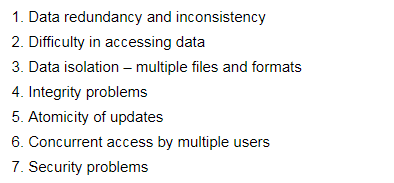
**What are advantages of DBMS over traditional file based systems?**  
  
Source: <http://cs.nyu.edu/courses/spring01/G22.2433-001/mod1.2.pdf>

**What are super, primary, candidate and foreign keys?**  
**Ans:**A [superkey](http://en.wikipedia.org/wiki/Superkey" \t "_blank)is a set of attributes of a relation schema upon which all attributes of the schema are functionally dependent. No two rows can have the same value of super key attributes.  
A [Candidate key](http://en.wikipedia.org/wiki/Candidate_key) is minimal superkey, i.e., no proper subset of Candidate key attributes can be a superkey.  
A [Primary Key](http://publib.boulder.ibm.com/infocenter/db2luw/v8/index.jsp?topic=/com.ibm.db2.udb.doc/admin/c0004799.htm) is one of the candidate keys. One of the candidate keys is selected as most important and becomes the primary key. There cannot be more that one primary keys in a table.  
[Foreign key](http://en.wikipedia.org/wiki/Foreign_key) is a field (or collection of fields) in one table that uniquely identifies a row of another table. See [this](http://www.w3schools.com/sql/sql_foreignkey.asp)for an example.

**What is the difference between primary key and unique constraints?**  
**Ans:** 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.

**What is database normalization?**  
**Ans:**It is a process of analyzing the given relation schemas based on their functional dependencies and primary keys to achieve the following desirable properties:  
1) Minimizing Redundancy  
2) Minimizing the Insertion, Deletion, And Update Anomalies  
Relation schemas that do not meet the properties are decomposed into smaller relation schemas that could meet desirable properties.  
Source: <http://cs.tsu.edu/ghemri/CS346/ClassNotes/Normalization.pdf>

**What is SQL?**  
SQL is Structured Query Language designed for inserting and modifying in a [relational database system](http://en.wikipedia.org/wiki/Relational_database_management_system).

**What are the differences between DDL, DML and DCL in SQL?  
Ans:** Following are some details of three.  
***DDL***stands for Data Definition Language. SQL queries like CREATE, ALTER, DROP and RENAME come under this.  
***DML***stands for Data Manipulation Language. SQL queries like SELECT, INSERT and UPDATE come under this.  
***DCL*** stands for Data Control Language. SQL queries like GRANT and REVOKE come under this.

**What is the difference between having and where clause?**  
**Ans:** HAVING is used to specify a condition for a group or an aggregate function used in select statement. The WHERE clause selects before grouping. The HAVING clause selects rows after grouping. Unlike HAVING clause, the WHERE clause cannot contain aggregate functions. (See [this](http://newtonapples.com/difference-clause-clause/)for examples).  
See [Having vs Where Clause?](http://quiz.geeksforgeeks.org/having-vs-where-clause/) for more details

**How to print duplicate rows in a table?**  
**Ans:**See <http://quiz.geeksforgeeks.org/how-to-print-duplicate-rows-in-a-table/>

**What is Join?**  
**Ans:** An SQL Join is used to combine data from two or more tables, based on a common field between them. For example, consider the following two tables.

Student Table

|  |  |  |
| --- | --- | --- |
| ***ENROLLNO*** | ***STUDENTNAME*** | ***ADDRESS*** |
| 1000 | geek1 | geeksquiz1 |
| 1001 | geek2 | geeksquiz2 |
| 1002 | geek3 | geeksquiz3 |

StudentCourse Table

|  |  |
| --- | --- |
| ***COURSEID*** | ***ENROLLNO*** |
| 1 | 1000 |
| 2 | 1000 |
| 3 | 1000 |
| 1 | 1002 |
| 2 | 1003 |

Following is join query that shows names of students enrolled in different courseIDs.

SELECT StudentCourse.CourseID, Student.StudentName

FROM StudentCourse

INNER JOIN Customers

ON StudentCourse.EnrollNo = Student.EnrollNo

ORDER BY StudentCourse.CourseID;

The above query would produce following result.

|  |  |
| --- | --- |
| ***COURSEID*** | ***STUDENTNAME*** |
| 1 | geek1 |
| 1 | geek2 |
| 2 | geek1 |
| 2 | geek3 |
| 3 | geek1 |

**What is Identity?**  
**Ans:** Identity (or AutoNumber) is a column that automatically generates numeric values. A start and increment value can be set, but most DBA leave these at 1. A GUID column also generates numbers; the value of this cannot be controlled. Identity/GUID columns do not need to be indexed.

**What is a view in SQL? How to create one**  
**Ans:** A [view](http://en.wikipedia.org/wiki/View_(SQL))is a virtual table based on the result-set of an SQL statement. We can create using create view syntax.

CREATE VIEW view\_name AS

SELECT column\_name(s)

FROM table\_name

WHERE condition

**What are the uses of view?**  
**1.** Views can represent a subset of the data contained in a table; consequently, a view can limit the degree of exposure of the underlying tables to the outer world: a given user may have permission to query the view, while denied access to the rest of the base table.  
**2.** Views can join and simplify multiple tables into a single virtual table  
**3.** Views can act as aggregated tables, where the database engine aggregates data (sum, average etc.) and presents the calculated results as part of the data  
**4.** Views can hide the complexity of data; for example a view could appear as Sales2000 or Sales2001, transparently partitioning the actual underlying table  
**5.** Views take very little space to store; the database contains only the definition of a view, not a copy of all the data which it presentsv.  
**6.** Depending on the SQL engine used, views can provide extra security  
Source: [Wiki Page](http://en.wikipedia.org/wiki/View_(SQL))

**What is a Trigger?**  
**Ans:**A [Trigger](http://en.wikipedia.org/wiki/Database_trigger) is a code that associated with insert, update or delete operations. The code is executed automatically whenever the associated query is executed on a table. Triggers can be useful to maintain integrity in database.

**What is a stored procedure?**  
**Ans:** A [stored procedure](http://en.wikipedia.org/wiki/Stored_procedure) is like a function that contains a set of operations compiled together. It contains a set of operations that are commonly used in an application to do some common database tasks.

**What is the difference between Trigger and Stored Procedure?**  
**Ans:** Unlike Stored Procedures, Triggers cannot be called directly. They can only be associated with queries.

**What is a transaction? What are ACID properties?**  
**Ans:** A [Database Transaction](http://en.wikipedia.org/wiki/Database_transaction) is a set of database operations that must be treated as whole, means either all operations are executed or none of them.  
An example can be bank transaction from one account to another account. Either both debit and credit operations must be executed or none of them.  
[ACID](http://en.wikipedia.org/wiki/ACID)(Atomicity, Consistency, Isolation, Durability) is a set of properties that guarantee that database transactions are processed reliably.

**What are indexes?**  
**Ans:** A [database index](http://en.wikipedia.org/wiki/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. These indexes need extra space on disk, but they allow faster search according to different frequently searched values.

**What are clustered and non-clustered Indexes?**  
**Ans:** Clustered indexes is the index according to which data is physically stored on disk. Therefore, only one clustered index can be created on a given database table.  
Non-clustered indexes don’t define physical ordering of data, but logical ordering. Typically, a tree is created whose leaf point to disk records. [B-Tree](http://en.wikipedia.org/wiki/B-tree) or [B+ tree](http://en.wikipedia.org/wiki/B+_tree) are used for this purpos

**Q. There is a table where only one row is fully repeated. Write a Query to find the Repeated row**

| **Name** | **Section** |
| --- | --- |
| abc | CS1 |
| bcd | CS2 |
| abc | CS1 |

In the above table, we can find duplicate row using below query.

SELECT name, section FROM tbl

GROUP BY name, section

HAVING COUNT(\*) > 1

**Q. Query to find 2nd highest salary of an employee?**

SELECT max(salary) FROM EMPLOYEES WHERE salary IN

(SELECT salary FROM EMPLOYEEs MINUS SELECT max(salary)

FROM EMPLOYEES);

OR

SELECT max(salary) FROM EMPLOYEES WHERE

salary <> (SELECT max(salary) FROM EMPLOYEES);

**Q.Consider the following Employee table. How many rows are there in the result of following query?**

**ID   salary   DeptName**  
1    10000      EC  
2    40000      EC  
3    30000      CS  
4    40000      ME  
5    50000      ME  
6    60000      ME  
7    70000      CS

**How many rows are there in the result of following query?**

SELECT E.ID

FROM  Employee E

WHERE  EXISTS  (SELECT E2.salary

FROM Employee E2

WHERE E2.DeptName = 'CS'

AND   E.salary > E2.salary)

Following 5 rows will be result of query as 3000 is the minimum salary of CS Employees and all these rows are greater than 30000.

2  
4  
5  
6  
7

**Q. Write a trigger to update Emp table such that, If an updation is done in Dep table then salary of all employees of that department should be incremented by some amount (updation)**

Assuming Table name are Dept and Emp, trigger can be written as –

CREATE OR REPLACE TRIGGER update\_trig

AFTER UPDATE ON Dept

FOR EACH ROW

DECLARE

CURSOR emp\_cur IS SELECT \* FROM Emp;

BEGIN

FOR i IN emp\_cur LOOP

IF i.dept\_no = :NEW.dept\_no THEN

DBMS\_OUTPUT.PUT\_LINE(i.emp\_no);  --  for printing those

UPDATE Emp                      -- emp number which are

SET sal = i.sal + 100           -- updated

WHERE emp\_no = i.emp\_no;

END IF;

END LOOP;

END;

**Q. There is a table which contains two column Student and Marks, you need to find all the students, whose marks are greater than average marks i.e. list of above average students.**

SELECT student, marks

FROM table

WHERE marks > SELECT AVG(marks) from table;

**Q.Name the student who has secured third highest marks using sub queries.**

SELECT Emp1.Name

FROM Employee Emp1

WHERE 2 = (SELECT COUNT(DISTINCT(Emp2.Salary))

FROM Employee Emp2

WHERE Emp2.Salary > Emp1.Salary

)

\*LOGIC- Number of people with salary higher than this person will be 2.

**Q. Why we cannot use WHERE clause with aggregate functions like HAVING ?**

The difference between the having and where clause in SQL is that the where clause canNOT be used with aggregates, but the having clause can. Please note : It is not a predefined rule but by and large you’ll see that in a good number of the SQL queries, we use WHERE prior to GROUP BY and HAVING after GROUP BY.

The Where clause acts as a pre filter where as Having as a post filter.

The where clause works on row’s data, not on aggregated data.

Let us consider below table ‘Marks’.

Student       Course      Score

a                c1             40  
a                c2             50  
b                c3             60  
d                c1             70  
e                c2             80

Consider the query

SELECT Student, sum(Score) AS total

FROM Marks

This would select data row by row basis. The having clause works on aggregated data.

For example,  output of below query

SELECT Student, sum(score) AS total FROM Marks

Student     Total  
a             90  
b             60  
d             70  
e             80

When we apply having in above query, we get

SELECT Student, sum(score) AS total

FROM Marks having total > 70

Student     Total  
a             90  
e 80

**Q. Difference between primary key and unique key and why one should use unique key if it allows only one null ?**

Primary key:

Only one in a row(tuple).

Never allows null value(only key field).

Unique key identifier and can not be null and must be unique.

Unique Key:

Can be more than one unique key in one row.

Unique key can have null values(only single null is allowed).

It can be a candidate key.

Unique key can be null and may not be unique.

**Q. What’s the difference between materialized and dynamic view?**

Materialized views

Disk based and are updated periodically based upon the query definition.

A materialized table is created or updated infrequently and it must be synchronized with its associated base tables.

Dynamic views

Virtual only and run the query definition each time they are accessed.

A dynamic view may be created every time that a specific view is requested by the user.

**Q. What is embedded and dynamic SQL?**

Static or Embedded SQL

SQL statements in an application that do not change at runtime and, therefore, can be hard-coded into the application.

Dynamic SQL

SQL statements that are constructed at runtime; for example, the application may allow users to enter their own queries.

|  |  |  |
| --- | --- | --- |
| **S.No.** | **Static (embedded) SQL** | **Dynamic (interactive) SQL** |
| 1. | In static SQL how database will be accessed is predetermined in the embedded SQL statement. | In dynamic SQL, how database will be accessed is determined at run time. |
| 2. | It is more swift and efficient. | It is less swift and efficient. |
| 3. | SQL statements are compiled at compile time. | SQL statements are compiled at run time. |
| 4. | Parsing, validation, optimization, and generation of application plan are done at compile time. | Parsing, validation, optimization, and generation of application plan are done at run time. |
| 5. | It is generally used for situations where data is distributed uniformly. | It is generally used for situations where data is distributed non-uniformly. |
| 6. | EXECUTE IMMEDIATE, EXECUTE and PREPARE statements are not used. | EXECUTE IMMEDIATE, EXECUTE and PREPARE statements are used. |
| 7. | It is less flexible. | It is more flexible. |

Dynamic SQL is a programming technique that enables you to buildSQL statements dynamically at runtime. You can create more general purpose, flexible applications by using dynamic SQL because the full text of a SQL statement may be unknown at compilation.

http://docs.oracle.com/cd/A87860\_01/doc/appdev.817/a76939/adg09dyn.htm

**Q. What is the difference between CHAR and VARCHAR?**

CHAR and VARCHAR are differ in storage and retrieval.

CHAR column length is fixed while VARCHAR length is variable.

The maximum no. of character CHAR data type can hold is 255 character while VARCHAR can hold up to 4000 character.

CHAR is 50% faster than VARCHAR.

CHAR uses static memory allocation while VARCHAR uses dynamic memory allocation.

**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.

**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.

**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.

**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 |

**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,

SELECT \* FROM Students

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.

**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,

**SELECT column1,column2 FROM table\_name WHERE column\_name LIKE pattern;**

**LIKE**: operator name

**pattern**: exact value extracted from the pattern to get related data in

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.

**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.

**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')

**What do you mean by data definition language?**  
Data definition language or DDL allows to execute queries like CREATE , ALTER and DELETE. That is, those queries which define the data.

**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.

**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.

**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:

CREATE VIEW view\_name AS

SELECT column1, column2.....

FROM table\_name

WHERE condition;

**view\_name**: Name for the View

**table\_name**: Name of the table

**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.

**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)

)

**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/).

**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. These indexes need extra space on disk, but they allow faster search according to different frequently searched values.

**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.

**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 |

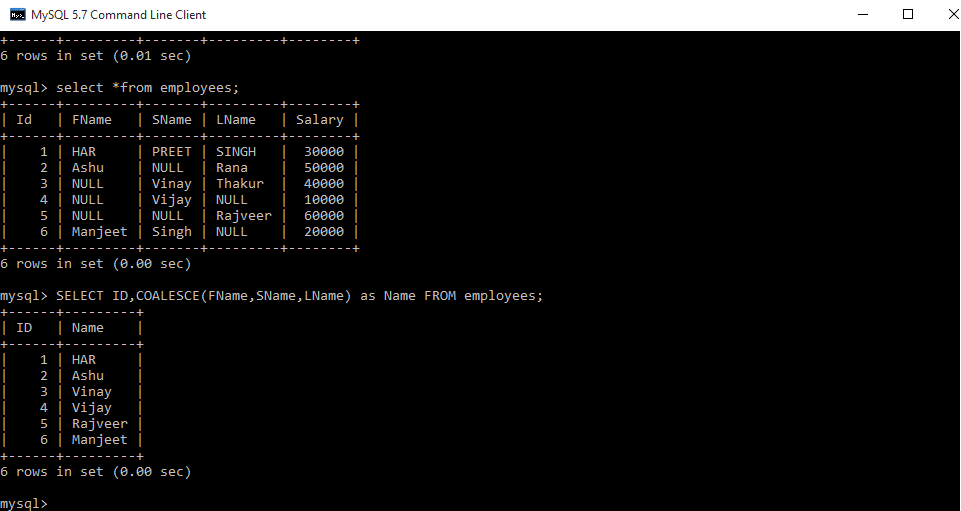
**Find the Name of Employees where First Name, Second Name and Last Name is given in table. Some Name is missing such as First Name, Second Name and may be Last Name. Here we will use**[**COALESCE()**](https://www.geeksforgeeks.org/sql-general-functions-nvl-nvl2-decode-coalesce-nullif-lnnvl-nanvl/)**function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


**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 |

**Query:**

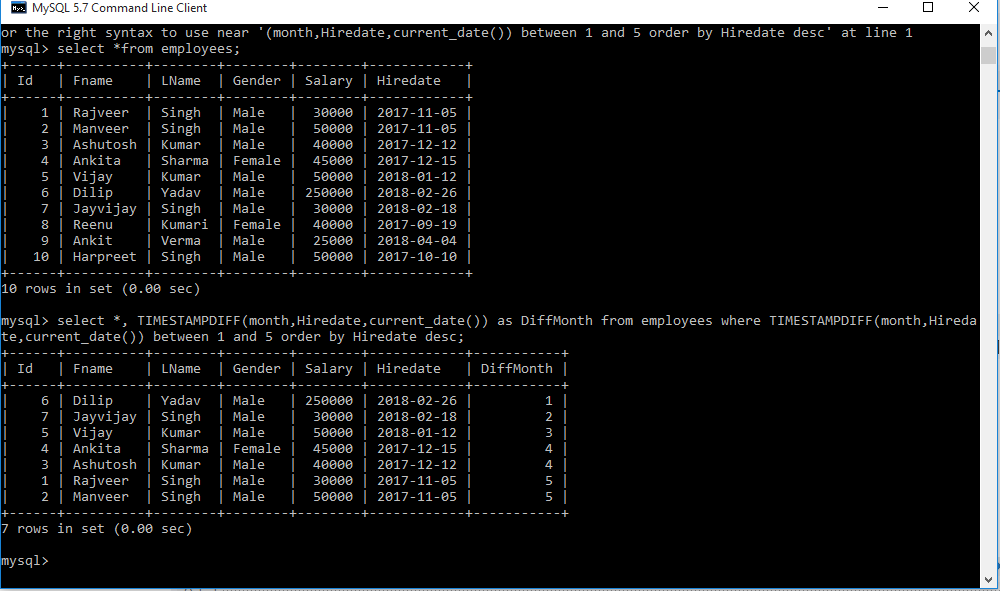
Select \*, TIMESTAMPDIFF(month, Hiredate, current\_date()) as

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:**  


**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:**

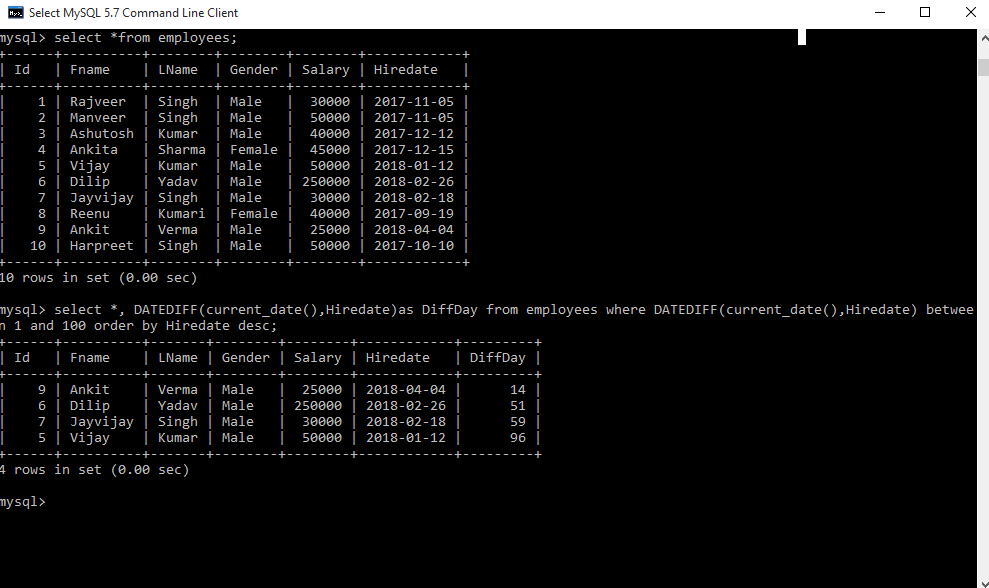
select \*, DATEDIFF(current\_date(), Hiredate)as

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:**  


**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:**

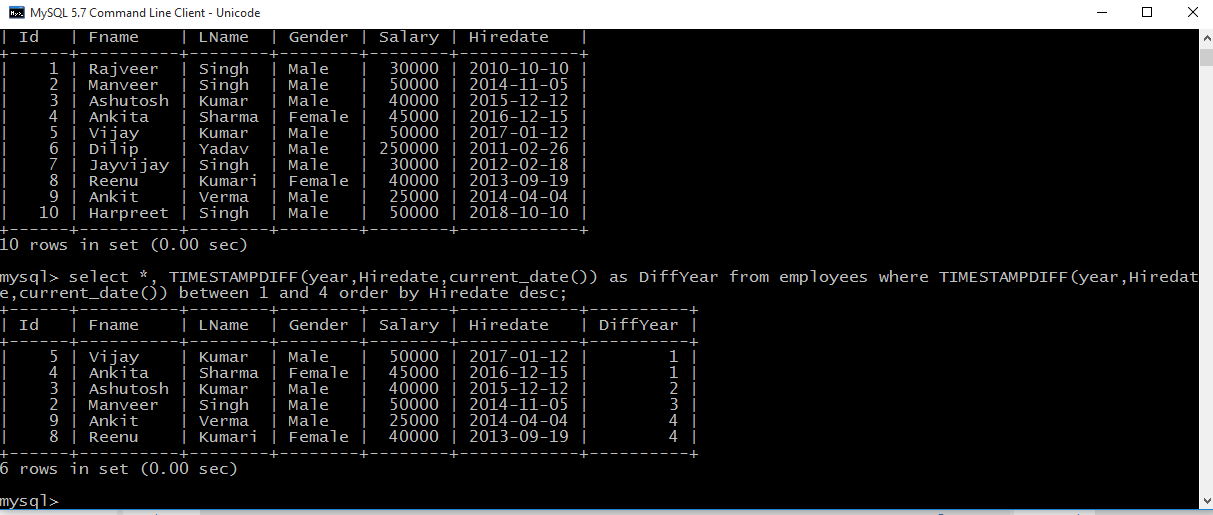
select \*, TIMESTAMPDIFF(year, Hiredate, current\_date()) as

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:**  


**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.