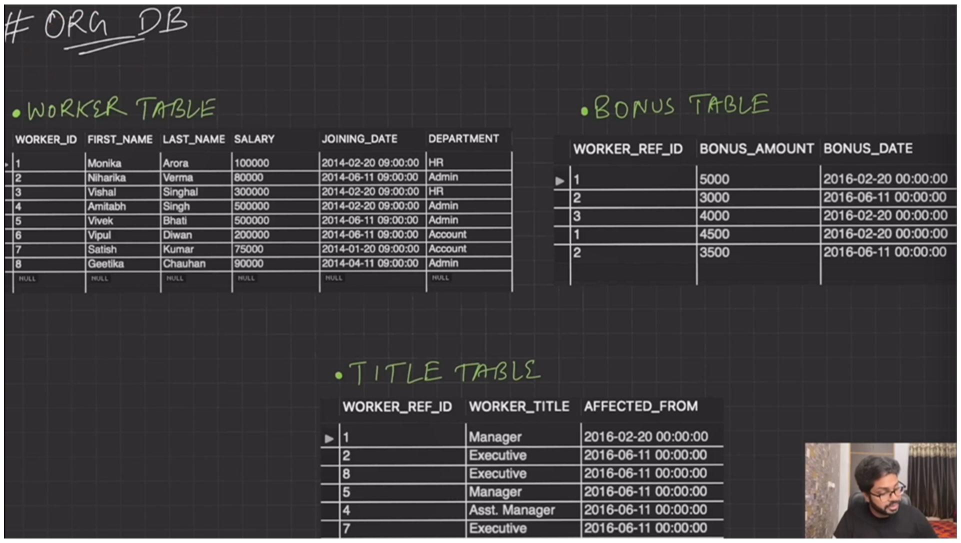
50 SQL Interview Questions

Practice SQL Questions →

Steps →

1. Database
2. Table/Relation
3. Data Relationship
4. Solving Question

Database



• WORKER TABLE

WORKER_ID	FIRST_NAME	LAST_NAME	SALARY	JOINING_DATE	DEPARTMENT
1	Monika	Arora	100000	2014-02-20 09:00:00	HR
2	Niharika	Verma	80000	2014-06-11 09:00:00	Admin
3	Vishal	Singhal	300000	2014-02-20 09:00:00	HR
4	Amitabh	Singh	500000	2014-02-20 09:00:00	Admin
5	Vivek	Bhati	500000	2014-06-11 09:00:00	Admin
6	Vipul	Diwan	200000	2014-06-11 09:00:00	Account
7	Satish	Kumar	75000	2014-01-20 09:00:00	Account
8	Geetika	Chauhan	90000	2014-04-11 09:00:00	Admin
NULL	NULL	NULL	NULL	NULL	NULL

• BONUS TABLE

WORKER_REF_ID	BONUS_AMOUNT	BONUS_DATE
1	5000	2016-02-20 00:00:00
2	3000	2016-06-11 00:00:00
3	4000	2016-02-20 00:00:00
1	4500	2016-02-20 00:00:00
2	3500	2016-06-11 00:00:00

• TITLE TABLE

WORKER_REF_ID	WORKER_TITLE	AFFECTED_FROM
1	Manager	2016-02-20 00:00:00
2	Executive	2016-06-11 00:00:00
8	Executive	2016-06-11 00:00:00
5	Manager	2016-06-11 00:00:00
4	Asst. Manager	2016-06-11 00:00:00
7	Executive	2016-06-11 00:00:00

```
CREATE DATABASE ORG;
SHOW DATABASES;
USE ORG;
```

```
CREATE TABLE Worker(
    WORKER_ID INT NOT NULL PRIMARY KEY AUTO_INCREMENT,
    FIRST_NAME VARCHAR(25),
    LAST_NAME VARCHAR(25),
    SALARY INT(15),
    JOINING_DATE DATETIME,
    DEPARTMENT CHAR(25)
);


INSERT INTO Worker VALUES (001, 'Monika', 'Arora', 100000, '14-02-20 09.00.00', 'HR');
INSERT INTO Worker VALUES (002, 'Niharika', 'Verma', 80000, '14-06-11 09.00.00', 'Admin');
INSERT INTO Worker VALUES (003, 'Vishal', 'Singhal', 300000, '14-02-20 09.00.00', 'HR');
INSERT INTO Worker VALUES (004, 'Amitabh', 'Singh', 500000, '14-02-20 09.00.00', 'Admin');
INSERT INTO Worker VALUES (005, 'Vivek', 'Bhati', 500000, '14-06-11 09.00.00', 'Admin');
INSERT INTO Worker VALUES (006, 'Vipul', 'Diwan', 200000, '14-06-11 09.00.00', 'Account');
INSERT INTO Worker VALUES (007, 'Satish', 'Kumar', 75000, '14-01-20 09.00.00', 'Account');
INSERT INTO Worker VALUES (008, 'Geetika', 'Chauhan', 90000, '14-04-11 09.00.00', 'Admin');

SELECT * FROM Worker;


CREATE TABLE Bonus(
    WORKER_REF_ID INT,
    BONUS_AMOUNT INT(10),
    BONUS_DATE DATETIME,
    FOREIGN KEY (WORKER_REF_ID)
        REFERENCES Worker(WORKER_ID)
        ON DELETE CASCADE
        ON UPDATE CASCADE
);

INSERT INTO Bonus
    (WORKER_REF_ID, BONUS_AMOUNT, BONUS_DATE) VALUES
        (001, 5000, '16-02-20'),
        (002, 3000, '16-06-11'),
        (003, 4000, '16-02-20'),
        (001, 4500, '16-02-20'),
        (002, 3500, '16-06-11');


CREATE TABLE Title(
    WORKER_REF_ID INT,
    WORKER_TITLE CHAR(25),
    AFFECTED_FROM DATETIME,
    FOREIGN KEY (WORKER_REF_ID)
        REFERENCES Worker(WORKER_ID)
        ON DELETE CASCADE
        ON UPDATE CASCADE
);


INSERT INTO Title
    (WORKER_REF_ID, WORKER_TITLE, AFFECTED_FROM) VALUES
        (001, 'Manager', '2016-02-20 00:00:00'),
        (002, 'Executive', '2016-06-11 00:00:00'),
        (008, 'Executive', '2016-06-11 00:00:00'),
        (005, 'Manager', '2016-06-11 00:00:00'),
        (004, 'Asst. Manager', '2016-06-11 00:00:00'),
        (007, 'Executive', '2016-06-11 00:00:00'),
        (006, 'Lead', '2016-06-11 00:00:00'),
        (003, 'Lead', '2016-06-11 00:00:00');

Select * FROM Title;
```

```
-- Q-1. Write an SQL query to fetch "FIRST_NAME" from Worker table using the alias name as <WORKER_NAME>.
SELECT FIRST_NAME AS WORKER_NAME FROM Worker;
```

```sql
-- Q-2. Write an SQL query to fetch "FIRST_NAME" from Worker table in upper case.
SELECT UPPER(FIRST_NAME) FROM Worker;


-- Q-3. Write an SQL query to fetch unique values of DEPARTMENT from Worker table.
SELECT DISTINCT DEPARTMENT FROM Worker;


-- Q-4. Write an SQL query to print the first three characters of FIRST_NAME from Worker table.
SELECT SUBSTRING(FIRST_NAME,1,3) FROM Worker;


-- Q-5. Write an SQL query to find the position of the alphabet ('b') in the first name column 'Amitabh' from Worker table.
SELECT INSTR(FIRST_NAME,'b') FROM Worker WHERE FIRST_NAME='Amitabh';


-- Q-6. Write an SQL query to print the FIRST_NAME from Worker table after removing white spaces from the right side.
SELECT RTRIM(FIRST_NAME) FROM Worker;


-- Q-7. Write an SQL query to print the DEPARTMENT from Worker table after removing white spaces from the left side.
SELECT LTRIM(FIRST_NAME) FROM Worker;


-- Q-8. Write an SQL query that fetches the unique values of DEPARTMENT from Worker table and prints its length.
SELECT DEPARTMENT, LENGTH(DEPARTMENT) FROM WORKER GROUP BY DEPARTMENT;


-- Q-9. Write an SOL query to print the FIRST_NAME from Worker table after replacing 'a' with 'A'.
SELECT REPLACE(FIRST_NAME,'a','A') FROM WORKER;


-- Q-10. Write an SQL query to print the FIRST_NAME and LAST_NAME from Worker table into a single column COMPLETE_NAME.
-- A space char should separate them.
SELECT CONCAT(FIRST_NAME,' ',LAST_NAME) AS COMPLETE_NAME FROM WORKER;


-- Q-11. Write an SQL query to print all Worker details from the Worker table order by FIRST_NAME Ascending.
SELECT * FROM WORKER ORDER BY FIRST_NAME ASC;


-- Q-12. Write an SQL query to print all Worker details from the Worker table order by
-- FIRST_NAME Ascending and DEPARTMENT Descending
SELECT * FROM WORKER ORDER BY FIRST_NAME ASC, DEPARTMENT DESC;


-- Q-13. Write an SQL query to print details for Workers with the first name as "Vipul" and "Satish" from Worker table.
SELECT * FROM WORKER WHERE FIRST_NAME IN ('Vipul','Satish');


-- Q-14. Write an SQL query to print details of workers excluding first names, "Vipul" and "Satish" from Worker table.
SELECT * FROM WORKER WHERE FIRST_NAME NOT IN ('Vipul','Satish');


-- Q-15. Write an SQL query to print details of Workers with DEPARTMENT name as "Admin*".
SELECT * FROM WORKER WHERE DEPARTMENT LIKE 'Admin%';


-- Q-16. Write an SQL query to print details of the Workers whose FIRST_NAME contains 'a'.
SELECT * FROM WORKER WHERE FIRST_NAME LIKE '%a%';


 -- 0-17. Write an SQL query to print details of the Workers whose FIRST_NAME ends with 'a'.
 SELECT * FROM WORKER WHERE FIRST_NAME LIKE '%a';


-- Q-18. Write an SQL query to print details of the Workers whose FIRST_NAME ends with 'h' and contains six alphabets.
SELECT * FROM WORKER WHERE FIRST_NAME LIKE '%h' AND LENGTH(FIRST_NAME)=6;


-- Q-19. Write an SQL query to print details of the Workers whose SALARY lies between 100000 and 500000.
```

```
SELECT * FROM WORKER WHERE SALARY BETWEEN 100000 AND 500000;

-- Q-20. Write an SOL query to print details of the Workers who have joined in Feb'2014.
SELECT * FROM WORKER WHERE YEAR(joining_date)=2014 AND MONTH(joining_date)=02;


-- Q-21. Write an SOL query to fetch the count of employees working in the department 'Admin'.
SELECT DEPARTMENT,COUNT(*) FROM WORKER WHERE DEPARTMENT='Admin';

-- Q-22. Write an SQL query to fetch worker full names with salaries >= 50000 and <= 100000.
SELECT CONCAT(FIRST_NAME,' ', LAST_NAME) AS COMPLETE_NAME, SALARY FROM WORKER
WHERE SALARY>=50000 AND SALARY<=100000;


-- Q-23. Write an SQL query to fetch the no. of workers for each department in the descending order.
SELECT DEPARTMENT, COUNT(WORKER_ID) AS NUMBER_OF_WORKER
FROM WORKER GROUP BY DEPARTMENT ORDER BY  NUMBER_OF_WORKER DESC;


-- Q-24. Write an SOL query to print details of the Workers who are also Managers.
SELECT WORKER.* FROM WORKER
INNER JOIN
TITLE ON WORKER.WORKER_ID=TITLE.WORKER_REF_ID
WHERE WORKER_TITLE='Manager';


-- Q-25. Write an SQL query to fetch number (more than 1) of different titles in the ORG.
SELECT WORKER_TITLE, COUNT(*) AS COUNT FROM TITLE GROUP BY WORKER_TITLE HAVING COUNT(*)>1;


-- Q-26. Write an SQL query to show only odd rows from a table.
SELECT * FROM WORKER WHERE NOT WORKER_ID%2=0;


-- Q-27. Write an SQL query to show only even rows from a table.
SELECT * FROM WORKER WHERE WORKER_ID%2=0;


-- Q-28. Write an SQL query to clone a new table from another table.
CREATE TABLE WORKER_CLONE LIKE WORKER;
INSERT INTO WORKER_CLONE SELECT * FROM WORKER;
SELECT * FROM WORKER_CLONE;


-- Q-29. Write an SQL query to fetch intersecting records of two tables.
-- INTERSECT
SELECT WORKER.* FROM WORKER
INNER JOIN WORKER_CLONE USING(WORKER_ID);


-- Q-30. Write an SQL query to show records from one table that another table does not have.
-- MINUS
SELECT WORKER.* FROM WORKER
LEFT JOIN WORKER_CLONE USING(WORKER_ID) WHERE WORKER_CLONE.WORKER_ID IS NULL;


-- Q-31. Write an SQL query to show the current date and time.
SELECT CURDATE();
SELECT NOW();

-- Q-32. Write an SQL query to show the top n (say 5) records of a table order by descending salary.
SELECT * FROM WORKER ORDER BY SALARY DESC LIMIT 5;

-- Q-33. Write an SQL query to determine the nth (say n=5) highest salary from a table.
SELECT * FROM WORKER ORDER BY SALARY DESC LIMIT 4,1;

-- Q-34. Write an SQL query to determine the 5th highest salary without using LIMIT keyword.
SELECT SALARY FROM WORKER  W1
WHERE 4=(
SELECT COUNT(DISTINCT (W2.SALARY))
FROM WORKER W2
WHERE W2.SALARY>=W1.SALARY
);


-- Q-35. Write an SQL query to fetch the list of employees with the same salary.
SELECT W1.* FROM WORKER W1, WORKER W2 WHERE W1.SALARY=W2.SALARY AND W1.WORKER_ID !=W2.WORKER_ID;


-- Q-36. Write an SQL query to show the second highest salary from a table using sub-query
SELECT MAX(SALARY) FROM WORKER
WHERE SALARY NOT IN (SELECT MAX(SALARY) FROM WORKER);

-- Q-37. Write an SQL query to show one row twice in results from a table.
SELECT * FROM WORKER
UNION ALL
SELECT * FROM WORKER ORDER BY WORKER_ID;

-- Q-38. Write an SQL query to list worker_id who does not get bonus.
SELECT WORKER_ID FROM WORKER WHERE
WORKER_ID NOT IN (SELECT WORKER_REF_ID FROM BONUS);


-- Q-39. Write an SQL query to fetch the first 50% records from a table.
SELECT * FROM WORKER WHERE WORKER_ID <=(SELECT COUNT(WORKER_ID)/2 FROM WORKER);


-- Q-40. Write an SQL query to fetch the departments that have less than 4 people in it.
SELECT DEPARTMENT,COUNT(*) AS NO_OF_PEOPLE FROM WORKER GROUP BY DEPARTMENT HAVING NO_OF_PEOPLE <4;


-- Q-41. Write an SQL query to show all departments along with the number of people in there.
SELECT DEPARTMENT,COUNT(*) AS NO_OF_PEOPLE FROM WORKER GROUP BY DEPARTMENT;


-- Q-42. Write an SQL query to show the last record from a table.
SELECT * FROM WORKER WHERE WORKER_ID=(SELECT MAX(WORKER_ID) FROM WORKER);


-- Q-43. Write an SQL query to fetch the first row of a table.
SELECT * FROM WORKER WHERE WORKER_ID=(SELECT MIN(WORKER_ID) FROM WORKER);

-- Q-44. Write an SQL query to fetch the last five records from a table.
(SELECT * FROM WORKER ORDER BY WORKER_ID DESC LIMIT 5) ORDER BY WORKER_ID;


-- Q-45. Write an SQL query to print the name of employees having the highest salary in each department.
SELECT W.DEPARTMENT, W.FIRST_NAME, W.SALARY FROM
(SELECT MAX(SALARY) AS MAXSALARY, DEPARTMENT FROM WORKER GROUP BY DEPARTMENT) TEMP
INNER JOIN WORKER W ON TEMP.DEPARTMENT=W.DEPARTMENT AND TEMP.MAXSALARY=W.SALARY;

-- Q-46. Write an SQL query to fetch three max salaries from a table using co-related query.
SELECT DISTINCT SALARY FROM WORKER W1
WHERE 3>=(
SELECT COUNT(DISTINCT SALARY) FROM WORKER  W2 WHERE W1.SALARY<=W2.SALARY)
ORDER BY W1.SALARY DESC;

-- OR

SELECT DISTINCT SALARY FROM WORKER ORDER BY SALARY DESC LIMIT 3;


-- Q-47. Write an SQL query to fetch three min salaries from a table using co-related query.
SELECT DISTINCT SALARY FROM WORKER W1
WHERE 3>=(
SELECT COUNT(DISTINCT SALARY) FROM WORKER  W2 WHERE W1.SALARY>=W2.SALARY)
ORDER BY W1.SALARY DESC;

-- OR

SELECT DISTINCT SALARY FROM WORKER ORDER BY SALARY ASC LIMIT 3;

-- Q-48. Write an SQL query to fetch nth max salaries from a table.
-- let n=4
```

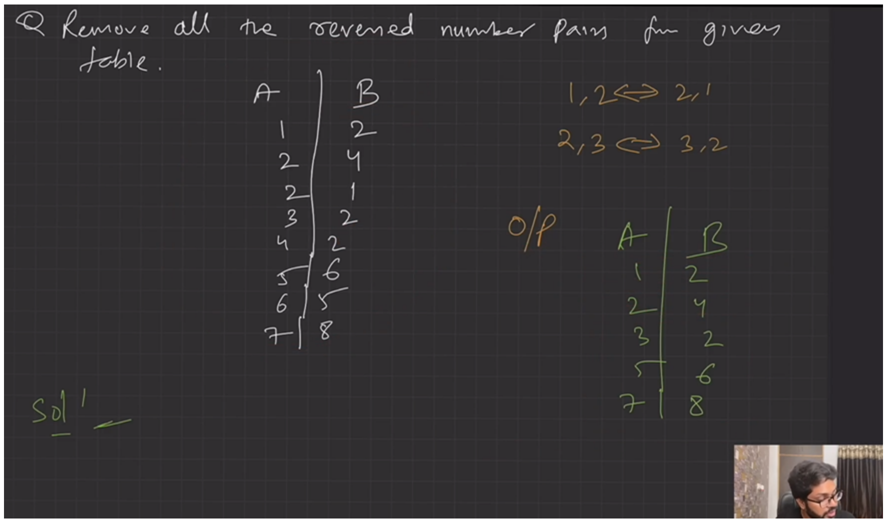
```
SELECT DISTINCT SALARY FROM WORKER W1
WHERE 4>=(
SELECT COUNT(DISTINCT SALARY) FROM WORKER  W2 WHERE W1.SALARY<=W2.SALARY)
ORDER BY W1.SALARY DESC;

-- Q-49. Write an SQL query to fetch departments along with the total salaries paid for each of them.
SELECT DEPARTMENT, SUM(SALARY) AS TOTAL_PAID_SALARY FROM WORKER
GROUP BY DEPARTMENT ORDER BY TOTAL_PAID_SALARY DESC;


-- Q-50. Write an SQL query to fetch the names of workers who earn the highest salary.
SELECT FIRST_NAME, SALARY FROM WORKER WHERE SALARY=(SELECT MAX(SALARY) FROM WORKER W2);
```

A correlated subquery is one way of reading every row in a table and comparing values in each row against related data.

Important Question



Q Remove all the reversed number pairs from given table.

A	B
1	2
2	4
2	1
3	2
4	2
5	6
6	5
7	8

1,2 ⟺ 2,1

2,3 ⟺ 3,2

O/P

A	B
1	2
2	4
3	2
5	6
7	8

Solⁿ

💡 Hint 1: Joins
Hint 2: Co-related Queries

```
CREATE DATABASE TEMP;
USE TEMP;
CREATE TABLE PAIRS(
    A INT,
    B INT
);

INSERT INTO PAIRS VALUES(1,2),(2,4), (2,1),(3,2), (4,2), (5,6), (6,5), (7,8);

SELECT * FROM PAIRS;


-- remove reversed pairs
-- Method 1: Joins
SELECT LT.* FROM PAIRS LT
LEFT JOIN
PAIRS RT ON LT.A=RT.B AND LT.B=RT.A
WHERE RT.A IS NULL OR LT.A<RT.A;


-- METHOD 2: CORRELATED SUBQUERIES
```

```
SELECT * FROM PAIRS P1 WHERE NOT EXISTS
(SELECT * FROM PAIRS P2 WHERE P1.B=P2.A AND P1.A=P2.B AND P1.A>P2.A);
```