

DDL – Data Definition Language

DML – Data Manipulation Language

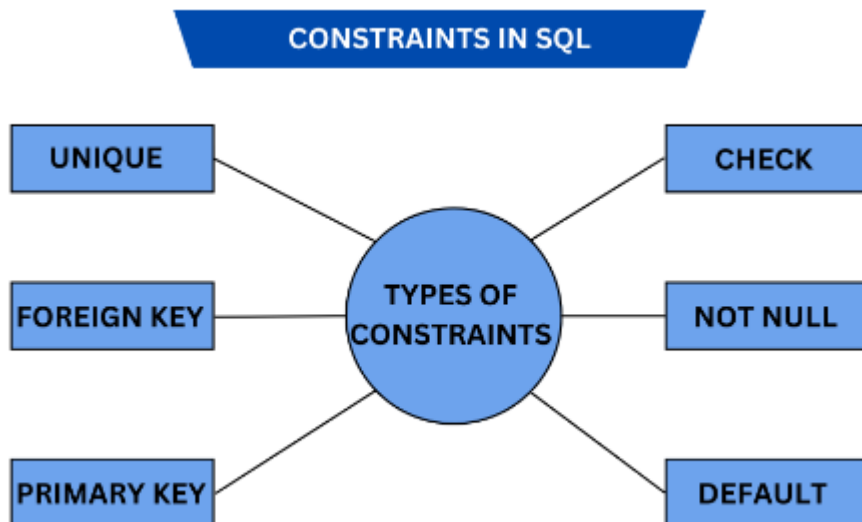
DQL – Data Query Language

DCL – Data Control Language

TCL – Transaction Control Language

Ques – Difference between Drop, delete & truncate

S. No	DELETE	TRUNCATE	DROP
1.	Delete is a DML command.	Truncate is a DDL command.	Drop is a DDL command.
2.	Delete is used to delete the records in the table.	Truncate is used to delete the data and keep the table structure as it is.	Drop is used to drop the table data as well as table structure.
3.	Delete statement use the where clause to delete particular records.	Truncate can't use where clause to delete particulate data.	Drop can't use where clause.
4.	Delete statement can be rollback before the commit.	Truncate statement can't rollback	Drop statement can't rollback.
5.	Delete is slower as compare to Truncate.	Truncate is faster as compare to Delete.	Drop removes the table from the database.
6.	Syntax: - delete from table_name;	Syntax: - Truncate Table table_name ;	Syntax: - Drop Table table_name;



## Different SQL Constraints and Usage

Constraint in SQL	Usage
NOT NULL	When a column shouldn't accept null values, use this.
UNIQUE	Used when every value in the column must be unique.
PRIMARY KEY	Used to provide a distinct identity for each row in a table.
FOREIGN KEY	Used to locate rows or entries kept in a different table.
CHECK	Used to verify that every piece of data in the column complies with a certain requirement.
DEFAULT	Used to establish the column's default value when the user doesn't enter one.
INDEX	Used to quickly obtain and search data from a database.

```
CREATE TABLE BoardInfinity_Emp
(
ID int(6) NOT NULL UNIQUE,
NAME varchar(10),
email varchar(20)
);
```

```
CREATE TABLE BoardInfinity_1
(
ID int NOT NULL,
ORDER_NO int,
C_ID int,
PRIMARY KEY (ID),
FOREIGN KEY (ID) REFERENCES BoardInfinity_2(ID)
)
```

```
CREATE TABLE BoardInfinity_3
(
ID int(6) NOT NULL,
NAME varchar(10) NOT NULL,
AGE int NOT NULL CHECK (AGE >= 18),
Company varchar(20) DEFAULT 'BoardInfinity'
);
```

Ques – Difference between Primary key, Unique key and Foreign key

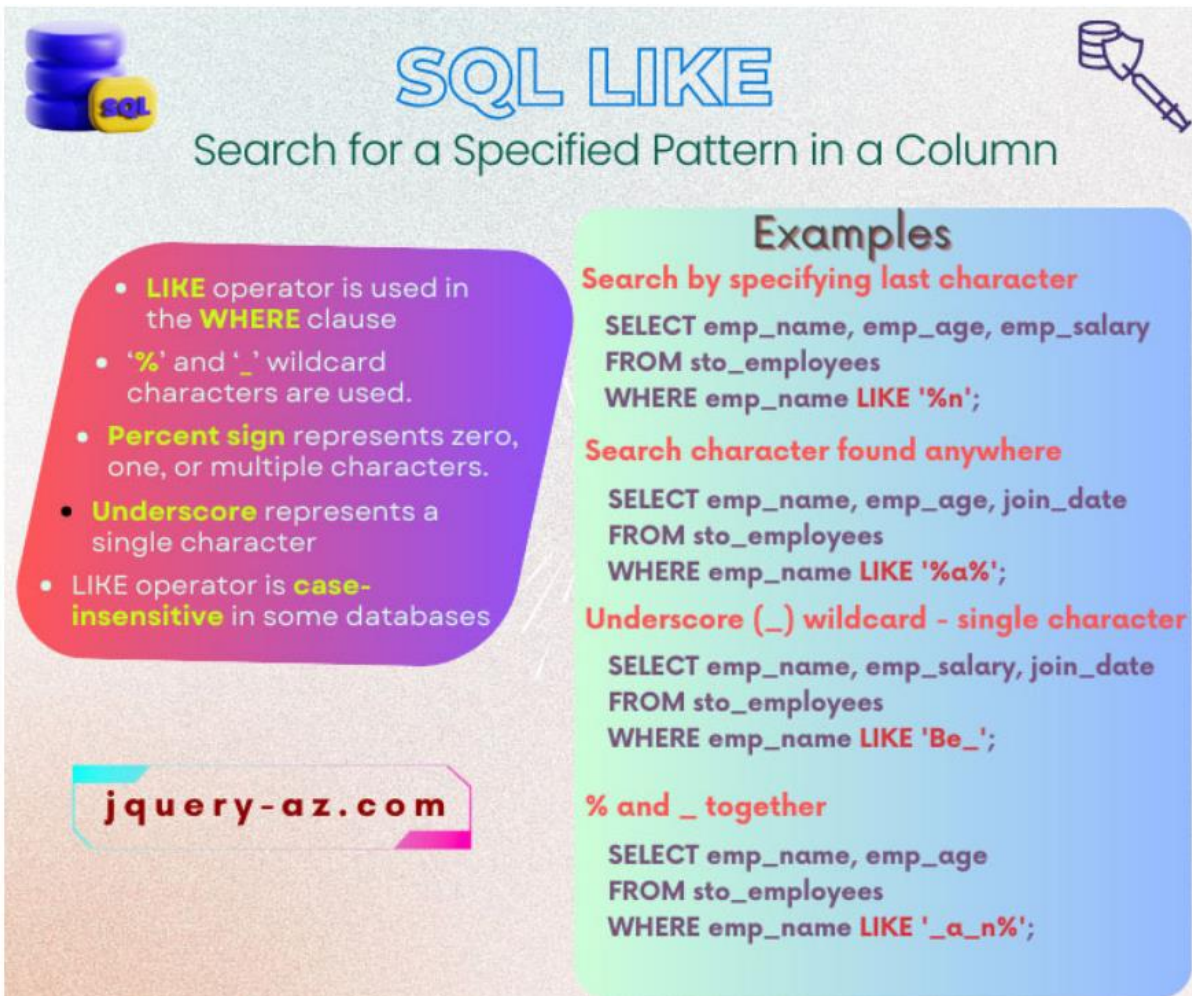
Sno.	PRIMARY KEY	UNIQUE KEY
1	Primary Key can't accept null values.	Unique key can accept only one null value.
2	We can have only one Primary key in a table.	We can have more than one unique key
3	Primary key can be made foreign key into another table.	Unique key can be made foreign key into another table.
4	By default it adds adclustered index	By default it adds a UNIQUE non-clustered index
5	defining a single column as a PRIMARY KEY column while creating a table: <pre>CREATE TABLE Employee1(Id INT NOT NULL PRIMARY KEY,FirstName VARCHAR(100), LastName VARCHAR(100),</pre>	defining a single column as a UNIQUE KEY column while creating a table: <pre>CREATE TABLE Employee1( Id INT NOT NULL UNIQUE,FirstName VARCHAR(100), LastName VARCHAR(100),</pre>
	<pre>City VARCHAR(50) )</pre>	<pre>City VARCHAR(50) )</pre>
6	By the sql command we can set the primry key like below <pre>ALTER TABLE Employee1 ADD Constraint primarykeycons_n ame PRIMARY KEY(ID)</pre>	By the sql command we can set the primry key like below <pre>ALTER TABLE Employee1 ADD Constraint UNIQUEKEY1 UNIQUE(Id)</pre>

	Primary Key	Foreign Key
1	Primary Key <b>Can Not</b> Accept Null Values.	Foreign Key <b>Can Accept</b> Multiple Null Values.
2	Only <b>One</b> Primary Key in a Table.	<b>More than One</b> Foreign Key in a Table.
3	Primary Key <b>Uniquely Identify</b> a Record in the Table	Foreign Key is a Field in the Table that is <b>Primary Key</b> in Another Table
4	Primary Key is <b>Clustered</b> Index. <b>Ex:</b> <pre> CREATE TABLE [country] (     [id] INT     IDENTITY (1, 1) NOT NULL,     [name] VARCHAR (50) NOT     NULL,     UNIQUE NONCLUSTERED ([name],         CONSTRAINT [PK_country]     PRIMARY KEY CLUSTERED ([id])     ); </pre>	Foreign Key is <b>Non-Clustered</b> Index. <b>Ex:</b> <pre> CREATE TABLE [reg] (     [country] VARCHAR (50) NOT     NULL,     [name] VARCHAR (50) NOT     NULL,     CONSTRAINT [FK_reg_country]     FOREIGN KEY ([name]) REFERENCES     [dbo].[country] ([name])     ); </pre>

For clustered Index and Non-clustered Index

<https://www.stratascratch.com/blog/clustered-vs-non-clustered-indexes-in-sql/>





**SQL LIKE**  
Search for a Specified Pattern in a Column

- **LIKE** operator is used in the **WHERE** clause
- **'%'** and **'\_'** wildcard characters are used.
- **Percent sign** represents zero, one, or multiple characters.
- **Underscore** represents a single character
- LIKE operator is **case-insensitive** in some databases

**Examples**

**Search by specifying last character**

```
SELECT emp_name, emp_age, emp_salary
FROM sto_employees
WHERE emp_name LIKE '%n';
```

**Search character found anywhere**

```
SELECT emp_name, emp_age, join_date
FROM sto_employees
WHERE emp_name LIKE '%a%';
```

**Underscore (\_) wildcard - single character**

```
SELECT emp_name, emp_salary, join_date
FROM sto_employees
WHERE emp_name LIKE 'Be_';
```

**% and \_ together**

```
SELECT emp_name, emp_age
FROM sto_employees
WHERE emp_name LIKE '_a_n%';
```

[jquery-az.com](http://jquery-az.com)

/\*LIKE: This operator is case-sensitive. It matches patterns while respecting case differences.

ILIKE: This operator is case-insensitive. It ignores case differences while matching patterns

Note that ILIKE is supported in certain SQL dialects, such as PostgreSQL and Trino, but might not be available in all databases like SQL Server, where only LIKE is supported by default. \*/

--Use of "\_" in like operator.>>"\_" means one character

--second character should be l

```
select Customer_ID,customer_Name from Orders
where customer_name like '_l%'
```

--this character should be a

```
select Customer_ID,customer_Name from Orders
where customer_name like '__a%'
```

--2nd character is **a or l** if we put any alphabet in [] it apply or funtion .

```
select Customer_ID,customer_Name from Orders
```

```
where customer_name like 'c[a]l%'
```

--2nd character is **not a or l** if we put any alphabet in [] it apply or funtion .

```
select Customer_ID,customer_Name from Orders
```

```
where customer_name like 'c[^a]l%'
```

--range a,b,c,d,e,f,g,h,i,j,k,l

```
select Customer_ID,customer_Name from Orders
```

```
where customer_name like 'c[a-l]l%'
```

### Assignment Questions (1-10) on the basis filtering ( where clause)

```
--1. write a sql to get all the orders where customers name has "a" as second character and "d" as fourth character (58 rows)
```

```
select * from Orders where customer_name like '_a_d%'
```

100 %												
Results Messages												
	row_id	order_id	order_date	ship_date	ship_mode	customer_id	customer_name	segment	country	city	state	p
1	24	US-2021-156909	2021-07-16 00:00:00.000	2021-07-18 00:00:00.000	Second Class	SF-20065	Sandra Flanagan	Consumer	United States	Philadelphia	Pennsylvania	1
2	145	CA-2021-155376	2021-12-22 00:00:00.000	2021-12-27 00:00:00.000	Standard Class	SG-20080	Sandra Glassco	Consumer	United States	Independence	Missouri	6
3	4887	CA-2020-163167	2020-11-28 00:00:00.000	2020-12-01 00:00:00.000	Second Class	RF-19345	Randy Ferguson	Corporate	United States	Marietta	Georgia	3
4	4888	CA-2020-163167	2020-11-28 00:00:00.000	2020-12-01 00:00:00.000	Second Class	RF-19345	Randy Ferguson	Corporate	United States	Marietta	Georgia	3
5	4889	CA-2020-163167	2020-11-28 00:00:00.000	2020-12-01 00:00:00.000	Second Class	RF-19345	Randy Ferguson	Corporate	United States	Marietta	Georgia	3

```
--2. write a sql to get all the orders placed in the month of dec 2020 (352 rows)
```

```
select * from Orders
where DATETRUNC(month,order_date) = '2020-12-01'
```

```
select * from orders
where month(order_date)=12 and year(order_date)=2020
```

```
select * from orders
where order_date between '2020-12-01' and '2020-12-31'
```

100 %

	row_id	order_id	order_date	ship_date	ship_mode	customer_id	customer_name	segment	country	city	state
1	14	CA-2020-161389	2020-12-05 00:00:00.000	2020-12-10 00:00:00.000	Standard Class	IM-15070	Irene Maddox	Consumer	United States	Seattle	Washington
2	22	CA-2020-137330	2020-12-09 00:00:00.000	2020-12-13 00:00:00.000	Standard Class	KB-16585	Ken Black	Corporate	United States	Fremont	Nebraska
3	23	CA-2020-137330	2020-12-09 00:00:00.000	2020-12-13 00:00:00.000	Standard Class	KB-16585	Ken Black	Corporate	United States	Fremont	Nebraska
4	36	CA-2020-117590	2020-12-08 00:00:00.000	2020-12-10 00:00:00.000	First Class	GH-14485	Gene Hale	Corporate	United States	Richardson	Texas

--3.write a query to get all the orders where ship\_mode is neither in 'Standard Class' nor in 'First Class' and ship\_date is after nov 2020 (944 rows).

```
select * from orders
where ship_mode not in ('Standard Class','First Class') and ship_date > '2020-11-30'
```

100 % ▾

Results Messages

	row_id	order_id	order_date	ship_date	ship_mode	customer_id	customer_name	segment	country	city	state	postal_code	region	product_id
1	24	US-2021-156909	2021-07-16 00:00:00.000	2021-07-18 00:00:00.000	Second Class	SF-20065	Sandra Flanagan	Consumer	United States	Philadelphia	Pennsylvania	19140	East	FUR-CH-10002774
2	35	CA-2021-107727	2021-10-19 00:00:00.000	2021-10-23 00:00:00.000	Second Class	MA-17560	Matt Abelman	Home Office	United States	Houston	Texas	77095	Central	OFF-PA-10000249
3	72	CA-2021-114440	2021-09-14 00:00:00.000	2021-09-17 00:00:00.000	Second Class	TB-21520	Tracy Blumstein	Consumer	United States	Jackson	Michigan	49201	Central	OFF-PA-10004675
4	86	CA-2021-140088	2021-05-28 00:00:00.000	2021-05-30 00:00:00.000	Second Class	PO-18865	Patrick O'Donnell	Consumer	United States	Columbia	South Carolina	29203	South	FUR-CH-10000863

--4- write a query to get all the orders where customer name neither start with "A" and nor ends with "n" (9815 rows)

```
select * from orders where customer_name not like 'A%n'
```

Results													
row_id	order_id	order_date	ship_date	ship_mode	customer_id	customer_name	segment	country	city	state	postal_code	region	product_id
1	1	CA-2020-152156	2020-11-08 00:00:00.000	2020-11-11 00:00:00.000	Second Class	CG-12520	Claire Gute	Consumer	United States	Henderson	Kentucky	42420	
2	2	CA-2020-152156	2020-11-08 00:00:00.000	2020-11-11 00:00:00.000	Second Class	CG-12520	Claire Gute	Consumer	United States	Henderson	Kentucky	42420	
3	3	CA-2020-138688	2020-06-12 00:00:00.000	2020-06-16 00:00:00.000	Second Class	DV-13045	Darrin Van Huff	Corporate	United States	Los Angeles	California	90036	

--5- write a query to get all the orders where profit is negative (1871 rows)

```
select * from Orders
where profit < 0

select * from orders where profit like '-%'
```

100 %														
Results Messages														
	segment	country	city	state	postal_code	region	product_id	category	sub_category	product_name	sales	quantity	discount	profit
1	Consumer	United States	Fort Lauderdale	Florida	33311	South	FUR-TA-10000577	Furniture	Tables	Bretford CR4500 Series Slim Rectangular Table	957.5775	5	0.45	-383.031
2	Home Office	United States	Fort Worth	Texas	76106	Central	OFF-AP-10002311	Office Supplies	Appliances	Holmes Replacement Filter for HEPA Air Cleaner, Ve...	68.81	5	0.8	-123.858
3	Home Office	United States	Fort Worth	Texas	76106	Central	OFF-BI-10000756	Office Supplies	Binders	Storex DuraTech Recycled Plastic Frosted Binders	2.544	3	0.8	-3.816
4	Consumer	United States	Philadelphia	Pennsylvania	19140	East	FUR-CH-10002774	Furniture	Chairs	Global Deluxe Stacking Chair, Gray	71.372	2	0.3	-1.0196
5	Consumer	United States	Philadelphia	Pennsylvania	19140	East	FUR-BO-10004834	Furniture	Bookcases	Riverside Palais Royal Lawyers Bookcase, Royale C...	3083.43	7	0.5	-1665.0522

--6- write a query to get all the orders where either quantity is less than 3 or profit is 0 (3348)

```
select * from orders where quantity<3 or profit=0
```

Results													
segment	country	city	state	postal_code	region	product_id	category	sub_category	product_name	sales	quantity	discount	profit
Consumer	United States	Henderson	Kentucky	42420	South	FUR-BO-10001798	Furniture	Bookcases	Bush Somerset Collection Bookcase	261.96	2	0	41.9136
Corporate	United States	Los Angeles	California	90036	West	OFF-LA-10000240	Office Supplies	Labels	Self-Adhesive Address Labels for Typewriters by Univ...	14.62	2	0	6.8714
Consumer	United States	Fort Lauderdale	Florida	33311	South	OFF-ST-10000760	Office Supplies	Storage	Eldon Fold 'N Roll Cart System	22.368	2	0.2	2.5164
Consumer	United States	West Jordan	Utah	84084	West	OFF-ST-10000107	Office Supplies	Storage	Fellowes Super StoriDrawer	55.5	2	0	9.989999999999999

--7- your manager handles the sales for South region and he wants you to create a report of all the orders in his region --where some discount is provided to the customers (815 rows)

```
select * from orders where region ='South' and discount > 0
```

Results													
segment	country	city	state	postal_code	region	product_id	category	sub_category	product_name	sales	quantity	discount	profit
Consumer	United States	Fort Lauderdale	Florida	33311	South	FUR-TA-10000577	Furniture	Tables	Bretford CR4500 Series Slim Rectangular Table	957.5775	5	0.45	
Consumer	United States	Fort Lauderdale	Florida	33311	South	OFF-ST-10000760	Office Supplies	Storage	Eldon Fold 'N Roll Cart System	22.368	2	0.2	
Consumer	United States	Concord	North Carolina	28027	South	OFF-PA-10002365	Office Supplies	Paper	Xerox 1967	15.552	3	0.2	
Corporate	United States	Melbourne	Florida	32935	South	OFF-ST-10003282	Office Supplies	Storage	Advantus 10-Drawer Portable Organizer, Chrome Me...	95.616	2	0.2	
Consumer	United States	Memphis	Tennessee	38109	South	FUR-CH-10000513	Furniture	Chairs	High-Back Leather Manager's Chair	831.936	8	0.2	



```
select top 5* from orders where category='Furniture' order by sales desc
```

100 % ▾														
Results Messages														
	name	segment	country	city	state	postal_code	region	product_id	category	sub_category	product_name	sales	quantity	discount
1	icott	Consumer	United States	Philadelphia	Pennsylvania	19134	East	FUR-CH-10002024	Furniture	Chairs	HON 5400 Series Task Chairs for Big and Tall	4416.174	9	0.3
2	ines	Corporate	United States	Burlington	Vermont	NULL		FUR-BO-10004834	Furniture	Bookcases	Riverside Palais Royal Lawyers Bookcase, Royale C...	4404.9	5	0
3	olt	Consumer	United States	Concord	North Carolina	28027	South	FUR-TA-10000198	Furniture	Tables	Chromcraft Bull-Nose Wood Oval Conference Tables ...	4297.644	13	0.4
4	er	Consumer	United States	New York City	New York	10035	East	FUR-BO-10004834	Furniture	Bookcases	Riverside Palais Royal Lawyers Bookcase, Royale C...	4228.704	6	0.2
5	l	Consumer	United States	Buffalo	New York	14215	East	FUR-BO-10002213	Furniture	Bookcases	DMI Eclipse Executive Suite Bookcases	4007.84	10	0.2

```
select * from orders
where category in ('technology','furniture') and order_date between '2020-01-01' and '2020-12-31'
```

100 Results Messages														
	customer_id	customer_name	segment	country	city	state	postal_code	region	product_id	category	sub_category	product_name	sales	quantity
1	CG-12500	Claire Guio	Consumer	United States	Henderson	Kentucky	42420	South	FUR-BQ-10001798	Furniture	Bookcases	South Somerset Collection Bookcase	261.96	2
2	CG-12520	Claire Guio	Consumer	United States	Henderson	Kentucky	42420	South	FUR-CH-10000454	Furniture	Chairs	Hon Deluxe Back Upholstered Stacking Chairs, Round	731.94	3
3	EH-13945	Eric Hoffmann	Consumer	United States	Los Angeles	California	90049	West	TEC-AC-10003027	Technology	Accessories	Imation SGB Mini Travel/Drive USB 2.0 Flash Drive	90.57	3
4	GH-14485	Gene Hale	Corporate	United States	Richardson	Texas	75080	Central	TEC-PH-10004977	Technology	Phones	EC 30524EE4	1097.544	7
5	GH-14485	Gene Hale	Corporate	United States	Richardson	Texas	75080	Central	FUR-FU-10003664	Furniture	Furnishings	Electrix Architect's Clamp-On Swing Arm Lamp, Black	190.92	2

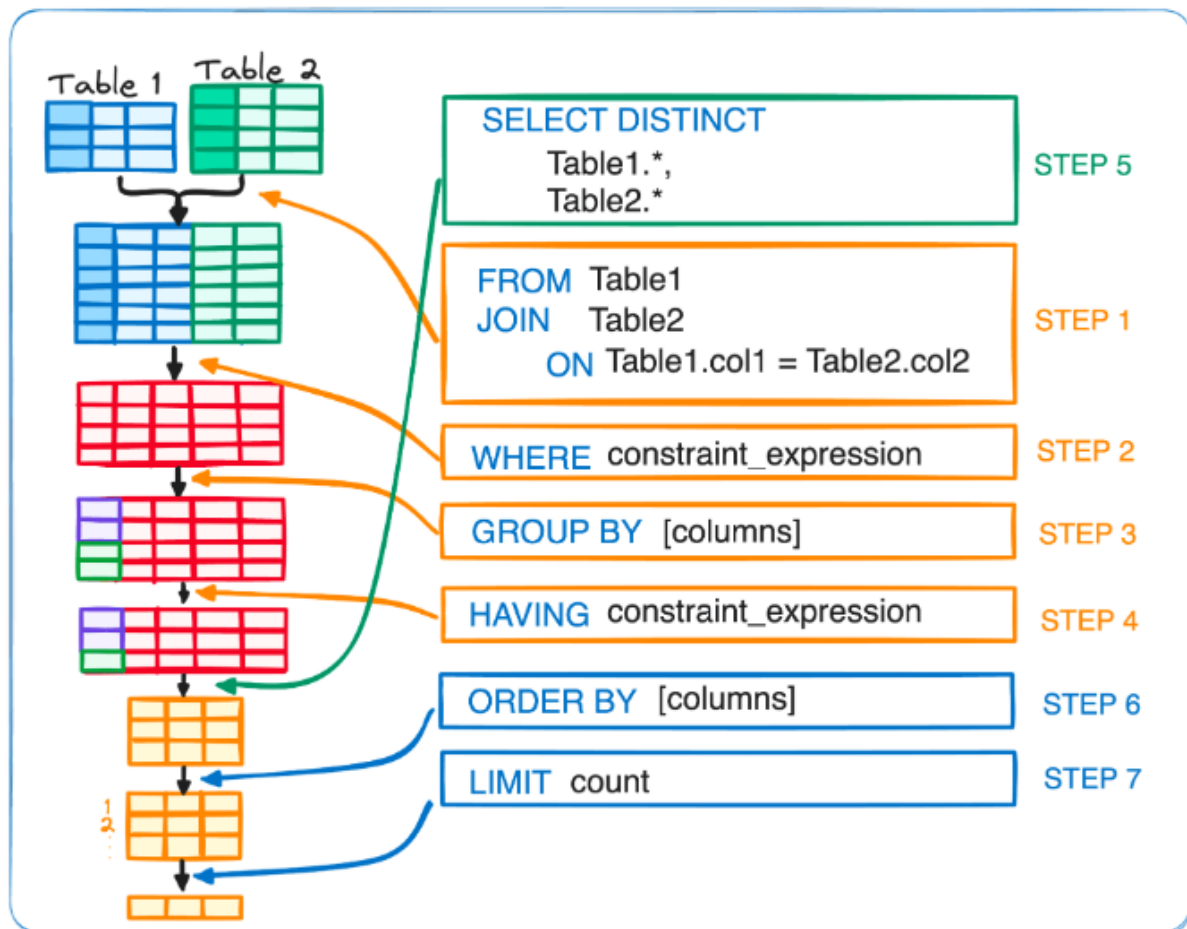
```
select * from Orders
where year(order_date)='2020' and year(ship_date) = '2021'

select * from orders
where order_date between '2020-01-01' and '2020-12-31' and ship_date between '2021-01-01' and '2021-12-31'
```

	row_id	order_id	order_date	ship_date	ship_mode	customer_id	customer_name	segment	country	city	state	postal_code	region	product_id
1	4762	CA-2020-117660	2020-12-30 00:00:00.000	2021-01-04 00:00:00.000	Standard Class	BM-11785	Bryan Mills	Consumer	United States	Columbus	Ohio	43229	East	OFF-LA-10003720
2	4763	CA-2020-117660	2020-12-30 00:00:00.000	2021-01-04 00:00:00.000	Standard Class	BM-11785	Bryan Mills	Consumer	United States	Columbus	Ohio	43229	East	OFF-SU-10001664
3	5033	CA-2020-155166	2020-12-26 00:00:00.000	2021-01-02 00:00:00.000	Standard Class	BB-11545	Brenda Bowman	Corporate	United States	Vineland	New Jersey	8360	East	FUR-CH-10003968
4	5034	CA-2020-155166	2020-12-26 00:00:00.000	2021-01-02 00:00:00.000	Standard Class	BB-11545	Brenda Bowman	Corporate	United States	Vineland	New Jersey	8360	East	OFF-AP-10002765

Note - null is not black and it is not 0, null is unknown

Qus) Execution order of SQL queries



## Assignment Questions

```
--3- write a query to get total profit, first order date and latest order date for each category
```

```
select category, Sum(profit) profit, max(order_date) as latest_order_date, min(order_date) as first_order_date from orders
group by category
```

100 %

Results Messages

	category	profit	latest_order_date	first_order_date
1	Office Supplies	122490.8008	2021-12-30 00:00:00.0000	2018-01-03 00:00:00.0000
2	Furniture	18451.2728	2021-12-30 00:00:00.0000	2018-01-06 00:00:00.0000
3	Technology	145454.9481	2021-12-30 00:00:00.0000	2018-01-06 00:00:00.0000

```
--4- write a query to find sub-categories where average profit is more than the half of the max profit in that sub-category

select sub_category from orders
group by sub_category
having avg(profit) > (max(profit)/2)
```

100 %

Results Messages

	sub_category	max_profit
1	Supplies	163.753
2	Storage	396.13455
3	Phones	614.08935
4	Fasteners	10.944

```
--write a query to find students who have got same marks in Physics and Chemistry. Imp

select student_id, marks
from exams
where subject in ('Physics','Chemistry')
group by student_id , marks
having count(1)=2 --COUNT(1) counts the number of rows in each group (for each unique student_id and marks combination)

SELECT e1.student_id
FROM exams e1
JOIN exams e2 ON e1.student_id = e2.student_id --self join
WHERE e1.subject = 'Physics'
AND e2.subject = 'Chemistry'
AND e1.marks = e2.marks;
```

100 %

Results Messages

	student_id	marks
1	1	91

```
--6- write a query to find total number of products in each category.
```

```
select category, count(product_name) no_of_products from orders
group by category
```

100 %

Results Messages

	category	no_of_products
1	Office Supplies	6026
2	Furniture	2121
3	Technology	1847

--7- write a query to find top 5 sub categories in west region by total quantity sold

```
select top 5 sub_category, sum(quantity) total_quantity from orders
where region = 'West'
group by sub_category
order by total_quantity desc
```

100 %

Results Messages

	sub_category	total_quantity
1	Binders	1868
2	Paper	1702
3	Furnishings	1175
4	Phones	1068
5	Storage	1039

--8- write a query to find total sales for each region and ship mode combination for orders in year 2020

```
select region, ship_mode, sum(sales) as total_sales
from orders
where order_date between '2020-01-01' and '2020-12-31'
group by region, ship_mode
```

100 %

Results Messages

	region	ship_mode	total_sales
1	West	Second Class	36881.81
2	East	First Class	25457.485
3	West	Standard Class	102334.6265

## Ques for Practice

```
--select * from exams
```

--write q query to find students who have same got marks in Physics and Chemistry.

```
--select student_id,marks from exams
--where subject in ('Chemistry','Physics')
--group by student_id, marks
--having count(*) >1
```

100 %

Results Messages

	student_id	subject	marks
1	1	Chemistry	91
2	1	Physics	91
3	1	Maths	92
4	2	Chemistry	80
5	2	Physics	90
6	3	Chemistry	80
7	3	Maths	80
8	4	Chemistry	71
9	4	Physics	54
10	5	Chemistry	79

	student_id	marks
1	1	91

```
--Find students who scored higher in Physics than in Chemistry.
```

```
--SELECT e1.student_id
--FROM exams e1
--JOIN exams e2 ON e1.student_id = e2.student_id
--WHERE e1.subject = 'Physics'
--AND e2.subject = 'Chemistry'
--AND e1.marks > e2.marks;
```

100 %

Results Messages

	student_id
1	2



```
--Find students who have taken all three subjects: Physics, Chemistry, and Maths.
```

```
SELECT student_id  
FROM exams  
WHERE subject IN ('Physics', 'Chemistry', 'Maths')  
GROUP BY student_id  
HAVING COUNT(DISTINCT subject) = 3;
```

100 %

Results Messages

	student_id
1	1

```
-- Find the average marks in each subject.
```

```
SELECT subject, AVG(marks) avg_m from exams  
group by subject
```

100 %

Results Messages

	subject	avg_m
1	Chemistry	80
2	Maths	86
3	Physics	78