DataType Conversion:

Implicit -> data conversion done automatically
Explicit -> data conversion done externally

select * from movie;

select * from movie where movid='1'; (implicit type)

select * from movie where movid=integer'1'; (explicit type)

Data Output Messages Notifications

	movid [PK] integer	movname character varying (50)	movlgth integer	movlang character varying (20)	ageinfo character varying (20)	releasedate date	d_id integer
1	1	A Clockwork Orange	112	English	18	1972-02-02	13

Both returning same output.

Using CAST for data conversions:

It supports all data type to typecast the data

From integer to string , string to integer , boolean to int , int to boolean etc.

Some of the syntax as follows:

select cast('1' as integer);

select cast(1 as char);

select cast('01-JAN-2023' as date);

select cast('0' as boolean),cast('1' as boolean);

select cast('t' as boolean),cast('f' as boolean);

select cast('10.2325' as double precision);

Another type of conversion instead using cast.

select * from movie where movid between '2'::integer and '5'::integer;

Data Output Messages Notifications

	movid [PK] integer	movname character varying (50)	movlgth integer	movlang character varying (20)	ageinfo character varying (20)	releasedate date	d_id integer
1	2	Apocalypse Now	168	English	15	1979-08-15	9
2	3	Battle Royale	111	Japanese	18	2001-01-04	10
3	4	Blade Runner	121	English	15	1982-06-25	27
4	5	Chungking Express	113	Chinese	15	1996-08-03	35

Implicit to Explicit conversions:

Making the implicit datatype data into explicit data as above queries.

Table data conversions:

```
select rid
case
where rating like '%U%' and '%A%' and '%U/A%'
cast (rating as integer)
else
0
end as rating from ratings;
```

Converting table data conversions

To_char conversion:
Converting the all dataype value to char such as int,date,timestamp etc.

select to_char(100900,'9,99999');

Data Output Messages Notifications

	to_char text
1	1,00900

Making this as format

select releasedate,to_char(releasedate,'DD-MM-YYYY') as converteddate from movie;

Data Output Messages Notifications

	releasedate date	converteddate text
1	1972-02-02	02-02-1972
2	1979-08-15	15-08-1979
3	2001-01-04	04-01-2001
4	1982-06-25	25-06-1982

Likewise it can convert all the data to the chartype

To_number Conversion
select to_number('1429.2332','9880.23') as numericcol;

Data Output Messages Notifications

	numericcol numeric
1	19

select to_number('$1,259.80','L9G999D99');

Data Output Messages Notifications

	to_number numeric
1	1259.80

To-Date conversion:

select to_date('22102022','DDMMYYYY');

Data Output Messages Notifications

	to_date date
1	2022-10-22

select to_date('October 10,2023','Month DD,YYYY');

Data Output Messages Notifications

	to_date date
1	2023-10-10

To_Timestamp conversion:

select to_timestamp('2022-10-15 12:00:05','yyyy-mm-dd hh:mi:ss') as ts;

Data Output Messages Notifications

	ts timestamp with time zone
1	2022-10-15 00:00:05+05:30

User Defined Data types;

create domain addr varchar(100) not null;

To create a domain so that we can access the datatype.

insert into testusdt values('mahesh'),('avinash'),('Nijanthan');
Inserted values

select * from testusdt;

Data Output Messages Notifications

	sname character varying (100)
1	mahesh
2	avinash
3	Nijanthan

create domain posnum int not null;
Created user defined datatype for numeric value

create table num (numid serial primary key,val posnum);
Created table and assigned the value of number

insert into num(val) values(20),(35),(78);
Inserted value

select * from num;

Data Output Messages Notifications

	numid [PK] integer	val integer
1	1	20
2	2	35
3	3	78

To create email validation:
create domain emailcheck varchar(250)
check(value ~* '^[A-Za-z0-9._%-]+@[A-Za-z0-9.-]+[.][A-Za-z]+$')
Created email validation check

create table emailinfo(cid serial primary key,email emailcheck);
Created table for to check email

insert into emailinfo(email) values('mahes@gmail.com'),('prade@hotmail.com');
Values inserted

select * from emailinfo;

Data Output Messages Notifications

	cid [PK] integer	email character varying (250)
1	1	mahes@gmail.com
2	2	prade@hotmail.com

Enumeration or set of values in User Defined data types:

create domain validcolor varchar(20)
check (value in ('white','black','grey'))
Created domain with 3 colors

create table color(colorname validcolor);
Created table

insert into color values('white'),('black'),('grey');
Inserted value

select * from color;

Data Output Messages Notifications

	colorname character varying (20)
1	white
2	black
3	grey

If i try to add other color
insert into color values('red');

Data Output Messages Notifications

```
ERROR:  value for domain validcolor violates check constraint "validcolor_check"

SQL state: 23514
```

It shows to check the constraints condition.

Get List of All Domains(list of all data types):
select typname from pg_catalog.pg_type join pg_catalog.pg_namespace on pg_namespace.oid = pg_type.typnamespace
where
typtype = 'd' and nspname='public';

Data Output Messages Notifications

	typname name
1	addr
2	posnum
3	postalcode
4	codepost
5	propemail
6	emailcheck
7	validcolor

How to drop domain ? ie User Defined Datatype:

drop domain postalcode;

Data Output Messages Notifications

```
DROP DOMAIN

Query returned successfully in 56 msec.
```

Create Type- Composite Data Type:

create type address as(city varchar(50),country varchar(50));
Created type as address

create table personinfo(pname varchar(50),address address);
Created table to include address type

insert into personinfo values('mahesh',row('Tindivanam','India'));
Inserted data

select * from personinfo;

Data Output Messages Notifications

	pname character varying (50)	address address
1	mahesh	(Tindivanam,India)

select (address).city as city from personinfo; (to return particular val from type)

Data Output Messages Notifications

	city character varying (50)
1	Tindivanam

Type as Enum:

create type countries as enum('USA','INDIA','SRI LANKA');
Created type for countries

alter type countries add value 'CHINA' after 'USA';
Adding one more type to enum

create table country(cid serial primary key,cname countries);

insert into country(cname) values('USA'),('CHINA'),('SRI LANKA');

Table created and inserted values

select * from country;

Data Output Messages Notifications

	cid [PK] integer	cname countries
1	1	INDIA
2	2	USA
3	3	CHINA
4	4	SRI LANKA

If we try to add other value it will return
insert into country(cname) values('RUSSIA');

Data Output Messages Notifications

```
ERROR:  invalid input value for enum countries: "RUSSIA"
LINE 1: insert into country(cname) values('RUSSIA');
                                          ^

SQL state: 22P02
Character: 35
```

Constraints:

Types: Not Null , Check , Default , Primary Key , Foreign Key and Unique.

Not Null Constraints

```
create table testnn(
nnid serial primary key,
        tag text not null
);
```

```
insert into testnn(tag) values('Mahesh is OP'),('GG Life');
select * from testnn;
```

Data Output Messages Notifications

	nnid [PK] integer	tag text
1	1	Mahesh is OP
2	2	GG Life

```
insert into testnn(tag) values(null);
```

Data Output Messages Notifications

```
ERROR:  Failing row contains (4, null).null value in column "tag" of relation "testnn" violates not-null constraint

ERROR:  null value in column "tag" of relation "testnn" violates not-null constraint
SQL state: 23502
Detail: Failing row contains (4, null).
```

When we try to insert null value it shows the constraints.

UNIQUE Constraints:

```
create table testunq(uid serial primary key,uname text unique);

insert into testunq(uname) values('pravin'),('nijanthan'),('avinash');

select * from testunq;
```

Data Output Messages Notifications

	uid [PK] integer	uname text
1	1	pravin
2	2	nijanthan
3	3	avinash

insert into testunq(uname) values('nijanthan');

Data Output Messages Notifications

```
ERROR:  Key (uname)=(nijanthan) already exists.duplicate key value violates unique constraint "testunq_uname_key"

ERROR:  duplicate key value violates unique constraint "testunq_uname_key"
SQL state: 23505
Detail: Key (uname)=(nijanthan) already exists.
```

When we try to add duplicate it will show the error of unique constraints.

We can also create multiple column for unique key.

create table testunq2(uid serial primary key,uname varchar(50),ucity varchar(50));

alter table testunq2 add constraint unqinfo unique(uname,ucity);
Making multiple column as unique constraints

insert into testunq2(uname,ucity) values('mahesh','tindivanam'),('pradeep','salem');

select * from testunq2;

insert into testunq2(uname,ucity) values('mahesh','salem');
It can run because different same name from different same places.

insert into testunq2(uname,ucity) values('prdaeep','salem');

Data Output Messages Notifications

```
ERROR:  Key (uname, ucity)=(prdaeep, salem) already exists.duplicate key value violates unique constraint "unqinfo"

ERROR:  duplicate key value violates unique constraint "unqinfo"
SQL state: 23505
Detail: Key (uname, ucity)=(prdaeep, salem) already exists.
```

It shows that the pair of data already exists.

Default Constraints:
create table testdef(dname varchar(50),gender char(1) default 'F');
Created table with default constraints

insert into testdef(dname) values('Ashi'),('Durga'),('Pravin');

select * from testdef;

Data Output Messages Notifications

	dname character varying (50)	gender character
1	Ashi	F
2	Durga	F
3	Pravin	F

alter table testdef alter column gender set default 'F';

It is another method to add default constraints

To drop the default
Alter table tname alter column column drop default.

Primary Key Constraints:
create table testpk(tid int primary key,tname varchar(100) not null);

insert into testpk values(101,'Tea'),(104,'Coffee'),(110,'Samosa'),(125,'Veg Roll');

select * from testpk;

Data Output Messages Notifications

	tid [PK] integer	tname character varying (100)
1	101	Tea
2	104	Coffee
3	110	Samosa
4	125	Veg Roll

How to add primary key to the existing table
alter table testpk add primary key(tname);

To add multiple primary key in table

create table testmpk(sid int,pid int,pname varchar(50));
Created table without primary key constraints.

insert into testmpk values(101,11022,'Urad dhal'),(105,12010,'Pappad'),(110,13201,'Pickle');

select * from testmpk;

Data Output Messages Notifications

	sid integer	pid integer	pname character varying (50)
1	101	11022	Urad dhal
2	105	12010	Pappad
3	110	13201	Pickle

To create composite key for multiple primary key.
alter table testmpk add constraint mpk primary key(sid,pid);

Foreign Key Contraints:

create table testcs(s_id int primary key,sname varchar(50));

create table testps(p_id int primary key,s_id int,pname varchar(50),foreign key(s_id) references testcs(s_id));

insert into testcs values(101,'mahesh ltd'),(102,'pravin & co'),(110,'nijan ltd');

insert into testps(p_id,pname) values(10122,'Tea'),(12110,'Coffee'),(11120,'Samosa');

insert into testps(s_id) values(101),(102),(110);

select * from testps;

How to drop a constraints:
Syntax: alter table tname drop constraints colname;

How to add Foreign key constraints in the existing table:

Syntax:
Alter table tname add constraint constrname foreign key(colname) references tname(colname)

Check Constraints :
create table testch(sid serial primary key,sname varchar(30),dob date check(dob>'1998-01-01'),sage int check(sage>15));
Created table with check constraints

insert into testch(sname,dob,sage) values('Ashi','1999-07-19',24),('Deeps','2000-10-11',23);

select * from testch;

Data Output Messages Notifications

	sid [PK] integer	sname character varying (30)	dob date	sage integer
1	1	Ashi	1999-07-19	24
2	2	Deeps	2000-10-11	23

insert into testch(sname,dob,sage) values('Vasanth','1995-12-07',28);

Data Output Messages Notifications

```
ERROR:  Failing row contains (6, Vasanth, 1995-12-07, 28).new row for relation "testch" violates check constraint
"testch_dob_check"

ERROR:  new row for relation "testch" violates check constraint "testch_dob_check"
SQL state: 23514
Detail: Failing row contains (6, Vasanth, 1995-12-07, 28).
```

insert into testch(sname,dob,sage) values('Asha','2010-12-08',13);

Data Output Messages Notifications

```
ERROR:  Failing row contains (5, Asha, 2010-12-08, 13).new row for relation "testch" violates check constraint
"testch_sage_check"

ERROR:  new row for relation "testch" violates check constraint "testch_sage_check"
SQL state: 23514
Detail: Failing row contains (5, Asha, 2010-12-08, 13).
```

These two wont work because it is not satisfies the constraints.

Check constraints on Add , Drop , Rename :

Rename the constraint name:
Alter table tname rename constraint constname1 to constname 2;

To drop the name:
Alter table tname drop constraint constname;

Postgresql Sequence:

To create sequence
create sequence if not exists testseq;

select nextval('testseq'); use to print next value random

select currval('testseq'); use to print current value

select setval('testseq',10); use to print value from set value.

Restart , Rename a sequence using pg admin:

Alter sequence seqname rename to newname;

Alter sequence seqname restart with val;

Create a Sequence with start with ,increment , minvalue,maxvalue syntax:

create sequence if not exists testseq2 as int;
Creating a sequence with the data type. It supports on;y integer data type.

Descending Sequence and Cycle Sequence:

Descending
create sequence descseq
increment -1
minvalue 1
maxvalue 3
start 3;

select nextval('descseq');

Once all the value done it will show error

Cycle
create sequence cycseq
increment -1
minvalue 1
maxvalue 3
start 3
cycle;

select nextval('cycseq');

It will do cyclic way.

How to drop sequence:

Drop sequence seqname;

List all sequences in the database:
select relname sequence_name from pg_class where relkind='S';

Data Output Messages Notifications

	sequence_name name
1	testseq
2	testseq2
3	descseq
4	cycseq

create sequence sroomno start with 200;

create table room(roomno int default nextval('sroomno') not null,rtype varchar(50));

create table keysinfo(keyno int default nextval('sroomno'));

insert into room(rtype) values('1BHK'),('3BHK'),('2BHK');

Created table to add no of rooms with specs.

select * from room;

Data Output Messages Notifications

	roomno integer	rtype character varying (50)
1	200	1BHK
2	201	3BHK
3	202	2BHK

To create alphanumeric Sequence:

create sequence numval start with 1;

create table empl(eid varchar(10) not null default('ID'||nextval('numval')),empname varchar(100));

insert into empl(empname) values('Ashi'),('Deeps'),('Durga'),('Elango'),('SaiTeja');

select * from empl;

Data Output Messages Notifications

	eid character varying (10)	empname character varying (100)
1	ID1	Ashi
2	ID2	Deeps
3	ID3	Durga
4	ID4	Elango
5	ID5	SaiTeja

Automatically by given condition in table the ids are created.

STRING FUNCTIONS in PostgreSQL:
Syntax:
Upper(str) , lower(str),initcap(str) : used to convert the string to the respective function.

left(str,frompos) right(str,frompos): is used to get the string from left right from the position.

reverse(str): returns the output as reversed string.

split_part(str,delimitter,position): Specifically split that char or input
--- split part
select split_part('1,2,3,10',',',4);

Trims: btrim,ltrim,rtrim,trim:
Use trim the spaces from left right and vice versa.

ltrim(str,char) for same all type of trim.

select ltrim('naman','n'); → aman

select rtrim('mahesh','h'); → mahes

select btrim('avinash','a'); → vinash

select trim(' mahesh '); → mahesh

LPAD RPAD Functions:
Syntax: lpad(str,length,fill) rpad(str,length,fill)

It is used to add space at left side or right side.

length(str): used to return a length of string

position(strword in str): used to return the position of the string.
select position('op' in 'mahesh is op');

Data Output Messages Notifications

	position integer
1	11

Strpos function:
strpos(str,substr): return true if it has the string
select strpos('op','mahesh is op');

Data Output Messages Notifications

	strpos integer
1	0

SUBSTRING Function:

substring(str,start,length(end)): return the string which has substring of the position.

select substring('gg life gaming' from 3 for 10);

Data Output Messages Notifications

	substring text
1	life gami

REPEAT function:

repeat(str,times)
select repeat('Sorry',10);

Data Output Messages Notifications

	repeat text
1	SorrySorrySorrySorrySorrySorrySorrySorrySorrySor...

Repeating the word for times given.

REPLACE Functions:
replace(str,fromstr,tostr): replace the given string in the place of string.

select replace('Valorant is Life','Life','OP');

Data Output Messages Notifications

	replace text
1	Valorant is OP

Aggregate Functions in postgreSQL:
Count: return the count of the record for ex: count(colname)
Count with distinct for ex; count(distinct(colname))

SUM: sum of all numerical values present in the column.
For ex: sum(colname)

Sum with distinct: for ex: sum(distinct(colname))

MAX and MIN functions:

It returns the min and max value from the column.

min(colname) , max(colname)

Greatest and Least Function:
For ex: greatest(colname) least(colname)

It accepts string also with the asci value of string.

Average Function; returns avg of the given input()

Select avg(col/val) from tname;

Using Date and Time Functions:

Date format: DD-MM-YYYY or YYYY-DD-MM or YYYY-MM-DD

DateTime format : YYYY-MM-DD HH:MM:SS (time of day formats)

Timestamp: YYYY-MM-DD HH:MM:SS

System Month date:

Set datestyle = ‘ISO’ ,’DMY’;
Show datestyle;

Using of to_date, to_timestamp and to_timestamptz are done earlier.

Make Date , Make Time and Make TimeStamp,Make interval and make timestamptz also done earlier.

Date Value Extractor Function:
Syntax:
Select extract(‘DAY’,from CURRENT_TIMESTAMP)

We can extract DAY , MONTH , YEAR from the Timestamp.

Math Operator in Dates are done earlier using Arithmetic Operations.

Age Function: use to calculate the difference.

age(date1,date2)

age(timestamp1,timestamp2)

Currentdate,currenttime,localdate,localtime,currenttimestamp and localtimestamp are done earlier.

Date Accuracy with Epochs:

It is the difference between the date and timestamp.

```
select
extract(epoch from timestamptz '2021-12-10')-
extract(epoch from timestamptz '2023-10-10')
as "diff_epoch";
```

Data Output Messages Notifications

	diff_epoch numeric
1	-57801600.000000

date_part(field,source)

Grouping in PostgreSQL:
Having , OrderBy,GroupBy,Where

Basic SQL Queries with the clause.

Groupby:
select movname,avg(movlgth) from movie group by movname;

Data Output Messages Notifications

	movname character varying (50)	avg numeric
1	Top Gun	121.0000000000000000
2	Gladiator	165.0000000000000000
3	Chungking Express	113.0000000000000000
4	Apocalypse Now	168.0000000000000000
5	Three Billboards Outside Ebbing, Missouri	134.0000000000000000
6	Let the Right One In	128.0000000000000000
7	V for Vendetta	140.0000000000000000
8	Way of the Dragon	99.0000000000000000
9	Oldboy	130.0000000000000000
10	Star Wars: Return of the Jedi	139.0000000000000000
11	City of Men	140.0000000000000000

Group By with multiple columns are done.

Order By - asc (ascending) desc(descending)

Having - condition that has to be inside the records.

Where - it is the condition to be there to perform certain operations.

Joins in PostgreSQL:

Right Join
Left Join
Natural Join
Cartesian Join/Cross join

For Examples:

--- joins (inner join)

```
select movie.movid,movie.movname,director.d_id from movie
inner join director on
movie.d_id = director.d_id;
```

Data Output Messages Notifications

	movid integer	movname character varying (50)	d_id integer
1	1	A Clockwork Orange	13
2	2	Apocalypse Now	9
3	3	Battle Royale	10
4	4	Blade Runner	27
5	5	Chungking Express	35
6	6	City of God	20
7	7	City of Men	22
8	8	Cold Fish	30
9	9	Crouching Tiger Hidden Dragon	15
10	10	Eyes Wide Shut	13

```
select movie.movid,movie.movname,director.d_id from movie
inner join director on
movie.d_id = director.d_id
where movie.movlgth>120;
```

Query with the condition

	movid integer	movname character varying (50)	d_id integer
1	20	Let the Right One In	1
2	46	There Will Be Blood	2
3	41	The Fifth Element	5
4	19	Leon	5
5	48	Titanic	6
6	13	Gone with the Wind	8
7	2	Apocalypse Now	9
8	42	The Lives of Others	11
9	21	Life of Brian	12
10	43	The Shining	13

```
select movie.*,director.* from movie
inner join director
on movie.d_id = director.d_id;
```

To retrieve all the data from the both table

Data Output Messages Notifications

	movid integer	movname character varying (50)	movlgth integer	movlang character varying (20)	ageinfo character varying (20)	releasedate date	d_id integer	d_ int
1	1	A Clockwork Orange	112	English	18	1972-02-02	13	
2	2	Apocalypse Now	168	English	15	1979-08-15	9	
3	3	Battle Royale	111	Japanese	18	2001-01-04	10	
4	4	Blade Runner	121	English	15	1982-06-25	27	
5	5	Chungking Express	113	Chinese	15	1996-08-03	35	
6	6	City of God	145	Portuguese	18	2003-01-17	20	
7	7	City of Men	140	Portuguese	15	2008-02-29	22	

Inner join ‘using’ query:

```
--- joins (inner join) 'Using'
select * from movie inner join director
using (d_id);
```

Same query as above with the common field.

```
--- joins (inner join) 'using' movie and movierev table

select * from movie inner join movierev
using (movid);
```

Data Output Messages Notifications

	movid integer	movname character varying (50)	movlgth integer	movlang character varying (20)	ageinfo character varying (20)	releasedate date	d_id integer
1	45	The Wizard of Oz	120	English	U	1939-08-25	8
2	13	Gone with the Wind	123	English	PG	1939-12-15	8
3	23	Mary Poppins	87	English	U	1964-08-29	32
4	44	The Sound of Music	91	English	U	1965-03-02	34
5	1	A Clockwork Orange	112	English	18	1972-02-02	13
6	53	Way of the Dragon	99	Chinese	12	1972-06-01	16
7	18	Jaws	134	English	12	1975-06-20	31
8	39	Taxi Driver	117	English	15	1976-02-07	26
9	35	Star Wars: A New Hope	123	English	PG	1977-05-25	17
10	2	Apocalypse Now	168	English	15	1979-08-15	9
11	21	Life of Brian	126	English	15	1979-08-17	12

Connecting 3 more tables in inner join

--- joins (inner join) 'using' 3 tables movie director and movierev

```
select * from movie
inner join director using (d_id)
inner join movierev using(movid);
```

Data Output Messages Notifications

	movid integer	d_id integer	movname character varying (50)	movlgth integer	movlang character varying (20)	ageinfo character varying (20)	releasedate date	fn ch
1	45	8	The Wizard of Oz	120	English	U	1939-08-25	Vi
2	13	8	Gone with the Wind	123	English	PG	1939-12-15	Vi
3	23	32	Mary Poppins	87	English	U	1964-08-29	R
4	44	34	The Sound of Music	91	English	U	1965-03-02	R
5	1	13	A Clockwork Orange	112	English	18	1972-02-02	St

```
select
mv.movname,d.fname,d.lname,mv.movlang,r.revenue_dom from movie mv
inner join director d using (d_id)
inner join movierev r using (movid)
where mv.movlang in ('English','Chinese','Japanese') and
r.revenue_dom>100;
```

Data Output Messages Notifications

	movname character varying (50)	fname character varying (100)	lname character varying (100)	movlang character varying (20)	revenue_dom numeric (10,2)
8	Star Wars: A New Hope	George	Lucas	English	461.20
9	Spider-Man 3	Sam	Raimi	English	337.00
10	Spider-Man 2	Sam	Raimi	English	374.10
11	Spider-Man	Sam	Raimi	English	404.10
12	Gladiator	Ridley	Scott	English	188.20
13	Top Gun	Tony	Scott	English	180.10
14	Watchmen	Zack	Snyder	English	107.50
15	Jaws	Steven	Spielberg	English	260.30
16	Mary Poppins	Robert	Stevenson	English	102.10
17	Pulp Fiction	Quentin	Tarantino	English	107.90

Left Joins:

Example:

```
create table testlft(pid serial primary key,pname varchar(100));

create table testrght(pid serial primary key,pname varchar(100));
```

```
insert into testlft(pname) values('Soap'),('Shampoo'),('Lotion'),('Face Cream');

insert into testrght(pname) values('Soap'),('Shampoo'),('Spray'),('Brush');

select tl.pid,tl.pname,tr.pname from testlft tl
left join testrght tr using(pid);

select * from testlft tl
left join testrght tr using(pid);
```

Data Output Messages Notifications

	pid integer	pname character varying (100)	pname character varying (100)
1	1	Soap	Soap
2	2	Shampoo	Shampoo
3	3	Lotion	Spray
4	4	Face Cream	Brush

Left join on movie database:

```
select d.fname,d.lname,mv.movname from director d
left join movie mv using (d_id) limit 5;
```

Data Output Messages Notifications

	fname character varying (100)	lname character varying (100)	movname character varying (50)
1	Tomas	Alfredson	Let the Right One In
2	Paul	Anderson	There Will Be Blood
3	Wes	Anderson	The Darjeeling Limited
4	Wes	Anderson	Rushmore
5	Wes	Anderson	Grand Budapest Hotel

All the examples with the conditions.

```
select d.d_id,d.fname,d.lname,mv.movname,

        (mr.revenue_dom+mr.revenue_int) as totalrev

        from movie mv

        left join director d using (d_id)
        left join movierev mr using (movid)

        limit 10;
```

Data Output Messages Notifications

	d_id integer	fname character varying (100)	lname character varying (100)	movname character varying (50)	totalrev numeric
1	1	Tomas	Alfredson	Let the Right One In	11.20
2	2	Paul	Anderson	There Will Be Blood	75.70
3	3	Wes	Anderson	Grand Budapest Hotel	174.80
4	3	Wes	Anderson	The Darjeeling Limited	35.10
5	3	Wes	Anderson	Rushmore	[null]
6	4	Richard	Ayoade	Submarine	0.90
7	5	Luc	Besson	The Fifth Element	264.40
8	5	Luc	Besson	Leon	[null]
9	6	James	Cameron	Titanic	2187.30
10	7	Guillermo	del Toro	Pans Labyrinth	84.20

Right Joins

```
select d.d_id,d.fname,d.lname,mv.movname,

        (mr.revenue_dom+mr.revenue_int) as totalrev

        from movie mv

        right join director d using (d_id)
        right join movierev mr using (movid)

        order by d.fname

        limit 10;
```

Data Output Messages Notifications

	d_id integer	fname character varying (100)	lname character varying (100)	movname character varying (50)	totalrev numeric
1	15	Ang	Lee	Life of Pi	609.00
2	15	Ang	Lee	Crouching Tiger Hidden Drag...	213.20
3	16	Bruce	Lee	Way of the Dragon	[null]
4	23	Chan-wook	Park	Oldboy	14.90
5	20	Fernando	Meirelles	City of God	31.70
6	11	Florian	Henckel von Donnersmarck	The Lives of Others	77.40
7	9	Francis	Ford Coppola	Apocalypse Now	[null]
8	17	George	Lucas	Star Wars: Return of the Jedi	475.30
9	17	George	Lucas	Star Wars: A New Hope	775.40
10	17	George	Lucas	Star Wars: Empire Strikes Back	538.10

Full Joins:
Return every row from all join table

select * from movie full join director using (d_id)
full join movierev using (movid) LIMIT 5;

Data Output Messages Notifications

	movid integer	d_id integer	movname character varying (50)	movlgth integer	movlang character varying (20)	ageinfo character varying (20)	releasedate date	fname character varying (100
1	20	1	Let the Right One In	128	Swedish	15	2008-10-24	Tomas
2	46	2	There Will Be Blood	168	English	15	2007-12-26	Paul
3	40	3	The Darjeeling Limited	119	English	PG	2007-09-29	Wes
4	30	3	Rushmore	104	English	12	1998-11-12	Wes
5	15	3	Grand Budapest Hotel	117	English	PG	2014-07-03	Wes

Self Join:
Use to join within the table column with same datatype.

Cross joins:
Used to join the table with rows probability.

select * from movie cross join director limit 5;

Data Output Messages Notifications

	movid integer	movname character varying (50)	movlgth integer	movlang character varying (20)	ageinfo character varying (20)	releasedate date	d_id integer	d_id integer	fnam chara
1	1	A Clockwork Orange	112	English	18	1972-02-02	13	1	Tom
2	2	Apocalypse Now	168	English	15	1979-08-15	9	1	Tom
3	3	Battle Royale	111	Japanese	18	2001-01-04	10	1	Tom
4	4	Blade Runner	121	English	15	1982-06-25	27	1	Tom
5	5	Chungking Express	113	Chinese	15	1996-08-03	35	1	Tom

Natural Joins:
select * from movie natural [left,right,inner] join
director limit 5;

Data Output Messages Notifications

	d_id integer	movid integer	movname character varying (50)	movlgth integer	movlang character varying (20)	ageinfo character varying (20)	releasedate date	fname character varying (100)
1	1	20	Let the Right One In	128	Swedish	15	2008-10-24	Tomas
2	2	46	There Will Be Blood	168	English	15	2007-12-26	Paul
3	3	40	The Darjeeling Limited	119	English	PG	2007-09-29	Wes
4	3	30	Rushmore	104	English	12	1998-11-12	Wes
5	3	15	Grand Budapest Hotel	117	English	PG	2014-07-03	Wes