Problem 1

> [Product Sales Analysis I - LeetCode]

Hint

Use square brackets `[ ]` for column names that are SQL keywords

✧ **Example:** `year` (SQL keyword) ---> `[year]`

Problem 2

> [Customer Who Visited but Did Not Make Any Transactions - LeetCode]

Hint

To check for missing values in SQL, use `IS NULL` instead of `= NULL`

✧ **Wrong:** `something = NULL`
✧ **Correct:** `something IS NULL`

Problem 3

> [Replace Employee ID With The Unique Identifier - LeetCode]

Hint

If you get a **TLE**, try **placing columns from the larger or primary table first** in the SELECT statement when using LEFT JOIN
This often improves performance without changing the query logic

---

Problem 4

> [Rising Temperature - LeetCode]

Hint

> Use **DATE functions** to work with dates in SQL:

✧ `DATEADD(unit, value, date)` → add or subtract time
  ✧ Example: `DATEADD(month, -1, '2025-06-03')` → '2025-05-03'
✧ `DATEDIFF(unit, startDate, endDate)` → difference between dates
  ✧ Example: `DATEDIFF(day, '2025-01-01', '2025-01-03')` → 2
  These help you compare or calculate dates easily.

Problem 5 – Employee Department Match with NULL Handling

> Find all employees and their department names. If an employee is not assigned to any department, show "Unassigned" instead of NULL

Table: **Employees**

COLUMN NAME	TYPE
emp_id	int
emp_name	varchar
dept_id	int

Table: **Departments**

COLUMN NAME	TYPE
dept_id	int
dept_name	varchar

Example Input: *Employees Table*

EMP_ID	EMP_NAME	DEPT_ID
1	Alice	101
2	Bob	NULL
3	Charlie	102
4	Diana	NULL

Departments Table

DEPT_ID	DEPT_NAME
101	Sales
102	Marketing
103	IT

Expected Output:

EMP_NAME	DEPT_NAME
Alice	Sales
Bob	Unassigned
Charlie	Marketing
Diana	Unassigned

Problem 6 – Product Search with Pattern Matching

> Find all products whose names contain "Phone" and match them with their suppliers. Include products even if they don't have a supplier yet

Table: **Products**

COLUMN NAME	TYPE
product_id	int
product_name	varchar
supplier_id	int

Table: **Suppliers**

COLUMN NAME	TYPE
supplier_id	int
supplier_name	varchar
country	varchar

Example Input: *Products Table*

PRODUCT_ID	PRODUCT_NAME	SUPPLIER_ID
1	iPhone 14	201
2	Laptop Pro	202
3	Samsung Phone	NULL
4	Headphones	201
5	Android Phone X	203

Suppliers Table

SUPPLIER_ID	SUPPLIER_NAME	COUNTRY
201	TechCorp	USA
202	CompuWorld	Canada
203	MobileTech	China

Expected Output:

PRODUCT_NAME	SUPPLIER_NAME
iPhone 14	TechCorp
Samsung Phone	NULL
Android Phone X	MobileTech

Problem 7 – Complete Customer Order Report

> Generate a report showing all customers and all orders, including customers who haven't ordered anything and orders without customer records. Display customer `full_name` as "FirstName LastName"

Table: **Customers**

COLUMN NAME	TYPE
customer_id	int
first_name	varchar
last_name	varchar
email	varchar

Table: **Orders**

COLUMN NAME	TYPE
order_id	int
customer_id	int
order_date	date
amount	decimal

Example Input:

Customers Table

CUSTOMER_ID	FIRST_NAME	LAST_NAME	EMAIL
1	John	Doe	john@email.com
2	Jane	Smith	jane@email.com
3	Mike	Johnson	mike@email.com

Orders Table

ORDER_ID	CUSTOMER_ID	ORDER_DATE	AMOUNT
101	1	2024-01-15	250.00
102	1	2024-02-20	180.50
103	999	2024-03-10	99.99
104	2	2024-03-15	320.00

Expected Output:

FULL_NAME	ORDER_ID	AMOUNT
John Doe	101	250.00
John Doe	102	180.50
Jane Smith	104	320.00
Mike Johnson	NULL	NULL
NULL	103	99.99

Problem 8 – Library Book Management Normalization

> Normalize a library database that tracks books, authors, and publishers. The table violates multiple normal forms and needs to be progressively normalized from 0NF to 3NF.

Table: `LibraryBooks` (0NF - Denormalized)

COLUMN NAME	TYPE
book_id	int
book_title	varchar
authors	varchar
publisher_id	int
publisher_name	varchar
publisher_city	varchar
genres	varchar

Example Input:

LibraryBooks (0NF)

BOOK_ID	BOOK_TITLE	AUTHORS	PUBLISHER_ID	PUBLISHER_NAME	PUBLISHER_CITY	GENRES
1	Database Systems	Silberschatz, Korth	101	McGraw Hill	New York	Computer, Education
2	Clean Code	Robert Martin	102	Prentice Hall	Boston	Computer
3	Design Patterns	Gang of Four	101	McGraw Hill	New York	Computer, Reference
4	The Pragmatic Programmer	Hunt, Thomas	102	Prentice Hall	Boston	Computer

Problem 9 – Student Course Registration Normalization

> Normalize a university database that tracks student enrollments, courses, and instructors. Progress through 1NF, 2NF, and 3NF to eliminate redundancy and anomalies.

Table: `StudentEnrollments` (0NF - Denormalized)

COLUMN NAME	TYPE
student_id	int
student_name	varchar
student_email	varchar
courses	varchar
instructors	varchar
instructor_dept	varchar
dept_building	varchar

Example Input:

StudentEnrollments (0NF)

STUDENT_ID	STUDENT_NAME	STUDENT_EMAIL	COURSES	INSTRUCTORS	INSTRUCTOR_DEPT	DEPT_BUILDING
1001	Alice Brown	alice@uni.edu	CS101, CS102	Dr. Smith, Dr. Lee	Computer Science, Computer Science	Tech Hall, Tech Hall
1002	Bob Chen	bob@uni.edu	MATH201	Dr. Johnson	Mathematics	Science Center
1003	Carol Davis	carol@uni.edu	CS101, MATH201	Dr. Smith, Dr. Johnson	Computer Science, Mathematics	Tech Hall, Science Center

Problem 10 – Hospital Patient Records Normalization

> Normalize a hospital database tracking patient appointments with doctors. The denormalized table contains patient info, doctor info, and appointment details with multiple violations.

Table: `PatientAppointments` (0NF - Denormalized)

COLUMN NAME	TYPE
appointment_id	int
patient_name	varchar
patient_phones	varchar
doctor_name	varchar
doctor_id	int
specialization	varchar
clinic_name	varchar
appointment_dates	varchar

Example Input:

PatientAppointments (0NF)

APPOINTMENT_ID	PATIENT_NAME	PATIENT_PHONES	DOCTOR_NAME	DOCTOR_ID	SPECIALIZATION	CLINIC_NAME	APPOINTMEI
1	John Smith	555-0101, 555-0102	Dr. Adams	201	Cardiology	Heart Care Center	2024-01-15, 2
2	Mary Johnson	555-0201	Dr. Brown	202	Dermatology	Skin Health Clinic	2024-01-18
3	John Smith	555-0101, 555-0102	Dr. Brown	202	Dermatology	Skin Health Clinic	2024-02-10