**C-406** DAY-1

**Foundations of SRE -RDBMS and SQL**

**Priyansh Kakani** 10-02-2025

* Today I have learned about the SQL queries and implemented through the MySql.
* Got a different view about table creation through a cupboard example which was given by the trainer. helped to identify how the SQL works in searching.
* To create a DATABASE and use it
  + CREATE DATABASE SCHOOL;
    - This creates a database SCHOOL.
  + USE SCHOOL;
    - This uses the specific database SCHOOL.
* To create a table under the DATABASE SCHOOL

CREATE TABLE STUDENT (

ID INT,

NAME VARCHAR2(20),

STREAM VARCHAR2(20),

PRIMARY KEY(ID)

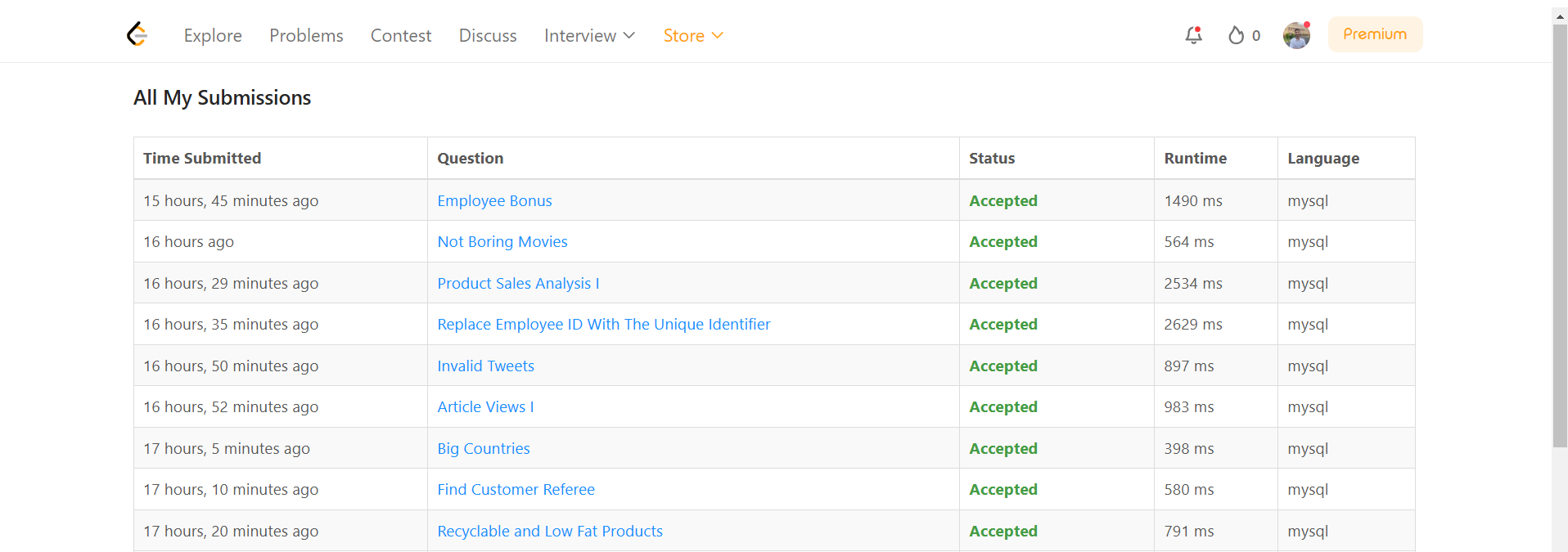
);

* In most of the cases DDL commands are faster than the DML commands
  + For e.g. TRUNCATE is faster than the DELETE just takes 0.000 sec to be executed.
* Used RENAME for renaming the table by 2 ways.
  + RENAME TABLE STUDENT TO NEW\_STUDENT;
  + ALTER TABLE STUDENT RENAME TO NEW\_STUDENT;
* Learned how the deep copies and shallow copies are created in the SQL.
  + 2 Ways for copying the old student relation data to new student relation.
  + For the Deep copy we use
    - CREATE TABLE OLD\_STUDENT SELECT \* FROM OLD\_STUDENT;
  + For the Shallow copy we use
    - CREATE TABLE NEW\_STUDENT LIKE OLD\_STUDENT;
      * It just creates the schema only no data is copied.
* To get a view of the table as per requirements we use SELECT.
  + SELECT \* FROM STUDENT;
    - This query will get all the columns of the STUDENT table.
* By default if we do sorting of table rows
  + it is ASCENDING ALWAYS just mention ORDER BY COLUMN\_NAME.
  + to get descending order use ORDER BY COLUMN\_NAME DESC.
* To get the description about the table e.g. columns, datatypes.
  + DESC STUDENT;
* To create a temporary table
  + CREATE TEMPORARY TABLE TEMP\_STUDENT(

… COLUMN AND DATATYPE DECLARATION

)

* Whenever we use HAVING It is mandatory to use GROUP BY otherwise it will not work. But when using GROUP BY it is optional to use HAVING.
* The primary key will be the first default key while retrieving the data from the table
* The data is aggregated by how it is spread across on the memory disk .if we specify the GROUP BY . then it will be of specific order.
* When we use GROUP BY so there are multiple table are created with the UNIQUE COLUMN values and have multiple related rows. And after that result are filtered out.
* Using the HAVING with GROUP BY is just like using the WHERE with FROM for the filter of the results.
* Used the wildcard for the searching patterns
  + e.g. SELECT NAME FROM STUDENT LIKE ‘P%’
    - it will print the student name starts with the P.
* Learned the aggregation functions such as COUNT,AVG,MIN,MAX with the GROUP BY and having
* Done today 9 SQL queries today on leetcode basics.



**C-406** DAY-2

**Foundations of SRE -RDBMS and SQL**

**Priyansh Kakani** 11-02-2025

* Today we have learned how to set the permission to the users for using the database as the different access is given to the different users so we provide a number that is decimal equivalent by the formula 2^n where n is the 1 present in the binary representation of the number.
* Sample program to demonstrate how to create permission table and insert data into it in decimal format.
  + CREATE TABLE permissions (  
        user\_id INT PRIMARY KEY,  
        username VARCHAR(50),  
        permission\_flags INT  -- Will store permission bits  
    );  
      
    -- Insert sample data  
    INSERT INTO permissions (user\_id, username, permission\_flags) VALUES  
    (1, 'admin', 7),     -- Binary: 111 (Read: 1, Write: 1, Execute: 1)  
    (2, 'developer', 6), -- Binary: 110 (Read: 1, Write: 1, Execute: 0)  
    (3, 'viewer', 4),    -- Binary: 100 (Read: 1, Write: 0, Execute: 0)  
    (4, 'guest', 1);     -- Binary: 001 (Read: 0, Write: 0, Execute: 1)
  + To give permission to user about execute we can change the values as
    - update permissions   
      set permission\_flags =permission\_flags ^ 1  
      where (permission\_flags & 1)=0;
* to give permission to user about write we can set values by
  + update permissions  
    set permission\_flags = permission\_flags | 2  
    where (permission\_flags & 2) =0
* to give permission to user about read we can set values by
  + select   
    username,  
    permission\_flags & 4 as has\_read\_permission,  
    case  
       when permission\_flags & 4 > 0 then 'Yes'  
       else 'No'  
    end as can\_read  
    from permissions;
* SET SQL\_SAFE\_UPDATES=0.
  + The error occurs because MySQL's **safe update mode** requires a key column in the WHERE clause to prevent accidental updates. Without it, MySQL blocks the update to ensure data safety.
* We can also do manipulation using bits and it applications are also the multiplication and division.
  + If we shift 1 bit to right then the number will be divided by 1.
  + If we shift 1 bit to left then the number will be multiplied by 2.
* Bit shifting operations example  
  CREATE TABLE bit\_shift\_demo (  
      id INT PRIMARY KEY,  
      value INT  
  );  
  INSERT INTO bit\_shift\_demo (id, value) VALUES  
  (1, 8),  -- Binary: 1000  
  (2, 12),  -- Binary: 1100  
  (3, 16);  -- Binary: 10000
* example to left shift the values will be double and 4 times

select id,value,  
value << 1 as left\_shft\_1,  
value << 2 as left\_shift\_2  
from bit\_shift\_demo;

* UNION operator is used to combine the results of two or more SELECT queries into a single result set. It removes duplicate rows from the result set by default.
  + SELECT column1, column2, ... FROM table1 WHERE condition1 UNION SELECT column1, column2, ... FROM table2 WHERE condition2;
* In UNION ALL it contains all rows even they are same i.e. redundant rows will be there in result.
* UNION ALL is faster than the UNION as it doesn’t checks for the repeated data into the result.
* SELECT 1 FROM CUSTOMERS;
  + This query selects 1 if there is a valid row so if we are applying into customers if customers has 5 rows so it will print 5 1’s in column each 1 denoting to each line used to check the no of rows present in table.
  + In practical scenarios if we want to count the no of rows indirectly then we can use this statement with where clause and a aggregate function e.g. COUNT so it will count no of rows.
* ANY keyword in SQL is typically used with a comparison operator (like =, >, <, etc.) to compare a value to any value in a set of values returned by a subquery. It allows you to compare a value with multiple values, and if any of those values meet the condition, the result is true.
  + SELECT column\_name FROM table\_name

WHERE column\_name operator ANY (subquery);

* + E.g. SELECT sale\_id, amount, salesperson FROM sales

WHERE amount = ANY (SELECT amount FROM sales WHERE salesperson = 'Bob');

This will select all amount if the salesperson is Bob.

* Sometime we use left join which is less expensive in terms of time than the inner join.
  + E.g. when creating a same column in the table representing the manager ‘s manager in the same table w/o using the INNER JOIN.
* DENSE\_RANK is a window function that assigns a unique rank to each distinct value in a result set, without leaving gaps in the rank values when there are ties.
* The PARTITION BY keyword in SQL is used in window functions to divide the result set into partitions (or groups) before performing the function.
  + E.g. we want the sales comparision between the salespersons and their monthly sales.
* the OVER keyword is used to define the windowing for the function, specifying how the data should be ordered or partitioned.
* RANK function is not good when there are 2 values same then it will give same rank to all 3 of them and continue to next with previous rank+2 to next which is not accurate when it comes to ranking. Example-

| **order\_id** | **sale\_amount** | **rank** |
| --- | --- | --- |
| 1 | 300 | 1 |
| 2 | 200 | 2 |
| 3 | 200 | 2 |
| 4 | 150 | 4 |
| 5 | 100 | 5 |

* LAG is a window function in SQL that allows you to access data from a previous row in the result set without the need for self-joins.
  + Its syntax is LAG(expression, offset, default\_value) OVER (PARTITION BY partition\_column ORDER BY order\_column).
  + E.g. if amazon has 5 products and we want to compare the products revenue according to the month. Then the LAG and DENSE\_RANK is used.