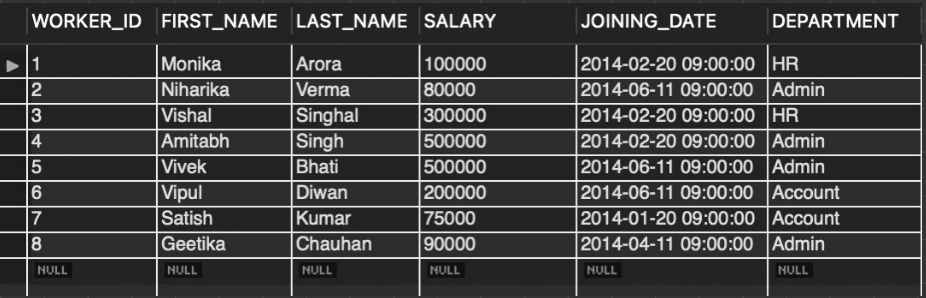
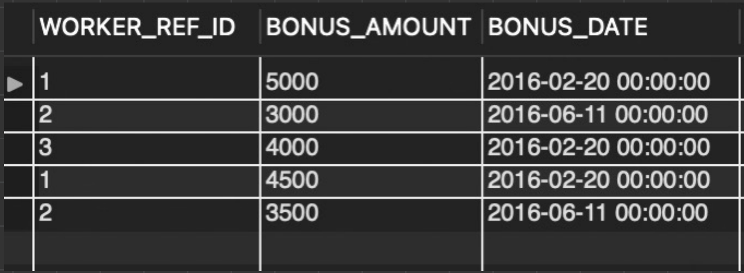
ORG_DB

• WORKER TABLE



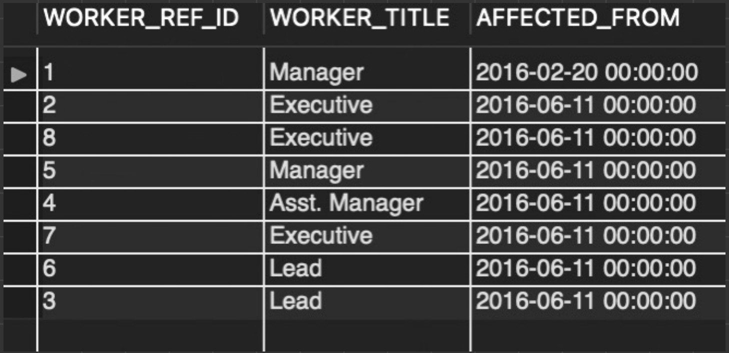
WORKER_ID	FIRST_NAME	LAST_NAME	SALARY	JOINING_DATE	DEPARTMENT
1	Monika	Arora	100000	2014-02-20 09:00:00	HR
2	Niharika	Verma	80000	2014-06-11 09:00:00	Admin
3	Vishal	Singhal	300000	2014-02-20 09:00:00	HR
4	Amitabh	Singh	500000	2014-02-20 09:00:00	Admin
5	Vivek	Bhati	500000	2014-06-11 09:00:00	Admin
6	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
6	Lead	2016-06-11 00:00:00
3	Lead	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 char(25),
last_name char(25),
salary
int(15),
joining_date datetime,
department char(25)
);

insert into worker
(worker_id,
first_name, last_name, salary, joining_date, department) values
(001, 'Monika', 'Arora',
100000, '14-02-20 09.00.00', 'HR'),
(002, 'Niharika', 'Verma', 80000, '14-06-11 09.00.00',
'Admin'),
(003, 'Vishal', 'Singhal', 300000, '14-02-20 09.00.00', 'HR'),
(004, 'Amitabh',
'Singh', 500000, '14-02-20 09.00.00', 'Admin'),
(005, 'Vivek', 'Bhati', 500000, '14-06-11
09.00.00', 'Admin'),
(006, 'Vipul', 'Diwan', 200000, '14-06-11 09.00.00', 'Account'),
(007,
'Satish', 'Kumar', 75000, '14-01-20 09.00.00', 'Account'),
(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
);

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');

select * from bonus;

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
);

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;

-- 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;  # or
select department from worker group by
department;

-- 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(department) 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;  # or
select distinct department, length(department) from
worker;

-- Q-9. Write an SQL 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%';

-- Q-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';

-- 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;  # or
select * from worker
where salary>=100000 and salary<=500000;

-- Q-20. Write an SQL 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
-- SQL query to fetch the count of employees
working in the department ‘Admin’.
select department, count(department) 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 full_name,
salary from worker where salary between 50000 and 100000;
```

```
-- Q-23. Write an SQL query
-- to
fetch the no. of workers for each department in the descending order.
select department,
count(department) as no_of_worker from worker
group by department
order by no_of_worker
desc;

-- Q-24. Write an SQL query to print details of the Workers who are also
--
Managers.
select w.* from worker as w
inner join
title as t
on w.worker_id=t.worker_ref_id and
t.worker_title="Manager";

-- Q-25. Write an SQL query to fetch number (more than 1)
of
-- same titles in the ORG of different types.
select worker_title, count(worker_title) 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 mod(worker_id,2)!=0;
select * from
worker where mod(worker_id,2)<>0;

# Alternative: using window function
select * from

(select *, Row_Number() over(order by worker_id) AS
RowNumber
from worker) as w
where
w.RowNumber % 2 = 1;

-- Q-27. Write an SQL query to show only even rows from a
--
table.
select * from worker where mod(worker_id,2)=0;

# Alternative: using window function#
Alternative: using window function
select * from
(select *, Row_Number() over(order by
worker_id) AS
RowNumber
from worker) as w
where w.RowNumber % 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.
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.
select worker.* from worker
left join
```

```
worker_clone
using(worker_id) where worker_clone.worker_id is null;  # here we using the join and it's
neccessary

-- Q-31. Write an SQL query to
show the
-- current date and time.
-- DUAL
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 1
offset 4;  # offset n-1

-- Q-34. Write an SQL query to determine the nth (say n=5)
-- highest
salary from a table.
select * from worker as w1
where N-1= (
          select
count(distinct(w2.salary))
          from worker as 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 as w1, worker as 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);

-- This gives the full detail of the worker who is having second
highest salary
select * from worker
where salary = (
    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);

-- this give the full detail of the worker who does not get bonus
select * from worker
where worker_id not in(select worker_ref_id from bonus);
```

```sql
-- 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(department) as dep_count from
worker group by department having dep_count<4;

-- Q-41. Write an SQL query to show all
--
departments along with the number of people in there.
select department, count(department) as
dep_count 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 max_sal, department from worker group by department) as temp
inner join worker as w
on
temp.department=w.department and temp.max_sal=w.salary;

-- Q-46. Write an SQL query to
--
fetch three max salaries from a table using co-related subquery
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;
-- DRY RUN AFTER REVISING THE CORELATED SUBQUERY
CONCEPT
-- FROM LEC-9.

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
subquery
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;

-- Q-48. Write an SQL
query to fetch nth
-- max salaries from a table.
select distinct salary from
worker w1
```

```
where n
>= (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) total_sal from worker group by
department order by total_sal 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);
```