DATABASE MID-TERM EXAM

**1/ Number Question Answer:**

A Primary Key uniquely identifies each record in a table, ensuring no duplicate values and enforcing entity integrity. In contrast, a Foreign Key is a field in one table that links to the Primary Key of another table, establishing a relationship between the two tables and enforcing referential integrity

**2/ Number Question Answer:**

A self join In SQL connects a table to itself, allowing comparison of rows within the same table. It is useful for hierarchical data, such as identifying employees who share the same manager by linking their IDs.

SELECT

E1.first\_name AS Employee\_First\_Name,

E1.manager\_id AS Manager\_ID

FROM

Employees e1

JOIN

Employees e2 ON e1.manager\_id = e2.employee\_id

ORDER BY

E1.manager\_id;

**3/ Number Question Answer:**

* **Create the Employees table**

CREATE TABLE Employees (

Employee\_Id INT PRIMARY KEY,

First\_Name VARCHAR(50) NOT NULL,

Last\_Name VARCHAR(50) NOT NULL,

Date\_of\_Birth DATE NOT NULL,

Department\_Id INT NOT NULL,

Salary DECIMAL(10, 2) NOT NULL

);

* **Create the Projects table**

CREATE TABLE Projects (

Project\_ID INT PRIMARY KEY,

Project\_Name VARCHAR(100) NOT NULL,

Start\_Date DATE NOT NULL,

End\_Date DATE NOT NULL,

Budget DECIMAL(15, 2) NOT NULL

);

* **Create the Employee\_Projects table to establish a many-to-many relationship**

CREATE TABLE Employee\_Projects (

Employee\_ID INT,

Project\_ID INT,

PRIMARY KEY (Employee\_ID, Project\_ID),

FOREIGN KEY (Employee\_ID) REFERENCES Employees(Employee\_Id),

FOREIGN KEY (Project\_ID) REFERENCES Projects(Project\_ID)

);

**4/ Number Question Answer:**

SELECT MAX(salary) AS Third\_Highest\_Salary

FROM employees

WHERE salary < (

SELECT MAX(salary)

FROM employees

WHERE salary < (

SELECT MAX(salary)

FROM employees

)

);

**5/ Number Question Answer:**

SELECT d.department\_name, COUNT(e.employee\_id) AS employee\_count

FROM departments d

LEFT JOIN employees e ON d.department\_id = e.department\_id

GROUP BY d.department\_name;

**6/ Number Question Answer:**

1. **INNER JOIN**

Description: Returns only the rows that have matching values in both tables.

**Example:**

SELECT e.employee\_id, e.first\_name, d.department\_name FROM employees e INNER JOIN departments d ON e.department\_id = d.department\_id;

**2. LEFT JOIN**

Description: Returns all rows from the left table (employees), and the matched rows from the right table (departments). If there is no match, NULL values are returned for columns from the right table.

**Example:**

SELECT e.employee\_id, e.first\_name, d.department\_name

FROM employees e

LEFT JOIN departments d ON e.department\_id = d.department\_id;

**3.RIGHT JOIN**

Description: Returns all rows from the right table (departments), and the matched rows from the left table (employees). If there is no match, NULL values are returned for columns from the left table.

**Example:**

SELECT e.employee\_id, e.first\_name, d.department\_name

FROM employees e

RIGHT JOIN departments d ON e.department\_id = d.department\_id;

**4.CROSS JOIN**

Description: Returns the Cartesian product of both tables, meaning every row from the first table is combined with every row from the second table.

**Example:**

SELECT e.employee\_id, e.first\_name, d.department\_name

FROM employees e

CROSS JOIN departments d;

**7/ Number Question Answer:**

A Common Table Expression (CTE) is a temporary result set in SQL that enhances query readability. It allows for organized calculations, such as identifying employees earning above their department’s average salary.

WITH department\_avg\_salary AS (

SELECT department\_id, AVG(salary) AS avg\_salary

FROM employees

GROUP BY department\_id

)

SELECT CONCAT(first\_name, ' ' ,last\_name) AS employee\_name

FROM employees e

JOIN department\_avg\_salary d ON e.department\_id = d.department\_id

WHERE e.salary > d.avg\_salary;

**8/ Number Question Answer:**

SELECT CONCAT(first\_name, ' ' ,last\_name) AS employee\_name FROM employees WHERE salary < (SELECT salary FROM employees WHERE first\_name = 'Steven' AND last\_name = 'King');

**9/ Number Question Answer:**

SELECT d.department\_name,

CONCAT(e.first\_name,' ' , e.last\_name) AS manager\_name

FROM departments d

JOIN employees e ON d.manager\_id = e.employee\_id;

**10/ Number Question Answer:**

SELECT DISTINCT l.city

FROM departments d

JOIN locations l ON d.location\_id = l.location\_id;