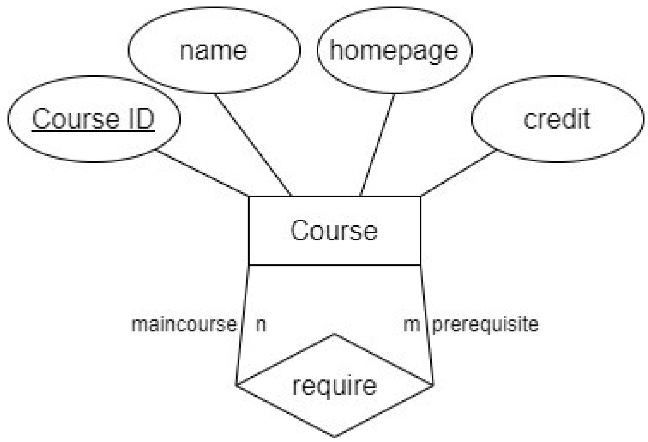
1. Draw the ER diagram for the entity type **Professor**. [3 points]
   Answer:



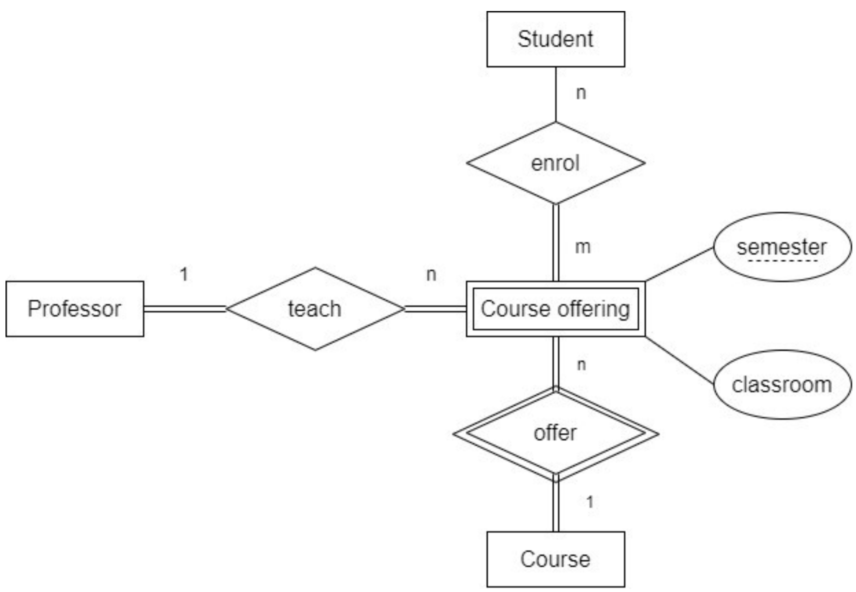
2. Draw the ER diagram for the entity type **Student**, assume that each student belongs to a single department but may have double majors. [3 points]
   Answer:



3. Draw the ER diagram for the entity type **Course** and its self-referencing relationship. Assume that each course can have zero or multiple prerequisite courses, and one course can be the prerequisite course of zero or many courses. [4 points]
   Answer:

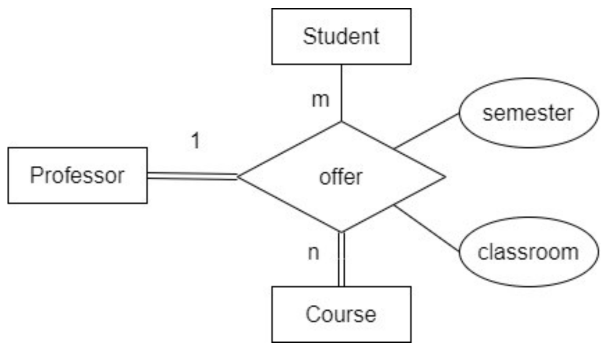


4. Suppose that in one semester, a course will be offered only once. This means that the combination of the course ID and the semester can uniquely identify one instance of a course offering. A course is taught by one professor in a semester but can be taught by different professors in different semesters. Each professor teaches one course in one semester and can teach different courses in different semesters. A student can enroll in zero or many courses in any semester, and a course is enrolled by many students every semester. A student is allowed to take the same course only once. Please draw the ER diagram for the weak entity '**Course Offering**' and all its associated relationships. (Hint: There's no need to include the attributes for Professor, Course, and Student in the diagram.) [5 points]
   Answer:

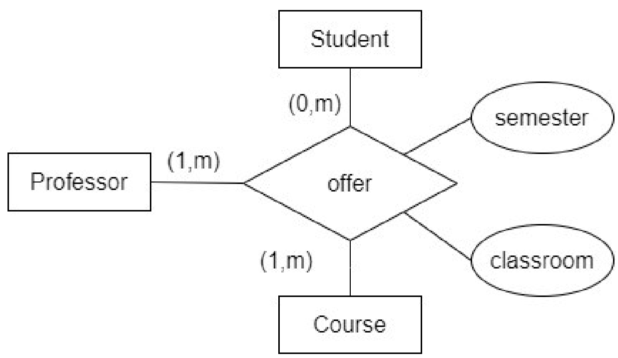


5. Based on the assumption in question 4, please convert the weak entity '**Course offering**' into a relationship among Professor, Course, and Student and draw its ER diagram. (Hint: no need to draw the attributes for Professor, Course, and Student.) [5 points]

   Answer:

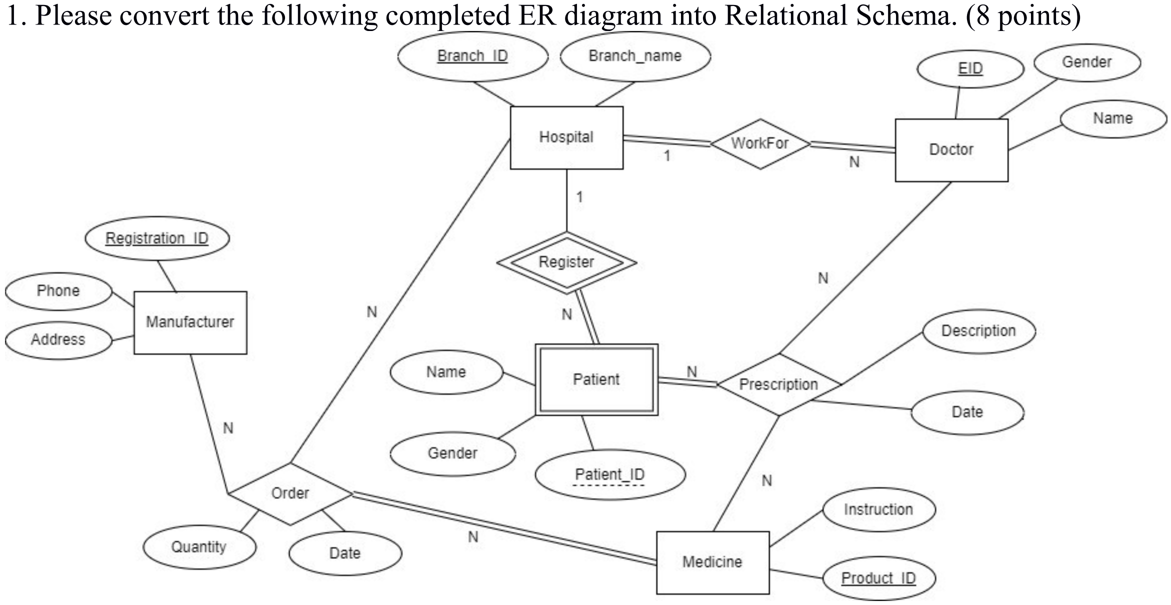


OR

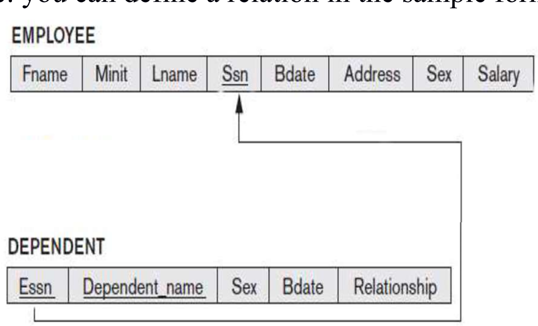


Problem TWO: Relational Model (20 points)

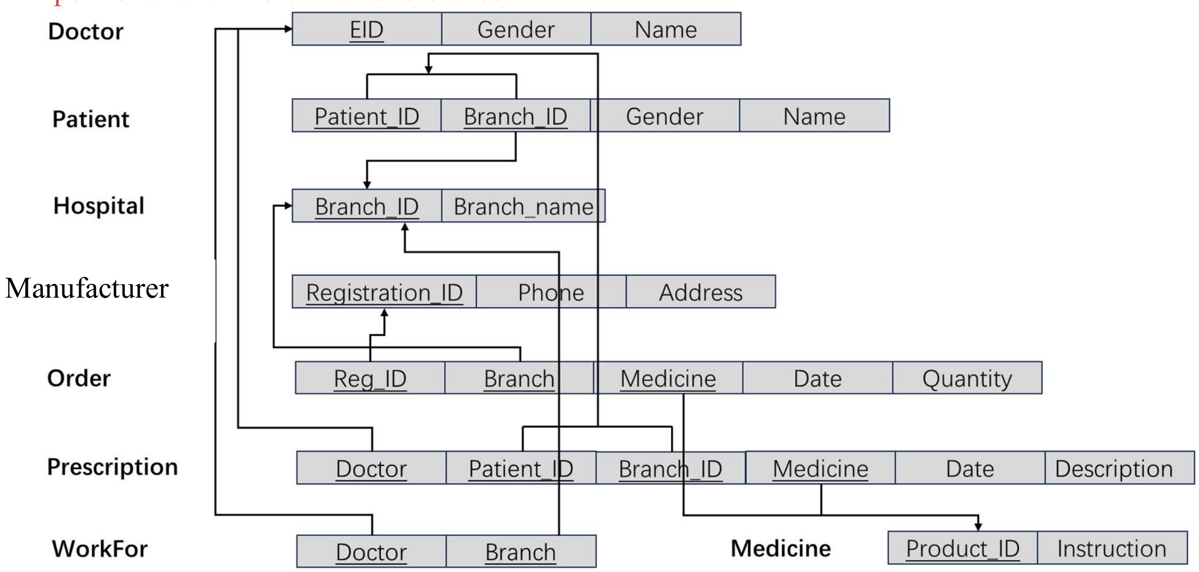
1. Please convert the following completed ER diagram into Relational Schema. (8 points)



Note: you can define a relation in the sample format below:



Answer:
1 points for each table and its reference



2. Assume the tables of the entities ‘Doctor’, ‘Hospital’ and ‘Medicine’ already exist. Create the tables of the **entity ‘Patient’** and the **relationship ‘Prescription’** and define all the primary keys and foreign keys with SQL statements. (Hint: you can define the datatype of attributes by yourself). [7 points]

Answer:

```
CREATE TABLE Patient
(
    Patient_ID    INT    NOT NULL,
    Branch_ID    INT    NOT NULL,
    Name    VARCHAR(20),
    Gender    CHAR(1),
    PRIMARY KEY(Patient_ID, Branch_ID),
    FOREIGN KEY (Branch_ID) REFERENCE Hospital (Branch_ID)
); (3 points)

CREATE TABLE Prescription
(
    Doctor    INT    NOT NULL,
    Patient_ID    INT    NOT NULL,
    Branch_ID INT, NOT NULL,
    Medicine    INT    NOT NULL,
    Date    DATE,
    Description    VARCHAR(500),
    PRIMARY KEY (Doctor, Patient, Medicine),
    FOREIGN KEY (Doctor) REFERENCES Doctor(EID),
    FOREIGN KEY (Patient_ID , Branch_ID) REFERENCES Patient(Patient_ID,
Branch_ID),
```

FOREIGN KEY (Medicine) REFERENCES Medicine(Product_ID)
); (4 points)

Problem Three: Integrity Constraints [20 points]
Suppose we have a relational database of E-commerce system which describe the orders of products made by customers. It contains three tables: Customer, Order and Product. The current state of the database is shown in the following tables.

Customer

User_id	User_name	Gender	Birth_date	Address
1	John Doe	Male	May. 5, 1990	123 Main St. Bridgeton
2	Emily Davis	Female	Jul. 27, 2006	18 Lancaster Ave. Plainview
3	Michael	Male	Aug. 13, 1990	123 Main St. Bridgeton
4	David Johnson	Male	Dec. 31, 1998	8907 Gonzales Ave. Ambler, PA
5	Luna	Female	Nov. 8, 1996	113 Old Lawrence Ave. Mishawaka, IN
6	John Doe	Male	Aug. 13, 1990	789 Oak St. Bridgeton
7	Sofia	Female	Aug. 13, 1995	381 Pearl Dr. Charlotte
8	David Johnson	Male	Apr. 30, 1990	37 Mill St. Bridgewater
9	Elizabeth	Female	Apr. 14, 1993	630 South Ave.

Order

Product_id	User_id	Quantity
1234	1	7
5566	5	48
21	7	99
4	3	56
643	2	1
25	3	10
1324	1	8
1234	6	1
1234	4	5

Product

Product_id	Brand_name	Product_name
1234	Apple	Cell phone
5566	Apple	Notebook
21	Samsung	Cell phone
4	Thinpad	Notebook
643	Huawei	Router
25	Huawei	Cell phone
1324	Thinkpad	Mouse

1. Analyze the primary keys of the provided relations and calculate the number of superkey(s) of each table. [6 points]

Answer:

For Customer table:
Primary key: User_id (1 point)
Superkey number: 2^4=16 (1 point)

For Order table:
Primary key: (User_id, Product_id) (1 point)
Superkey number: 2^1=2 (1 point)

For Product table:
Primary key: Product_id (1 point)
Superkey number: 2^2=4 (1 point)

2. Suppose all three tables are already created, write corresponding SQL statements to define all foreign keys. [2 points]

Answer:

ALTER TABLE Order ADD CONSTRAINT FK_order_user FOREIGN KEY (User_id) REFERENCES (Customer(User_id)); (1 point)

ALTER TABLE Order ADD CONSTRAINT FK_order_product FOREIGN KEY (Product_id) REFERENCES (Product(Product_id)); (1 point)

3. For 3.1 and 3.2 below, suppose each of the following Update operations is applied directly to the database. Discuss all integrity constraints violated by each operation if any, and the different ways of enforcing these constraints.

3.1) Delete tuple <1234, ‘Apple’, ‘Cell Phone’> from Product. [4 points]

Answer: Violates the referential integrity constraint. Because many existing tuples in Order referred to this tuple to be deleted. After deletion, these Order tuples will refer to non-existing product. (2 point)

We may enforce the constraint by: (i) rejecting the deletion operation, (1 point) or (ii) cascaded deletion, deleting all Order tuples which refer to the deleted product tuple as well. (1 point)

3.2) Insert <null, 'Tony', 'Male' , 2000, '73 Fifth Ave. Potomac'> into User. [4 points]

Answer:
Violates the entity integrity constraint because User_id of User is the key attribute which does not allow to be NULL. (1 point) Violates domain constraint because the domain of Birth_date should not be integer. (1 point)

We may enforce the constraint by: (i) rejecting the insertion, (1 point) or (ii) changing the value of User_id to a value that is not null and doesn't exist in the table User. Meanwhile changing Birth_date into char/date type. (1 point)

4. Given the following relation, list all the nontrivial functional dependencies satisfied in it. [4 points]

A	B	C
a1	b1	c2
a1	b2	c2
a2	b1	c1
a2	b2	c1

Answer:
A→C
C→A
AB→C
BC→A

Problem Four: Normalization [20 points]

1. Suppose we have a relation R with attributes A, B, C, D, E, F, G, H and the functional dependencies are: AD→BH, C→AF, B→EG. Please prove that FD: CD→E holds. [5 points]

Answer:

1. C→A (decomposition by C→AF) (1 point)

2. CD→AD (augmentation by 1) (1 point)
3. CD→BH (transitivity by AD→BH) (1 point)
4. CD→B (decomposition by 3) (1 point)
5. CD→EG (transitivity by B→EG) (1 point)
6. CD→E (decomposition by 5)

2. Let's consider the following relation R storing the information about airline reservations. R(ReservationNo, PassportNo, Birthplace, FlightNo, Origin, Destination, Lounge, Membership, Nationality).
It has following functional dependencies:
PassportNo → Birthplace
Birthplace → Nationality
FlightNo → {Origin, Destination}
{ReservationNo, PassportNo} → FlightNo
Membership → Lounge

2.1) Identify all the candidate keys in this table. [2 Points]
Answer: {PassportNo, ReservationNo, Membership}

2.2) Is the relation R in 2NF and why? If not, decompose it into **Four** tables which satisfy 2NF. [5 Points]
Answer: Not in 2NF, because there exists partial function dependency on primary keys, PassportNo → {Birthplace, Nationality}, {ReservationNo, PassportNo} → {FlightNo, Destination, Origin}, Membership → Lounge. (2 marks)

R1 (<u>PassportNo</u>, Birthplace, Nationality)
R2 (<u>ReservationNo</u>, <u>PassportNo</u>, FlightNo, Origin, Destination)
R3 (<u>Membership</u>, Lounge)
R4 (<u>PassportNo</u>, <u>ReservationNo</u>, <u>Membership</u>) (3 marks)

2.3) Does your decomposition in 2.2) satisfy 3NF and why? If not, normalize it into 3NF. [5 Points]

Answer: Not in 3NF, because there exists transitive function dependency on primary keys: PassportNo → Birthplace → Nationality, {ReservationNo, PassportNo} → FlightNo → {Origin, Destination} (2 marks)

R1A (<u>PassportNo</u>, Birthplace)
R1B (<u>Birthplace</u>, Nationality)
R2A (<u>ReservationNo</u>, <u>PassportNo</u>, FlightNo)
R2B (<u>FlightNo</u>, Origin, Destination)
R3 (<u>Membership</u>, Lounge)
R4 (<u>PassportNo</u>, <u>ReservationNo</u>, <u>Membership</u>) (3 marks)

2.4) Does your decomposition in 2.3) satisfy BCNF and why? If not, normalize it into BCNF. [3 Points]
Answer: Yes, it already satisfies BCNF. Because in each table, for each functional dependency, the left-hand side is a super key.

Problem FIVE: SQL [25 points]

Given the following relations about the information of course offerings in a university.

- Student (**StudentID**: integer, **Name**: string, **Age**: integer, **Gender**: string)
- Teacher (**TeacherID**: integer, **Name**: string, **Department**: string)
- Course (**CourseID**: integer, **Name**: string, **Semester**: char, **Department**: string, **TeacherID**: integer)
- Grade (**StudentID**: integer, **CourseID**: integer, **Score**: integer)

Suppose now we have a valid database state. Answer the following questions by completing missing parts of given SQL statement.

1. Query the ID, name, maximum, and minimum scores of courses in semester '2023 A' that have more than 100 students enrolled. [5 points]

   SELECT ____

   FROM Course, Grade

   WHERE ____

   ____

   ____;

   Answer:
   SELECT CourseID, Name, MAX(Score), MIN(Score)
   From Course, Grade
   WHERE Course.CourseID=Grade.CourseID AND Semester='2023 A'
   GROUP BY CourseID, Name
   HAVING Count(Distinct StudentID) > 100; //count (*) or count(student ID) are also correct

2. Create a view "**CourseTopScores**" that has the following three columns: **CourseID**, **CourseName**, **HighestScore**. The **HighestScore** means the historical highest score of each course. [5 points]

   CREATE____

   AS SELECT ______________________________________

   FROM ____

WHERE ____

____;

Answer:
CREATE VIEW CourseTopScores (CourseID, CourseName, HighestScore)
AS SELECT C.CourseID, C.Name, MAX(G.Score)
FROM Course AS C, Grade AS G
WHERE C.CourseID=G.CourseID
GROUP BY C.CourseID, C.Name;

3. Query the ID of students who have taken all courses offered by the 'CS' department. [5 points]

SELECT S.StudentID

FROM Student AS S

WHERE NOT EXISTS (

SELECT ____

FROM ____

WHERE ____

____

SELECT ____

FROM ____

WHERE ____

);

Answer:
```
SELECT S.StudentID
FROM Student AS S
WHERE NOT EXISTS (
        SELECT C.CourseID
        FROM Course AS C
        WHERE C.Department='CS'
        EXCEPT
        SELECT G.CourseID
        FROM Grade AS G
        WHERE G.StudentID=S.StudentID
);
```

4. Find the names of students who are classmates with the student with Student

ID=1234. [5 points]

SELECT ____

FROM Student AS S

WHERE EXSIST (

SELECT ____________________________________

FROM ______________________________________

WHERE ____________________________________

) AND StudentID<>1234;

Answer:

```
SELECT S.Name
FROM Student AS S
WHERE EXIST
(
    SELECT *
    FROM Grade AS G1, Grade AS G2
    WHERE S.StudentID=G1.StudentID AND G1.CourseID = G2.CourseID and
G1.Semester = G2. Semester AND G2.StudentID=1234
)
AND S. StudentID<>1234;
```

5. Query the names of teachers who have taught the most courses in semester ‘2022 B’. Sort the results in the descending order of the name of teachers. Assume that teacher’s name is unique. [5 points]

SELECT T.Name

FROM Teacher AS T, Course AS C

WHERE ______________________________________

GROUP BY ____________________________________

HAVING________________ = (

SELECT MAX(CourseCount)

FROM (

SELECT ______________________________

FROM _______________________________

WHERE ______________________________

GROUP BY ______________________________

)

)

____;

Answer:

```
SELECT T.Name
FROM Teacher AS T, Course AS C
WHERE T.TeacherID=C.TeacherID AND semester='2022 B'
GROUP BY T.Name
HAVING Count(*) = (
        SELECT MAX(CourseCount)
        FROM (
                SELECT COUNT(*) AS CourseCount
                FROM Course AS C2
                WHERE C2.semester='2022 B'
                GROUP BY C2.TeacherID
        )
)
ORDER BY T.Name DESC;
```