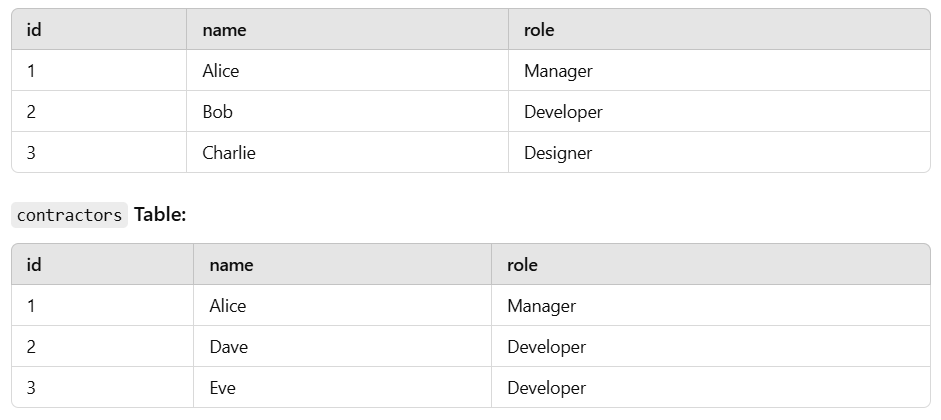
**SQL Interviews Queries**

1. **Second Highest salary**
2. **Select distinct(salary) from employee order by salary desc limit 1 offset 1;**
3. **Select max(salary) as second\_max from employee where salary <(select max(salary) from employee).**

**UNION and UNIONALL**

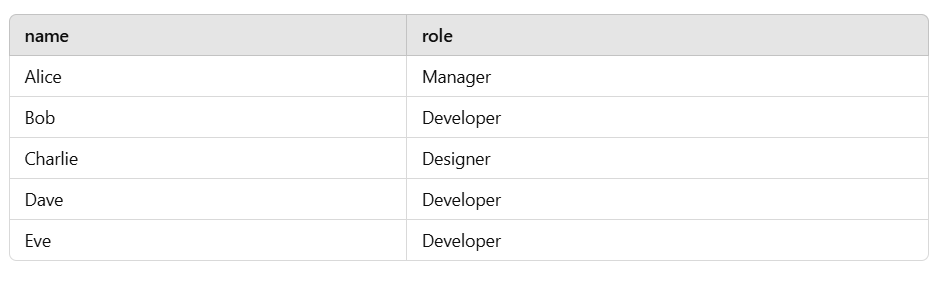
****

**UINON:- It is used to combining the result of two or more query by removing the duplicate row data.**

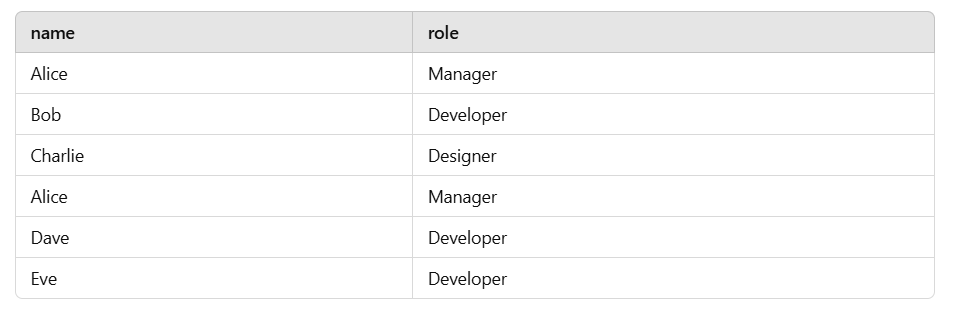
**It is slower because it is filtering the duplicate data which takes some time to execute as compare to the UNIONALL**

**Query:-**

**Select name, role from employees UNION select name, role from contractor;**

**Output:-** ****

**UNIONALL:- It is used to combining the result of two or more quires by including the duplicate row data.It is faster as compared to the UNION**

****

**Query:- select name, role from employees UNIONALL select name, role from contractor.**

**Super key:-**

**Candidate key:- The key fields which can be uniquely identify the record of the table.**

**Candidate key could be any column in the table which qualifies the as primary key.**

**Primary key can select among the candidate key, A table can have more than one candidate key.**

**Composite Key:- This key is combination(consist ) of two or more column in a table. The combination of these columns uniquely identifies a record in the table as primary key and it could not set to be as null.**

**Alternate key :-**

**Those candidate key which is eligible for the primary key but not primary key then those keys is know as the alternate key**

**IN other word those column which is satisfying all the condition of primary key but not primary in the table is know as Candidate key.**

**Ex- let suppose we have order table so by combination of order\_id and customer\_id we can make the composite key as primary key .**

**CREATE TABLE student\_courses (**

**student\_id INT,**

**course\_id INT,**

**enrollment\_date DATE,**

**PRIMARY KEY (student\_id, course\_id) -- Composite key consisting of student\_id and course\_id**

**);**

**Querys:-  
  
Making group using group by column clause.**

1. **Select grade\_name,count(\*) from grades group by grade\_name;**

**Group with having clause:**

**SELECT** Department ,**COUNT**(\*) **FROM** salaryprocesses **GROUP** **BY** Department **HAVING** **COUNT**(\*)>10

**String Starts with**

**SELECT** \* **FROM** salaryprocesses **WHERE** Department **LIKE** 'A%' **OR** Department **like** 'I%'

**When we have to perform updation simultaneously the we use case**

**UPDATE** categories

**SET** category =**case**

**when** category='IT' **then** 'iit'

**when** category='ITT' **then** 'itt'

**ELSE** category

**END**;