What Is SQL's Order of Execution?

FROM/JOIN. The FROM and/or JOIN clauses are executed first to determine the data of interest.

WHERE. The WHERE clause is executed to filter out records that do not meet the constraints.

GROUP BY. ...

HAVING. ...

SELECT. ...

ORDER BY. ...

LIMIT/OFFSET.

175

Combine Two Tables

SELECT

P.first\_name,P.Last\_name,A.city,A.State

FROM

Person P INNER JOIN

Address A

ON P.personId = A.PersonId

------------------------------------------\

176

Second Highest Salary

SELECT

distinct Salary

From Employee

order by Salary desc

limit 1 offset 1

// offset will skip the first row

or

Second Highest Salary

SELECT

distinct Salary

From Employee

order by Salary desc

limit 1,1 first 1: no of rows to skip

seconf 1: no. of rows to display

2nd approch

select salary, dense rank(order by salary desc) as r

from employee where r = 2

The above one is wrong bcz

you cannot reference an alias (r) in the WHERE clause in the same query.

This is because the logical order of query execution dictates that the

WHERE clause is evaluated before the SELECT clause

select salary from(

SELECT

Salary,

dense\_rank()

over(order by Salary desc) as r

from Employee)as T1

where r = 2;

-----------------------------------\

178

Rank Scores

select

scores,

dense\_rank()

over(order by scores desc) as r

from table

-----------------------------------\

180

Write a SQL query to find all numbers that appear at least three times

consecutively.

SELECT DISTINCT num

FROM (

SELECT num,

LAG(num) OVER (ORDER BY id) AS prev\_num,

LEAD(num) OVER (ORDER BY id) AS next\_num

FROM YourTable

) AS subquery

WHERE num = prev\_num AND num = next\_num;

2nd approch :

SELECT DISTINCT num

FROM numbers

WHERE num = LAG(num) OVER (ORDER BY id)

AND num = LEAD(num) OVER (ORDER BY id);

181

Employees Earning More Than Their Managers

select e.Name as Employee

from Employee as e inner join Employee as m on e.ManagerId = m.id

where e.Salary > m.Salary;

182

Write a SQL query to find all duplicate emails in a table named Person.

select email,count(email)

from person

group by email

having count(email) >1

------------------------------------\

183

Customers Who Never Order

select name from Customers

where id not in (select CustomerId

from Orders);

or

select C.name from Customers C

left join orders o on c.id = o.customer\_Id

where o.order\_ID is null;

---------------------------------\

197

Given a Weather table, write a SQL query to find all dates' Ids with higher

temperature compared to its previous (yesterday's) dates.

select w1.id

from weather w1 , weather w2

where datediff(w1.recordDate,w2.RecordDate)=1

and w1.Temperature> w2.Temperature

find all pairs of employees from the same department

SELECT e1.employee\_name AS employee1, e2.employee\_name AS employee2, e1.department

FROM employees e1

INNER JOIN employees e2 ON e1.department = e2.department

WHERE e1.employee\_id <> e2.employee\_id;

Q Write a SQL query to find the

names of employees who have not been assigned to any project.

select name from employees

where id not in (select emp\_id from project)

2nd approch ( using join is good computaionally use less power)

select E.name from Employee E

Left JOIN Project P ON

E.ID = P.Emp\_id

where P.Emp\_id is NUll

-------------------------------------------------------/

In the Student table, the marks column contains a list of

values separated by commas.How can you determine the number

of values in this comma-separated list?

StudentID StudentName Marks

1 Alice 90,85,78,92

select studentId , StudentName , Marks,

LENGTH(Marks)-LENGTH(REPLACE(Marks,',','')) +1 as Count

from Students

---------------------------------------------------------------------------

Q Find out even emp ids in table

select emp\_id from employee

where MOD(emp\_id,2) == 0

Q Find out odd emp ids in table

select emp\_id from employee

where MOD(emp\_id,2) == 1

Q Find the last record of the table

select \* from emp

where empid = (select max(empid) from emp)

or

select \* from emp order by empid desc limit 1

Q list of emp working in same department ?

SELECT e1.EmployeeName AS Employee1, e2.EmployeeName AS Employee2, e1.DepartmentID AS Department

FROM Employees e1

JOIN Employees e2 ON e1.DepartmentID = e2.DepartmentID

WHERE e1.EmployeeID <> e2.EmployeeID;

================Strings=====================

-- concat

select concat(first\_name," ",last\_name) from customers1;

-- Lower case

select lcase(first\_name) from customers1;

-- or

select lower(first\_name) from customers1;

-- uper case

select upper(first\_name) from customers1;

-- length

select length("Arun");

-- LOCATE(): This function is used to find the nth position of the given word in a string.

select locate("arun","The boy arun is good arun",1);

-- return the index no. of 1st letter of the string (9 in this case)

-- 1 means check from 1st string

select locate("arun","The boy arun is good arun",10);

-- here i used 10 so that it start seraching after first occurence of arun

-- so that it can locate the second occurence

-- ltrim , rtrim

select LTRIM(' geeks');

-- POSITION(): This function is used to find position of the

-- first occurrence of the given alphabet.

SELECT POSITION('e' IN 'geeksforgeeks');

-- REPEAT(): This function is used to write the given string again

-- and again till the number of times mentioned.

SELECT REPEAT('geeks', 2);

-- STRCMP():

-- This function is used to compare 2 strings.

-- If string1 and string2 are the same, the STRCMP function will return 0.

-- If string1 is smaller than string2, the STRCMP function will return -1.

-- If string1 is larger than string2, the STRCMP function will return 1.

SELECT STRCMP('google.com', 'geeksforgeeks.com');

-- SUBSTR():

-- This function is used to find a sub string from the a string from the given position.

select SUBSTR('geeksforgeeks', 1, 5);

-- Trim

select Trim(" Arun ");

-- DATE

SELECT CURRENT\_TIMESTAMP;

CREATE TABLE Orders\_\_ (

OrderID INT NOT NULL AUTO\_INCREMENT,

OrderNumber VARCHAR(255) NOT NULL,

OrderedAt DATETIME NOT NULL,

CustomerID INT NOT NULL,

PRIMARY KEY (OrderID)

);

INSERT INTO Orders\_\_ (OrderNumber, OrderedAt, CustomerID)

VALUES ('1234567890', '2023-09-15 00:00:00', 12345),

('9876543210', '2023-09-14 23:59:59', 98765),

('0123456789', '2023-09-14 23:59:58', 01234);

select \* from Orders\_\_;

-- DATE(): This function extracts the date part from a datetime expression. For example,

-- the following query will extract the date from the OrderDate column:

-- if the col habe date and time bith then it extract date only

select date(OrderedAt) from orders\_\_;

-- MONTH()

select extract(month from OrderedAt) from orders\_\_;

-- year()

select extract(year from OrderedAt) from orders\_\_;

-- DAY

select extract(day from OrderedAt) from orders\_\_;

-- weekday

select extract(weekday from OrderedAt) from orders\_\_;

-- Hour minute and second

select extract(hour from OrderedAt), extract(minute from OrderedAt), extract(second from OrderedAt) from orders\_\_;

-- DateAdd : This function adds a specified number of days, months, years,

-- hours, minutes, or seconds to a date expression.

-- For example, the following query will add 30 days to the OrderDate column:

select DATE\_ADD(OrderedAt,interval 10 day) as o from orders\_\_;

select datediff(now(),OrderedAt)as diff , OrderedAt, now() as now from orders\_\_;

select datediff('month',now(),OrderedAt) as month\_diff from orders\_\_;

-- this will add 2 day to current day

select date\_add(now(),interval 2 day)

-- this will subtract 2 day from now

select date\_add(now(),interval -2 day)

------------------------------------------------------------------------

Questions on Strings

Q1 Select those name whoese 2nd character is a vowel

select first\_name from Customers

where substr(first\_name,2,1) in ('a','e','i','o','u');

Q2: Select names from the Employees table where the 3rd character is a consonant,

and the last character is the same as the first character.

select first\_name from customers where

substr(first\_name,3,1) not in ('a','e','i','o','u') and

substr(first\_name,-1,1) = substr(first\_name,1,1);

more accurate ans is

select first\_name from customers where

substr(lower(first\_name),3,1) not in ('a','e','i','o','u') and

substr(lower(first\_name),-1,1) = substr(lower(first\_name),1,1);

Q3: Select names from the Employees table where

the 2nd and 4th characters are vowels (a, e, i, o, u), and

the name has at least one consonant between the 2nd and 4th characters.

select f\_name from customers

where substr(first\_name,2,1) in ('a','e','i','o','u')

and substr(first\_name,4,1) in ('a','e','i','o','u')

and substr(first\_name,3,1) not in ('a','e','i','o','u');

Q4: Select the names from the Employees table that contain a

palindrome within their name.

select first\_name from customers

where left(first\_name) = right(first\_name);

Q5 : Write a query to extract username(characters before @ symbol)

from the Email\_ID column.

select substr(email,1,locate('@',email,1)-1) as user\_name from employee;

Q6: Write a query to extract domain name like .com, .in, .au etc.

from the Email\_ID column.

Select substr(email,locate('.',email,1)) from employee;

Q7: Write a query to extract email service provider names like google,

yahoo, outlook, etc. from the Email\_ID column.

select substr(email,locate('@',email,1)+1,locate('.',email,1)-1) from employee;

Write a query to extract the employees’ details who joined in the year 1981.

SELECT \*

FROM employees

WHERE SUBSTR(BirthDate,3,2) = '81';

WINDOW FUNCTIONS

QLead by 2 lag by 2:

SELECT new\_id,

LEAD(new\_id, 2) OVER( ORDER BY new\_id) AS "LEAD\_by2",

LAG(new\_id, 2) OVER( ORDER BY new\_id) AS "LAG\_by2"

FROM test\_data

Q Update the name of the employee as xyz where eid =905

update employee set e\_name = 'Ram Sahu'

where eid = 905;

Q Find the first 5 chracters from a column

select left(col\_name,5) from table

Q Mask a col such that last 5 characters

are converted to \*

let col length is 9

select concat(left(col,length(col)-5,'\*\*\*\*\*') from table

Q find the 3rd to 6th character of a col

select substr(col,3,3) from table;

Q reterive age of emp who have age grater

than what ravi has

select name from employee where

age >(

select age from employee

where name = 'ravi')

\*\* you can use join to solve those subquery question

where u can use primary key as condition like here age is not pk so we

cant use join here eg: if we want to find customer who never order so we can use cid

as condition which is avaiable in both customer and order table.

Difference between view and CTE

let's consider a scenario where we have a database with two tables: Orders and

Customers.We want to create a query that lists all orders along with the customer names.

We'll demonstrate how this can be done using both a View and a CTE.

-- Create a View to join Orders and Customers tables

CREATE VIEW OrderDetails AS

SELECT o.OrderID, o.OrderDate, c.CustomerName

FROM Orders o

JOIN Customers c ON o.CustomerID = c.CustomerID;

-- Query the View

SELECT \* FROM OrderDetails;

In this code, we first create a View named OrderDetails

that encapsulates the logic of joining the Orders and Customers tables.

Then, we query the View to retrieve the desired information.

-- Using a CTE to join Orders and Customers tables

WITH OrderDetails AS (

SELECT o.OrderID, o.OrderDate, c.CustomerName

FROM Orders o

JOIN Customers c ON o.CustomerID = c.CustomerID

)

SELECT \* FROM OrderDetails;

In this code,we use a CTE named OrderDetails defined with the WITH clause.

The CTE contains the same logic as the View, joining the Orders and Customers

tables. After defining the CTE, we immediately select from it in the same query.

Both the View and the CTE produce the same result,

but the CTE is defined and used within a single query, making it suitable

for scenarios where you don't need to reuse the logic in other queries.

Views, on the other hand, provide a named, reusable structure

for encapsulating query logic that can be used in multiple queries

or by different users.

Q Find the list of employee that are assigned to some project

(two table employee and project)

select e.emp\_name from

employee e inner join project p

on e.eid = p.emp\_id;

Find the list of emp along with their respective project names

emp with no project should have their project asa null

select E.emp\_name, p.project\_name

from employee E left join project P

on E.id = P.emp\_id

Q let say you have wifi\_id , wifi\_speed , wifi\_latency

date

-- find the average wifi speed for each wifi

-- find the avg wifi speed of each wifi in last 2 days

select avg(wifi\_speed) from wifi

where dt>= date\_add(now(),interval -2 day)

group by wifi\_id

order by dt desc

---------------------------------------------------------------------

Group by vs distinct:

select distinct(name) from Employee

Select name from Employee group by name

both will give the same result

Dont do : Select distinct(name) from employee group by(name);

--------------------------------------------------------------------------

"Suppose you have a table named orders with the following columns: order\_id,

customer\_id, order\_date, and order\_amount. Write a SQL query to find the

cumulative sum of order\_amount for each customer, ordered by customer\_id

and order\_date. Use the PARTITION BY clause to achieve this."

You can use the PARTITION BY clause along with the SUM()

window function to calculate the cumulative sum of order\_amount for each customer.

Suppose you have a sequence of numbers: 1, 2, 3, 4, 5.

The cumulative sum of this sequence would be:

The first number (1) is 1.

The sum of the first two numbers (1 + 2) is 3.

The sum of the first three numbers (1 + 2 + 3) is 6.

The sum of the first four numbers (1 + 2 + 3 + 4) is 10.

The sum of all five numbers (1 + 2 + 3 + 4 + 5) is 15.

select sum(order\_amount) over (partiton by customer\_id

order by order\_date )from orders

order by customer\_id , order\_date

---------------------------------------------------------------------

\*\*\* GROUP BY:

Use GROUP BY when you want to aggregate data based on one or more columns

and create summary statistics for each group.

It's typically used for operations that result in a

single row per group, such as calculating sums, averages, counts,

or other aggregation functions for each group.

Example: Calculating the total sales by product category,

finding the average age of customers in each city, or counting

the number of orders per customer.

PARTITION BY (Window Functions):

Use PARTITION BY when you want to perform calculations or

aggregations within partitions of your data while maintaining all

rows in the result set.

It's often used for tasks that require comparing values within a specific

window or partition of data, such as calculating moving averages,

cumulative sums, or ranking items within groups.

Window functions allow you to perform calculations on subsets of

your data without collapsing the result to a single row per group.

Example: Calculating the running total of sales for each product over time,

ranking employees by salary within each department, or finding the lead and

lag values within a partition.

In summary, GROUP BY is used for summarizing data at the group level,

resulting in one row per group, while PARTITION BY (window functions)

is used for performing calculations within partitions or windows,

preserving all rows in the result set.

----------------------------------------------------------------------------

Q)) Retrieve the departments where the total salary expenditure

exceeds the average total salary expenditure across all departments.

SELECT department\_id

FROM employees

GROUP BY department\_id

HAVING SUM(salary) > (

SELECT AVG(total\_salary)

FROM (

SELECT department\_id, SUM(salary) as total\_salary

FROM employees

GROUP BY department\_id

) AS subquery

);

Q) Find the employee with the third highest salary without using the LIMIT clause.

select salary from (

select dense\_rank() over(order by salary desc )as r from employee

) where r =3;

Q) Identify departments that have less than the company-wide median number of employees

select department\_id from employee

group by department\_id

having count(id)<(

select Avg(emp\_cnt) from(

select department\_id,count(id)as emp\_cnt

from employee

group by department\_id

) sub\_query

)

Q Get the most common job title among employees who earn above the company average.

SELECT job\_title

FROM employee

WHERE salary > (SELECT AVG(salary) FROM employee)

GROUP BY job\_title

ORDER BY COUNT(\*) DESC

LIMIT 1;

--------------------------------------------------------------------------------

Q Find the total number of employees in the 'Sales' department and the 'Marketing'

department combined in a company database. Assume you have an 'employees' table with

columns 'employee\_id,' 'department,' and other relevant information.

Write an SQL query to calculate the sum of employees in these two departments."

select count(\*) from employee

where dept in ('Sales','Marketing')

---------------------------------------------------------------------------

Identify employees who earn more than the average

salary in both their department and the company.

SELECT id, name, salary

FROM employees e1

WHERE salary > (

SELECT AVG(salary)

FROM employees

WHERE department\_id = e1.department\_id

)

AND salary > (

SELECT AVG(salary)

FROM employees

);

Retrieve the month (in numbers) with the

highest total sales from a table of daily sales.

SELECT MONTH(date) as sales\_month

FROM sales

GROUP BY MONTH(date)

ORDER BY SUM(amount) DESC

LIMIT 1;

To find CustomerID who purchased both products 'A' and 'B,'

you can use the INTERSECT operator. Here's the SQL query:

-- Select CustomerID who purchased product 'A'

SELECT CustomerID FROM #Purchase WHERE ProductCode = 'A'

-- Intersect with select CustomerID who purchased product 'B'

INTERSECT

SELECT CustomerID FROM #Purchase WHERE ProductCode = 'B';

To find CustomerID who purchased both products 'A' and 'B' but not 'C',

you can use a combination of INTERSECT and EXCEPT operators. Here's the SQL query:

-- Select CustomerID who purchased product 'A'

SELECT CustomerID FROM #Purchase WHERE ProductCode = 'A'

-- Intersect with select CustomerID who purchased product 'B'

INTERSECT

SELECT CustomerID FROM #Purchase WHERE ProductCode = 'B'

-- Exclude CustomerID who purchased product 'C'

EXCEPT

SELECT CustomerID FROM #Purchase WHERE ProductCode = 'C';

print 1 to 10 number

with CTE as

(

select 1 Number

union all

select Number +1 from CTE where Number<10

)

Q1. Product Sales Analysis - Consider two tables, Products and Sales:

Table Schemas -

CREATE TABLE Products ( product\_id INT PRIMARY KEY, product\_name VARCHAR(255), category VARCHAR(255) );

CREATE TABLE Sales ( sale\_id INT PRIMARY KEY, product\_id INT, sale\_date DATE, quantity\_sold INT );

Assume necessary INSERT statements are already executed.

Write an SQL query to find the names of products that have sold more

than 100 units in the last quarter. The result should be ordered by product\_name.

SELECT product\_name

FROM Product P INNER JOIN SALES S

ON P.id = S.id

where S.quantity\_sold >100 and

EXTRACT(QUATER FROM sales\_date) = 4

Write a query to find the average working hours per day for

each employee. Include only employees who have clocked more than 20 days.

Order the results by average working hours ascending.

SELECT

employee\_id,

AVG(EXTRACT(HOUR FROM clock\_out\_time - clock\_in\_time)) AS working\_hours

FROM

EmployeeAttendance

GROUP BY

employee\_id

HAVING

COUNT(employee\_id) > 20

ORDER BY

working\_hours;

**Write a query to retrieve duplicate records from a table without using the DISTINCT keyword.**

Select id , count(id) from department

Group by id

Having count(id)>1

**How can you fetch alternate records from a table?**

**For** odd rows

Select \* from department

WHERE mod(id,2) =1

For even rows

SELECT \* FROM department

WHERE MOD(ID,2) = 0

**10. Write a query to find the second highest salary from a table.**

SELECT salary (

SELECT salary, DENSE\_RANK(ORDER BY salary DESC) AS r FROM department)

WHERE r=2

SELECT DISTINCT salary FROM department

ORDER BY salary LIMIT 1,1

**Write a query to find all employees who started after Jan 1, 2020, but before Jan 1, 2023.**

SELECT EMP FOM DEPARTMENT

WHERE joining\_date BETWEEN ‘2020-01-01’ AND ‘2022-12-31’

**How would you find the three most frequent values in a column along with their counts?**

SELECT NAME , COUNT(NAME) FROM Student

GROUP BY NAME

ORDER BY NAME DESC

LIMIT 3

**Write a query to get the monthly sales amount for the last 12 months.**

Select amount , month(sale\_date),year(sale\_date) from sales

Where SALE\_DATE BETWEEN( date\_sub(now(),INTERVAL 12 MONTH) AND NOW()

GROUP BY YEAR(sale\_date), MONTH(sale\_date)

ORDER BY YEAR(sale\_date) DESC, MONTH(sale\_date) DESC;

**Write a query to find employees who have managers with a salary greater than $100,000.**