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)



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;



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

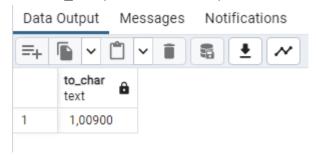
end as rating from ratings;

Converting table data conversions

To_char conversion:

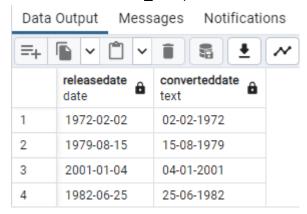
Converting the all dataype value to char such as int,date,timestamp etc.

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



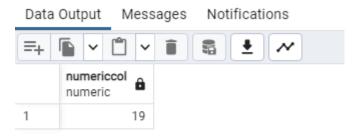
Making this as format

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

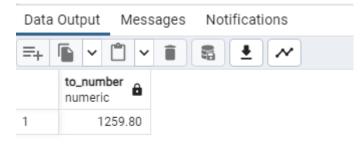


Likewise it can convert all the data to the chartype

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



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

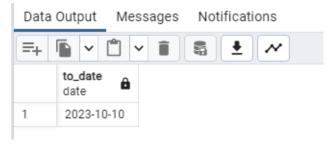


To-Date conversion:

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

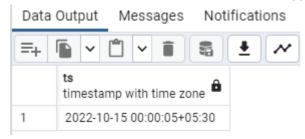


select to_date('October 10,2023','Month DD,YYYY');



To Timestamp conversion:

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



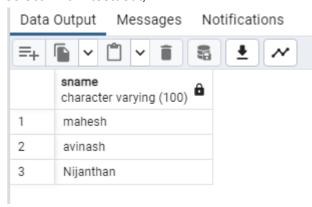
User Defined Data types;

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

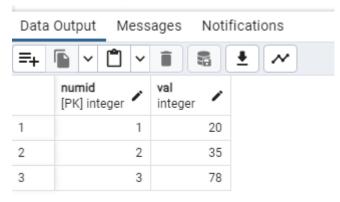
select * from testusdt;



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;

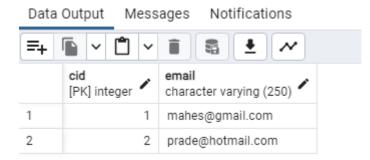


To create email validation: create domain emailcheck varchar(250) check(value ~* '^[A-Za-z0-9._%-]+@[A-Za-z0-9.-]+[.][A-Za-z]+\$') Created email validation check

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;



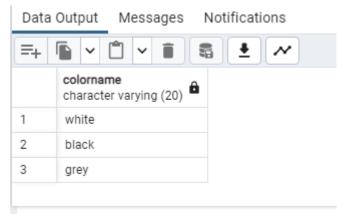
Enumeration or set of values in User Defined data types:

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;



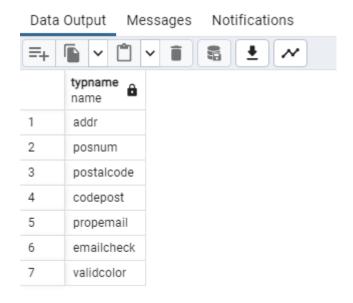
If i try to add other color insert into color values('red');



It shows to check the constraints condition.

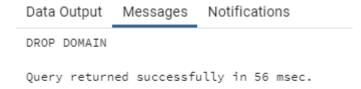
Get List of All Domains(list of all data types): select typname from pg_catalog.pg_type join pg_catalog.pg_namespace on pg_namespace.oid = pg_type.typnamespace where

typtype = 'd' and nspname='public';



How to drop domain? ie User Defined Datatype:

drop domain postalcode;



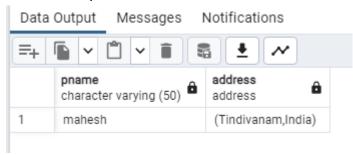
Create Type- Composite Data Type:

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

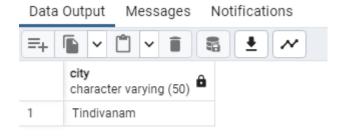
create table personinfo(pname varchar(50),address address); Created table to include address type

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

select * from personinfo;



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



Type as Enum:

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

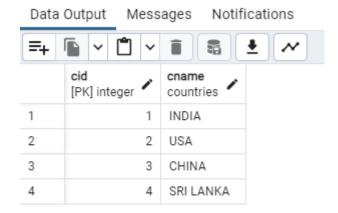
alter type countries add value 'CHINA' after 'USA'; Adding one more type to enum

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;



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');

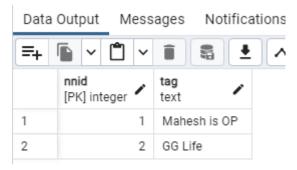
SQL state: 22P02 Character: 35

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;



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).
```

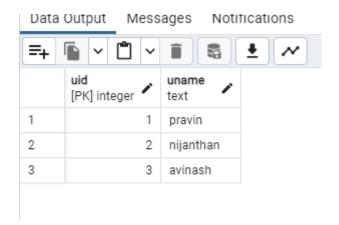
When we try to insert null value it shows the constraints.

UNIQUE Constraints:

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

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

select * from testung;



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 testung2(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 testung2(uname,ucity) values('mahesh','tindivanam'),('pradeep','salem');

select * from testung2;

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;



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

It is another method to add default constraints

To drop the default

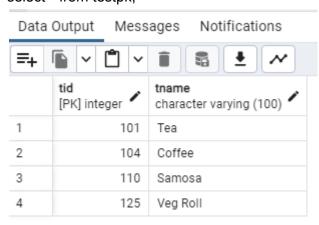
Alter table tname alter column column drop default.

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;



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		otifications
=+	• • [<u> </u>		• ~
	sid integer	â	pid integer 6	pname character varying (50) 6
1	1	01	11022	Urad dhal
2	1	05	12010	Pappad
3	1	10	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:

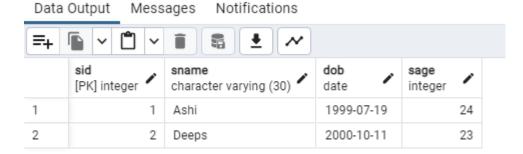
Alter table tname add constraint constrname foreign key(colname) references tname(colname)

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;



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 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('testseg'); 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 segname rename to newname;
Alter sequence segname restart with val;
Create a Sequence with start with ,increment , minvalue,maxvalue syntax:
create sequence if not exists testseg2 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;

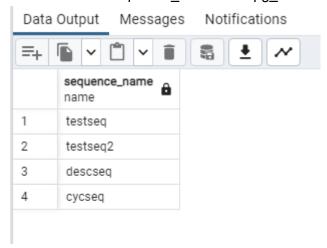
select nextval('cycseq');

It will do cyclic way.

How to drop sequence:

Drop sequence segname;

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



create sequence sroomno start with 200;

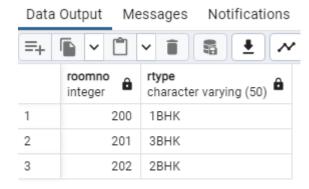
create table room(roomno int default nextval('sroomno') not null,rtype varchar(50));

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;

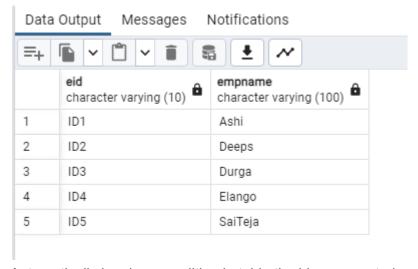


To create alphanumeric Sequence:

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;



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.

Itrim(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

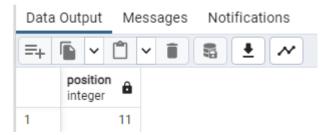
LPAD RPAD Functions:

Syntax: lpad(str,length,fill) rpad(str,length,fill)

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

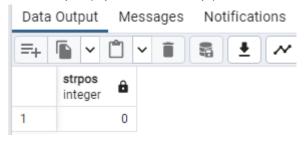
length(str): used to return a length of string

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



Strpos function:

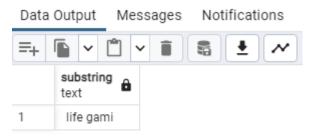
strpos(str,substr): return true if it has the string select strpos('op','mahesh is op');



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);



REPEAT function:

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

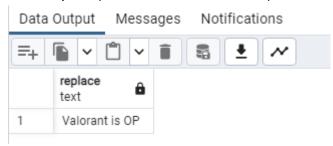


Repeating the word for times given.

REPLACE Functions:

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

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



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)

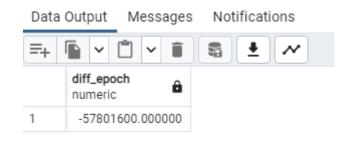
Currentdate, currenttime, localdate, localtime, currenttimestamp and localtimestamp are done earlier.

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";



date_part(field,source)

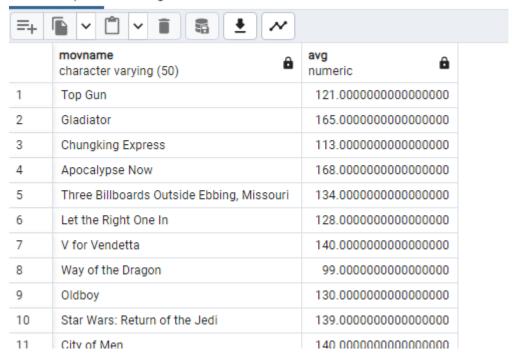
Grouping in PostgreSQL: Having , OrderBy,GroupBy,Where

Basic SQL Queries with the clause.

Groupby:

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

Data Output Messages Notifications



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.

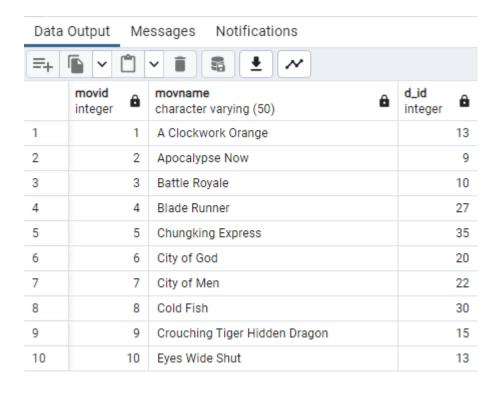
Joins in PostgreSQL:

Right Join Left Join Natural Join Cartesian Join/Cross join

For Examples:

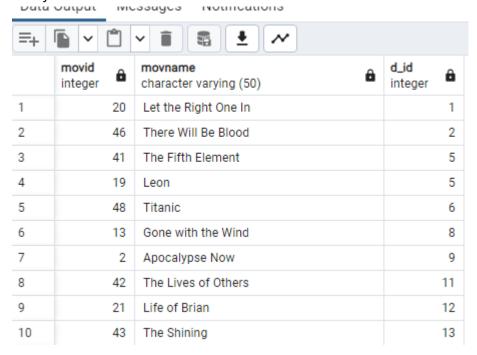
--- joins (inner join)

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



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



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

To retrieve all the data from the both table

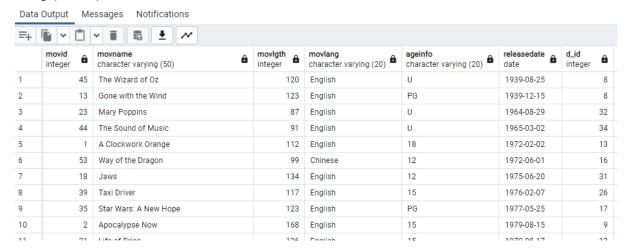


Inner join 'using' query:

--- 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);



Connecting 3 more tables in inner join

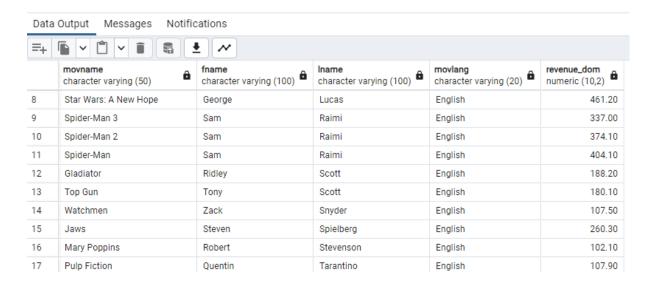
--- joins (inner join) 'using' 3 tables movie director and movierev

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



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;



Left Joins:

Example:

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

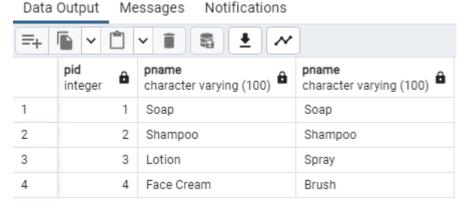
create table testrght(pid serial primary key,pname varchar(100));

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

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

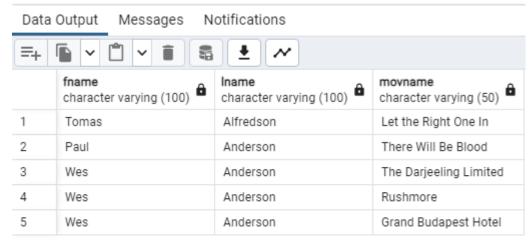
select tl.pid,tl.pname,tr.pname from testlft tl left join testrght tr using(pid);

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



Left join on movie database:

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



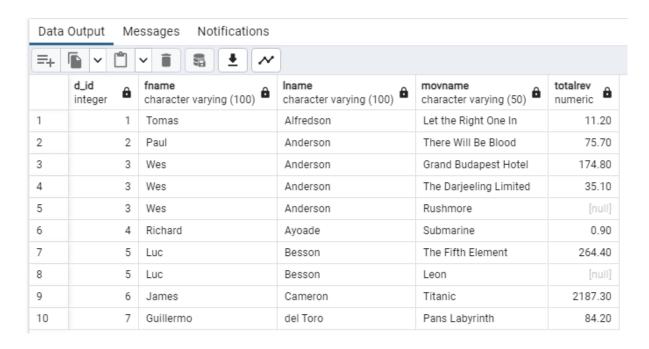
All the examples with the conditions. select d.d_id,d.fname,d.lname,mv.movname,

(mr.revenue_dom+mr.revenue_int) as totalrev

from movie my

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

limit 10;



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 **P** ~ + =+ d_id fname Iname movname totalrev â character varying (100) integer character varying (100) character varying (50) numeric Ang Lee Life of Pi 609.00 1 15 2 15 Ang Lee Crouching Tiger Hidden Drag... 213.20 3 Way of the Dragon 16 Bruce Lee 4 Oldboy 14.90 23 Chan-wook Park 5 20 Fernando Meirelles City of God 31.70 Henckel von Donnersmarck The Lives of Others 77.40 6 11 Florian 7 9 Francis Ford Coppola Apocalypse Now 8 17 Star Wars: Return of the Jedi 475.30 George Lucas 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;



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;



Natural Joins:

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

