Final Exam

Classroom: Online

Assignment Points: 25 points

Name: Michael Ghattas
Student ID: 109200649

Exam rules:

- You MUST submit this final exam by **5/4/2023, 11:59 pm.** There will not be any extension or late submission.
- Submission: submit in **Canvas in pdf or word doc**.
- This is open book exam, and any kind of resource materials are allowed.
- Collaborations and consultations are NOT allowed. Do your own work.

Section 1: 3 points

1. During normalization process, we tend end up with more tables than we started?

A. True
B. False

2. You should remove partial dependency of non-Key attributes on the entire key in which normalization form?

A. First Normal Form
B. Second Normal
C. Third Normal Form
D. Fourth Normal Form

3. Which one of the following is used to create/define table and alter the of the structure in relational database system?

A. DML (Data Manipulation Language)
B. DDL (Data Definition Language)
C. Query
D. Relational Schema

4. A composite primary key may have a null value so long as other candidate keys remain not null.

A. True
B. False

5. Which of the following clause is used to filter the conditions in the SQL statement?

A. THEN
B. WHILE
C. WHERE
D. IF

6. Which SQL keyword is used to sort the result-set?

A. ORDER BY
B. ORDER
C. SORT BY
D. SORT

7. The BETWEEN operator is inclusive: begin and end values are included.

A. True
B. False

8. The **LIKE** operator is used to match a specified pattern in a SQL statement.

9. **Normalization** is the process to eliminate data redundancy and to make functional dependency of attributes.

10. The complete SQL statement may be complicated but it must include at least these two keywords: **SELECT**, **FROM**

11. Write a SQL statement to select the unique "LastName" column from the "Students" table?

```
SELECT DISTINCT LastName
FROM Students;
```

12. Write a SQL statement to select all the records from a table named "Characters" where the ‘FirstName’ starts from ‘A’ or ‘B’.

```
SELECT *
FROM Characters
WHERE FirstName LIKE ‘A%'
        OR FirstName LIKE 'B%';
```

13. Write a SQL statement to select all the records from a table “Customers” where the “LastName” ends with “c”.

```
SELECT *
FROM Customers
WHERE LastName LIKE '%c';
```

14. DELETE FROM supplier statement does the following:

A. Delete all rows in the supplier table, including the table structure
B. Delete all rows in the supplier table
C. No data is deleted without WHERE clause
D. Delete the first row in the supplier table

15. In GROUP BY SQL statements, you remove groups that do not meet the conditions with the use of:

A. WHERE clause
B. HAVING clause after GROUP BY
C. WHERE clause first, and then use the HAVING clause
D. HAVING clause first, and then use the WHERE clause

Section 2: 3 points

Normalize the following form into **3NF.** Only your 3rd NF will be graded.

University Department Sample Form

Dept Name	Hollywood
Phone 1	123-456-1111
Phone 2	123-456-1112
Phone 3	123-456-1113

CourseID	Course Name	InstructorID	Instructor Name	Gender
1	Mission Impossible	EH_123	Ethan Hunt	M
2	Star Wars	RS_456	Rey Skywalker	F
3	Iron Man	TS_789	Tony Stark	M
4	Black Widow	NR_012	Natasha Romanoff	F

This is the University departments sample form used by many departments.
If there is no concatenated key or many to many relationships, you may not need 2nd NF.
Do not carried away with unnecessary normalization.

Assumptions:
Department may have many phones and it offers many courses.
Each course belongs to only one department.
Each course is taught by only one instructor.

Hints:
List all attributes.
Identify repeating group of attributes.
Create proper entities and keys e.g., PK/FK to form relationship.
Make sure to create PK, if there is no obvious candidate key for PK.
Resolve transitive dependency, if any.

1st NF

Department
DeptID (PK)
DeptName

Phone
PhoneID (PK)
PhoneNumber

Course
CourseID (PK)
CourseName
InstructorID
InstructorName
Gender

2nd NF

Department
DeptID (PK)
DeptName

Phone
PhoneID (PK)
PhoneNumber

Course
CourseID (PK)
CourseName
InstructorID
InstructorFirstName
InstructorLastName
InstructorGender

3rd NF

Department
DeptID (PK)
DeptName

Phone
PhoneID (PK)
PhoneNumber
DeptID (FK)

Course
CourseID (PK)
CourseName
DeptID (FK)
InstructorID (FK)

Instructor
InstructorID
InstructorFirstName
InstructorLastName
InstructorGender

Section 3: 3 points

Create **ERD design** for following scenario:

Your data model design (ERD) should include relationships between tables with primary keys, foreign keys, optionality, and cardinality relationships. Captions are NOT required.

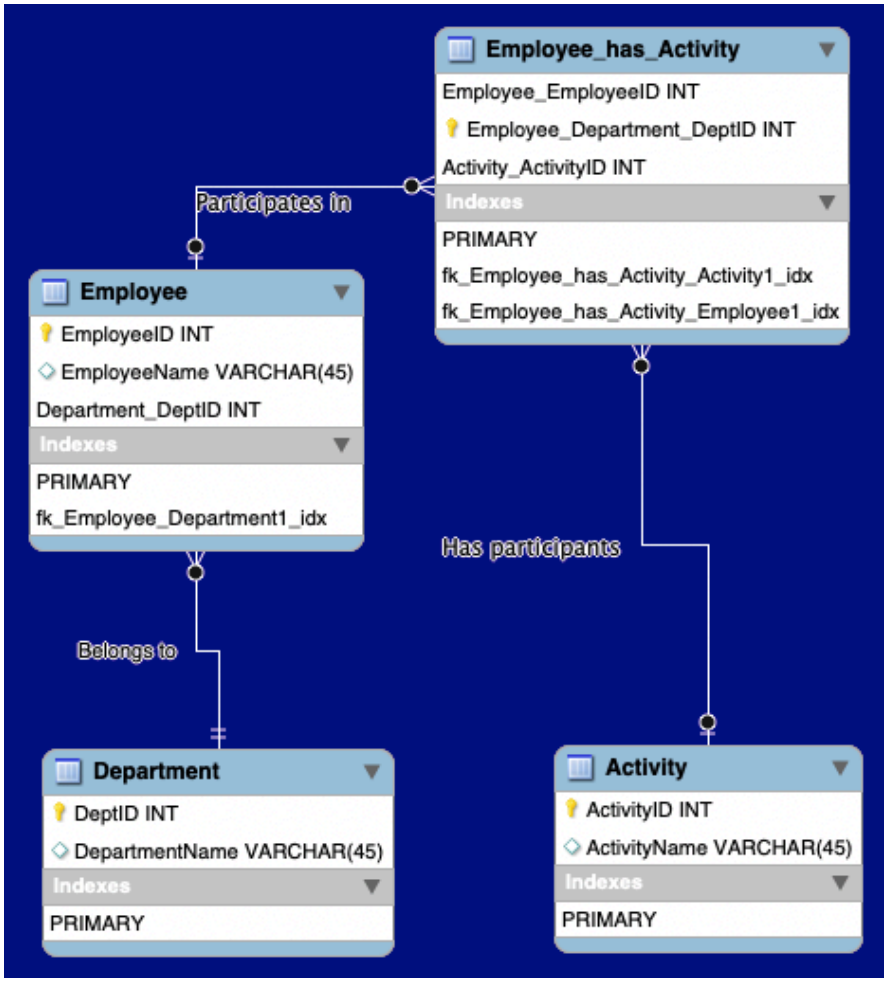
Scenario: There are 3 tables with 2 columns in each table:

Department (Dept ID, Department Name)

Employee (Employee ID, Employee Name)

Activity (Activity ID, Activity Name)

Each Employee must belong to ONLY ONE Department.
Department may have ZERO, ONE OR MORE Employees, i.e., Department may exist without any employee.
Each Employee may participate in ZERO, ONE OR MORE Activities
Each Activity may be performed by ZERO, ONE OR MORE Employees.



Section 4: 2 points

a. Create table **T1** with following columns and constraints.
Note: DO NOT use alter table, list all constraints while creating table.

C1 INT Primary key
C2 INT
C3 INT
C4 VARCHAR (40)

Constraints: C3 NON-ZERO, C2 greater than C3, C4 default value of 'HR'

```
CREATE TABLE T1 (
        C1 INT PRIMARY KEY,
        C2 INT CHECK (C2 > C3),
        C3 INT NOT NULL CHECK (C3 <> 0),
        C4 VARCHAR(40) DEFAULT 'HR'
);
```

b. Create table **T2** with following columns and Foreign Key.
Note: DO NOT use alter table, create FK while creating table.

C5 INT Primary key
C6 INT
FK on C6 column referencing to C1 column in table T1 above.

```
CREATE TABLE T2 (
        C5 INT PRIMARY KEY,
        C6 INT,
        CONSTRAINT FK_T2_T1_C6_C1 FOREIGN KEY (C6)
                REFERENCES T1 (C1)
);
```

Section 5: 7 points

All questions are based on below **Employee** table:

EmpId	ManagerId	Name	Department	Salary	City
1	0	Alex Smith	Admin	$90,000	Boulder
2	1	Amy Mars	Admin	$50,000	Longmont
3	1	Logan Mars	Admin	$70,000	Longmont
4	1	James Mont	Marketing	$55,000	
5	6	John Smith	Marketing	$60,000	Boulder
6	1	Lily Mars	Marketing	$95,000	
7	6	Ravi Grace	Database	$75,000	Longmont
8	6	Tara Frank	Database	$80,000	Longmont
9	6	Tom Ford	Database	$65,000	
10	6	William Cruze	Database	$85,000	Longmont

a. Write a SQL statement to find the Name and Salary who has **5th HIGHEST** Salary in the entire Employee table.

```
SELECT Name, Salary
FROM (
        SELECT Name, Salary,
                RANK() OVER (ORDER BY Salary DESC) AS salary_rank
        FROM Employee
        ) AS subquery
WHERE salary_rank = 5;
```

b. Write a SQL statement to find the Department and their count whose count is more than 3.

```
SELECT Department, COUNT(*) AS count
FROM Employee
GROUP BY Department
HAVING COUNT(*) > 3;
```

c. Write a SQL statement to show Name, Department and City.
However, if City is NULL, then display 'Broomfield' otherwise display City itself.

```
SELECT Name, Department,
        CASE
                WHEN City IS NULL THEN 'Broomfield'
                ELSE City
        END AS City
FROM Employee;
```

d. Write a SQL statement to find distinct employee Name who is also a manager.

```
SELECT DISTINCT e.Name
FROM Employee e
INNER JOIN Employee m ON e.EmpId = m.ManagerId;
```

e. Write a SQL statement to show Name, Department and Salary who earn MORE THAN the Average Salary in **THEIR department**. You must use sub-query.

```
SELECT Name, Department, Salary
FROM Employee e1
WHERE Salary > (
        SELECT AVG(Salary)
        FROM Employee e2
WHERE e1.Department = e2.Department
);
```

f. Write a SQL statement to show Name, Department, Salary and their Rank **WITHIN Department** from highest to lowest salary.
i.e., Salary rank must reset and re-rank start from 1 for EACH Department.

```
SELECT Name, Department, Salary,
     RANK() OVER (PARTITION BY Department ORDER BY Salary DESC) AS Salary_Rank
FROM Employee
ORDER BY Department, Salary DESC;
```

g. Write a SQL statement to find HIGHEST paying employee's Name and Salary from the entire Employee table. You must use sub-query.

```
SELECT Name, Salary
FROM Employee
WHERE Salary = (
        SELECT MAX(Salary)
        FROM Employee
);
```

Section 6: 5 points

All questions are based on below **Student** table:

Student

Attribute	**Datatype**
StudentID	Int
FirstName	Varchar(40)
LastName	Varchar(40)
DOB	Date
City	Varchar(40)
Country	Varchar(40)

a. Write a SQL to select all records (*) from student table but exclude TEST students. TEST Students define as studentid ends with 999 (e.g., 123999) OR (firstName = TEST_Student and lastname = TEST_Student).

```
SELECT *
FROM Student
WHERE StudentID NOT LIKE '%999'
        AND (FirstName <> 'TEST_Student' OR LastName <> 'TEST_Student');
```

b. Write a SQL to find the difference between total count of city and unique (distinct) count of city from Student table. Show the difference as 'difference'.

```
SELECT COUNT(City) - COUNT(DISTINCT City) AS difference
FROM Student;
```

c. Write a SQL to find longest and shortest length cities and their length from Student table. Your sql result should display city and city_length columns, sort by longest first and shortest second.

```
SELECT City, LENGTH(City) AS city_length
FROM Student
ORDER BY city_length DESC, city_length ASC
LIMIT 2;
```

d. Write a SQL to show unique (distinct) city that starts with a vowel (a,e,i.o,u).

```
SELECT DISTINCT City
FROM Student
WHERE City LIKE ‘a%'
        OR City LIKE ‘e%'
        OR City LIKE ‘i%'
        OR City LIKE ‘o%'
        OR City LIKE ‘u%';
```

e. Write a SQL to show students count by age_group from student table.
age_group: if students age < 18 is ‘minor’ else ‘adult’. Your SQL result should show age_group (minor/adult) and their count.

```
SELECT
        CASE
                WHEN YEAR(GETDATE()) - YEAR(DOB) < 18 THEN ‘minor'
                ELSE ‘adult'
        END AS age_group,
        COUNT(*) AS count
FROM Student
GROUP BY age_group;
```

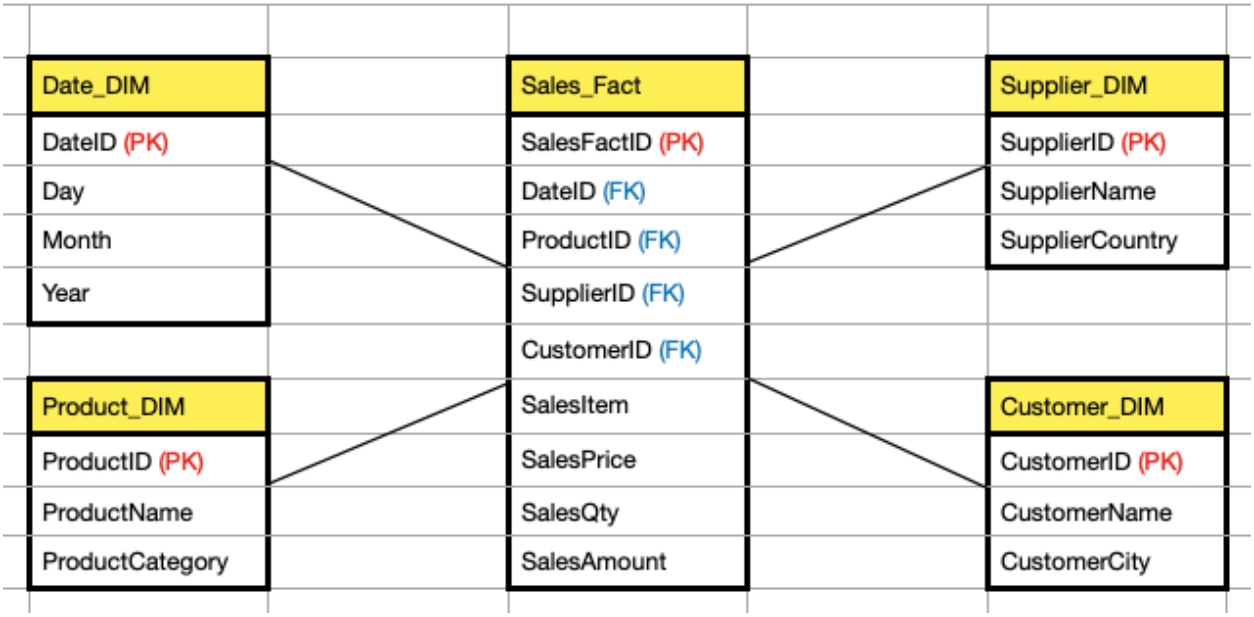
Section 7: 2 points

Create a Retail Sales Company **Data Warehouse design** using **STAR schema** from following info. Make sure to indicate proper _DIM and _Fact tables and their PKs/FKs. You need to join those tables using JUST straight lines (optionality and cardinality relationships are NOT required).

Date, Month, Year, SupplierName, SupplierCountry, ProductName, ProductCategory, CustomerName, CustomerCity, SalesItem, SalesPrice, SalesQty, SalesAmount

Note: You may use any tool or just handwritten to create STAR schema data warehouse design.

Retail Sales Company (STAR schema)



☺☺☺ The END ☺☺☺