1. Junções necessárias (inner join, left join e right join)

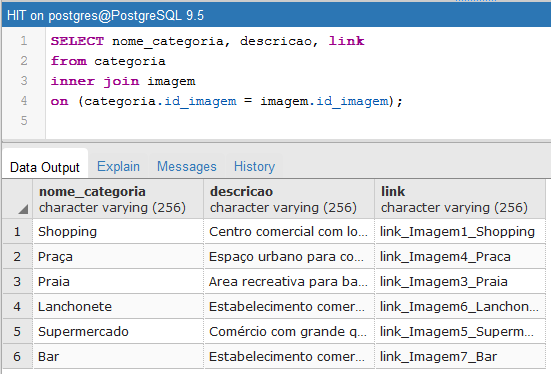
* Ao acessar as categorias de lugares é exibido o **nome**, a **descrição** e o **link** da imagem referente a categoria. Logo, são acessados dados das tabelas categoria e imagem:

SELECT nome\_categoria, descricao, link

from categoria

inner join imagem

on (categoria.id\_imagem = imagem.id\_imagem);



* Ao selecionar uma categoria são exibidos os lugares, sendo apresentados o **nome**, **avaliação geral** quando possuir (obtida da média das avaliações individuais), **imagem** do lugar e a **localização** (latitude e longitude do lugar):

SELECT nome\_lugar, link, lat, long, AVG(nota) as Nota

from lugar

inner join imagem

on (lugar.id\_lugar = imagem.id\_lugar)

left outer join avaliacao

on (avaliacao.id\_lugar = lugar.id\_lugar)

WHERE lugar.id\_categoria IN (

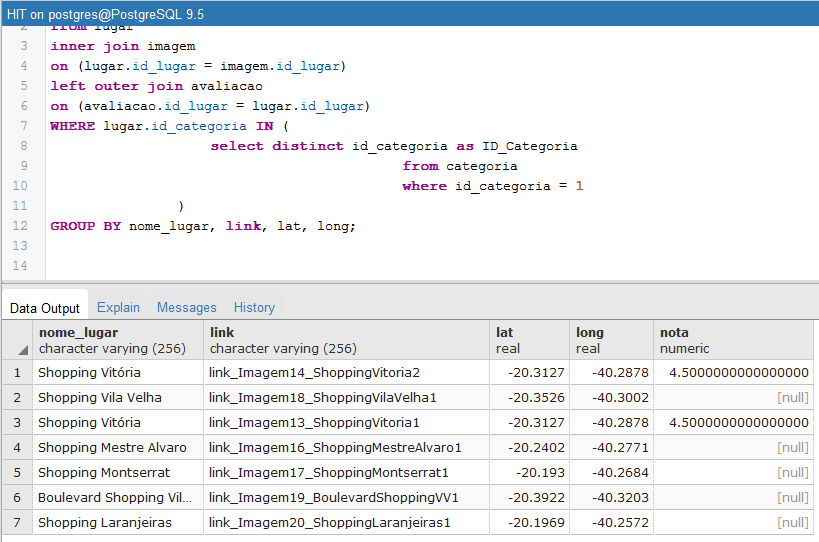
select distinct id\_categoria as ID\_Categoria

from categoria

where id\_categoria = 1

)

GROUP BY nome\_lugar, link, lat, long;



* Ao selecionar o lugar, é exibida a tela de avaliação do lugar sendo exibido os comentários individuais, sendo compostos de **nome da pessoa**, **imagem da pessoa**, **nota** atribuída ao lugar pela pessoa e o **comentário**:

SELECT nome\_pessoa, link, comentario, nota

from pessoa

inner join imagem

on(pessoa.id\_imagem = imagem.id\_imagem)

inner join avaliacao

on(avaliacao.id\_pessoa = pessoa.id\_pessoa)

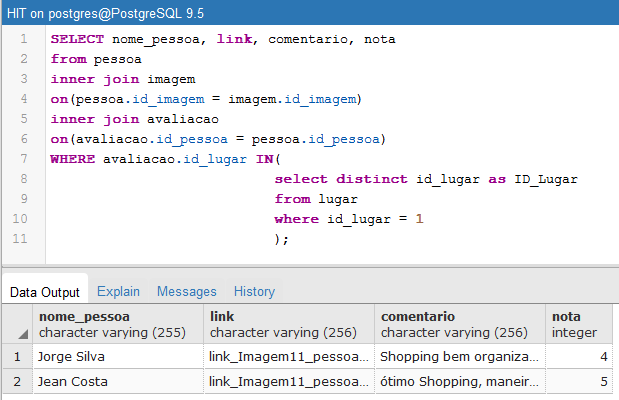
WHERE avaliacao.id\_lugar IN(

select distinct id\_lugar as ID\_Lugar

from lugar

where id\_lugar = 1

);



* Ao acessar a timeline do lugar são apresentadas as postagens, sendo mostradas informações de **nome da pessoa**, **imagem** da pessoa, **postagem** (texto e **imagem** quando houver) e **data** da postagem:

SELECT nome\_pessoa, link, postagem, data

from postagem

inner join pessoa

on(postagem.id\_pessoa = pessoa.id\_pessoa)

inner join imagem

on(pessoa.id\_imagem = imagem.id\_imagem)

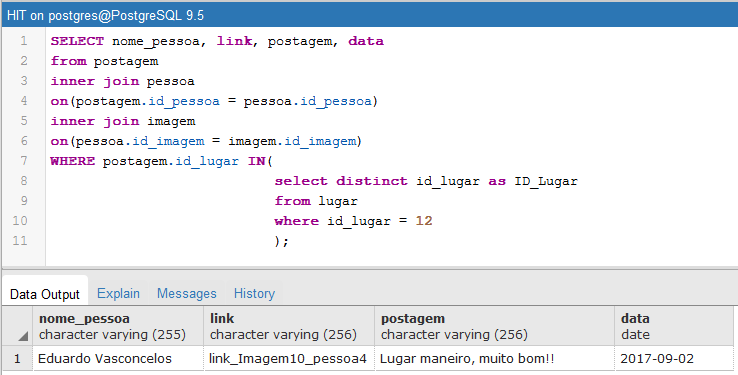
WHERE postagem.id\_lugar IN(

select distinct id\_lugar as ID\_Lugar

from lugar

where id\_lugar = 12

);



**f)Check constraint**

**\* Tabela Pessoa**

CREATE TABLE public.pessoa

(

email character varying(255) COLLATE pg\_catalog."default" CONSTRAINT proper\_email CHECK (email ~\* '^[A-Za-z0-9.\_%-]+@[A-Za-z0-9.-]+[.][A-Za-z]+$'),

colaboracao integer,

nome\_pessoa character varying(255) COLLATE pg\_catalog."default",

autenticidade integer,

senha character varying(20) COLLATE pg\_catalog."default",

login character varying(20) COLLATE pg\_catalog."default",

bairro character varying(100) COLLATE pg\_catalog."default",

cidade character varying(100) COLLATE pg\_catalog."default",

estado character varying(100) COLLATE pg\_catalog."default",

rua character varying(100) COLLATE pg\_catalog."default",

numero character varying(5) COLLATE pg\_catalog."default",

cep character varying(8) COLLATE pg\_catalog."default" CONSTRAINT proper\_cep CHECK (cep ~\* '^[0-9]{2}.[0-9]{3}-[0-9]{3}$'),

ativo character(1) COLLATE pg\_catalog."default",

id\_pessoa integer NOT NULL,

id\_imagem integer,

)

CONSTRAINT pessoa\_pkey PRIMARY KEY (id\_pessoa),

CONSTRAINT pessoa\_id\_imagem\_fkey FOREIGN KEY (id\_imagem)

REFERENCES public.imagem (id\_imagem) MATCH SIMPLE

ON UPDATE NO ACTION

ON DELETE NO ACTION

)

WITH (

OIDS = FALSE

)

TABLESPACE pg\_default;

ALTER TABLE public.pessoa

OWNER to postgres;

**\* Tabela Avaliação**

CREATE TABLE public.avaliacao

(

comentario character varying(256) COLLATE pg\_catalog."default",

id\_avaliacao integer NOT NULL,

nota integer check (nota >= 1 and nota <=5),

id\_lugar integer,

id\_pessoa integer,

CONSTRAINT avaliacao\_pkey PRIMARY KEY (id\_avaliacao),

CONSTRAINT avaliacao\_id\_lugar\_fkey FOREIGN KEY (id\_lugar)

REFERENCES public.lugar (id\_lugar) MATCH SIMPLE

ON UPDATE NO ACTION

ON DELETE NO ACTION,

CONSTRAINT avaliacao\_id\_pessoa\_fkey FOREIGN KEY (id\_pessoa)

REFERENCES public.pessoa (id\_pessoa) MATCH SIMPLE

ON UPDATE NO ACTION

ON DELETE NO ACTION

)

WITH (

OIDS = FALSE

)

TABLESPACE pg\_default;

ALTER TABLE public.avaliacao

OWNER to postgres;

**g)Views**

create view categorias as

SELECT nome\_categoria, descricao, link

from categoria

inner join imagem

on (categoria.id\_imagem = imagem.id\_imagem);

create view lugares as

SELECT nome\_lugar, link, lat, long, AVG(nota) as Nota

from lugar

inner join imagem

on (lugar.id\_lugar = imagem.id\_lugar)

left outer join avaliacao

on (avaliacao.id\_lugar = lugar.id\_lugar)

WHERE lugar.id\_categoria IN (

select distinct id\_categoria as ID\_Categoria

from categoria

where id\_categoria = 1

)

GROUP BY nome\_lugar, link, lat, long;

create view avaliacao\_lugar as

SELECT nome\_pessoa, link, comentario, nota

from pessoa

inner join imagem

on(pessoa.id\_imagem = imagem.id\_imagem)

inner join avaliacao

on(avaliacao.id\_pessoa = pessoa.id\_pessoa)

WHERE avaliacao.id\_lugar IN(

select distinct id\_lugar as ID\_Lugar

from lugar

where id\_lugar = 1

);