SQL Interview Questions

1. **What is SQL?**  
   SQL stands for Structured Query Language. It is a language used to interact with the database, i.e to create a database, to create a table in the database, to retrieve data or update a table in the database etc. SQL is an ANSI(American National Standards Institute) standard. Using SQL, we can do many things, for example – we can execute queries, we can insert records in a table, we can update records, we can create a database, we can create a table, we can delete a table etc.
2. **What is a Database?**  
   A Database is defined as a structured form of data which is stored in a computer or data in an organised manner and can be accessed in various ways. It is also the collection of schemas, tables, queries, views etc. Database helps us in easily storing, accessing and manipulation of data held in a computer. The Database Management System allows a user to interact with the database.

1. **Does SQL support programming language features ?**  
   It is true that SQL is a language but it does not support programming as it is not a programming language, it is a command  language. We do not have conditional statements in SQL like for loops or if..else, we only have commands which we can use to query, update , delete etc. data in the database. SQL allows us to manipulate data in a database.
2. **What are the differences between SQL and PL/SQL?**  
   Ans: Some common differences between SQL and PL/SQL are as shown below:

|  |  |
| --- | --- |
| **SQL** | **PL/SQL** |
| SQL is a query execution or commanding language | PL/SQL is a complete programming language |
| SQL is data oriented language | PL/SQL is a procedural language |
| SQL is very declarative in nature | PL/SQL has a procedural nature |
| It is used for manipulating data | It is used for creating applications |
| We can execute one statement at a time in SQL | We can execute block of statements in PL/SQL |
| SQL tells database, what to do? | PL/SQL tells database how to do |
| We can embed SQL in PL/SQL | We can not embed PL/SQL in SQL |

1. **What is the difference between BETWEEN and IN operators in SQL?**  
   **BETWEEN**  
   The **BETWEEN** operator is used to fetch rows based on a range of values.  
   For example,
2. SELECT \* FROM Students
3. WHERE ROLL\_NO BETWEEN 20 AND 30;

This query will select all those rows from the table Students where the value of the field ROLL\_NO lies between 20 and 30.  
**IN**  
The **IN**operator is used to check for values contained in specific sets.  
For example,

SELECT \* FROM Students

WHERE ROLL\_NO IN (20,21,23);

This query will select all those rows from the table Students where the value of the field ROLL\_NO is either 20 or 21 or 23.

1. **Write an SQL query to find names of employee start with ‘A’?**  
   The LIKE operator of SQL is used for this purpose. It is used to fetch filtered data by searching for a particular pattern in where clause.  
   The Syntax for using LIKE is,
2. **SELECT column1,column2 FROM table\_name WHERE column\_name LIKE pattern;**
3. **LIKE**: operator name
4. **pattern**: exact value extracted from the pattern to get related data in
5. result set.

The required query is:

SELECT \* FROM Employees WHERE EmpName like 'A%' ;

You may refer to this article on [WHERE clause](https://www.geeksforgeeks.org/sql-where-clause/) for more details on LIKE operator.

1. **What is the difference between CHAR and VARCHAR2 datatype in SQL?**  
   Both of these datatypes are used for characters but varchar2 is used for character strings of variable length whereas char is used for character strings of fixed length. For example, if we specify the type as char(5) then we will not be allowed to store string of any other length in this variable but if we specify the type of this variable as varchar2(5) then we will be allowed to store strings of variable length, we can store a string of length 3 or 4 or 2 in this variable.
2. **Name different types of case manipulation functions available in SQL.**  
   There are three types of case manipulation functions available in SQL. They are,
   * **LOWER**: The purpose of this function is to return the string in lowercase. It takes a string as argument and returns the string by converting it into lower case.  
     Syntax:
   * LOWER('string')
   * **UPPER**:The purpose of this function is to return the string in uppercase. It takes a string as argument and returns the string by converting it into uppercase.  
     Syntax:
   * UPPER('string')
   * **INITCAP**:The purpose of this function is to return the string with first letter in uppercase and rest of the letters in lowercase.  
     Syntax:
   * INITCAP('string')
3. **What do you mean by data definition language?**  
   Data definition language or DDL allows to execute queries like CREATE, DROP and ALTER. That is, those queries which define the data.

1. **What do you mean by data manipulation language?**  
   Data manipulation Language or DML is used to access or manipulate data in the database.  
   It allows us to perform below listed functions:
   * Insert data or rows in database
   * Delete data from database
   * Retrieve or fetch data
   * Update data in database.
2. **What is the difference between primary key and unique constraints?**  
   Primary key cannot have NULL value, the unique constraints can have NULL values. There is only one primary key in a table, but there can be multiple unique constrains. The primary key creates the cluster index automatically but the Unique key does not.
3. **What is a view in SQL?**  
   Views in SQL are kind of virtual tables. A view also has rows and columns as they are in a real table in the database. We can create a view by selecting fields from one or more tables present in the database. A View can either have all the rows of a table or specific rows based on certain condition.  
   The CREATE VIEW statement of SQL is used for creating Views.  
   Basic Syntax:
4. CREATE VIEW view\_name AS
5. SELECT column1, column2.....
6. FROM table\_name
7. WHERE condition;
8. **view\_name**: Name for the View
9. **table\_name**: Name of the table
10. **condition**: Condition to select rows

For more details on how to create and use view, please refer to [this](https://www.geeksforgeeks.org/sql-views/) article.

1. **What do you mean by foreign key?**  
   A Foreign key is a field which can uniquely identify each row in another table. And this constraint is used to specify a field as Foreign key. That is, this field points to primary key of another table. This usually creates a kind of link between the two tables.  
   Consider the two tables as shown below:

**Orders**

|  |  |  |
| --- | --- | --- |
| **O\_ID** | **ORDER\_NO** | **C\_ID** |
| 1 | 2253 | 3 |
| 2 | 3325 | 3 |
| 3 | 4521 | 2 |
| 4 | 8532 | 1 |

**Customers**

|  |  |  |
| --- | --- | --- |
| **C\_ID** | **NAME** | **ADDRESS** |
| 1 | RAMESH | DELHI |
| 2 | SURESH | NOIDA |
| 3 | DHARMESH | GURGAON |

As we can see clearly that the field C\_ID in Orders table is the primary key in Customers table, i.e. it uniquely identifies each row in the Customers table. Therefore, it is a Foreign Key in Orders table.  
Syntax:

CREATE TABLE Orders

(

O\_ID int NOT NULL,

ORDER\_NO int NOT NULL,

C\_ID int,

PRIMARY KEY (O\_ID),

FOREIGN KEY (C\_ID) REFERENCES Customers(C\_ID)

)

1. **What is a join in SQL? What are the types of joins?**  
   An SQL Join statement is used to combine data or rows from two or more tables based on a common field between them. Different types of Joins are:
   * **INNER JOIN**: The INNER JOIN keyword selects all rows from both the tables as long as the condition satisfies. This keyword will create the result-set by combining all rows from both the tables where the condition satisfies i.e value of the common field will be same.
   * **LEFT JOIN**:This join returns all the rows of the table on the left side of the join and matching rows for the table on the right side of join. The rows for which there is no matching row on right side, the result-set will contain null. LEFT JOIN is also known as LEFT OUTER JOIN
   * **RIGHT JOIN**:RIGHT JOIN is similar to LEFT JOIN. This join returns all the rows of the table on the right side of the join and matching rows for the table on the left side of join. The rows for which there is no matching row on left side, the result-set will contain null. RIGHT JOIN is also known as RIGHT OUTER JOIN.
   * **FULL JOIN**: FULL JOIN creates the result-set by combining result of both LEFT JOIN and RIGHT JOIN. The result-set will contain all the rows from both the tables. The rows for which there is no matching, the result-set will contain NULL values.

To know about each of these joins in details, refer [this](https://www.geeksforgeeks.org/sql-join-set-1-inner-left-right-and-full-joins/).

1. **What is an index?**  
   A database index is a data structure that improves the speed of data retrieval operations on a database table at the cost of additional writes and the use of more storage space to maintain the extra copy of data. Data can be stored only in one order on disk. To support faster access according to different values, faster search like binary search for different values is desired. For this purpose, indexes are created on tables.

SQL Interview Questions | Set 1

**Que-1:** Difference between blocking and deadlocking.

1. **Blocking:**  
   Blocking occurs is a transaction tries to acquire an incompatible lock on a resource that another transaction has already locked. The blocked transaction remain blocked until the blocking transaction releases the lock.
2. **Deadlocking:**  
   Deadlocking occurs when two or more transactions have a resource locked, and each transaction requests a lock on the resource that another transaction has already locked. Neither of the transactions here can more forward, as each one is waiting for the other to release the lock.

**Que-2:** Delete duplicate data from table only first data remains constant.

**Managers –**

|  |  |  |
| --- | --- | --- |
| **ID** | **NAME** | **SALARY** |
| 1 | Harpreet | 20000 |
| 2 | Ravi | 30000 |
| 3 | Vinay | 10000 |
| 4 | Ravi | 30000 |
| 5 | Harpreet | 20000 |
| 6 | Vinay | 10000 |
| 7 | Rajeev | 40000 |
| 8 | Vinay | 10000 |
| 9 | Ravi | 30000 |
| 10 | Sanjay | 50000 |

**Query –**

DELETE M1

From managers M1, managers M2

Where M2.Name = M1.Name AND M1.Id>M2.Id;

**Output –**

|  |  |  |
| --- | --- | --- |
| **ID** | **NAME** | **SALARY** |
| 1 | Harpreet | 20000 |
| 2 | Ravi | 30000 |
| 3 | Vinay | 10000 |
|  |  |  |
| 7 | Rajeev | 40000 |
| 10 | Sanjay | 50000 |

**Que-3:** Find the Name of Employees.

Finding the name of Employees where First Name, Second Name and Last Name is given in table. Some Name is missing such as First Name, Second Name and may be Last Name. Here we will use COALESCE() function which will return first Non Null values.

**Employees –**

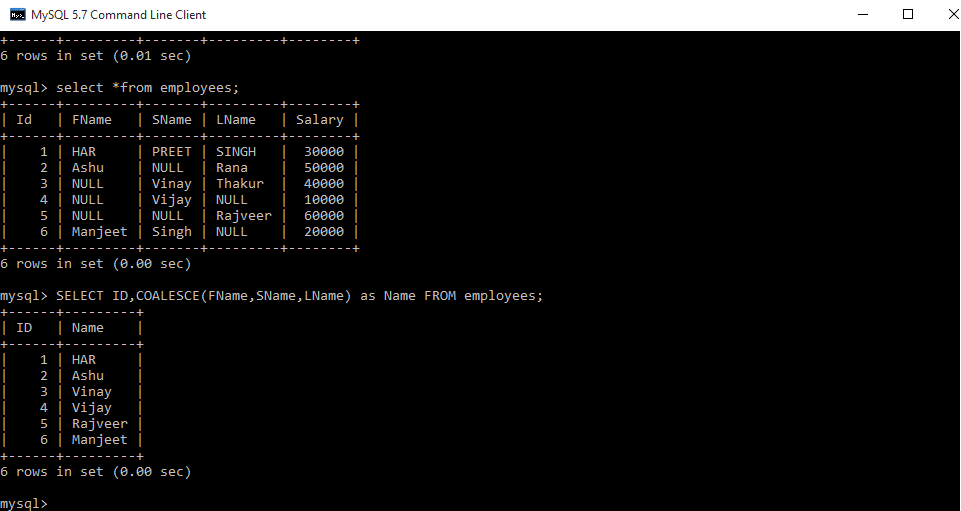
| **ID** | **FNAME** | **SNAME** | **LNAME** | **SALARY** |
| --- | --- | --- | --- | --- |
| 1 | Har | preet | Singh | 30000 |
| 2 | Ashu | NULL | Rana | 50000 |
| 3 | NULL | Vinay | Thakur | 40000 |
| 4 | NULL | Vinay | NULL | 10000 |
| 5 | NULL | NULL | Rajveer | 60000 |
| 6 | Manjeet | Singh | NULL | 60000 |

**Query –**

SELECT ID, COALESCE(FName, SName, LName) as Name

FROM employees;

**Output –**



**Que-4:** Find the Employees who hired in the Last n months.

Finding the Employees who have been hire in the last n months. Here we get desire output by using TIMESTAMPDIFF() mysql function.

**Employees –**

| **ID** | **FNAME** | **LNAME** | **GENDER** | **SALARY** | **HIREDATE** |
| --- | --- | --- | --- | --- | --- |
| 1 | Rajveer | Singh | Male | 30000 | 2017/11/05 |
| 2 | Manveer | Singh | Male | 50000 | 2017/11/05 |
| 3 | Ashutosh | Kumar | Male | 40000 | 2017/12/12 |
| 4 | Ankita | Sharma | Female | 45000 | 2017/12/15 |
| 5 | Vijay | Kumar | Male | 50000 | 2018/01/12 |
| 6 | Dilip | Yadav | Male | 25000 | 2018/02/26 |
| 7 | Jayvijay | Singh | Male | 30000 | 2018/02/18 |
| 8 | Reenu | Kumari | Female | 40000 | 2017/09/19 |
| 9 | Ankit | Verma | Male | 25000 | 2018/04/04 |
| 10 | Harpreet | Singh | Male | 50000 | 2017/10/10 |

**Query –**

Select \*, TIMESTAMPDIFF (month, Hiredate, current\_date()) as DiffMonth

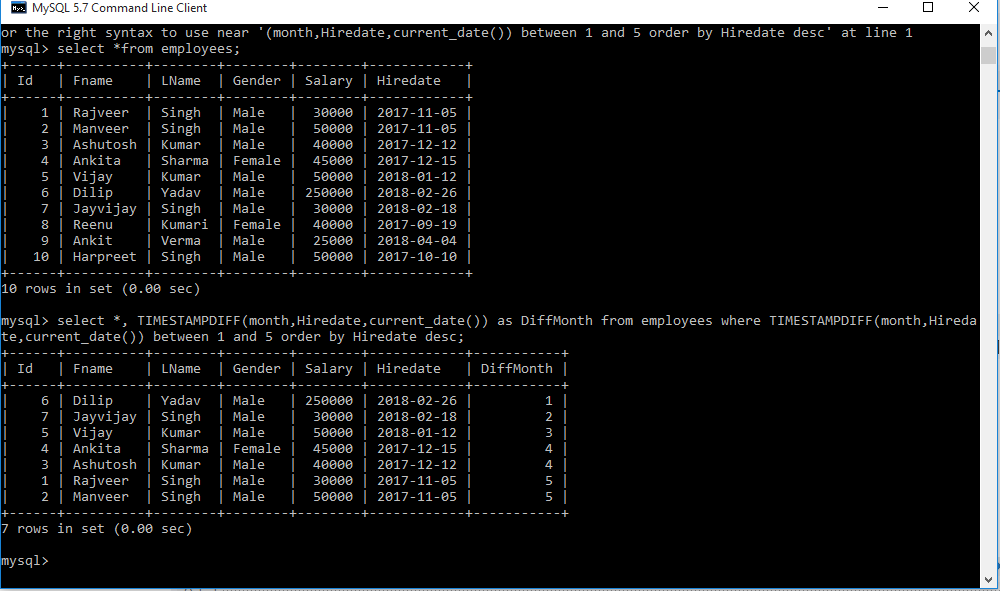
From employees

Where TIMESTAMPDIFF (month, Hiredate, current\_date())

Between 1 and 5 Order by Hiredate desc;

**Note –** Here in query 1 and 5 are indicates 1 to n months.which show the Employees who have hired last 1 to 5 months. In this query DiffMonth is a extra column for our understanding which show the Nth months.

**Output –**



**Que-5:** Find the Employees who hired in the Last n days.

Finding the Employees who have been hire in the last n days. Here we get desire output by using DATEDIFF() mysql function.

**Employees –**

| **ID** | **FNAME** | **LNAME** | **GENDER** | **SALARY** | **HIREDATE** |
| --- | --- | --- | --- | --- | --- |
| 1 | Rajveer | Singh | Male | 30000 | 2017/11/05 |
| 2 | Manveer | Singh | Male | 50000 | 2017/11/05 |
| 3 | Ashutosh | Kumar | Male | 40000 | 2017/12/12 |
| 4 | Ankita | Sharma | Female | 45000 | 2017/12/15 |
| 5 | Vijay | Kumar | Male | 50000 | 2018/01/12 |
| 6 | Dilip | Yadav | Male | 25000 | 2018/02/26 |
| 7 | Jayvijay | Singh | Male | 30000 | 2018/02/18 |
| 8 | Reenu | Kumari | Female | 40000 | 2017/09/19 |
| 9 | Ankit | Verma | Male | 25000 | 2018/04/04 |
| 10 | Harpreet | Singh | Male | 50000 | 2017/10/10 |

**Query –**

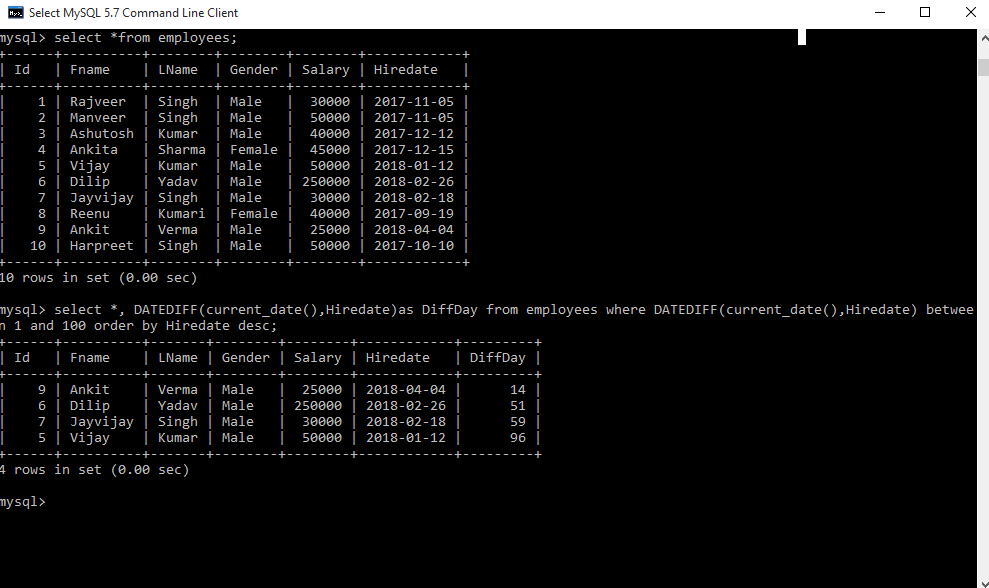
Select \*, DATEDIFF (current\_date(), Hiredate) as DiffDay

From employees

Where DATEDIFF (current\_date(), Hiredate) between 1 and 100 order by Hiredate desc;

**Note –** Here in query 1 and 100 are indicates 1 to n days.which show the Employees who have hired last 1 to 100 days. In this query DiffDay is a extra column for our understanding which show the Nth days.

**Output –**



**Que-6:** Find the Employees who hired in the Last n years.

Finding the Employees who have been hire in the last n years. Here we get desire output by using TIMESTAMPDIFF() mysql function.

**Employees –**

| **ID** | **FNAME** | **LNAME** | **GENDER** | **SALARY** | **HIREDATE** |
| --- | --- | --- | --- | --- | --- |
| 1 | Rajveer | Singh | Male | 30000 | 2010/11/05 |
| 2 | Manveer | Singh | Male | 50000 | 2017/11/05 |
| 3 | Ashutosh | Kumar | Male | 40000 | 2015/12/12 |
| 4 | Ankita | Sharma | Female | 45000 | 2016/12/15 |
| 5 | Vijay | Kumar | Male | 50000 | 2017/01/12 |
| 6 | Dilip | Yadav | Male | 25000 | 2011/02/26 |
| 7 | Jayvijay | Singh | Male | 30000 | 2012/02/18 |
| 8 | Reenu | Kumari | Female | 40000 | 2013/09/19 |
| 9 | Ankit | Verma | Male | 25000 | 2017/04/04 |
| 10 | Harpreet | Singh | Male | 50000 | 2017/10/10 |

**Query –**

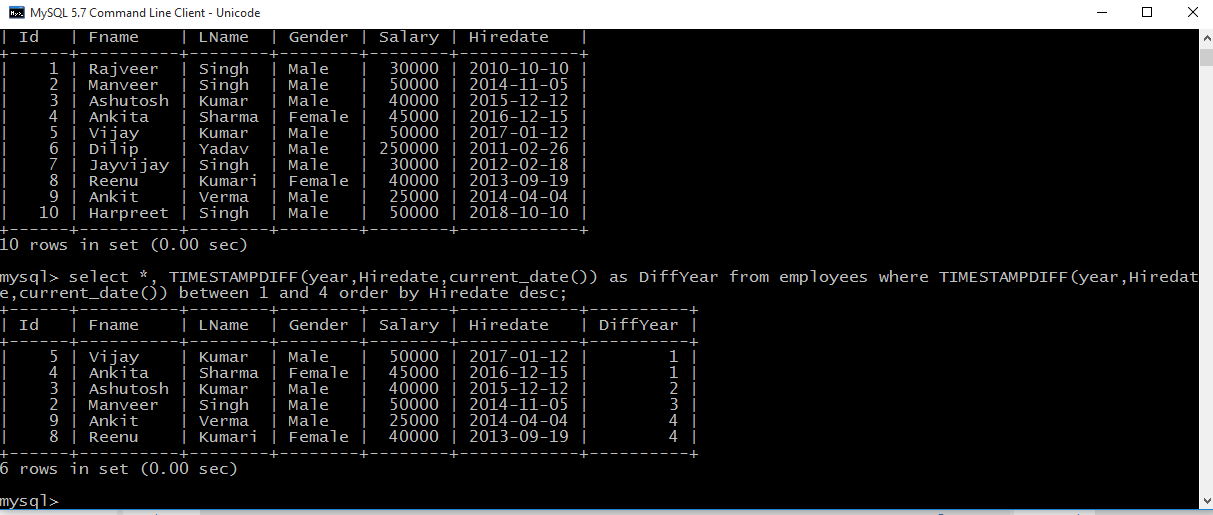
Select \*, TIMESTAMPDIFF (year, Hiredate, current\_date()) as DiffYear

From employees

Where TIMESTAMPDIFF (year, Hiredate, current\_date()) between 1 and 4 order by Hiredate desc;

**Note –** Here in query 1 and 4 are indicates 1 to n years.which show the Employees who have hired last 1 to 4 years. In this query DiffYear is a extra column for our understanding which show the Nth years.

**Output –**



**Que-7:** Select all names that start with a given letter.

Here we get desire output by using three different query.

**Employees –**

| **ID** | **FNAME** | **LNAME** | **GENDER** | **SALARY** | **HIREDATE** |
| --- | --- | --- | --- | --- | --- |
| 1 | Rajveer | Singh | Male | 30000 | 2010/11/05 |
| 2 | Manveer | Singh | Male | 50000 | 2017/11/05 |
| 3 | Ashutosh | Kumar | Male | 40000 | 2015/12/12 |
| 4 | Ankita | Sharma | Female | 45000 | 2016/12/15 |
| 5 | Vijay | Kumar | Male | 50000 | 2017/01/12 |
| 6 | Dilip | Yadav | Male | 25000 | 2011/02/26 |
| 7 | Jayvijay | Singh | Male | 30000 | 2012/02/18 |
| 8 | Reenu | Kumari | Female | 40000 | 2013/09/19 |
| 9 | Ankit | Verma | Male | 25000 | 2017/04/04 |
| 10 | Harpreet | Singh | Male | 50000 | 2017/10/10 |

**Query –**

Select \*

From employees

Where Fname like 'A%';

Select \*

From employees

Where left(FName, 1)='A';

Select \*

From employees

Where substring(FName, 1, 1)='A';

**Note –** Here every query will give same output and the list of Employees who’s FName start with letter A.

# SQL Interview Questions | Set 2

1. **Difference between Locking, Blocking and Deadlocking**
   * **Locking :**Locking occurs when a connection needs access to a piece of data in database and it locks it for certain use so that no other transaction is able to access it.
   * **Blocking :**Blocking occurs when a transaction tries to acquire an incompatible lock on a resource that another transaction has already locked. The blocked transaction remain blocked until the blocking transaction releases the lock.
   * **Deadlocing :**Deadlocking occurs when two or more transactions have a resource locked, and each transaction requests a lock on the resource that another transaction has already locked. Neither of the transactions here can more forward, as each one is waiting for the other to release the lock.
2. **Delete duplicate data from table so that only first data remains constant**

**Managers**

|  |  |  |
| --- | --- | --- |
| **ID** | **NAME** | **SALARY** |
| 1 | Harpreet | 20000 |
| 2 | Ravi | 30000 |
| 3 | Vinay | 10000 |
| 4 | Ravi | 30000 |
| 5 | Harpreet | 20000 |
| 6 | Vinay | 10000 |
| 7 | Rajeev | 40000 |
| 8 | Vinay | 10000 |
| 9 | Ravi | 30000 |
| 10 | Sanjay | 50000 |

**Query:**

DELETE M1 from managers M1, managers M2 where M2.Name=M1.Name AND M1.Id>M2.Id;

**Output:**

|  |  |  |
| --- | --- | --- |
| **ID** | **NAME** | **SALARY** |
| 1 | Harpreet | 20000 |
| 2 | Ravi | 30000 |
| 3 | Vinay | 10000 |
|  |  |  |
| 7 | Rajeev | 40000 |
| 10 | Sanjay | 50000 |

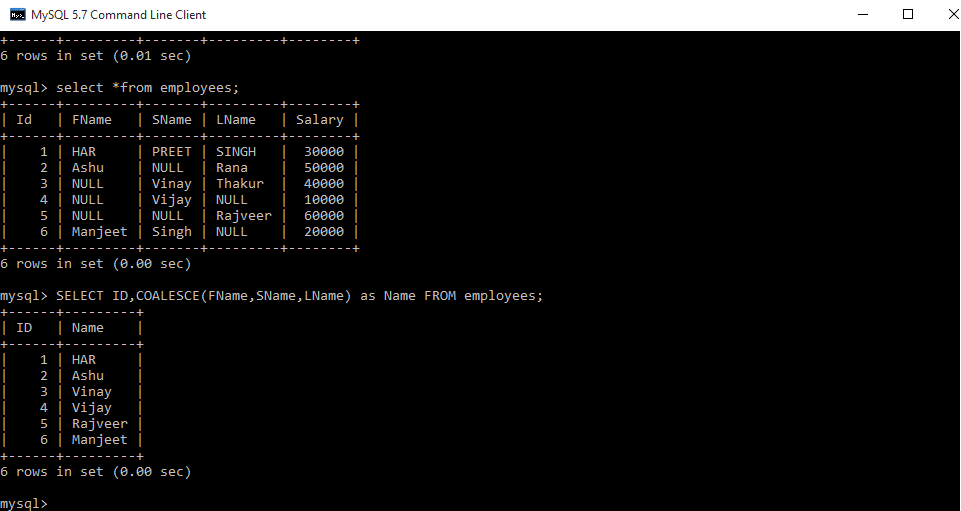
1. **Find the Name of Employees where First Name, Second Name and Last Name is given in table. Some Name is missing such as First Name, Second Name and may be Last Name. Here we will use**[**COALESCE()**](https://www.geeksforgeeks.org/sql-general-functions-nvl-nvl2-decode-coalesce-nullif-lnnvl-nanvl/)**function which will return first Non Null values.**

**Employees**

| **ID** | **FNAME** | **SNAME** | **LNAME** | **SALARY** |
| --- | --- | --- | --- | --- |
| 1 | Har | preet | Singh | 30000 |
| 2 | Ashu | NULL | Rana | 50000 |
| 3 | NULL | Vinay | Thakur | 40000 |
| 4 | NULL | Vinay | NULL | 10000 |
| 5 | NULL | NULL | Rajveer | 60000 |
| 6 | Manjeet | Singh | NULL | 60000 |

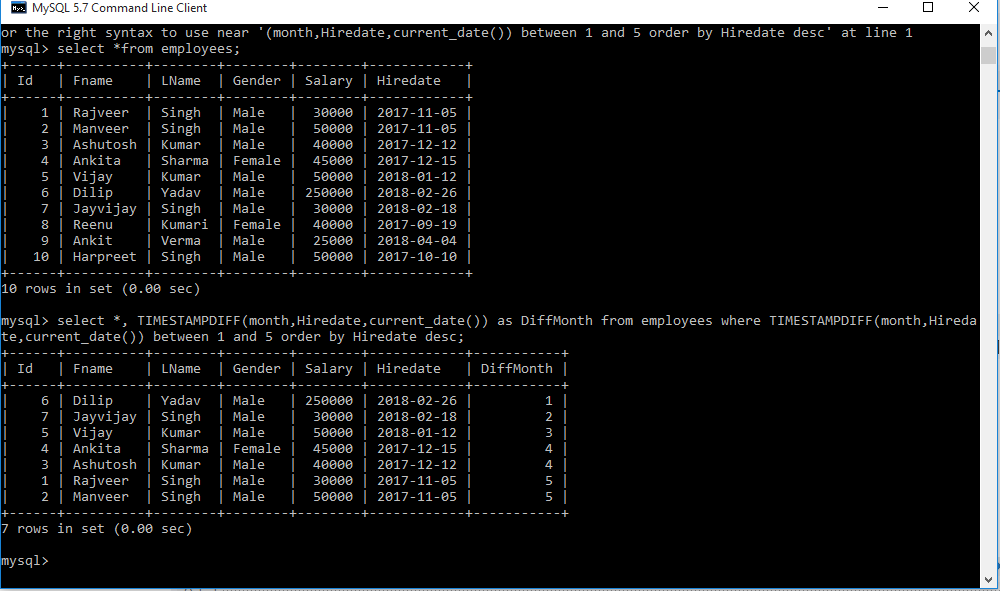
**Query :**

SELECT ID, COALESCE(FName, SName, LName) as Name FROM employees;

**Output:**  


1. **Find the Emloyees who were hired in the Last n months**  
   Finding the Employees who have been hire in the last n months. Here we get desired output by using TIMESTAMPDIFF() mysql function  
   **Employees**

| **ID** | **FNAME** | **LNAME** | **GENDER** | **SALARY** | **HIREDATE** |
| --- | --- | --- | --- | --- | --- |
| 1 | Rajveer | Singh | Male | 30000 | 2017/11/05 |
| 2 | Manveer | Singh | Male | 50000 | 2017/11/05 |
| 3 | Ashutosh | Kumar | Male | 40000 | 2017/12/12 |
| 4 | Ankita | Sharma | Female | 45000 | 2017/12/15 |
| 5 | Vijay | Kumar | Male | 50000 | 2018/01/12 |
| 6 | Dilip | Yadav | Male | 25000 | 2018/02/26 |
| 7 | Jayvijay | Singh | Male | 30000 | 2018/02/18 |
| 8 | Reenu | Kumari | Female | 40000 | 2017/09/19 |
| 9 | Ankit | Verma | Male | 25000 | 2018/04/04 |
| 10 | Harpreet | Singh | Male | 50000 | 2017/10/10 |

1. **Query:**
2. Select \*, TIMESTAMPDIFF(month, Hiredate, current\_date()) as
3. DiffMonth from employees
4. where TIMESTAMPDIFF(month, Hiredate, current\_date()) between
5. 1 and 5 order by Hiredate desc;
6. **Note:**Here in query 1 and 5 are indicates 1 to n months which show the Employees who have hired last 1 to 5 months. In this query, DiffMonth is a extra column for our understanding which show the Nth months.
7. **Output:**  
   
8. **Find the Emloyees who hired in the Last n days**  
   **Finding the Employees who have been hired in the last n days. Here we get desired output by using**[**DATEDIFF()**](https://www.geeksforgeeks.org/sql-date-functions/)**mysql function**

**Employees**

| **ID** | **FNAME** | **LNAME** | **GENDER** | **SALARY** | **HIREDATE** |
| --- | --- | --- | --- | --- | --- |
| 1 | Rajveer | Singh | Male | 30000 | 2017/11/05 |
| 2 | Manveer | Singh | Male | 50000 | 2017/11/05 |
| 3 | Ashutosh | Kumar | Male | 40000 | 2017/12/12 |
| 4 | Ankita | Sharma | Female | 45000 | 2017/12/15 |
| 5 | Vijay | Kumar | Male | 50000 | 2018/01/12 |
| 6 | Dilip | Yadav | Male | 25000 | 2018/02/26 |
| 7 | Jayvijay | Singh | Male | 30000 | 2018/02/18 |
| 8 | Reenu | Kumari | Female | 40000 | 2017/09/19 |
| 9 | Ankit | Verma | Male | 25000 | 2018/04/04 |
| 10 | Harpreet | Singh | Male | 50000 | 2017/10/10 |

**Query:**

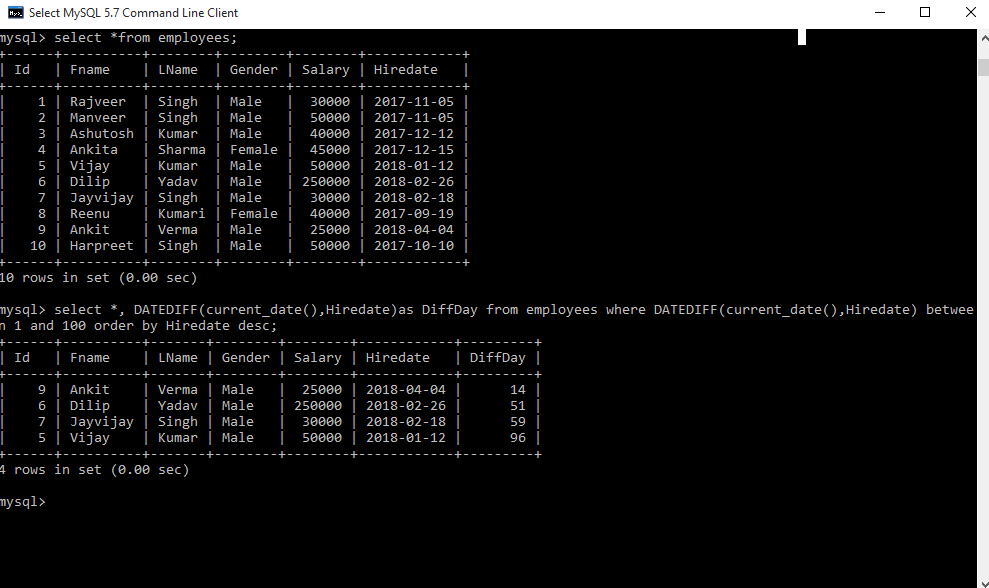
select \*, DATEDIFF(current\_date(), Hiredate)as

DiffDay from employees

where DATEDIFF(current\_date(), Hiredate) between

1 and 100 order by Hiredate desc;

**Note :** Here in query 1 and 100 are indicate 1 to n days which show the Employees who have hired last 1 to 100 days.In this query DiffDay is a extra column for our understanding which show the Nth days.

**Output:**  


1. **Find the Employees who were hired in the Last n years**  
   **Finding the Employees who have been hire in the last n years. Here we get desired output by using TIMESTAMPDIFF() mysql function**

**Employees**

| **ID** | **FNAME** | **LNAME** | **GENDER** | **SALARY** | **HIREDATE** |
| --- | --- | --- | --- | --- | --- |
| 1 | Rajveer | Singh | Male | 30000 | 2010/11/05 |
| 2 | Manveer | Singh | Male | 50000 | 2017/11/05 |
| 3 | Ashutosh | Kumar | Male | 40000 | 2015/12/12 |
| 4 | Ankita | Sharma | Female | 45000 | 2016/12/15 |
| 5 | Vijay | Kumar | Male | 50000 | 2017/01/12 |
| 6 | Dilip | Yadav | Male | 25000 | 2011/02/26 |
| 7 | Jayvijay | Singh | Male | 30000 | 2012/02/18 |
| 8 | Reenu | Kumari | Female | 40000 | 2013/09/19 |
| 9 | Ankit | Verma | Male | 25000 | 2017/04/04 |
| 10 | Harpreet | Singh | Male | 50000 | 2017/10/10 |

**Query:**

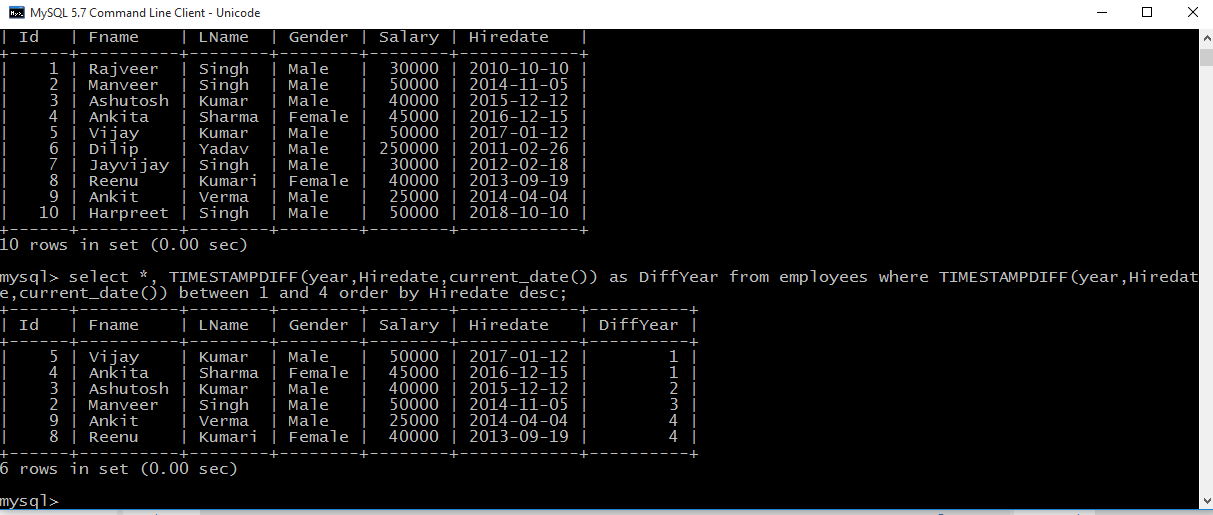
select \*, TIMESTAMPDIFF(year, Hiredate, current\_date()) as

DiffYear from employees

where TIMESTAMPDIFF(year, Hiredate, current\_date())

between 1 and 4 order by Hiredate desc;

**Note:** Here in query 1 and 4 are indicates 1 to n years which shows the Employees who have hired last 1 to 4 years. In this query, DiffYear is a extra column for our understanding which show the Nth years.

**Output:**  


1. **Select all names that start with a given letter**  
   **Here we get desired output by using three different queries**

**Employees**

| **ID** | **FNAME** | **LNAME** | **GENDER** | **SALARY** | **HIREDATE** |
| --- | --- | --- | --- | --- | --- |
| 1 | Rajveer | Singh | Male | 30000 | 2010/11/05 |
| 2 | Manveer | Singh | Male | 50000 | 2017/11/05 |
| 3 | Ashutosh | Kumar | Male | 40000 | 2015/12/12 |
| 4 | Ankita | Sharma | Female | 45000 | 2016/12/15 |
| 5 | Vijay | Kumar | Male | 50000 | 2017/01/12 |
| 6 | Dilip | Yadav | Male | 25000 | 2011/02/26 |
| 7 | Jayvijay | Singh | Male | 30000 | 2012/02/18 |
| 8 | Reenu | Kumari | Female | 40000 | 2013/09/19 |
| 9 | Ankit | Verma | Male | 25000 | 2017/04/04 |
| 10 | Harpreet | Singh | Male | 50000 | 2017/10/10 |

**Query:**

select \*from employees where Fname like 'A%';

select \*from employees where left(FName, 1)='A';

select \*from employees where substring(FName, 1, 1)='A';

**Note:** Here every query will give same output and the list of Employees who’s FName start with letter A.