https://sqlzoo.net/wiki/SELECT\_within\_SELECT\_Tutorial

UNIQUE Constraint prevents identical values in a column from appearing in two records.

CASE

WHEN condition1 THEN result1

WHEN condition2 THEN result2

WHEN conditionN THEN resultN

ELSE result

END;

CREATE TABLE #Employee (id INT, name VARCHAR(25))  
INSERT INTO #Employee VALUES (01, ‘Ashish’), (02, ‘Atul’)

BLOB may be used to store binary data, which includes images, movies, audio, and applications.

DuplicateDelete

DELETE FROM table WHERE ID IN (SELECT ID, COUNT(ID) FROM   table GROUP BY  ID HAVING COUNT (ID) > 1);

ELECT

employee\_id,first\_name, last\_name,

salary,

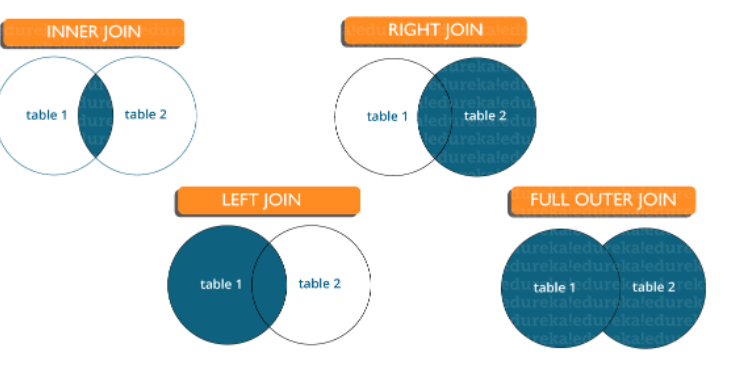
DENSE\_RANK() OVER (

ORDER BY salary DESC

) salary\_rank

FROM

employees;



* Foreign key maintains referential integrity by enforcing a link between the data in two tables.
* The foreign key in the child table references the primary key in the parent table.
* The [foreign key constraint](https://www.edureka.co/blog/foreign-key-sql/) prevents actions that would destroy links between the child and parent tables.
* Denormalization refers to a technique which is used to access data from higher to lower forms of a database. It helps the database managers to increase the performance of the entire infrastructure as it introduces redundancy into a table. It adds the redundant data into a table by incorporating database queries that combine data from various tables into a single table.

[DROP command](https://www.edureka.co/blog/sql-commands) removes a table and it cannot be rolled back from the database whereas TRUNCATE command removes all the rows from the table.

AVG(): returns the average value from specified columns.  
COUNT(): returns the number of table rows, including rows with null values.  
MAX(): returns the largest value among the group.  
MIN(): returns the smallest value among the group.  
SUM(): returns the total summed values(non-null) of the specified column.  
FIRST(): returns the first value of an expression.  
LAST(): returns the last value of an expression.

The DISTINCT clause in SQL is used to eliminate duplicates from a SELECT statement’s result set.

* **Atomicity:**Changes to data are performed as a single, unified operation
* **Consistency:**Data values are consistent at the start and end of the transaction
* **Isolation:**The intermediate state of a transaction is hidden from other transactions
* **Durability:** Changes to data remain the same after the transaction is completed

**List the ways to get the count of records in a table?**

SELECT \* FROM table1

SELECT COUNT(\*) FROM table1

**Write a SQL query to find the names of employees that begin with ‘A’?**

|  |  |
| --- | --- |
| 1 | SELECT \* FROM Table\_name WHERE EmpName like 'A%' |

**Write a SQL query to get the third-highest salary of an employee from employee\_table?**

1. SELECT \* FROM `employees` ORDER BY `salary` DESC LIMIT 1 OFFSET 2

|  |  |
| --- | --- |
| 1  2  3  4  5  6 | SELECT TOP 1 salary  FROM(  SELECT TOP 3 salary  FROM employee\_table  ORDER BY salary DESC) AS emp  ORDER BY salary ASC; |

**Example of BETWEEN:**

SELECT \* FROM Students where ROLL\_NO BETWEEN 10 AND 50;

**Example of IN:**

SELECT \* FROM students where ROLL\_NO IN (8,15,25)

HAVING clause can be used only with SELECT statement. It is usually used in a GROUP BY clause and whenever GROUP BY is not used, HAVING behaves like a WHERE clause.  
Having Clause is only used with the GROUP BY function in a query whereas WHERE Clause is applied to each row before they are a part of the GROUP BY function in a query.

### different set operators available in SQL?

Some of the available set operators are – Union, Intersect or Minus operators.

You can select unique records from a table by using the DISTINCT keyword.

Select DISTINCT studentID from Student

### most important SQL constraints and how are they used?

**Sample answer**:

Here are some of the most commonly used SQL constraints:

* **NOT NULL** ensures a column cannot contain a NULL value
* **UNIQUE**ensures all values in a column are different
* **DEFAULT**provides a default value for a column when none is specified
* **INDEX**creates an index for data retrieval purposes
* **CHECK**checks values in a column against certain specified conditions

**all even or odd numbers in a table?**

**Sample answer**:

The MOD function can be used in most RDBMSs as part of the WHERE statement in a select query to retrieve odd or even data entries in a table.

The formatting is as follows:

* For even numbers, use ‘MOD (*column name*, 2) = 1’
* For odd numbers, use ‘MOD (*column name*, 2) = 0’

To find employees with the same salary, the following solution can be used:

‘WHERE *salary*IN

(SELECT *salary*

FROM *employee*

WHERE *employee.employee\_id* <> *employee.employee\_id*)’

* Using the GROUP BY clause with the COUNT function, and then replacing SELECT with DELETE FROM

SELECT TOP 100 user\_id FROM dbo.users WHERE user\_id % 2 = 1 ORDER BY user\_id

**How do you copy data from one table to another table ?**

Hide answer

INSERT INTO table2 (column1, column2, column3, ...)

SELECT column1, column2, column3, ...

FROM table1

WHERE condition;

**Write a query to add 2 where**Nmbr**is 0 and add 3 where**Nmbr**is 1.**

Hide answer

This can be done as follows:

update TBL set Nmbr = case when Nmbr = 0 then Nmbr+2 else Nmbr+3 end;

SELECT name FROM world

WHERE population >

(SELECT population FROM world

WHERE name='Russia')

SELECT name FROM world

WHERE continent='Europe' AND gdp/population >

(SELECT gdp/population FROM world

WHERE name='United Kingdom')

SELECT DISTINCT salesman\_id FROM orders;

SELECT customer\_id,ord\_date,MAX(purch\_amt)

FROM orders

GROUP BY customer\_id,ord\_date

HAVING MAX(purch\_amt)>2000.00;

From the following tables write a SQL query to find those salespersons who received a commission from the company more than 12%. Return Customer Name, customer city, Salesman, commission.

SELECT a.cust\_name AS "Customer Name",

a.city, b.name AS "Salesman", b.commission

FROM customer a

INNER JOIN salesman b

ON a.salesman\_id=b.salesman\_id

WHERE b.commission>.12;

SELECT a.ord\_no,a.ord\_date,a.purch\_amt,

b.cust\_name AS "Customer Name", b.grade,

c.name AS "Salesman", c.commission

FROM orders a

INNER JOIN customer b

ON a.customer\_id=b.customer\_id

INNER JOIN salesman c

ON a.salesman\_id=c.salesman\_id;

SELECT \* FROM orders NATURAL JOIN customer NATURAL JOIN salesman;

SELECT a.cust\_name,a.city, b.ord\_no, b.ord\_date,b.purch\_amt AS "Order Amount", c.name,c.commission FROM customer a

LEFT OUTER JOIN orders b ON a.customer\_id=b.customer\_id

LEFT OUTER JOIN salesman c ON c.salesman\_id=b.salesman\_id;

1. **SELECT** customer\_id, cust\_name, cellphone, homephone
2. **FROM** customers
3. LEFT JOIN contacts **ON** customer\_id = contact\_id
4. **WHERE** cellphone **IS** NULL ;

**ON** customers.customer\_id = orders.customer\_id  === USING(customer\_id)