Tricky SQL queries, advanced SQL queries, Interesting SQL queries

This is a collection of tricky sql queries, usually asked inthe interviews. Some of the questions have been taken from other websites and the original links have been provided. The sole purpose is to provide a collection of queries in one place.

The solutions will work in ORACLE database. It may or may not work in other databases(depending on the portability of the SQL queries). In cases where the solutions are not working, the question or the logic may be taken and an attempt be made to solve it.

The solutions provided in no case may be optimal. Better(efficient/alternate) solutions can be shared as we are all trying to learn and grow.

Any error that may have crept in inadvertently may be pointed out.

For more queries :http://oddabout.com/?page_id=1907

and http://oddabout.com/?page_id=2210

1. **A column has some negative values and some positive values. It is required to find the sum of negative numbers and the sum of the positive numbers in two separate columns.**

```
create table neg_pos(num number);
insert into neg_pos values(-1);
insert into neg_pos values(-2);
insert into neg_pos values(-3);
insert into neg_pos values(-4);
insert into neg_pos values(1);
insert into neg_pos values(2);
insert into neg_pos values(3);
insert into neg_pos values(4);
commit;
```

```
select * from neg_pos ;
num
-1
-2
-3
-4
1
2
3
4
```

Answer :

```
SELECT
SUM(CASE WHEN num < 0 THEN num ELSE 0 END) neg,
SUM(CASE WHEN num > 0 THEN num ELSE 0 END)pos
FROM neg_pos;
```

gives

———-

neg | pos

———-

-10 | 10

———-

****

2. **How to search for strings containing ‘%’ in Oracle? Search for columns containing ‘%’ in Oracle.**

In ORACLE , you can use the ESCAPE keyword to search for strings containing ‘%’. Otherwise it would be considered as a META CHARACTER .

Using the escape character (to search for strings containing like ‘ABC %%TRF’, ‘TR%FF’ or ‘%GH’)

Answer :

```
SELECT col_name FROM tbl_name
   WHERE col_name LIKE '%?%%' ESCAPE '?';
```

Here ‘?’ can be replaced with any other character.

Another solution:

```
SELECT col_name FROM tbl_name
WHERE instr(col_name,'%') > 0
```

****

3. **How does one remove special characters in ORACLE?**

To replace special characters in a string with null use translate :

translate(‘string’,'to_replace’,'replace_with’)

for eg:

```
SELECT translate
('asdfsd@#@$#$%$sdfg&;','!@#$%^&;*()_+=-`~?><:/.,',' ') FROM dual;
```

will return—asdfsdsdfg

To remove quotes, use two quotes for every single quote as shown below:

```
CREATE TABLE test_quote(quote VARCHAR2(5));
INSERT INTO test_quote VALUES ('L''uck');
SELECT * FROM test_quote;
SELECT REPLACE(quote,'''','') from test_quote;
```

A table has columns with numbers and numbers with alphabets. Write a query to select only those rows which contains alphanumeric values.

```
create table alpha_numeric(col1 varchar2(20));
insert into alpha_numberic values ('1000');
insert into alpha_numberic values ('a1093b');
insert into alpha_numberic values ('19b45');
insert into alpha_numberic values ('231');
insert into alpha_numberic values ('1000cc');
insert into alpha_numberic values ('a1000');
commit;
```

Answer:

```
SELECT * from alpha_numeric
where length(trim(translate(col1,'1234567890',' '))>0);
```

```
col1

——

a1093b

19b45

1000cc

a1000
```

****

4. **Give a string of format ‘NN/NN’, verify that the first and last two characters are numbers and that the middle character is’/’. Print the expression ‘NUMBER’ if valid, ‘NOT NUM’ if not valid. Use the following values to test your solution. ‘12/34’,’01/1a’, ‘99/98’**.

Answer:

```
SELECT CASE WHEN ascii(substr('99/98',1,1)) BETWEEN 48 AND 57
AND ascii(substr('99/98',2,1)) BETWEEN 48 AND 57
AND substr('99/98',3,1) ='/'
AND ascii(substr('99/98',4,1)) BETWEEN 48 AND 57
AND ascii(substr('99/98',5,1)) BETWEEN 48 AND 57
THEN 'number' ELSE 'not num' END FROM dual;
```

****

5. **From the given table, find those employees who are more than 21 years of age.**

```
 CREATE TABLE find_age(NAME Varchar2(10), dob DATE);
 INSERT INTO find_age VALUES('AAA',to_date('06/02/1983','DD/MM/YYYY'));
 INSERT INTO find_age VALUES('BBB',to_date('06/02/1967','DD/MM/YYYY'));
 INSERT INTO find_age VALUES('CCC',to_date('02/06/1983','DD/MM/YYYY'));
 INSERT INTO find_age VALUES('DDD',to_date('03/06/1983','DD/MM/YYYY'));
 INSERT INTO find_age VALUES('EEE',to_date('04/06/1999','DD/MM/YYYY'));
 INSERT INTO find_age VALUES('FFF',to_date('04/06/1999','DD/MM/YYYY'));
 INSERT INTO find_age VALUES('GGG',to_date('02/06/1999','DD/MM/YYYY'));
 INSERT INTO find_age VALUES('HHH',to_date('02/06/1990','DD/MM/YYYY'));
 INSERT INTO find_age VALUES('III',to_date('03/06/1990','DD/MM/YYYY'));
 COMMIT;
```

Answer:

```
SELECT NAME FROM find_age
    WHERE dob < (SELECT add_months(SYSDATE,-(12*21)) FROM dual);
```

*corrected after the comment.

****

6. **There are two tables stu_dept and dept_cap. stu_dept contains the student name and the department(consider distinct values). dept_cap contains the capacity for each department. We need to find those departments(DEPT) where the number of students is less than the total capacity of the department.**

```
CREATE TABLE stu_dept(stu_name VARCHAR2(30), dept VARCHAR2(30));
 INSERT INTO stu_dept VALUES('AAA','D1');
 INSERT INTO stu_dept VALUES('BBB','D1');
 INSERT INTO stu_dept VALUES('CCC','D1');
 INSERT INTO stu_dept VALUES('DDD','D1');
 INSERT INTO stu_dept VALUES('EEE','D2');
 INSERT INTO stu_dept VALUES('FFF','D2');
 INSERT INTO stu_dept VALUES('GGG','D2');
 INSERT INTO stu_dept VALUES('HHH','D3');
 INSERT INTO stu_dept VALUES('III','D3');
 INSERT INTO stu_dept VALUES('JJJ','D3');
 INSERT INTO stu_dept VALUES('KKK','D3');
 INSERT INTO stu_dept VALUES('LLL','D3');

CREATE TABLE dept_cap(dept VARCHAR2(5),capacity NUMBER);
 INSERT INTO dept_cap VALUES('D1',5);
 INSERT INTO dept_cap VALUES('D2',5);
 INSERT INTO dept_cap VALUES('D3',5);
```

Answer:

```
SELECT a.dept,a.capacity-t.cap remaining_seats
 FROM dept_cap a,
(SELECT dept,COUNT(dept)cap FROM stu_dept
GROUP BY dept)t
WHERE a.dept=t.dept
AND t.cap<a.capacity;
```

****

7. **Some questions on the dual table.**

Select two rows from dual

```
select dummy from dual
union all
select dummy from dual
```

To dispaly the numbers 1..10 from dual

```
select level from dual
connect by level <=10
```

or

```
SELECT ROWNUM FROM dual
CONNECT BY ROWNUM <=10
```

Another tricky question on dual involves the use of decode with NULL.

```
SELECT decode(null,null,1,0) from dual;
```

OUTPUT—1

Although two NULL values are not equal, the output is 1, as decode checks for the existence of NULL and does not compare the two values.

****

8. **Find the missing sequence. Table test_number contains the sequence for each id. Table test_number_min_max contains the minimum and maximum number for each id. We need to find the missing number between the minimum and maximum number for each id. text column can be ignored.**

```


 CREATE TABLE test_number(id NUMBER,SEQ NUMBER,text VARCHAR2(5)) ;
  INSERT INTO test_number VALUES(1,1,'AA');
  INSERT INTO test_number VALUES(1,3,'CC');
  INSERT INTO test_number VALUES(1,4,'DD');
  INSERT INTO test_number VALUES(1,5,'EE');
  INSERT INTO test_number VALUES(1,6,'FF');
  INSERT INTO test_number VALUES(1,7,'GG');
  INSERT INTO test_number VALUES(1,8,'HH');
  INSERT INTO test_number VALUES(1,10,'JJ');
  INSERT INTO test_number VALUES(2,1,'KK');
  INSERT INTO test_number VALUES(2,2,'LL');
  INSERT INTO test_number VALUES(2,3,'MM');
  INSERT INTO test_number VALUES(2,4,'NN');
  INSERT INTO test_number VALUES(2,6,'PP');
  INSERT INTO test_number VALUES(2,7,'QQ');
  INSERT INTO test_number VALUES(3,1,'TT');
  INSERT INTO test_number VALUES(3,4,'ZZ');
  INSERT INTO test_number VALUES(3,5,'XX');

 create tabel test_number_min_max(id number,mn number,mx  number);
  INSERT INTO test_number_min_max VALUES(1,1,12);
  INSERT INTO test_number_min_max VALUES(2,1,9);
  INSERT INTO test_number_min_max VALUES(3,1,5);
```

Answer:

```
SELECT r id,rn seq FROM (SELECT ROWNUM rn FROM all_objects WHERE ROWNUM
<13),
(SELECT ROWNUM r FROM all_objects
WHERE ROWNUM <4),test_number_min_max m
WHERE r=id
AND rn >= mn
AND rn <= mx
AND (r,rn) NOT IN
(SELECT id,seq FROM test_number)
```

OR

```
SELECT r id ,l seq FROM
(SELECT LEVEL l FROM dual
CONNECT BY LEVEL <13),
(SELECT LEVEL r  FROM dual  CONNECT BY LEVEL <4),
test_number_min_max
WHERE r=id
AND  l>=mn
AND l<=mx
AND (r,l)  NOT IN (SELECT id,seq  FROM test_number)
```

OUTPUT :

```
ID SEQ

1 2

1 9

1 11

1 12

2 5

2 8

2 9

3 2

3 3
```

****

9. **Get the following OUTPUT using dual**

```
1 L R
_________

1 1 1

1 1 2

1 1 3

1 2 1

1 2 2

1 2 3

1 3 1

1 3 2

1 3 3
```

Answer:

```
SELECT *  FROM
(SELECT 1  FROM dual),
(SELECT LEVEL l  FROM dual  CONNECT BY LEVEL <4),
(SELECT LEVEL r  FROM dual  CONNECT BY LEVEL <4);
```

****

10. Check the Input and Output and try to figure out the question.

```
CREATE TABLE test_output(NAME VARCHAR2(5), city VARCHAR2(6), num NUMBER);
       INSERT INTO test_output VALUES ('AN','TTT',5);
       INSERT INTO test_output VALUES ('AN','TTT',6);
       INSERT INTO test_output VALUES ('AN','TTT',7);
       INSERT INTO test_output VALUES ('BB','RRR',8);
       INSERT INTO test_output VALUES ('BB','RRR',9);
       INSERT INTO test_output VALUES ('BB','RRR',10);
```

Input :

NAME	CITY	NUM
AN	TTT	5
AN	TTT	6
AN	TTT	7
BB	RRR	8
BB	RRR	9
BB	RRR	10

Answer :

```
SELECT (CASE WHEN rn=1 THEN NAME ELSE NULL END) NAME,
 (CASE WHEN rn=1 THEN CITY ELSE NULL END ) CITY,
 num
 FROM
 (SELECT NAME,city,num,
 row_number() over(PARTITION BY NAME,city ORDER BY NAME) rn
 FROM test_output);
```

Output :

NAME	CITY	NUM
AN	TTT	5
		6
		7
BB	RRR	8
		9
		10

****

11.

****

12. **Beginner question based on the above logic. From the table given below, all the numbers should be on the first column and the alphabets on the second column.**

ALPA RANK

———-

a 1

b 2

c 4

x 5

y 6

z 8

9 g

0 f

7 e

3 d

All the alphabets on column B and all numbers in column A

OUTPUT:

A B

_____

0 f

1 a

2 b

3 d

4 c

5 x

6 y

7 e

8 z

9 g

Answer:

```
SELECT least(alpa,rank) a,greatest(alpa,rank) b FROM test_b
ORDER BY 1
```

****

13. **One of the most common question asked in interviews. To find the second (or third or fourth…) nth highest number in each group.**

```
CREATE TABLE find_rank(group_id VARCHAR2(2),val NUMBER);
INSERT INTO find_rank VALUES ('G1',11);
INSERT INTO find_rank VALUES ('G1',12);
INSERT INTO find_rank VALUES ('G1',13);
INSERT INTO find_rank VALUES ('G1',14);
INSERT INTO find_rank VALUES ('G2',8);
INSERT INTO find_rank VALUES ('G2',10);
INSERT INTO find_rank VALUES ('G2',10);
INSERT INTO find_rank VALUES ('G2',19);
INSERT INTO find_rank VALUES ('G2',21);
INSERT INTO find_rank VALUES ('G3',1);
INSERT INTO find_rank VALUES ('G3',2);
INSERT INTO find_rank VALUES ('G3',4);
INSERT INTO find_rank VALUES ('G4',0);
INSERT INTO find_rank VALUES ('G5',-1);
INSERT INTO find_rank VALUES ('G5',-2);
INSERT INTO find_rank VALUES ('G5',-3);
COMMIT;
```

Answer:

```
SELECT DISTINCT * FROM
(SELECT group_id,val,
dense_rank() over
(PARTITION BY group_id ORDER BY val DESC) rn
FROM find_rank) t
WHERE t.rn=&rank
```

with &rank = 2

```
GROUP_ID VAL RN

——————-

G1 13 2

G2 19 2

G3 2 2

G5 -2 2
```

If we need to have G4 also in the output even though it does not have a second/third highest value then :

```
SELECT DISTINCT f.group_id,
CASE WHEN o.val > 0 THEN to_char(o.val) ELSE 'N/A' END val
FROM find_rank f
LEFT OUTER JOIN
 (SELECT DISTINCT group_id,val FROM
 (SELECT group_id,val,
dense_rank() over (PARTITION BY group_id ORDER BY val DESC) rn
FROM find_rank) t
WHERE t.rn=&rank)o
ON f.group_id=o.group_id
```

with &rank =3

```
GROUP_ID VAL

—————-

G1 12

G2 10

G3 1

G4 N/A

G5 -3
```

****

14. **Another common interview question. To transform column into rows.**

```
CREATE TABLE col_to_rows(stu_name VARCHAR2(30),
subject VARCHAR2(10),marks NUMBER);
INSERT INTO col_to_rows VALUES('GEORGE','ECO',77);
INSERT INTO col_to_rows VALUES('GEORGE','HIS',99);
INSERT INTO col_to_rows VALUES('GEORGE','MAT',64);
INSERT INTO col_to_rows VALUES('GEORGE','GEO',85);
INSERT INTO col_to_rows VALUES('GEORGE','SCI',98);
INSERT INTO col_to_rows VALUES('ROBERT','ECO',71);
INSERT INTO col_to_rows VALUES('ROBERT','HIS',90);
INSERT INTO col_to_rows VALUES('ROBERT','MAT',84);
INSERT INTO col_to_rows VALUES('ROBERT','GEO',95);
INSERT INTO col_to_rows VALUES('ROBERT','SCI',58);
INSERT INTO col_to_rows VALUES('TIMOTHY','ECO',56);
INSERT INTO col_to_rows VALUES('TIMOTHY','HIS',55);
INSERT INTO col_to_rows VALUES('TIMOTHY','MAT',67);
INSERT INTO col_to_rows VALUES('TIMOTHY','GEO',54);
INSERT INTO col_to_rows VALUES('TIMOTHY','SCI',45);
COMMIT;
```

Answer :

```
SELECT stu_name,
max(CASE WHEN subject='ECO' THEN marks ELSE 0 END) ECO,
max(CASE WHEN subject='HIS' THEN marks ELSE 0 END) HIS,
max(CASE WHEN subject='MAT' THEN marks ELSE 0 END) MAT,
max(CASE WHEN subject='GEO' THEN marks ELSE 0 END) GEO,
max(CASE WHEN subject='SCI' THEN marks ELSE 0 END) SCI
FROM col_to_rows
GROUP BY stu_name
```

OR

```
SELECT stu_name,
MAX(decode(subject,'ECO',marks,0)) ECO,
MAX(decode(subject,'HIS',marks,0)) HIS,
MAX(decode(subject,'MAT',marks,0)) MAT,
MAX(decode(subject,'GEO',marks,0)) GEO,
MAX(decode(subject,'SCI',marks,0)) SCI
FROM col_to_rows GROUP BY stu_name
```

OR

```
SELECT stu_name,
max(CASE WHEN rn=1 THEN marks ELSE 0 END) ECO,
max(CASE WHEN rn=2 THEN marks ELSE 0 END) GEO,
max(CASE WHEN rn=3 THEN marks ELSE 0 END) HIS,
max(CASE WHEN rn=4 THEN marks ELSE 0 END) MAT,
max(CASE WHEN rn=5 THEN marks ELSE 0 END) SCI FROM
(SELECT stu_name,subject,marks, rank() over (PARTITION BY stu_name ORDER BY subject )rn
FROM col_to_rows)
GROUP BY stu_name
```

Output :

STU_NAME	ECO	HIS	MAT	GEO	SCI
GEORGE	77	99	64	85	98
ROBERT	71	90	84	95	58
TIMOTHY	56	55	67	54	45

****

15. **Another question from http://asktom.oracle.com/pls/asktom/f?p=100:11:0::::P11_QUESTION_ID:65356113852721 . This question teaches the trick to use decode with order by to select your own ordering rule . In this case the minimum value should always be at the last row. The other values are sorted in ascending order. You can create your own ordering rules.**

```
SQL> CREATE TABLE TEST1  (c1  NUMBER(2));

Table created

SQL> INSERT INTO TEST1  ( C1 )  VALUES (
  2  2);

1 row inserted

SQL> INSERT INTO TEST1  ( C1 )  VALUES (
  2  3);

1 row inserted

SQL> INSERT INTO TEST1  ( C1 )  VALUES (
  2  1);

1 row inserted

SQL> INSERT INTO TEST1  ( C1 )  VALUES (
  2  5);

1 row inserted

SQL> INSERT INTO TEST1  ( C1 )  VALUES (
  2  4);

1 row inserted

SQL> COMMIT;

Commit complete

SQL>

C1
2
```

```
3
1
5
4

select * from test1
order by decode( c1, (select min(c1) from test1), to_number(null), c1);

C1
2
3
4
5
1
```

****

- Post to Blogger
- Post to Delicious
- Post to Digg
- Post to Facebook

5

5

5

5

5

5

5

5

5

- Post to StumbleUpon
- Post to Twitter

1

1

1

1

1

1

1

1

1

40 Responses to “Tricky SQL queries, advanced SQL queries, Interesting SQL queries”

1.

admin says:
July 1, 2011 at 10:02 pm

for more queries visit

http://oddabout.com/?page_id=1907

reply

2.

unknown says:
September 4, 2012 at 10:44 am

very nice collection of queries….helped me alot…thanks

reply

3.

jatin says:
October 16, 2012 at 1:58 pm

very tricky, took time to understand them and very soon use them, quite helpful too

reply

4.

Ram says:
December 5, 2012 at 11:25 pm

Very Useful…….Appreciate efforts for putting very useful info.

reply

5.

Vimal says:
March 4, 2013 at 8:01 pm

Hello,

Above Help is Great.

reply

6.

Nishu says:
April 7, 2013 at 6:25 pm

Buddy nice collection of queries helpde me a lot..keep finding more things like this..Thanks!!!

reply

7.

Sankar says:
May 16, 2013 at 10:33 pm

Thanks a lot for your effort.It helped me a lot.

reply

8.

dipu says:
May 17, 2013 at 8:42 pm

10. How to solve it

CREATE TABLE test_output(NAME VARCHAR2(5), city VARCHAR2(6), num NUMBER);

INSERT INTO test_output VALUES (‘AN’,'TTT’,5);

INSERT INTO test_output VALUES (‘AN’,'TTT’,6);

INSERT INTO test_output VALUES (‘AN’,'SSS’,7);

INSERT INTO test_output VALUES (‘BB’,'SSS’,8);

INSERT INTO test_output VALUES (‘BB’,'GGG’,9);

INSERT INTO test_output VALUES (‘BB’,'GGG’,10);

The required output as follows

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

NAME CITY NUM

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

AN TTT 5

—————————

6

____

SSS 7

____

BB SSS 8

____

GGG 9

____

10

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

reply

- *dipu says:*
  May 17, 2013 at 8:49 pm

  Output :

  NAMECITYNUM

  ANTTT5

  6

  SSS7

  BBSSS8

  GGG9

  10

  reply

- *pradeep says:*
  May 20, 2013 at 1:07 pm

  SELECT (CASE WHEN rn=1 and num=5 or num=8 THEN NAME ELSE ” END) NAME,

  (CASE WHEN rn=1 or city=’sss’ and name=’bb’ THEN CITY ELSE ” END) CITY,

  num

  FROM

```
(SELECT NAME,city,num,

row_number() over(PARTITION BY city,city ORDER BY NAME) rn

FROM test_outputed) as tabl
```

reply

- pradeep says:
  May 20, 2013 at 1:12 pm

```
SELECT (CASE WHEN rn=1 and num=5 or num=8 THEN NAME ELSE ” END) NAME,

(CASE WHEN rn=1 or city=’sss’ and name=’bb’ THEN CITY ELSE ” END ) CITY,

num

FROM

(SELECT NAME,city,num,

row_number() over(PARTITION BY city,city ORDER BY NAME) rn

FROM test_output) as tabl
```

reply

- Garuda says:
  January 29, 2014 at 6:25 pm

Thanks a lot for your effort.

Nice questions…. keep posting …

- Tanvi Garg says:
  March 26, 2014 at 2:40 pm

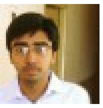
a more generic approach, without the hard coded city name could be as follows:-

```
select case when rn_name = 1 then name else ‘ ‘ end as name, case when rn_city = 1 then city else ‘ ‘ end as

city,num from

(select name, CITY,num,row_number() over ( partition by name order by city desc) rn_name,

row_number() over ( partition by name,city order by city desc) rn_city from test_output) a;
```



- *SIva says:*
  February 18, 2015 at 2:47 pm

Another generalised approach: hope it might help

```
from(select row_number() over (partitioned by name,city) as rn_1, row_number() over (partitioned by name) as rn_2
,name,city,num) select (case when rn_1 =1 and rn_2 =1 then Name when rn_1 = 1 and rn_2!=1 then city else num
end) as name,
(case when rn_1=1 and rn_2 = 1 then city when rn_1=1 and rn_2 !=1 then num else null end) as city,
(case when rn_1=1 and rn_2 =1 then num else null end) as num;
```

- *Anudeep Jaiswal says:*
  March 16, 2015 at 11:44 am

```
WITH CTE
AS (SELECT Name,
City,
NUM,
Row_number()
OVER(
Partition BY Name, City
ORDER BY Name, City, NUM) AS id
FROM test_output)
SELECT Ltrim(Rtrim(CASE WHEN ID = 1 THEN Name ELSE ” END + ‘ ‘ + CASE WHEN ID = 1 THEN City ELSE ”
END + ‘ ‘ + CONVERT(VARCHAR, Num))) Output
FROM CTE
```