**Explain DML, DDL, DCL and TCL statements with examples?**

**DML**: DML stands for Data Manipulation Language. DML is used to retrieve, store, modify, delete, insert and update data in database.  
**Examples**of DML statements: SELECT, UPDATE, INSERT, DELETE statements.  
  
  
**DDL**: DDL stands for Data Definition Language. DDL is used to create and modify the structure of database objects.  
  
**Examples**: CREATE, ALTER, DROP statements.  
  
  
**DCL**: DCL stands for Data Control Language. DCL is used to create roles, grant and revoke permissions, establish referential integrity etc.  
**Examples**: GRANT, REVOKE statements  
  
  
**TCL**: TCL stands for Transactional Control Language. TCL is used to manage transactions within a database.  
**Examples**: COMMIT, ROLLBACK statements

**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

**Primary key is basically used to identify uniquely each record in the table.**

**Foreign Key:**

* Foreign keys are used to enforce data integrity.
* A FOREIGN KEY is a key used to link two tables together. A FOREIGN KEY in a table is linked with the PRIMARY KEY of another table.
* In layman’s term, a foreign key in one table points to a primary key in another table.
* The foreign key constraint prevents invalid data from being inserted into the foreign key column. The values you enter into the foreign key column, has to be one of the values contained in the table it points to.

**How to find nth highest salary in sql**

1. How to find nth highest salary in SQL Server using a Sub-Query  
2. How to find nth highest salary in SQL Server using a CTE  
3. How to find the 2nd, 3rd or 15th highest salary

1)Using a Sub-Query:

SELECT TOP 1 SALARY

FROM (

SELECT DISTINCT TOP 3 SALARY

FROM EMPLOYEES

ORDER BY SALARY DESC

) RESULT

Order by Salary

1)when we execute the above query, first, the sub query will be executed and stored the result with Name RESULT

|  |
| --- |
| Salary |
| 80000 |
| 70000 |

2)Then we are using Order by clause for the RESULT to arrange the output in Ascending order

|  |
| --- |
| Salary |
| 70000 |
| 80000 |

3)Then the outer query will be executed with the result from the sub query

SELECT TOP 1 SALARY

FROM (

SELECT DISTINCT TOP N SALARY

FROM EMPLOYEES

ORDER BY SALARY DESC

) RESULT

Order by Salary

2)Using CTE(Common table Expression) with Dense\_Rank() keyword:

Dense\_Rank() function is used to give the rank i.e. 1,2,3 to all rows for the column

When we use Dense\_Rank() function we have to specify the result set upon which we want that Dense\_)Rank to be applied so we want the dense rank to be applied over

Select Salary , DENSE\_RANK() over (Order By Salary Desc) from Employees

When we run the above query , we get the result as follows.

Salary (No column name)

80000 1

70000 2

70000 2

60000 3

45000 4

45000 4

35000 5

30000 6

Now we we are going to make this common table expression by using “WITH” keyword and let’s call this common table expression result and give it name as “RESULT”

Then we need to write query for the RESULT Table as follows.

WITH RESULT AS

(

SELECT SALARY,

DENSE\_RANK() OVER (ORDER BY SALARY DESC) AS DENSERANK

FROM EMPLOYEES

)

select top 1 Salary from RESULT WHERE RESULT.DENSERANK = 2

WITH RESULT AS

(

    SELECT SALARY,

           DENSE\_RANK() OVER (ORDER BY SALARY DESC) AS DENSERANK

    FROM EMPLOYEES

)

SELECT TOP 1 SALARY

FROM RESULT

WHERE DENSERANK = N

3)

Please Note: On many of the websites, you may have seen that, the following query can be used to get the nth highest salary. The below query will only work if there are no duplicates.

WITH RESULT AS

(

    SELECT SALARY,

           ROW\_NUMBER() OVER (ORDER BY SALARY DESC) AS ROWNUMBER

    FROM EMPLOYEES

)

SELECT SALARY

FROM RESULT

WHERE ROWNUMBER = 3

**SQL Query to find nth lowest salary**

**SELECT** **TOP** 1 Salary **AS** 'Lowest Salary',**Name** **FROM** (**SELECT** **DISTINCT** **TOP** 3 salary,**Name** **FROM** tblSalary **ORDER** **BY** Salary **ASC**) a **ORDER** **BY** Salary **DESC**

select email from tblPerson

select CHARINDEX('@',Email)+1 from tblPerson -- returns the index of @ in each row for Email column. If you dont use +1 then it will give the index including @.

select LEN(Email) from tblPerson --return the length of the email column for each row

select CHARINDEX('@',Email) from tblPerson -- returns the returns the index of @ in each row for Email column including @

select LEN(Email) - CHARINDEX('@',Email) from tblPerson --

**To Print the Domains Names from the Email column**

--To Print the domain names from the Email column

select SUBSTRING(Email,CHARINDEX('@',Email)+1, LEN(Email) - CHARINDEX('@',Email))

as DomainName

from tblPerson

output :

DomainName

j.com

m.com

s.com

sam.com

may.com

k.com

Ray.com

**To Print the Domains Names with count from the Email column**

--To print the domain names and count of the email column

select SUBSTRING(Email,CHARINDEX('@',Email)+1, LEN(Email) - CHARINDEX('@',Email)) as DomainName,COUNT(Email)

as DomainNameCount from tblPerson

Group by SUBSTRING(Email,CHARINDEX('@',Email)+1, LEN(Email) - CHARINDEX('@',Email))

Output:

DomainName DomainNameCount

j.com 1

k.com 1

m.com 1

may.com 1

Ray.com 1

s.com 1

sam.com 1

**What is the difference between Drop, Delete and Truncate statements in SQL Server?**

The difference between the Delete, Truncate and Drop command is

**Delete** command is a DML command, it is used to delete rows from a table. It can be rolled back.

**Truncate** is a DDL command, it is used to delete all the rows from the table and free the space containing the table. It cant be rolled back.

**Drop** is a DDL command, it removes the complete data along with the table structure(unlike truncate command that removes only the rows). All the tables’ rows, indexes, and privileges will also be removed.

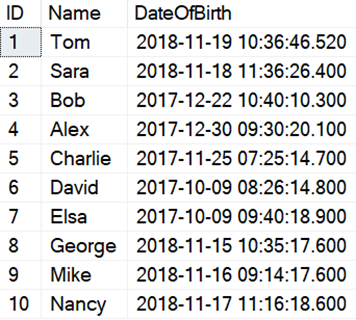


**DIfference between primary key and unique key in SQL Server?**

1. A table can have only one primary key. On the other hand a table can have more than one unique key.  
2. Primary key column does not accept any null values, where as a unique key column accept one null value

Date interview questions.

Sample Table



Here is the SQL Script to create the table and populate it with data

Create Table Employees

(

       ID int identity primary key,

       Name nvarchar(50),

       DateOfBirth DateTime

)

Insert Into Employees Values ('Tom', '2018-11-19 10:36:46.520')

Insert Into Employees Values ('Sara', '2018-11-18 11:36:26.400')

Insert Into Employees Values ('Bob', '2017-12-22 10:40:10.300')

Insert Into Employees Values ('Alex', '2017-12-30 9:30:20.100')

Insert Into Employees Values ('Charlie', '2017-11-25 7:25:14.700')

Insert Into Employees Values ('David', '2017-10-09 8:26:14.800')

Insert Into Employees Values ('Elsa', '2017-10-09 9:40:18.900')

Insert Into Employees Values ('George', '2018-11-15 10:35:17.600')

Insert Into Employees Values ('Mike', '2018-11-16 9:14:17.600')

Insert Into Employees Values ('Nancy', '2018-11-17 11:16:18.600')

**Write a SQL query to retrieve all people who are born on a given date (For example, 9th October 2017)**

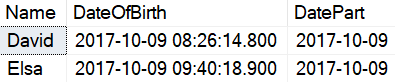
Notice we are using the SQL CAST() method to convert DateTime data type to Date data type which returns just the date part and that is compared with the DATE string in the WHERE clause.

SELECT Name, DateOfBirth, CAST(DateOfBirth AS DATE) AS [DatePart]

FROM     Employees

WHERE  CAST(DateOfBirth AS DATE) = '2017-10-09'

Result:



**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

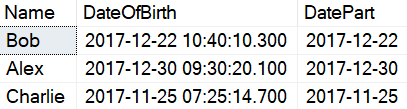
**Write a SQL query to retrieve all people who are born between 2 given dates (For example, all people born between Nov 1, 2017 and Dec 31, 2017)** 

SELECT Name, DateOfBirth, CAST(DateOfBirth AS DATE) AS [DatePart]

FROM     Employees

WHERE  CAST(DateOfBirth AS DATE) BETWEEN '2017-11-01' AND '2017-12-31'

**Result**



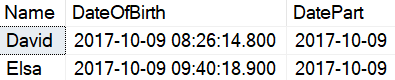
**Write a SQL query to retrieve all people who are born on the same day and month excluding the year (For example, 9th October)** 

SELECT Name, DateOfBirth, CAST(DateOfBirth AS DATE) AS [DatePart]

FROM     Employees

WHERE  DAY(DateOfBirth) = 9 AND Month(DateOfBirth) = 10

**Result**



**Write a SQL query to get all people who are born in a given year (Example, all people born in the year 2017)** 

SELECT Name, DateOfBirth, CAST(DateOfBirth AS DATE) AS [DatePart]

FROM    Employees

WHERE  YEAR(DateOfBirth) = 2017

**To get all people who are born yesterday**

SELECT Name, DateOfBirth, CAST(DateOfBirth AS DATE) AS [DatePart]

FROM    Employees

WHERE CAST(DateOfBirth AS DATE) = DATEADD(DAY, -1, CAST(GETDATE() ASDATE))

To get yesterday's date, we are adding -1 to the DAY part of the DATE. So if today is Nov 10, 2017, the following DATEADD() function will give us Nov 9, 2017

**To get all people who will be born tomorrow, add 1 to the DAY part of the DATE as shown below.**

SELECT Name, DateOfBirth, CAST(DateOfBirth AS DATE) AS [DatePart]

FROM    Employees

WHERE  CAST(DateOfBirth AS DATE) = DATEADD(DAY, 1, CAST(GETDATE() ASDATE))

**To get all people who are born since yesterday (i.e all the people who are born yesterday and today)**

SELECT           Name, DateOfBirth, CAST(DateOfBirth AS DATE) AS [DatePart]

FROM              Employees

WHERE           CAST(DateOfBirth AS DATE)

BETWEEN       DATEADD(DAY, -1, CAST(GETDATE() AS DATE))

AND                 CAST(GETDATE() AS DATE)

**To get all people who are born in the last 7 days (excluding today)**

SELECT         Name, DateOfBirth, CAST(DateOfBirth AS DATE) AS [DatePart]

FROM            Employees

WHERE         CAST(DateOfBirth AS DATE)

BETWEEN     DATEADD(DAY, -7, CAST(GETDATE() AS DATE))

AND               DATEADD(DAY, -1, CAST(GETDATE() AS DATE))

**To get all people who are born today**

SELECT        Name, DateOfBirth, CAST(DateOfBirth AS DATE) AS [DatePart]

FROM            Employees

WHERE         CAST(DateOfBirth AS DATE) = CAST(GETDATE() AS DATE)

**--SQL Query to find department with highest number of employees**

select top 1 DepartmentName

from InterviewEmployees

join InterviewDepartments

on InterviewEmployees.DepartmentID = InterviewDepartments.DepartmentID

Group by DepartmentName

order by COUNT(\*) desc

--SQL Query to to find employees hired in last n months

Select \* from employees

Where DATEDIFF(Month,HIREDATE,getdate()) between 1 and 3

Order by HireDate desc

Here hiredate is column