### DAY ☀️: \*\*Task – Creating Tables in SQL\*\*

```

Created by :

Name: Dhiraj Kr.

Profession: Data Scientist & GenAI Developer

```

---

### ⚡ Task 1: Create a Basic Student Table

\*\*Question:\*\* Create a table `students` with columns for ID, Name, and Age.

\*\*Requirements:\*\*

- `id` **(integer,Primary Key)**

- `name` (text)

- `age` (integer)

**QUERY:**

**CREATE TABLE tasks.Students\_Table (**

**ID INT PRIMARY KEY,**

**Name VARCHAR(100),**

**Age INT**

**);**

---

### ⚡ Task 2: Add City and Gender Columns

\*\*Question:\*\* Modify the `students` table to also include `city` and `gender`.

\*\*Requirements:\*\*

- `city` (text)

- `gender` (text)

**QUERY:**

**ALTER TABLE tasks.Students\_Table**

**ADD COLUMN City VARCHAR(100),**

**ADD COLUMN Gender VARCHAR(10);**

---

### ⚡ Task 3: Create a Departments Table

\*\*Question:\*\* Create a `departments` table with department ID and name.

\*\*Requirements:\*\*

- `dept\_id` (integer, primary key)

- `dept\_name` (text)

**Query : CREATE TABLE tasks.Department\_Table (**

**Dept\_id INT PRIMARY KEY,**

**Dept\_Name VARCHAR(20)**

**);**

---

### ⚡ Task 4: Create a Courses Table

\*\*Question:\*\* Create a `courses` table linked to `departments`.

\*\*Requirements:\*\*

- `course\_id` (integer, primary key)

- `course\_name` (text)

- `dept\_id` (foreign key)

**Query: CREATE TABLE tasks.Course\_Table (**

**Course\_ID INT PRIMARY KEY,**

**Course\_Name VARCHAR(100),**

**Dept\_id INT,**

**FOREIGN KEY(Dept\_id) REFERENCES tasks.department\_table(Dept\_id)**

**);**

---

### ⚡ Task 5: Create an Enrollments Table

\*\*Question:\*\* Create an `enrollments` table that links students with courses and stores their marks.

\*\*Requirements:\*\*

- `enroll\_id` (integer, primary key)

- `student\_id` (foreign key)

- `course\_id` (foreign key)

- `marks` (integer)

**QUERY : CREATE TABLE tasks.Enrollments\_Table (**

**Enroll\_ID INT PRIMARY KEY,**

**Student\_ID INT,**

**Course\_ID INT,**

**Marks INT,**

**FOREIGN KEY(Student\_ID) REFERENCES tasks.students\_table(ID),**

**FOREIGN KEY(Course\_ID) REFERENCES tasks.course\_table(Course\_ID)**

**);**

---

### ⚡ Task 6: Create a Table with Default Value

\*\*Question:\*\* Create a `fees` table with default value for `status` column.

\*\*Requirements:\*\*

- `fee\_id` (integer)

- `student\_id` (integer)

- `amount` (integer)

- `status` (default: 'Pending')

**Query : CREATE TABLE tasks.Fee\_Table (**

**Fee\_ID INT,**

**Student\_ID INT,**

**Amount INT,**

**Status VARCHAR(50) DEFAULT 'Pending'**

**);**

---

### ⚡ Task 7: Create a Table with NOT NULL Constraint

\*\*Question:\*\* Create a `teachers` table where `name` and `subject` cannot be NULL.

\*\*Requirements:\*\*

- `teacher\_id` (integer)

- `name` (text, not null)

- `subject` (text, not null)

**Query : CREATE TABLE tasks.teacher\_table(**

**TeacherID INT,**

**Name VARCHAR(100) NOT NULL,**

**Subject VARCHAR(50) NOT NULL**

**);**

---

### ⚡ Task 8: Create a Table with UNIQUE Constraint

\*\*Question:\*\* Create a `library` table where book title must be unique.

\*\*Requirements:\*\*

- `book\_id` (integer)

- `title` (text, unique)

- `author` (text)

**Query : CREATE TABLE tasks.library\_table(**

**BookID INT PRIMARY KEY,**

**Title VARCHAR(100) UNIQUE,**

**Author VARCHAR(100)**

**);**

---

### ⚡ Task 9: Create a Table with CHECK Constraint

\*\*Question:\*\* Create a `hostel` table where `room\_capacity` must be greater than 0.

\*\*Requirements:\*\*

- `hostel\_id` (integer)

- `hostel\_name` (text)

- `room\_capacity` (integer, CHECK > 0)

**Query : CREATE TABLE tasks.hostel\_table (**

**HostelID INT PRIMARY KEY,**

**Room\_Capacity INT CHECK (Room\_Capacity> 0)**

**);**

---

### ⚡ Task 10: Create Table Using Multiple Data Types

\*\*Question:\*\* Create a `staff` table using multiple data types.

\*\*Requirements:\*\*

- `staff\_id` (integer)

- `name` (text)

- `dob` (date)

- `salary` (float)

**Query : CREATE TABLE tasks.staff\_table (**

**StaffID INT PRIMARY KEY,**

**Name VARCHAR(100) NOT NULL,**

**DOB DATE,**

**Salary FLOAT**

**);**

---

\*\*End of Document\*\* ✨