

Consultas das tabelas da Biblioteca Virtual

Instalando biblioteca de acesso com postgres

In [1]:

```
!pip install --user psycopg2-binary  
!pip install --user seaborn
```

Requirement already satisfied: psycopg2-binary in c:\users\20171tiimi0017\appdata\roaming\python\python36\site-packages

You are using pip version 9.0.1, however version 18.1 is available.

You should consider upgrading via the 'python -m pip install --upgrade pip' command.

Requirement already satisfied: seaborn in c:\users\20171tiimi0017\appdata\roaming\python\python36\site-packages

Requirement already satisfied: numpy>=1.9.3 in c:\users\20171tiimi0017\appdata\roaming\python\python36\site-packages (from seaborn)

Requirement already satisfied: matplotlib>=1.4.3 in c:\users\20171tiimi0017\appdata\roaming\python\python36\site-packages (from seaborn)

Requirement already satisfied: scipy>=0.14.0 in c:\users\20171tiimi0017\appdata\roaming\python\python36\site-packages (from seaborn)

Requirement already satisfied: pandas>=0.15.2 in c:\users\20171tiimi0017\appdata\roaming\python\python36\site-packages (from seaborn)

Requirement already satisfied: cycler>=0.10 in c:\users\20171tiimi0017\appdata\roaming\python\python36\site-packages (from matplotlib>=1.4.3->seaborn)

Requirement already satisfied: python-dateutil>=2.1 in c:\users\20171tiimi0017\appdata\roaming\python\python36\site-packages (from matplotlib>=1.4.3->seaborn)

Requirement already satisfied: kiwisolver>=1.0.1 in c:\users\20171tiimi0017\appdata\roaming\python\python36\site-packages (from matplotlib>=1.4.3->seaborn)

Requirement already satisfied: pyparsing!=2.0.4,!=2.1.2,!=2.1.6,>=2.0.1 in c:\users\20171tiimi0017\appdata\roaming\python\python36\site-packages (from matplotlib>=1.4.3->seaborn)

Requirement already satisfied: pytz>=2011k in c:\users\20171tiimi0017\appdata\roaming\python\python36\site-packages (from pandas>=0.15.2->seaborn)

Requirement already satisfied: six in c:\users\20171tiimi0017\appdata\roaming\python\python36\site-packages (from cycler>=0.10->matplotlib>=1.4.3->seaborn)

Requirement already satisfied: setuptools in c:\program files (x86)\python36-32\lib\site-packages (from kiwisolver>=1.0.1->matplotlib>=1.4.3->seaborn)

You are using pip version 9.0.1, however version 18.1 is available.

You should consider upgrading via the 'python -m pip install --upgrade pip' command.

Importando biblioteca psycopg2 e Configurando conexão e usando cursosr

In [5]:

```
import psycopg2
conn = psycopg2.connect(host="localhost", database="bibliotecavirtual", user="postgres", pas
```

Obtendo dados do database com Pandas

In [3]:

```
!pip install --user pandas
```

```
Requirement already satisfied: pandas in c:\users\gustavo\appdata\roaming\python\python37\site-packages (0.23.4)
Requirement already satisfied: numpy>=1.9.0 in c:\users\gustavo\appdata\roaming\python\python37\site-packages (from pandas) (1.15.4)
Requirement already satisfied: pytz>=2011k in c:\users\gustavo\appdata\roaming\python\python37\site-packages (from pandas) (2018.7)
Requirement already satisfied: python-dateutil>=2.5.0 in c:\users\gustavo\appdata\roaming\python\python37\site-packages (from pandas) (2.7.5)
Requirement already satisfied: six>=1.5 in c:\users\gustavo\appdata\roaming\python\python37\site-packages (from python-dateutil>=2.5.0->pandas) (1.11.0)
```

```
You are using pip version 10.0.1, however version 18.1 is available.
You should consider upgrading via the 'python -m pip install --upgrade pip' command.
```

In [1]:

```
import pandas as pd
```

9 - TABELAS E PRINCIPAIS CONSULTAS

9.1 CONSULTAS DAS TABELAS COM TODOS OS DADOS INSERIDOS (Todas)

Pessoa

In [6]:

```
result1 = pd.read_sql_query("""
    select * from pessoa
""", conn)
```

In [7]:

result1

Out[7]:

	cod_pessoa	nome	fk_cidade_cod_cidade
0	1	João	1
1	2	Pedro	2
2	3	José	2
3	4	Maria	3
4	5	Joana	4
5	6	Mário	4
6	7	Plínio	5
7	8	Fernando	5
8	9	Rafael	5
9	10	Carlos	5
10	11	Leonardo	5
11	12	John	6
12	13	Bob	7
13	14	Fred	8
14	15	Haru	10
15	16	Machado de Assis	3
16	17	Stephen King	7
17	18	Victor Hugo	9
18	19	Franz Kafka	11
19	20	Haruki Murakami	10
20	21	John Green	6
21	22	Robert Lawrence	8

Usuário

In [14]:

```
result2 = pd.read_sql_query("""
    select * from usuario
    """
    ,conn)
```

In [15]:

result2

Out[15]:

	email	senha	fk_pessoa_cod_pessoa
0	joao@gmail	aca3ec7278a47144c0a863e60c595abe	1
1	pedro@gmail	77fe0359ee910002d306e2991d8ec27f	2
2	jose@gmail	e7d80ffefafa212b7c5c55700e4f7193e	3
3	maria@gmail	f8461b554d59b3014e8ff5165dc62fac	4
4	joana@gmail	f153d147e459b3ebca47d59fdf179ecd	5
5	mario@gmail	190661875f0470d6fd4ed9811b07322d	6
6	plinio@gmail	0740aeab01c11bc05bd503652d3be878	7
7	fernando@gmail	fbc4fef99cdd459c958f92ef36c9fe2b	8
8	rafa@gmail	2c13817fca846f6f9a7d934d71a668ee	9
9	carlos@gmail	a261108357a8dbc6a891b1efd40a794	10
10	leo@gmail	657b298b04e033810343842f993c9817	11
11	john@gmail	6e0b7076126a29d5dfcbd54835387b7b	12
12	bob@gmail	2acba7f51acfd4fd5102ad090fc612ee	13
13	fred@gmail	77064f5bd13e417f564e7d880dc7a536	14
14	haru@gmail	aecab1503021d470bc5cde7ae0e87032	15

Evento

In [7]:

```
result3 = pd.read_sql_query("""
    select * from evento
    """",conn)
```

In [8]:

result3

Out[8]:

	cod_evento	nome_evento
0	1	Leitura Coletiva
1	2	Encontro de Leitores
2	3	Doação de Livros
3	4	Livros Antigos
4	5	Feira de livros
5	6	Sebo Veredas
6	7	Venda de livros usados

Tipo_evento

In [22]:

```
result4 = pd.read_sql_query("""
    select * from tipo_evento
""",conn)
```

In [23]:

```
result4
```

Out[23]:

	tipo_desc	cod_tipo
0	Encontro	1
1	Vendas	2
2	Outros	3

Livro

In [20]:

```
result5 = pd.read_sql_query("""
    select * from livro
""",conn)
```

In [21]:

```
result5
```

Out[21]:

	cod_livro	nome_livro
0	1	Helena
1	2	O Iluminado
2	3	Goosebumps
3	4	Os Miseráveis
4	5	Dom Casmurro
5	6	A Metamorfose
6	7	Norwegian Wood
7	8	Kafka a beira-mar
8	9	A Culpa é das Estrelas
9	10	Tartarugas até lá embaixo

Gênero

In [24]:

```
result6 = pd.read_sql_query("""
    select * from genero
""", conn)
```

In [25]:

result6

Out[25]:

	cod_genero	genero_desc
0	1	Terror
1	2	Romance
2	3	Ficção

Postagem_posta_comentario

In [26]:

```
result7 = pd.read_sql_query("""
    select * from postagem_posta_comentario
""", conn)
```

In [27]:

result7

Out[27]:

	cod_postagem	data_postagem	hora_postagem	fk_livro_cod_livro	fk_usuario_fk_pessoa_cod
0	1	2018-02-04	12:00:00	1	
1	2	2018-02-05	11:30:00	2	
2	3	2018-04-06	08:45:00	3	
3	4	2018-05-06	18:00:00	4	
4	5	2018-05-06	20:00:00	5	
5	6	2018-06-07	07:00:00	6	
6	7	2018-06-07	09:30:00	7	
7	8	2018-06-07	15:00:00	7	
8	9	2018-07-07	18:00:00	7	
9	10	2018-08-07	20:00:00	8	

País

In [29]:

```
result8 = pd.read_sql_query("""
    select * from pais
""",conn)
```

In [30]:

```
result8
```

Out[30]:

	cod_pais	pais_desc
0	1	Brasil
1	2	Estados Unidos
2	3	França
3	4	Japão
4	5	Áustria

Estado

In [31]:

```
result9 = pd.read_sql_query("""
    select * from estado
""",conn)
```

In [32]:

```
result9
```

Out[32]:

	cod_estado	estado_desc	fk_pais_cod_pais
0	1	Espírito Santo	1
1	2	Rio de Janeiro	1
2	3	São Paulo	1
3	4	Indiana	2
4	5	Maine	2
5	6	Nova Iorque	2
6	7	Doubs	3
7	8	Kyoto	4
8	9	Boemia	5

Cidade

In [33]:

```
result10 = pd.read_sql_query("""
    select * from cidade
    """",conn)
```

In [34]:

result10

Out[34]:

	cod_cidade	cidade_desc	fk_estado_cod_estado
0	1	Vitória	1
1	2	Serra	1
2	3	Campos	2
3	4	Campinas	3
4	5	São Paulo	3
5	6	Indianápolis	4
6	7	Portland	5
7	8	Nova Iorque	6
8	9	Besançon	7
9	10	Kyoto	8
10	11	Praga	9

9.2 CONSULTAS DAS TABELAS COM FILTROS WHERE (Mínimo 4)

select * from pessoa where cod_pessoa <= 5

In [35]:

```
result11 = pd.read_sql_query("""
    select * from pessoa where cod_pessoa <= 5
    """",conn)
```

In [36]:

result11

Out[36]:

	cod_pessoa	nome	fk_cidade_cod_cidade
0	1	João	1
1	2	Pedro	2
2	3	José	2
3	4	Maria	3
4	5	Joana	4

select * from postagem_posta_comentario where data_postagem > '2018-05-06'

In [37]:

```
result12 = pd.read_sql_query("""
    select * from postagem_posta_comentario where data_postagem > '2018-05-06'
    """, conn)
```

In [38]:

```
result12
```

Out[38]:

	cod_postagem	data_postagem	hora_postagem	fk_livro_cod_livro	fk_usuario_fk_pessoa_cod
0	6	2018-06-07	07:00:00	6	
1	7	2018-06-07	09:30:00	7	
2	8	2018-06-07	15:00:00	7	
3	9	2018-07-07	18:00:00	7	
4	10	2018-08-07	20:00:00	8	



select nome_livro from livro where cod_livro > 4

In [39]:

```
result13 = pd.read_sql_query("""
    select nome_livro from livro where cod_livro > 4
    """, conn)
```

In [40]:

```
result13
```

Out[40]:

	nome_livro
0	Dom Casmurro
1	A Metamorfose
2	Norwegian Wood
3	Kafka a beira-mar
4	A Culpa é das Estrelas
5	Tartarugas até lá embaixo

select * from usuario where fk_pessoa_cod_pessoa < 10

In [41]:

```
result14 = pd.read_sql_query("""
    select * from usuario where fk_pessoa_cod_pessoa < 10
    """ ,conn)
```

In [42]:

result14

Out[42]:

	email	senha	fk_pessoa_cod_pessoa
0	joao@gmail	aca3ec7278a47144c0a863e60c595abe	1
1	pedro@gmail	77fe0359ee910002d306e2991d8ec27f	2
2	jose@gmail	e7d80ffefafa212b7c5c55700e4f7193e	3
3	maria@gmail	f8461b554d59b3014e8ff5165dc62fac	4
4	joana@gmail	f153d147e459b3ebca47d59fdf179ecd	5
5	mario@gmail	190661875f0470d6fd4ed9811b07322d	6
6	plinio@gmail	0740aeab01c11bc05bd503652d3be878	7
7	fernando@gmail	fbc4fef99cdd459c958f92ef36c9fe2b	8
8	rafa@gmail	2c13817fca846f6f9a7d934d71a668ee	9

9.3 CONSULTAS QUE USAM OPERADORES LÓGICOS, ARITMÉTICOS E TABELAS OU CAMPOS RENOMEADOS (Mínimo 11)

a) Criar 5 consultas que envolvam os operadores lógicos AND, OR e Not

1) select * from pessoa where cod_pessoa > 3 and estado cod_pessoa <= 15 and nome <> 'Leonardo'

In [48]:

```
result15 = pd.read_sql_query("""
    select * from pessoa where cod_pessoa > 3 and cod_pessoa <= 15
    """ ,conn)
```

In [49]:

result15

Out[49]:

	cod_pessoa	nome	fk_cidade_cod_cidade
0	4	Maria	3
1	5	Joana	4
2	6	Mário	4
3	7	Plínio	5
4	8	Fernando	5
5	9	Rafael	5
6	10	Carlos	5
7	12	John	6
8	13	Bob	7
9	14	Fred	8
10	15	Haru	10

2) select * from usuario where fk_pessoa_cod_pessoa = 1 or fk_pessoa_cod_pessoa = 4

In [50]:

```
result16 = pd.read_sql_query("""
    select * from usuario where fk_pessoa_cod_pessoa = 1 or fk_pess
    """,conn)
```

In [51]:

result16

Out[51]:

	email	senha	fk_pessoa_cod_pessoa
0	joao@gmail.com	aca3ec7278a47144c0a863e60c595abe	1
1	maria@gmail.com	f8461b554d59b3014e8ff5165dc62fac	4

3) select * from evento where not cod_evento = 1 and not cod_evento = 3

In [52]:

```
result17 = pd.read_sql_query("""
    select * from evento where not cod_evento = 1 and not cod_event
    """,conn)
```

In [53]:

result17

Out[53]:

	cod_evento	nome_evento
0	2	Encontro de Leitores
1	4	Livros Antigos
2	5	Feira de livros
3	6	Sebo Veredas
4	7	Venda de livros usados

4) select * from postagem_posta_comentario where cod_postagem > 3 and hora_postagem >= '09:00:00'

In [58]:

```
result18 = pd.read_sql_query("""
    select * from postagem_posta_comentario where cod_postagem > 3
    """ ,conn)
```

In [59]:

result18

Out[59]:

	cod_postagem	data_postagem	hora_postagem	fk_livro_cod_livro	fk_usuario_fk_pessoa_cod_
0	4	2018-05-06	18:00:00	4	
1	5	2018-05-06	20:00:00	5	
2	7	2018-06-07	09:30:00	7	
3	8	2018-06-07	15:00:00	7	
4	9	2018-07-07	18:00:00	7	
5	10	2018-08-07	20:00:00	8	

5) select * from livro where not cod_livro > 7 and nome_livro <> 'Helena'

In [60]:

```
result19 = pd.read_sql_query("""
    select * from livro where not cod_livro > 7 and nome_livro <>
    """ ,conn)
```

In [61]:

```
result19
```

Out[61]:

	cod_livro	nome_livro
0	2	O Iluminado
1	3	Goosebumps
2	4	Os Miseráveis
3	5	Dom Casmurro
4	6	A Metamorfose
5	7	Norwegian Wood

b) Criar no mínimo 3 consultas com operadores aritméticos

1) select * from pessoa where cod_pessoa <> 5

In [62]:

```
result20 = pd.read_sql_query("""  
    select * from pessoa where cod_pessoa <> 5  
""",conn)
```

In [63]:

```
result20
```

Out[63]:

	cod_pessoa	nome	fk_cidade_cod_cidade
0	1	João	1
1	2	Pedro	2
2	3	José	2
3	4	Maria	3
4	6	Mário	4
5	7	Plínio	5
6	8	Fernando	5
7	9	Rafael	5
8	10	Carlos	5
9	11	Leonardo	5
10	12	John	6
11	13	Bob	7
12	14	Fred	8
13	15	Haru	10
14	16	Machado de Assis	3
15	17	Stephen King	7
16	18	Victor Hugo	9
17	19	Franz Kafka	11
18	20	Haruki Murakami	10
19	21	John Green	6
20	22	Robert Lawrence	8

2) select * from evento where cod_evento <= 3

In [64]:

```
result21 = pd.read_sql_query("""
    select * from evento where cod_evento <= 3
    """,conn)
```

In [65]:

result21

Out[65]:

	cod_evento	nome_evento
0	1	Leitura Coletiva
1	2	Encontro de Leitores
2	3	Doação de Livros

3) select * from usuario where fk_pessoa_cod_pessoa >= 5

In [71]:

```
result22 = pd.read_sql_query("""
    select * from usuario where fk_pessoa_cod_pessoa >= 5
    """, conn)
```

In [72]:

result22

Out[72]:

	email	senha	fk_pessoa_cod_pessoa
0	joana@gmail	f153d147e459b3ebca47d59fdf179ecd	5
1	mario@gmail	190661875f0470d6fd4ed9811b07322d	6
2	plinio@gmail	0740aeab01c11bc05bd503652d3be878	7
3	fernando@gmail	fbc4fef99cdd459c958f92ef36c9fe2b	8
4	rafa@gmail	2c13817fca846f6f9a7d934d71a668ee	9
5	carlos@gmail	a261108357a8dbc6a891b1efd40a794	10
6	leo@gmail	657b298b04e033810343842f993c9817	11
7	john@gmail	6e0b7076126a29d5dfcbd54835387b7b	12
8	bob@gmail	2acba7f51acf4fd5102ad090fc612ee	13
9	fred@gmail	77064f5bd13e417f564e7d880dc7a536	14
10	haru@gmail	aecab1503021d470bc5cde7ae0e87032	15

c) Criar no mínimo 3 consultas com operação de renomear nomes de campos ou tabelas**1) select nome_livro as livro from livro where livro <> 'Os Miseráveis'**

In [76]:

```
result23 = pd.read_sql_query("""
    select nome_livro as nome_do_livro from livro  where nome_livro
    """, conn)
```

In [77]:

result23

Out[77]:

nome_do_livro

0	Helena
1	O Iluminado
2	Goosebumps
3	Dom Casmurro
4	A Metamorfose
5	Norwegian Wood
6	Kafka a beira-mar
7	A Culpa é das Estrelas
8	Tartarugas até lá embaixo

2) select nome_evento as Encontros from evento where cod_evento > 3

In [78]:

```
result24 = pd.read_sql_query("""
    select nome_evento as encontros from evento where cod_evento >
    """",conn)
```

In [79]:

result24

Out[79]:

encontros

0	Livros Antigos
1	Feira de livros
2	Sebo Veredas
3	Venda de livros usados

3) select nome as nome_da_pessoa,cod_pessoa as codigo_da_pessoa from pessoa where nome = 'Fernando' or cod_pessoa < 5

In [82]:

```
result25 = pd.read_sql_query("""
    select nome as nome_da_pessoa,cod_pessoa as codigo_da_pessoa
    from pessoa where nome = 'Fernando' or cod_pessoa < 5
    """",conn)
```

In [83]:

result25

Out[83]:

	nome_da_pessoa	codigo_da_pessoa
0	João	1
1	Pedro	2
2	José	3
3	Maria	4
4	Fernando	8

9.4 CONSULTAS QUE USAM OPERADORES LIKE E DATAS (Mínimo 12)

a) Criar outras 5 consultas que envolvam like ou ilike

1) select * from pessoa where nome like 'e%'

In [88]:

```
result26 = pd.read_sql_query("""
    select * from pessoa where nome like 'J%'
""", conn)
```

In [89]:

result26

Out[89]:

	cod_pessoa	nome	fk_cidade_cod_cidade
0	1	João	1
1	3	José	2
2	5	Joana	4
3	12	John	6
4	21	John Green	6

2) select * from usuario where email ilike 'jo%'

In [90]:

```
result27 = pd.read_sql_query("""
    select * from usuario where email ilike 'jo%'
""", conn)
```

In [91]:

result27

Out[91]:

	email	senha	fk_pessoa_cod_pessoa
0	joao@gmail	aca3ec7278a47144c0a863e60c595abe	1
1	jose@gmail	e7d80ffeeafa212b7c5c55700e4f7193e	3
2	joana@gmail	f153d147e459b3ebca47d59fdf179ecd	5
3	john@gmail	6e0b7076126a29d5dfcbd54835387b7b	12

3) select nome_livro from livro where nome_livro ilike 'o%' or nome_livro like "%s'

In [96]:

```
result28 = pd.read_sql_query("""
    select nome_livro from livro where nome_livro ilike 'o%' or nome_livro like "%s"
    """, conn)
```

In [97]:

result28

Out[97]:

	nome_livro
0	O Iluminado
1	Goosebumps
2	Os Miseráveis
3	A Culpa é das Estrelas

4) select * from pessoa where nome ilike "%an%"

In [98]:

```
result29 = pd.read_sql_query("""
    select * from pessoa where nome ilike "%an%"
    """, conn)
```

In [99]:

result29

Out[99]:

	cod_pessoa	nome	fk_cidade_cod_cidade
0	5	Joana	4
1	8	Fernando	5
2	19	Franz Kafka	11

5) select * from evento where nome_evento ilike '%livro%'

In [100]:

```
result30 = pd.read_sql_query("""
    select * from evento where nome_evento ilike '%livro%'
    """, conn)
```

In [101]:

```
result30
```

Out[101]:

	cod_evento	nome_evento
0	3	Doação de Livros
1	4	Livros Antigos
2	5	Feira de livros
3	7	Venda de livros usados

6) select * from usuario where email like '%r%'

In [120]:

```
result31 = pd.read_sql_query("""
    select * from usuario where email like '%r%'
    """, conn)
```

In [117]:

```
result31
```

Out[117]:

	email	senha	fk_pessoa_cod_pessoa
0	pedro@gmail	77fe0359ee910002d306e2991d8ec27f	2
1	maria@gmail	f8461b554d59b3014e8ff5165dc62fac	4
2	mario@gmail	190661875f0470d6fd4ed9811b07322d	6
3	fernando@gmail	fbc4fef99cdd459c958f92ef36c9fe2b	8
4	rafa@gmail	2c13817fca846f6f9a7d934d71a668ee	9
5	carlos@gmail	a261108357a8dbcb6a891b1efd40a794	10
6	fred@gmail	77064f5bd13e417f564e7d880dc7a536	14
7	haru@gmail	aecab1503021d470bc5cde7ae0e87032	15

b) Criar uma consulta para cada tipo de função data apresentada.

9.5 ATUALIZAÇÃO E EXCLUSÃO DE DADOS (Mínimo 6)

1) update pessoa set nome = 'Sérgio' where cod_pessoa = 1

2) update usuario set email = 'sergio@gmail' where fk_pessoa_cod_pessoa = 1

3) update livro set nome_livro = 'It - A Coisa' where cod_livro = 2

4) delete from pessoa where cod_pessoa = 1

5) delete from livro where codigo >= 8

6) delete from evento where nome_evento = 'Feira de livros'

9.6 CONSULTAS COM JUNÇÃO E ORDENAÇÃO (Mínimo 6)

a) Uma junção que envolva todas as tabelas possuindo no mínimo 3 registros no resultado

In [44]:

```
resultx = pd.read_sql_query("""
    select * from pessoa pe
    inner join usuario us on (pe.cod_pessoa = us.fk_pessoa_cod_pess
    inner join cidade ci on (ci.cod_cidade = pe.fk_cidade_cod_cidad
    inner join estado es on (es.cod_estado = ci.fk_estado_cod_estad
    inner join pais pa on (pa.cod_pais = es.fk_pais_cod_pais)
    inner join postagem_posta_comentario ppc
    on (us.fk_pessoa_cod_pessoa = ppc.fk_usuario_fk_pessoa_cod_pess
    inner join livro lv on (us.fk_pessoa_cod_pessoa = lv.cod_livro)
    inner join livro_genero lg on (lv.cod_livro = lg.fk_livro_cod_l
    inner join genero ge on (lg.fk_genero_cod_genero = ge.cod_gener
    inner join comparece_evento ce on (us.fk_pessoa_cod_pessoa = ce
    inner join evento ev on (ce.fk_evento_cod_evento = ev.cod_event
    inner join evento_possui_tipo ept on (ev.cod_evento = ept.fk_ev
    inner join tipo_evento te on (ept.fk_tipo_evento_cod_tipo = te.
    """",conn)
```

In [45]:

resultx

Out[45]:

	cod_pessoa	nome	fk_cidade_cod_cidade	email	
0	1	João	1	joao@gmail	aca3ec7278a47144c0a863e60c5
1	1	João	1	joao@gmail	aca3ec7278a47144c0a863e60c5
2	1	João	1	joao@gmail	aca3ec7278a47144c0a863e60c5
3	2	Pedro	2	pedro@gmail	77fe0359ee910002d306e2991d8
4	2	Pedro	2	pedro@gmail	77fe0359ee910002d306e2991d8
5	2	Pedro	2	pedro@gmail	77fe0359ee910002d306e2991d8
6	2	Pedro	2	pedro@gmail	77fe0359ee910002d306e2991d8
7	3	José	2	jose@gmail	e7d80ffefafa212b7c5c55700e4f
8	3	José	2	jose@gmail	e7d80ffefafa212b7c5c55700e4f
9	4	Maria	3	maria@gmail	f8461b554d59b3014e8ff5165d0
10	6	Mário	4	mario@gmail	190661875f0470d6fd4ed9811b0
11	6	Mário	4	mario@gmail	190661875f0470d6fd4ed9811b0
12	8	Fernando	5	fernando@gmail	fbc4fef99cdd459c958f92ef36c
13	8	Fernando	5	fernando@gmail	fbc4fef99cdd459c958f92ef36c
14	8	Fernando	5	fernando@gmail	fbc4fef99cdd459c958f92ef36c
15	9	Rafael	5	rafa@gmail	2c13817fca846f6f9a7d934d71a

16 rows × 33 columns

b) Outras junções que o grupo considere como sendo as de principal importância para o trabalho

1) Join evento e tipo_evento

In [130]:

```
result32 = pd.read_sql_query("""
    select nome_evento,tipo_desc from evento ev inner join evento_p
    on (ev.cod_evento = ep.fk_evento_cod_evento)
    inner join tipo_evento tp
    on (ep.fk_tipo_evento_cod_tipo = tp.cod_tipo)
    order by tipo_desc
""",conn)
```

In [131]:

```
result32
```

Out[131]:

	nome_evento	tipo_desc
0	Leitura Coletiva	Encontro
1	Encontro de Leitores	Encontro
2	Doação de Livros	Outros
3	Livros Antigos	Outros
4	Feira de livros	Outros
5	Sebo Veredas	Outros
6	Venda de livros usados	Outros

2) Join livro e gênero

In [136]:

```
result33 = pd.read_sql_query("""
    select nome_livro,genero_desc from livro lv inner join livro_ge
    on (lv.cod_livro = lg.fk_livro_cod_livro)
    inner join genero ge
    on (lg.fk_genero_cod_genero = ge.cod_genero)
    order by genero_desc
""",conn)
```

In [137]:

```
result33
```

Out[137]:

	nome_livro	genero_desc
0	Tartarugas até lá embaixo	Ficção
1	A Metamorfose	Ficção
2	A Culpa é das Estrelas	Romance
3	Os Miseráveis	Romance
4	Dom Casmurro	Romance
5	Helena	Romance
6	Norwegian Wood	Romance
7	Kafka a beira-mar	Romance
8	O Iluminado	Terror
9	Goosebumps	Terror

3) Join pessoa , cidade, estado e país

In [150]:

```
result34 = pd.read_sql_query("""
    select nome,cidade_desc,estado_desc,pais_desc from pessoa pe
    inner join cidade ci on (pe.fk_cidade_cod_cidade = ci.cod_cidade)
    inner join estado es on (ci.fk_estado_cod_estado = es.cod_estado)
    inner join pais pa on (es.fk_pais_cod_pais = pa.cod_pais)
    order by cod_pessoa
""",conn)
```

In [151]:

result34

Out[151]:

	nome	cidade_desc	estado_desc	pais_desc
0	João	Vitória	Espírito Santo	Brasil
1	Pedro	Serra	Espírito Santo	Brasil
2	José	Serra	Espírito Santo	Brasil
3	Maria	Campos	Rio de Janeiro	Brasil
4	Joana	Campinas	São Paulo	Brasil
5	Mário	Campinas	São Paulo	Brasil
6	Plínio	São Paulo	São Paulo	Brasil
7	Fernando	São Paulo	São Paulo	Brasil
8	Rafael	São Paulo	São Paulo	Brasil
9	Carlos	São Paulo	São Paulo	Brasil
10	Leonardo	São Paulo	São Paulo	Brasil
11	John	Indianápolis	Indiana	Estados Unidos
12	Bob	Portland	Maine	Estados Unidos
13	Fred	Nova Iorque	Nova Iorque	Estados Unidos
14	Haru	Kyoto	Kyoto	Japão
15	Machado de Assis	Campos	Rio de Janeiro	Brasil
16	Stephen King	Portland	Maine	Estados Unidos
17	Victor Hugo	Besançon	Doubs	França
18	Franz Kafka	Praga	Boemia	Áustria
19	Haruki Murakami	Kyoto	Kyoto	Japão
20	John Green	Indianápolis	Indiana	Estados Unidos
21	Robert Lawrence	Nova Iorque	Nova Iorque	Estados Unidos

4) Join pessoa (autor), escreve_livro e livro

In [158]:

```
result35 = pd.read_sql_query("""
    select nome as autor,nome_livro as livro_escrito from pessoa pe
    on (pe.cod_pessoa = el_fk_pessoa_cod_pessoa)
    inner join livro lv
    on (el_fk_livro_cod_livro = lv.cod_livro)
    order by nome
    """ ,conn)
```

In [159]:

result35

Out[159]:

	autor	livro_escrito
0	Franz Kafka	A Metamorfose
1	Haruki Murakami	Norwegian Wood
2	Haruki Murakami	Kafka a beira-mar
3	John Green	Tartarugas até lá embaixo
4	John Green	A Culpa é das Estrelas
5	Machado de Assis	Helena
6	Machado de Assis	Dom Casmurro
7	Robert Lawrence	Goosebumps
8	Stephen King	O Iluminado
9	Victor Hugo	Os Miseráveis

5) Join evento e cidade (onde acontece)

In [164]:

```
result36 = pd.read_sql_query("""
    select nome_evento,cidade_desc,data_evento,hora_evento
    from evento ev inner join evento_acontece ea
    on (ev.cod_evento = ea.fk_evento_cod_evento)
    inner join cidade ci
    on (ea.fk_cidade_cod_cidade = ci.cod_cidade)
    order by cidade_desc
    """,conn)
```

In [165]:

result36

Out[165]:

	nome_evento	cidade_desc	data_evento	hora_evento
0	Doação de Livros	Campos	2019-03-04	12:00:00
1	Feira de livros	Indianápolis	2019-08-10	15:00:00
2	Venda de livros usados	Portland	2019-09-12	20:00:00
3	Sebo Veredas	Portland	2019-01-11	19:00:00
4	Livros Antigos	São Paulo	2019-05-05	11:00:00
5	Encontro de Leitores	Serra	2019-02-01	18:00:00
6	Leitura Coletiva	Vitória	2019-01-01	10:00:00

9.7 CONSULTAS COM GROUP BY E FUNÇÕES DE AGRUPAMENTO (Mínimo 6)

1) Quantidade de usuários que leram um livro

In [20]:

```
result37 = pd.read_sql_query("""
    select nome_livro, count(fk_usuario_fk_pessoa_cod_pessoa) as qt
    inner join leu on (livro.cod_livro = leu.fk_livro_cod_livro) gr
    """,conn)
```

In [21]:

```
result37
```

Out[21]:

	nome_livro	qtd_usu_leram
0	Goosebumps	8
1	Kafka a beira-mar	1
2	Helena	8
3	Norwegian Wood	3
4	A Culpa é das Estrelas	3
5	Dom Casmurro	4
6	O Iluminado	8
7	Os Miseráveis	4
8	Tartarugas até lá embaixo	3
9	A Metamorfose	4

2) Quantidade de pessoas cadastradas por estado

In [38]:

```
result38 = pd.read_sql_query("""
    select estado_desc,count(cidade_desc) from cidade inner join es
    on (cidade.fk_estado_cod_estado = estado.cod_estado)
    group by cod_estado,estado_desc
    """,conn)
```

In [39]:

```
result38
```

Out[39]:

	estado_desc	count
0	Indiana	1
1	Espírito Santo	2
2	São Paulo	2
3	Doubs	1
4	Rio de Janeiro	1
5	Maine	1
6	Boemia	1
7	Nova Iorque	1
8	Kyoto	1

3) Mostra a quantidade de estados cadastrados em cada país

In [36]:

```
result39 = pd.read_sql_query("""
    select pais_desc,count(estado_desc) from estado inner join pais
    (estado.fk_pais_cod_pais = pais.cod_pais)
    group by cod_pais,pais_desc
    """ ,conn)
```

In [37]:

```
result39
```

Out[37]:

	pais_desc	count
0	Estados Unidos	3
1	Brasil	3
2	França	1
3	Japão	1
4	Áustria	1

4) Mostra o número de livros escritos por autor

In [47]:

```
result40 = pd.read_sql_query("""
    select cod_pessoa,nome,count(fk_livro_cod_livro) as num_livros_
    (pessoa.cod_pessoa = escreve_livro.fk_pessoa_cod_pessoa)
    inner join livro on
    (livro.cod_livro = escreve_livro.fk_livro_cod_livro)
    group by cod_pessoa
    order by cod_pessoa desc
    """ ,conn)
```

In [48]:

result40

Out[48]:

	cod_pessoa	nome	num_livros_escritos
0	22	Robert Lawrence	1
1	21	John Green	2
2	20	Haruki Murakami	2
3	19	Franz Kafka	1
4	18	Victor Hugo	1
5	17	Stephen King	1
6	16	Machado de Assis	2

5) Mostra o número de amigos de cada usuário

In [42]:

```
result41 = pd.read_sql_query("""
    select nome, count(fk_usuario_fk_pessoa_cod_pessoa_) as num_amigos
    inner join pessoa on (usuario.fk_pessoa_cod_pessoa = pessoa.cod_pessoa)
    inner join amizade on (usuario.fk_pessoa_cod_pessoa = amizade.fk_pessoa_cod_pessoa)
    group by nome order by num_amigos desc
    """ ,conn)
```

In [43]:

result41

Out[43]:

	nome	num_amigos
0	José	6
1	Pedro	5
2	Joana	5
3	Maria	5
4	João	4
5	John	2
6	Leonardo	2
7	Mário	2
8	Plínio	2
9	Carlos	2
10	Bob	2
11	Rafael	1
12	Fred	1
13	Haru	1

6) Mostra o número de livros lidos por cada usuário

In [46]:

```
result42 = pd.read_sql_query("""
    select nome, count(fk_livro_cod_livro) as num_livros_lidos from
    inner join pessoa on (usuario.fk_pessoa_cod_pessoa = pessoa.cod_pessoa)
    inner join leu on (usuario.fk_pessoa_cod_pessoa = leu.fk_usuario)
    group by nome order by num_livros_lidos desc
    """ ,conn)
```

In [47]:

result42

Out[47]:

	nome	num_livros_lidos
0	Haru	10
1	John	4
2	Fred	4
3	Joana	4
4	Plínio	4
5	Pedro	3
6	Fernando	3
7	João	3
8	José	3
9	Maria	2
10	Carlos	2
11	Bob	2
12	Mário	1
13	Rafael	1

9.8 CONSULTAS COM LEFT E RIGHT JOIN (Mínimo 4)

1) Mostra os dados relacionados à tabela pessoa (quem não for usuário não possui email ou senha)

In [54]:

```
result43 = pd.read_sql_query("""
    select * from pessoa left outer join usuario on (pessoa.cod_pes
    """",conn)
```

In [55]:

result43

Out[55]:

	cod_pessoa	nome	fk_cidade_cod_cidade	email	
0	1	João	1	joao@gmail	aca3ec7278a47144c0a863e60c5
1	2	Pedro	2	pedro@gmail	77fe0359ee910002d306e2991d
2	3	José	2	jose@gmail	e7d80ffefafa212b7c5c55700e41
3	4	Maria	3	maria@gmail	f8461b554d59b3014e8ff5165d
4	5	Joana	4	joana@gmail	f153d147e459b3ebca47d59fdf1
5	6	Mário	4	mario@gmail	190661875f0470d6fd4ed9811b0
6	7	Plínio	5	plinio@gmail	0740aeab01c11bc05bd503652d3
7	8	Fernando	5	fernando@gmail	fbc4fef99cdd459c958f92ef36
8	9	Rafael	5	rafa@gmail	2c13817fca846f6f9a7d934d71a
9	10	Carlos	5	carlos@gmail	a261108357a8dbcb6a891b1efd4
10	11	Leonardo	5	leo@gmail	657b298b04e033810343842f993
11	12	John	6	john@gmail	6e0b7076126a29d5dfcbd548353
12	13	Bob	7	bob@gmail	2acba7f51acf4fd5102ad090fc
13	14	Fred	8	fred@gmail	77064f5bd13e417f564e7d880dc
14	15	Haru	10	haru@gmail	aecab1503021d470bc5cde7ae0e
15	16	Machado de Assis	3	None	
16	17	Stephen King	7	None	
17	18	Victor Hugo	9	None	
18	19	Franz Kafka	11	None	
19	20	Haruki Murakami	10	None	
20	21	John Green	6	None	
21	22	Robert Lawrence	8	None	

2) Mostra em quais cidades ocorrerão eventos (há cidades que não possuem eventos)

In [65]:

```
result44 = pd.read_sql_query("""
    select cod_cidade,cidade_desc,cod_evento,nome_evento from evento
    on (evento.cod_evento = evento_acontece.fk_evento_cod_evento)
    right outer join cidade on
    (cidade.cod_cidade = evento_acontece.fk_evento_cod_evento)
    """",conn)
```

In [66]:

result44

Out[66]:

	cod_cidade	cidade_desc	cod_evento	nome_evento
0	1	Vitória	1.0	Leitura Coletiva
1	2	Serra	2.0	Encontro de Leitores
2	3	Campos	3.0	Doação de Livros
3	4	Campinas	4.0	Livros Antigos
4	5	São Paulo	5.0	Feira de livros
5	6	Indianápolis	6.0	Sebo Veredas
6	7	Portland	7.0	Venda de livros usados
7	11	Praga	NaN	None
8	10	Kyoto	NaN	None
9	8	Nova Iorque	NaN	None
10	9	Besançon	NaN	None

3) Mostra quais livros foram escritos por cada pessoa (usuários não possuem livros escritos)

In [85]:

```
result45 = pd.read_sql_query("""
    select cod_pessoa,nome,cod_livro,nome_livro from pessoa left ou
    (pessoa.cod_pessoa = escreve_livro.fk_pessoa_cod_pessoa)
    left outer join livro on
    (livro.cod_livro = escreve_livro.fk_livro_cod_livro)
    order by cod_pessoa desc
    """",conn)
```

In [86]:

result45

Out[86]:

	cod_pessoa	nome	cod_livro	nome_livro
0	22	Robert Lawrence	3.0	Goosebumps
1	21	John Green	9.0	A Culpa é das Estrelas
2	21	John Green	10.0	Tartarugas até lá embaixo
3	20	Haruki Murakami	7.0	Norwegian Wood
4	20	Haruki Murakami	8.0	Kafka a beira-mar
5	19	Franz Kafka	6.0	A Metamorfose
6	18	Victor Hugo	4.0	Os Miseráveis
7	17	Stephen King	2.0	O Iluminado
8	16	Machado de Assis	1.0	Helena
9	16	Machado de Assis	5.0	Dom Casmurro
10	15	Haru	NaN	None
11	14	Fred	NaN	None
12	13	Bob	NaN	None
13	12	John	NaN	None
14	11	Leonardo	NaN	None
15	10	Carlos	NaN	None
16	9	Rafael	NaN	None
17	8	Fernando	NaN	None
18	7	Plínio	NaN	None
19	6	Mário	NaN	None
20	5	Joana	NaN	None
21	4	Maria	NaN	None
22	3	José	NaN	None
23	2	Pedro	NaN	None
24	1	João	NaN	None

4) Mostra as postagens referentes a tal livro (alguns livros não possuem postagens)

In [89]:

```
result46 = pd.read_sql_query("""
    select * from postagem_posta_comentario right outer join livro
    on (livro.cod_livro = postagem_posta_comentario.fk_livro_cod_li
    """",conn)
```

In [90]:

result46

Out[90]:

	cod_postagem	data_postagem	hora_postagem	fk_livro_cod_livro	fk_usuario_fk_pessoa_cod_pessoa
0	1.0	2018-02-04	12:00:00	1.0	
1	2.0	2018-02-05	11:30:00	2.0	
2	3.0	2018-04-06	08:45:00	3.0	
3	4.0	2018-05-06	18:00:00	4.0	
4	5.0	2018-05-06	20:00:00	5.0	
5	6.0	2018-06-07	07:00:00	6.0	
6	7.0	2018-06-07	09:30:00	7.0	
7	8.0	2018-06-07	15:00:00	7.0	
8	9.0	2018-07-07	18:00:00	7.0	
9	10.0	2018-08-07	20:00:00	8.0	
10	NaN	None	None	NaN	
11	NaN	None	None	NaN	

9.9 CONSULTAS COM SELF JOIN E VIEW (Mínimo 6)

a) Uma junção que envolva Self Join

Mostra os amigos de cada usuário

In [14]:

```
result47 = pd.read_sql_query("""
    select p1.nome,p2.nome amigo from amizade am
    inner join pessoa p1 on (p1.cod_pessoa = am.fk_usuario_fk_pesso
    inner join pessoa p2 on (p2.cod_pessoa = am.fk_amigo_fk_pesso
    order by p1.nome
    """",conn)
```

In [15]:

result47

Out[15]:

	nome	amigo
0	Bob	Plínio
1	Carlos	Pedro
2	Fred	João
3	Fred	José
4	Fred	Pedro
5	Haru	Mário
6	Haru	Rafael
7	Haru	Plínio
8	Joana	Haru
9	Joana	Carlos
10	Joana	John
11	João	Pedro
12	João	José
13	João	Maria
14	John	Joana
15	José	Maria
16	José	Mário
17	José	Joana
18	Leonardo	John
19	Leonardo	Maria
20	Leonardo	Bob
21	Maria	Bob
22	Maria	José
23	Maria	Joana
24	Maria	Leonardo
25	Maria	João
26	Maria	Fred
27	Maria	Pedro
28	Mário	João
29	Mário	José
30	Pedro	José
31	Pedro	Joana
32	Pedro	Maria
33	Plínio	Joana

	nome	amigo
34	Plínio	Maria
35	Plínio	José
36	Plínio	Pedro
37	Plínio	João
38	Rafael	Leonardo
39	Rafael	Carlos

b) Outras junções com views que o grupo considere como sendo de relevante importância para o trabalho**1) Visão informacoes_livro**

```
create view informacoes_livro as select cod_livro,nome_livro,genero_desc,
cod_genero,pe.nome,cod_pessoa
from livro lv inner join livro_genero lg on (lv.cod_livro = lg.fk_livro_cod_livro)
inner join genero ge on (ge.cod_genero = lg.fk_genero_cod_genero)
inner join escreve_livro el on (lv.cod_livro = el.fk_livro_cod_livro)
inner join pessoa pe on (pe.cod_pessoa = el.fk_pessoa_cod_pessoa)
order by cod_livro
```

In [26]:

```
result48 = pd.read_sql_query("""
                            select * from informacoes_livro
                            """",conn)
```

In [27]:

result48

Out[27]:

	cod_livro	nome_livro	genero_desc	cod_genero	nome	cod_pessoa
0	1	Helena	Romance	2	Machado de Assis	16
1	2	O Iluminado	Terror	1	Stephen King	17
2	3	Goosebumps	Terror	1	Robert Lawrence	22
3	4	Os Miseráveis	Romance	2	Victor Hugo	18
4	5	Dom Casmurro	Romance	2	Machado de Assis	16
5	6	A Metamorfose	Ficção	3	Franz Kafka	19
6	7	Norwegian Wood	Romance	2	Haruki Murakami	20
7	8	Kafka a beira-mar	Romance	2	Haruki Murakami	20
8	9	A Culpa é das Estrelas	Romance	2	John Green	21
9	10	Tartarugas até lá embaixo	Ficção	3	John Green	21

2) Visão informacoes_evento

```
create view informacoes_evento as select cod_evento,nome_evento,tipos_desc,cod_tipo,cidade_desc,
cidade_desc,pais_desc,data_evento,hora_evento
from evento ev inner join evento_possui_tipo ep on (ev.cod_evento = ep.fk_evento_cod_evento)
inner join tipo_evento te on (ep.fk_tipo_evento_cod_tipo = te.cod_tipo)
inner join evento_acaente ea on (ea.fk_evento_cod_evento = ev.cod_evento)
inner join cidade ci on (ci.cod_cidade = ea.fk_cidade_cod_cidade)
inner join estado es on (ci.fk_estado_cod_estado = es.cod_estado)
inner join pais pa on (es.fk_pais_cod_pais = pa.cod_pais)
```

In [21]:

```
result49 = pd.read_sql_query("""
    select * from informacoes_evento
""",conn)
```

In [22]:

result49

Out[22]:

	cod_evento	nome_evento	tipo_desc	cod_tipo	cidade_desc	estado_desc	pais_desc	data_
0	1	Leitura Coletiva	Encontro	1	Vitória	Espírito Santo	Brasil	2019
1	2	Encontro de Leitores	Encontro	1	Serra	Espírito Santo	Brasil	2019
2	3	Doação de Livros	Outros	3	Campos	Rio de Janeiro	Brasil	2019
3	4	Livros Antigos	Outros	3	São Paulo	São Paulo	Brasil	2019
4	5	Feira de livros	Outros	3	Indianápolis	Indiana	Estados Unidos	2019
5	6	Sebo Veredas	Outros	3	Portland	Maine	Estados Unidos	2019
6	7	Venda de livros usados	Outros	3	Portland	Maine	Estados Unidos	2019

3) Visão cidade_estado_pais

```
create view cidade_estado_pais as select cod_cidade,cidade_desc,cod_estado,estado_desc,cod_pais,pais_desc
from cidade ci inner join estado es on (ci.fk_estado_cod_estado = es.cod_estado)
inner join pais pa on (es.fk_pais_cod_pais = pa.cod_pais)
```

In [24]:

```
result50 = pd.read_sql_query("""
    select * from cidade_estado_pais
    """,conn)
```

In [25]:

```
result50
```

Out[25]:

	cod_cidade	cidade_desc	cod_estado	estado_desc	cod_pais	pais_desc
0	1	Vitória	1	Espírito Santo	1	Brasil
1	2	Serra	1	Espírito Santo	1	Brasil
2	3	Campos	2	Rio de Janeiro	1	Brasil
3	4	Campinas	3	São Paulo	1	Brasil
4	5	São Paulo	3	São Paulo	1	Brasil
5	6	Indianápolis	4	Indiana	2	Estados Unidos
6	7	Portland	5	Maine	2	Estados Unidos
7	8	Nova Iorque	6	Nova Iorque	2	Estados Unidos
8	9	Besançon	7	Doubs	3	França
9	10	Kyoto	8	Kyoto	4	Japão
10	11	Praga	9	Boemia	5	Áustria

4) Visão pessoa_usuario

```
create view pessoa_usuario as select cod_pessoa, nome, email, senha  
from pessoa p inner join usuario u on (p.cod_pessoa = u.fk_pessoa_cod_pessoa)
```

In [28]:

```
result51 = pd.read_sql_query("""  
    select * from pessoa_usuario  
""", conn)
```

In [29]:

result51

Out[29]:

	cod_pessoa	nome	email	senha
0	1	João	joao@gmail	aca3ec7278a47144c0a863e60c595abe
1	2	Pedro	pedro@gmail	77fe0359ee910002d306e2991d8ec27f
2	3	José	jose@gmail	e7d80ffefea212b7c5c55700e4f7193e
3	4	Maria	maria@gmail	f8461b554d59b3014e8ff5165dc62fac
4	5	Joana	joana@gmail	f153d147e459b3ebca47d59fdf179ecd
5	6	Mário	mario@gmail	190661875f0470d6fd4ed9811b07322d
6	7	Plínio	plinio@gmail	0740aeab01c11bc05bd503652d3be878
7	8	Fernando	fernando@gmail	fbc4fef99cdd459c958f92ef36c9fe2b
8	9	Rafael	rafa@gmail	2c13817fca846f6f9a7d934d71a668ee
9	10	Carlos	carlos@gmail	a261108357a8dbcb6a891b1efd40a794
10	11	Leonardo	leo@gmail	657b298b04e033810343842f993c9817
11	12	John	john@gmail	6e0b7076126a29d5dfcbd54835387b7b
12	13	Bob	bob@gmail	2acba7f51acf4fd5102ad090fc612ee
13	14	Fred	fred@gmail	77064f5bd13e417f564e7d880dc7a536
14	15	Haru	haru@gmail	aecab1503021d470bc5cde7ae0e87032

5) Visão curtiu_postagem

```
create view curtiu_postagem as select cod_pessoa , nome, cod_postagem as curtiu_postagem_cod
from pessoa p inner join curte c on (p.cod_pessoa = c.fk_usuario_fk_pessoa_cod_pessoa)
inner join postagem_posta_comentario post on (c.fk_postagem_posta_comentario_cod_postagem = post.cod_postagem)
```

In [30]:

```
result52 = pd.read_sql_query("""
    select * from curtiu_postagem
    """, conn)
```

In [31]:

result52

Out[31]:

	cod_pessoa	nome	curtiu_postagem_cod
0	1	João	1
1	1	João	2
2	2	Pedro	2
3	3	José	1
4	4	Maria	3
5	4	Maria	4
6	5	Joana	2
7	6	Mário	2
8	7	Plínio	5
9	7	Plínio	6
10	7	Plínio	7
11	8	Fernando	1
12	8	Fernando	8
13	9	Rafael	9
14	9	Rafael	10
15	10	Carlos	5
16	10	Carlos	6
17	10	Carlos	7
18	11	Leonardo	3
19	11	Leonardo	9
20	12	John	1
21	13	Bob	4
22	13	Bob	5
23	14	Fred	7
24	14	Fred	9
25	15	Haru	6
26	15	Haru	8

9.10 SUBCONSULTAS (Mínimo 3)

1) Nome dos livros de um determinado gênero

In [38]:

```
result53 = pd.read_sql_query("""
    select nome_livro nome_livro_genero from livro_genero lg inner
    where fk_genero_cod_genero in (select cod_genero from genero wh
    """",conn)
```

In [39]:

result53

Out[39]:

	nome_livro_genero
0	Helena
1	Os Miseráveis
2	Dom Casmurro
3	Norwegian Wood
4	Kafka a beira-mar
5	A Culpa é das Estrelas

2) Lista os nomes de eventos, tipo de evento onde o nome contém a palavra "livro"

In [40]:

```
result54 = pd.read_sql_query("""
    select nome_evento,(select tipo_desc from tipo_evento where cod
    from evento
    inner join evento_possui_tipo on (evento.cod_evento = evento_pc
    where nome_evento ilike '%livro%'
    """",conn)
```

In [41]:

result54

Out[41]:

	nome_evento	tipo_desc
0	Doação de Livros	Outros
1	Livros Antigos	Outros
2	Feira de livros	Outros
3	Venda de livros usados	Outros

3) Mostra livros que possuem apenas uma postagem

In [42]:

```
result55 = pd.read_sql_query("""
    select nome_livro from livro where (select count(cod_postagem)
    from postagem_posta_comentario pp where pp.fk_livro_cod_livro =
    """",conn)
```

In [43]:

result55

Out[43]:

	nome_livro
0	Helena
1	O Iluminado
2	Goosebumps
3	Os Miseráveis
4	Dom Casmurro
5	A Metamorfose
6	Kafka a beira-mar

In []: