CompanyDB Staff Schema Documentation

Schema: Staff

Table: Projects

Description: Stores information about projects within the organization.

The following tables are helpful for tasking Projects' Company with its related Assignments for maintaining projects' progress.

Columns:

- **ProjectID:** `INT` (Primary Key) - Unique identifier for each project.
- **ProjectName:** `VARCHAR(50)` - Name of the project. Cannot be null.
- **StartDate:** `DATE` - Start date of the project. Cannot be null.
- **EndDate:** `DATE` - End date of the project. Can be null.
- **Budget:** `MONEY` - Budget allocated for the project. Must be a non-negative value.

Constraints:

- **Primary Key:** ProjectID
- **Check Constraint:** Budget must be greater than or equal to 0.

Table: Assignments

Description: Stores information about employee assignments for projects.

Columns:

- **AssignmentID:** `INT` (Primary Key) - Unique identifier for each assignment.
- **EmployeeID:** `INT` - Unique identifier for each employee. Cannot be null.
- **ProjectID:** `INT` - Unique identifier for the project. Cannot be null.
- **Role:** `VARCHAR(50)` - Role of the employee in the project.
- **AssignmentDate:** `DATE` - Date the assignment was made.

Constraints:

- **Primary Key:** AssignmentID
- **Foreign Key:** ProjectID references Projects(ProjectID)

Task Plan Queries (For Analysis [After Database Creation])

- Query 1: Employee Details with Function Manipulation

Run the following query:

```
SELECT UPPER(e.FirstName) AS FIRST_NAME,
    LOWER(e.LastName) AS last_name,
    LEN(Position) AS position_length,
    d.DepartmentName
FROM Staff.Employees AS e
JOIN Staff.Departments AS d
ON d.DepartmentID = e.DepartmentID;
```

- Query 2: Department Budget Summary

Run the following query:

```
SELECT d.DepartmentName,
    ROUND(SUM(e.Salary), -3) AS total_salary,
    COUNT(d.DepartmentID) AS NumberOfEmployees
FROM Staff.Employees AS e
JOIN Staff.Departments AS d
ON d.DepartmentID = e.DepartmentID
GROUP BY d.DepartmentName;
```

- Query 3: Project Assignments

Run the following query:

```
SELECT UPPER(p.ProjectName) AS PROJECTNAME,
    CONCAT(FirstName,' ', LastName) AS full_name,
    a.Role
FROM Staff.Employees AS e
JOIN Staff.Assignments AS a
ON a.EmployeeID = e.EmployeeID
JOIN Staff.Projects AS p
ON p.ProjectID = a.ProjectID;
```

- Query 4: Customer Order Analysis

Run the following query:

```
SELECT LOWER(c.CustomerName) AS customer_name,
      COUNT(o.OrderID) AS number_of_orders,
      SUM(o.TotalAmount) AS total_amount_spent
FROM Sales.Customers AS c
JOIN Sales.Orders AS o
ON c.CustomerID = o.CustomerID
GROUP BY LOWER(c.CustomerName);
```

- Query 5: Product Details Extraction

Run the following query:

```
SELECT LEFT(p.ProductName, 10) AS TruncatedProductName,
      LEFT(p.ProductName, 2) AS ProductCategory,
      SUM(od.Quantity) AS TotalQuantityOrdered
FROM Sales.Products AS p
JOIN OrderDetails AS od
ON p.ProductID = od.ProductID
GROUP BY LEFT(p.ProductName, 10),
        LEFT(p.ProductName, 2);
```

- Query 6: High Salary Employees in Specific Departments

Run the following query:

```
SELECT CONCAT(e.FirstName, ' ', e.LastName),
      e.Salary,
      d.DepartmentName
FROM Staff.Employees AS e
JOIN Staff.Departments AS d
ON d.DepartmentID = e.DepartmentID
WHERE e.Salary > (
  SELECT AVG(e2.Salary)
  FROM Staff.Employees AS e2
  WHERE e2.DepartmentID = e.DepartmentID);
```