1. Character manipulation function
2. Substr

select SUBSTR('abcdefgh',3,4) from dual

m = 3

n =4

sql string starts from 1.

Default value of m and n is 1

m = starting index, n = length of the substring

here for 3,4 is starts from c to length of 4 i.e. ‘cdef’

where m and n are inclusive

If m is negative then indexing starts from end

a b c d e f g h

-8 -7 -6 -5 -4 -3 -2 -1

select SUBSTR('abcdefgh',-3,2) from dual

starts from g and length of 2 means ‘fg’

If length is 3 or 4 or 5 in this case the result is ‘fgh’

1. trim

select TRIM(' abcdefg ') from dual ‘abcdefg’ only remove preceding and ending spaces, not spaces in the middle

select TRIM('g' from 'abcdefg') from dual 🡺 ‘abcdef’

select TRIM('fg' from 'abcdefg') from dula 🡺 error cannot use more than one character

1. Number function
2. Difference between ROUND and TRUNC

ROUND(46.267, 2) 🡺 46.27

TRUNC(46.267, 2) 🡺 46.26 Truncate only truncate the value

1. Instr function

select instr(name, 'a') from employee

It will give the position of ‘a’ in every name field if not exist the return 0.

Fore ‘Ram’ result is 2 for ‘Jodu’ result is 0

1. Example of case

https://www.sqlshack.com/case-statement-in-sql/

1. Write a sql query starts ends with ‘m’ and having 6 letters

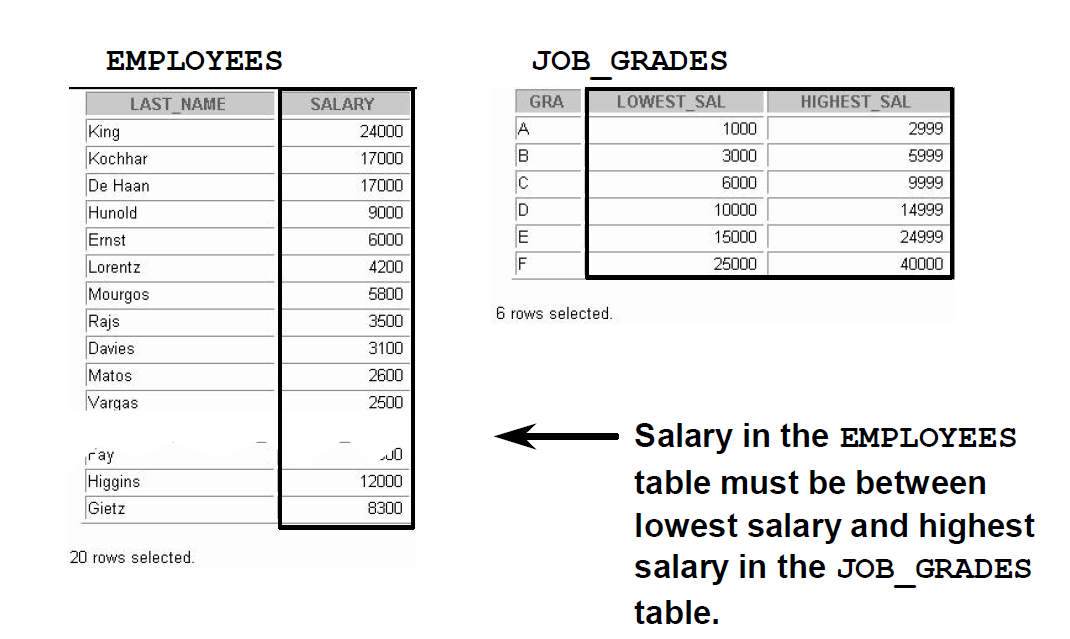
Add 5 ‘\_’ and m

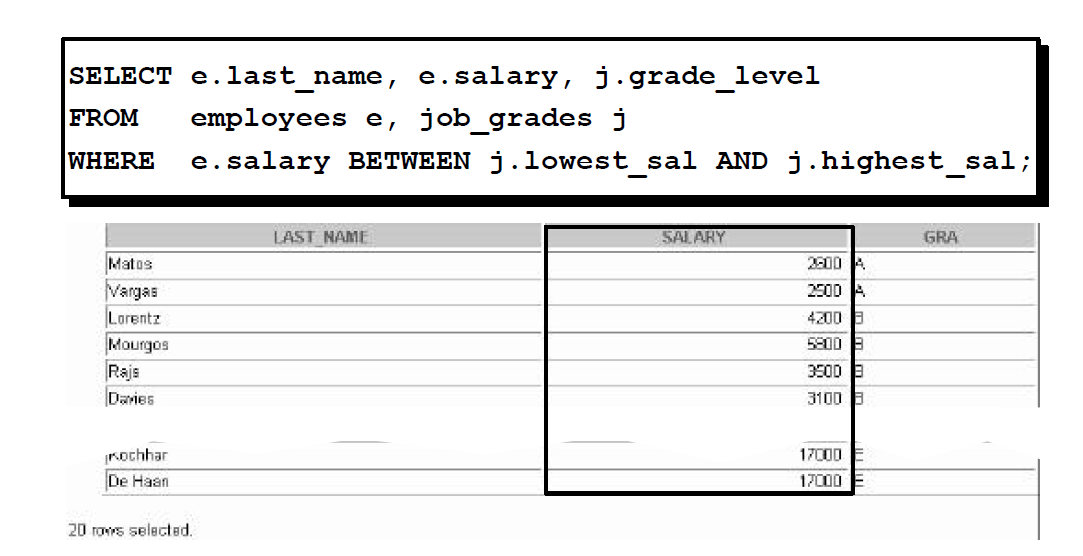
select name from employee where name like '\_\_\_\_m'

1. What is non equijoin ?

A non equijoin is a join condition containing something other than equality operator,

Here in this case it is using between and operator





1. **Cross join**  is Cartesian product of two table if employee has 8 rows, dept has 3 rows, it will give 24 rows in result. **Natural join** only returns matching rows, Both does not need to give any join condition. But the foreign key column name should be equal and cannot select the foreign key.

Like select dept\_id is not possible.

1. Use **JOIN** keyword with ON keyword with condition, without on it will not work.
2. Left outer join or outer join will choose all matching rows + left rows with right null and right outer join or right join do the vice versa. we need to give on condtion as well.
3. How to write left and right outer join using where clause

select e.emp\_id, e.name, d.dept\_id from employee e , department d where e.dept\_id(+) = d.dept\_id

But these statement always give full outer join

1. Relation between All, any and In

In works on list , suppose a subquery returns 98,94,95 then we can use In clause to check whether our id is one of these three id.

For ‘all’ and ‘any’ we can use other operators like <, > , >=, <=

Any will check with any value, All only returns true when condition satisfies for all the values.

select \* from employee where salary > all(1000, 18000)

will return only those employees whose salary is greater than 1000 AND 18000

select \* from employee where salary > all(1000, 18000)

will return only those employees whose salary is greater than 1000 OR 18000

1. Group by function on multiple group by column

EMP\_ID NAME SALARY DEPT\_ID MANAGER\_ID

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

7 Jodu 14000 3 0

1 Alice 10000 3 0

8 Modhu 50000 3 7

10 Ram 30000 4 7

select manager\_id, dept\_id , sum(salary) from employee group by(dept\_id,manager\_id)

This will group all the department with 3 and 4 seperately.

so it will pick Jodu , alice and Modhu for dept\_id =3

Then group by manager id 0 and 7

0 has 2 and 7 has 1 employee

The result will be

summation of 10000+ 14000

avg will be (10000+ 14000) /2

1. Example of a group by clause with having and where

select manager\_id,sum(salary) from employee where manager\_id > 0 group by manager\_id having sum(salary) > 2000

where has to be placed before having and group by both

Having can be placed before or after group by but recommended way is to place after group by for readability

1. Can we use aggregate functions in where clause ?

No, sum avg all has to be in having function

1. Sequence of execution in sql statement

FROM clause

ON clause

OUTER clause

WHERE clause

GROUP BY clause

HAVING clause

SELECT clause

DISTINCT clause

ORDER BY clause

TOP clause

1. **What is the difference between rowid and rownumber**

<https://oraclefrontovik.com/2014/05/16/using-oracle-sql-doesnt-rownum-2-work/>

select \* from employee where rownum < 3;

rowid is an unique address associated with each row of a table and it is not a number, it is a hexadecimal value.

rownum is assciated with each row after projection.

Above query will return first 2 rows of the employee table order by default

Find the second highest salary

SELECT Salary FROM EMPLOYEE WHERE salary < (SELECT MAX(SALARY) from EMPLOYEE) and rownum<2;

Order of execution

<https://sqlbolt.com/lesson/select_queries_order_of_execution>

1. **Some date related functions**

update employee set hire\_date = TO\_DATE('2016/01/03 10:10:44', 'yyyy/mm/dd hh24:mi:ss') where emp\_id = 8;

Select trunc(months\_between(TO\_DATE('2020/01/02', 'yyyy/mm/dd'), hire\_date)) as months FROM Employee

Select hire\_date , trunc(months\_between(sysdate, hire\_date)/12) as years FROM Employee

Select hire\_date , trunc(sysdate - hire\_date) as days FROM Employee

select to\_char(hire\_date, 'yyyy/mm/dd hh24:mi:ss') from employee where emp\_id = 9

1. **Find number of hour between two hire date**

select e2.name,to\_char(e2.hire\_date, 'yyyy/mm/dd hh24:mi:ss'), e1.name, to\_char(e1.hire\_date, 'yyyy/mm/dd hh24:mi:ss'),

(e1.hire\_date - e2.hire\_date)\*24 from employee e1, employee e2 where e1.emp\_id = 8 and e2.emp\_id = 9

1. **Write a SQL query to find all numbers that appear at least three times consecutively**.

+----+-----+

| Id | Num |

+----+-----+

| 1 | 1 |

| 2 | 1 |

| 3 | 1 |

| 4 | 2 |

| 5 | 1 |

| 6 | 2 |

| 7 | 2 |

+----+-----+

For example, given the above Logs table, 1 is the only number that appears consecutively for at least three times.

+-----------------+

| ConsecutiveNums |

+-----------------+

| 1 |

+-----------------+

SELECT

DISTINCT a.Num AS ConsecutiveNums

FROM

Logs AS a,

Logs AS b,

Logs AS c

WHERE

a.Num = b.Num

AND b.Num = c.Num

AND a.Id = b.Id + 1

AND b.Id = c.Id + 1;

1. Rank() and dense\_rank()

<https://stackoverflow.com/questions/11183572/whats-the-difference-between-rank-and-dense-rank-functions-in-oracle>

21. The Employee table holds all employees. Every employee has an Id, and there is also a column for the department Id.

+----+-------+--------+--------------+

| Id | Name | Salary | DepartmentId |

+----+-------+--------+--------------+

| 1 | Joe | 85000 | 1 |

| 2 | Henry | 80000 | 2 |

| 3 | Sam | 60000 | 2 |

| 4 | Max | 90000 | 1 |

| 5 | Janet | 69000 | 1 |

| 6 | Randy | 85000 | 1 |

| 7 | Will | 70000 | 1 |

+----+-------+--------+--------------+

Write a SQL query to find employees who earn the top three salaries in each of the department. For the above tables, your SQL query should return the following rows (order of rows does not matter).

+------------+----------+--------+

| Department | Employee | Salary |

+------------+----------+--------+

| IT | Max | 90000 |

| IT | Randy | 85000 |

| IT | Joe | 85000 |

| IT | Will | 70000 |

| Sales | Henry | 80000 |

| Sales | Sam | 60000 |

+------------+----------+--------+

with cte\_employee as (select e.Name , e.Salary, e.DepartmentId,

dense\_rank() OVER (PARTITION by e.DepartmentId order by e.salary desc) rn

from employee e)

select d.Name "Department", e.NAME "Employee", e. Salary "Salary"

from cte\_employee e

inner join Department d on e.DepartmentId = d.Id

where e.rn <= 3

order by d.name, salary desc

22. The Trips table holds all taxi trips. Each trip has a unique Id, while Client\_Id and Driver\_Id are both foreign keys to the Users\_Id at the Users table. Status is an ENUM type of (‘completed’, ‘cancelled\_by\_driver’, ‘cancelled\_by\_client’).

+----+-----------+-----------+---------+--------------------+----------+

| Id | Client\_Id | Driver\_Id | City\_Id | Status |Request\_at|

+----+-----------+-----------+---------+--------------------+----------+

| 1 | 1 | 10 | 1 | completed |2013-10-01|

| 2 | 2 | 11 | 1 | cancelled\_by\_driver|2013-10-01|

| 3 | 3 | 12 | 6 | completed |2013-10-01|

| 4 | 4 | 13 | 6 | cancelled\_by\_client|2013-10-01|

| 5 | 1 | 10 | 1 | completed |2013-10-02|

| 6 | 2 | 11 | 6 | completed |2013-10-02|

| 7 | 3 | 12 | 6 | completed |2013-10-02|

| 8 | 2 | 12 | 12 | completed |2013-10-03|

| 9 | 3 | 10 | 12 | completed |2013-10-03|

| 10 | 4 | 13 | 12 | cancelled\_by\_driver|2013-10-03|

+----+-----------+-----------+---------+--------------------+----------+

The Users table holds all users. Each user has an unique Users\_Id, and Role is an ENUM type of (‘client’, ‘driver’, ‘partner’).

+----------+--------+--------+

| Users\_Id | Banned | Role |

+----------+--------+--------+

| 1 | No | client |

| 2 | Yes | client |

| 3 | No | client |

| 4 | No | client |

| 10 | No | driver |

| 11 | No | driver |

| 12 | No | driver |

| 13 | No | driver |

+----------+--------+--------+

Write a SQL query to find the cancellation rate of requests made by unbanned users (both client and driver must be unbanned) between **Oct 1, 2013** and **Oct 3, 2013**. The cancellation rate is computed by dividing the number of canceled (by client or driver) requests made by unbanned users by the total number of requests made by unbanned users.

For the above tables, your SQL query should return the following rows with the cancellation rate being rounded to *two* decimal places.

+------------+-------------------+

| Day | Cancellation Rate |

+------------+-------------------+

| 2013-10-01 | 0.33 |

| 2013-10-02 | 0.00 |

| 2013-10-03 | 0.50 |

+------------+-------------------+

Answer :

tm.Request\_at as Day

,round(cancelled\_trips/total\_requests,2) as "Cancellation Rate"

FROM

(

SELECT

t.Request\_at

,sum(case when Status = 'cancelled\_by\_driver' then 1 when Status = 'cancelled\_by\_client' then 1 else 0 END) as cancelled\_trips

,count(t.Id) as total\_requests

FROM

Trips t

JOIN

Users u1

ON

t.Client\_Id = u1.Users\_Id

AND u1.Banned = 'No'

JOIN

Users u2

ON

t.Driver\_Id = u2.Users\_Id

AND u2.Banned = 'No'

WHERE t.Request\_at between '2013-10-01' AND '2013-10-03'

GROUP BY

t.Request\_at ) as tm

**23. Duplicate elements :**

STUDENTID NAME ADDRESS

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

1 A Kolkata

2 B Bangalore

3 C Chennai

4 D Delhi

5 A Kolkata

6 B Banglaore

**/\*\*Find number of duplicate rows \*\*/**

select name, address, count(\*) from student group by (name,address) having count(\*) >1

**/\*\* Remove duplicates which are recent \*\*/**

delete from student where studentid not in(select min(studentid) from student group by name, address)

**/\*\* Remove duplicates which are older \*\*/**

delete from student where studentid not in(select max(studentid) from student group by name, address)

**/\*\* Find all duplicate rows \*\*/**

ID CATEGORY NAME

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

1 A Abir

2 B Balbir

3 A Abir

4 C Abir

select \* from product where (name, category) in(select name,category from product group by (name,category) having count(\*) >1)

**24. Hierarchy statements :**

EMP\_ID NAME SALARY DEPT\_ID MANAGER\_ID

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

5 Ram 20000 2 0

6 Shyam 15000 2 5

7 Jodu 14000 3 0

1 Alice 10000 1 0

2 Bob 8000 1 1

3 Tom 1 2

4 Harry 6000 1 3

8 Modhu 3 7

**/\* find upper level hierarchy or ancester \*/**

select EMP\_ID, NAME from employee start with EMP\_ID = 4 connect by prior MANAGER\_ID =EMP\_ID

select EMP\_ID, NAME from employee start with EMP\_ID = 3 connect by prior MANAGER\_ID =EMP\_ID

**/\* find lower hierarchy or descendent \*/**

select EMP\_ID, NAME from employee start with EMP\_ID = 1 connect by prior EMP\_ID = MANAGER\_ID

select EMP\_ID, NAME from employee start with EMP\_ID = 2 connect by prior EMP\_ID = MANAGER\_ID

select e.emp\_id, e.name, e.manager\_id, m.name as manager\_name from employee e, employee m where e.manager\_id = m.emp\_id and e.emp\_id in

(select EMP\_ID from employee start with EMP\_ID = 4 connect by prior MANAGER\_ID =EMP\_ID)

**/\* find number of descendent level \*/**

select count(distinct level) from employee start with emp\_id =1 connect by prior emp\_id = manager\_id

**/\* Find number of ancester level \*/**

select count(distinct level) from employee start with emp\_id =4 connect by prior manager\_id = emp\_id

**25. Join queries :**

/\*\* Find 3rd maximum salary from employee table \*\*/

/\*\* 2 = 3-1 \*\*/

select e1.\* from employee e1 where 2 = (select count(distinct e2.id) from employee e2 where e2.salary > e1.salary)

/\*\* Find top 3 highest paid employees\*\*/

select \* from (select \* from employee where salary is not null order by salary desc) where rownum <=3

===========================================

/\* How to join multiple tables using join statement\*/

SELECT A.order\_number, A.header\_id, B.line\_id, B.quantity,

C.hold\_price\_id, C.released\_flag

FROM Table\_A a JOIN

Table\_B b

ON a.header\_id = b.header\_id JOIN

Table\_C c

ON c.header\_id = b.header\_id AND c.line\_id = b.line\_id

WHERE a.order\_number = '123';

/\*\* Return highest slary of the department , emp\_name, emp\_id \*\*/

select e.name,e.salary,d.dept\_name from employee e join department d on e.dept\_id = d.dept\_id where

e.salary in(select max(salary) from employee group by dept\_id)

/\*\* What is hash join ? \*\*/

select /\* +use\_hash( a b) \*/ a.emp\_no,b.dept\_no,b.dept\_name from emp a,dept b where a.dept\_no=b.dept\_no;

https://techgoeasy.com/hash-join/