

Para acessar o documento no Overleaf, basta clicar neste link:

<https://www.overleaf.com/read/qhcqyyhpcsqw>

Alunos: Lucas Carrijo Ferrari e Marco Antônio da Silva Alves

Grupo: 10

1. Junções Internas [1]

```
1 SELECT
2   O.fname AS Nome_Administrador ,
3   O.lname AS Sobrenome_Administrador ,
4   B.name AS Nome_Empresa ,
5   C.city AS Cidade
6 FROM
7   officer O
8 JOIN business B ON
9   O.cust_id = B.cust_id
10 JOIN customer C ON
11   B.cust_id = C.cust_id
12 WHERE
13   O.title IS NOT NULL;
```

- Estamos selecionando as colunas `fname` e `lname` da tabela `officer` para obter o nome e sobrenome do administrador.
- Estamos selecionando a coluna `name` da tabela `business` para obter o nome da empresa.
- Estamos selecionando a coluna `city` da tabela `customer` para obter a cidade onde a empresa está presente.
- Estamos usando `JOIN` para combinar as tabelas `officer`, `business`, e `customer` com base nas chaves `cust_id`.
- Estamos usando a cláusula `WHERE` para garantir que estamos apenas selecionando os administradores que têm um título (`O.title IS NOT NULL`), ou seja, estamos excluindo aqueles sem título.

2. Junções Internas, União e Seleção [2]

```
1 SELECT
2   CASE
3     WHEN c.cust_type_cd = 'I' THEN CONCAT(i.fname, ' ', i.lname)
4     WHEN c.cust_type_cd = 'B' THEN b.name
5   END AS customer_name
6 FROM
7   account a
8 JOIN customer c ON
9   a.cust_id = c.cust_id
```

```
10 LEFT JOIN individual i ON
11     c.cust_id = i.cust_id
12     AND c.cust_type_cd = 'I'
13 LEFT JOIN business b ON
14     c.cust_id = b.cust_id
15     AND c.cust_type_cd = 'B'
16 JOIN branch br ON
17     a.open_branch_id = br.branch_id
18 WHERE
19     br.city <> c.city;
```

- O script usa uma instrução **SELECT** para criar uma lista de nomes de clientes.
- Ele usa a cláusula **CASE** para verificar o tipo de cliente (**cust_type_cd**) e, com base nisso, construir o nome do cliente. Se o tipo de cliente for 'I' (individual), ele concatena o primeiro nome (**fname**) e o sobrenome (**lname**) do cliente. Se o tipo de cliente for 'B' (business), ele usa o nome da empresa (**name**).
- A consulta está vinculando as tabelas **account**, **customer**, **individual**, **business** e **branch** usando várias junções (**JOIN**).
- O **JOIN** entre a tabela **account** e **customer** é feito com base no campo **cust_id**, que é comum às duas tabelas e relaciona uma conta a um cliente.
- As tabelas **individual** e **business** são vinculadas à tabela **customer** usando junções **LEFT JOIN**. Isso permite que o script traga informações sobre clientes individuais e comerciais, dependendo do valor de **cust_type_cd**.
- Além disso, o script vincula a tabela **branch** para verificar se a cidade da agência (**br.city**) é diferente da cidade do cliente (**c.city**). Isso é feito para encontrar clientes que possuem uma conta em uma cidade diferente da cidade de estabelecimento.
- A cláusula **WHERE** impõe a condição em que a cidade da agência (**br.city**) deve ser diferente da cidade do cliente (**c.city**).

3. Alternativa na 3 [3]

```
1 SELECT
2     e.fname,
3     e.lname,
4     YEAR(t.txn_date) AS ano,
5     COUNT(t.txn_id) AS num_transacoes
6 FROM
7     employee e
8 LEFT JOIN account a ON
9     e.emp_id = a.open_emp_id
10 LEFT JOIN 'transaction' t ON
11     a.account_id = t.account_id
12 GROUP BY
13     e.emp_id,
```

```
14 ano
15 ORDER BY
16 e.lname,
17 e.fname,
18 ano;
```

- A consulta seleciona os campos `fname` e `lname` da tabela `employee` para obter os nomes dos funcionários.
- Utiliza a função `YEAR()` para extrair o ano da coluna `txn_date` da tabela `transaction`. Isso agrupa as transações por ano.
- Usa a função `COUNT()` para contar o número de transações (representadas pelo `txn_id`) para cada funcionário em cada ano.
- A consulta realiza um `LEFT JOIN` entre as tabelas `employee`, `account`, e `transaction`. O `LEFT JOIN` é usado para garantir que todos os funcionários sejam incluídos na lista, mesmo que não tenham contas ou transações associadas a eles.
- Os `JOINS` são feitos com base nas relações entre as tabelas, usando as colunas `emp_id` de `employee`, `open_emp_id` de `account`, e `account_id` de `transaction`.
- Os resultados da consulta são agrupados por `emp_id` (identificador do funcionário) e pelo ano da transação (`ano`).
- Por fim, os resultados são ordenados alfabeticamente pelos sobrenomes (`lname`) e, em seguida, pelo ano em ordem crescente.

4. Junções Internas, Agrupamento, Agregação, União e Concatenação [4]

```
1 SELECT
2   b.branch_id AS agencia_id,
3   b.name AS nome_agencia,
4   a.account_id AS conta_id,
5   CASE
6     WHEN c.cust_type_cd = 'I' THEN CONCAT(i.fname, ' ', i.lname)
7     WHEN c.cust_type_cd = 'B' THEN b2.name
8   END AS nome_titular,
9   a.avail_balance AS saldo
10 FROM
11   account AS a
12 INNER JOIN branch AS b ON
13   a.open_branch_id = b.branch_id
14 INNER JOIN customer AS c ON
15   a.cust_id = c.cust_id
16 LEFT JOIN individual AS i ON
17   c.cust_type_cd = 'I'
18   AND c.cust_id = i.cust_id
19 LEFT JOIN business AS b2 ON
20   c.cust_type_cd = 'B'
```

```

21  AND c.cust_id = b2.cust_id
22  WHERE
23  a.avail_balance = (
24  SELECT
25      MAX(avail_balance)
26  FROM
27      account
28  WHERE
29      open_branch_id = b.branch_id
30  );

```

- Selecionamos as colunas que queremos na saída, incluindo o identificador da agência (**branch_id**), o nome da agência (**name**), o identificador da conta (**account_id**), o nome do titular (**nome_titular**), e o saldo disponível da conta (**avail_balance**).
- Usamos uma junção interna (**INNER JOIN**) para combinar as tabelas **account** e **branch** com base no identificador da agência.
- Usamos outra junção interna para combinar a tabela **account** com a tabela **customer** com base no identificador do cliente (**cust_id**).
- Usamos junções à esquerda (**LEFT JOIN**) para combinar a tabela **customer** com as tabelas **individual** e **business** com base no tipo de cliente (**cust_type_cd**). Isso nos permite obter o nome do titular, que pode ser uma pessoa física ou uma empresa.
- Na cláusula **WHERE**, selecionamos apenas as linhas em que o saldo disponível da conta é igual ao maior saldo disponível daquela agência. Isso nos dá a conta com o maior saldo de dinheiro em cada agência.

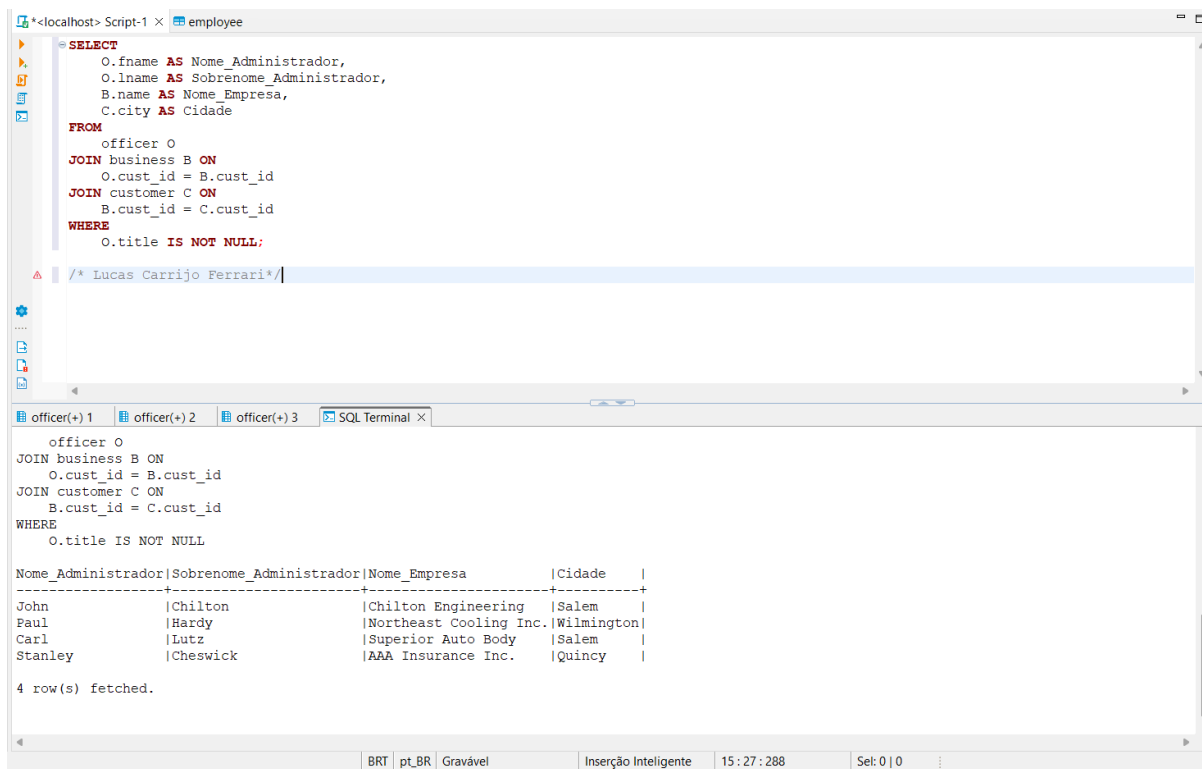
5. Visualização

```

1  /* 2 Modularizada */
2  CREATE VIEW customer_names_view AS
3  SELECT
4      CASE
5          WHEN c.cust_type_cd = 'I' THEN CONCAT(i.fname, ' ', i.lname)
6          WHEN c.cust_type_cd = 'B' THEN b.name
7      END AS customer_name
8  FROM
9      account a
10     JOIN customer c ON a.cust_id = c.cust_id
11     LEFT JOIN individual i ON c.cust_id = i.cust_id AND c.cust_type_cd = 'I'
12     LEFT JOIN business b ON c.cust_id = b.cust_id AND c.cust_type_cd = 'B'
13     JOIN branch br ON a.open_branch_id = br.branch_id
14  WHERE
15      br.city <> c.city;
16
17  /* 4 Modularizada */
18  CREATE VIEW view_saldo_max_agencia AS

```

```
19 SELECT
20     b.branch_id AS agencia_id,
21     b.name AS nome_agencia,
22     a.account_id AS conta_id,
23     CASE
24         WHEN c.cust_type_cd = 'I' THEN CONCAT(i.fname, ' ', i.lname)
25         WHEN c.cust_type_cd = 'B' THEN b2.name
26     END AS nome_titular,
27     a.avail_balance AS saldo
28 FROM account AS a
29 INNER JOIN branch AS b ON a.open_branch_id = b.branch_id
30 INNER JOIN customer AS c ON a.cust_id = c.cust_id
31 LEFT JOIN individual AS i ON c.cust_type_cd = 'I' AND c.cust_id = i.
    cust_id
32 LEFT JOIN business AS b2 ON c.cust_type_cd = 'B' AND c.cust_id = b2.
    cust_id
33 WHERE a.avail_balance = (
34     SELECT MAX(avail_balance)
35     FROM account
36     WHERE open_branch_id = b.branch_id
37 );
```



The screenshot shows a SQL IDE window with a script editor and a results pane. The script editor contains a SQL query that performs an internal join between the officer, business, and customer tables. The results pane displays the output of the query, showing four rows of data.

```
SELECT
  O.fname AS Nome_Administrador,
  O.lname AS Sobrenome_Administrador,
  B.name AS Nome_Empresa,
  C.city AS Cidade
FROM
  officer O
JOIN business B ON
  O.cust_id = B.cust_id
JOIN customer C ON
  B.cust_id = C.cust_id
WHERE
  O.title IS NOT NULL;
```

/* Lucas Carrijo Ferrari*/

```
officer O
JOIN business B ON
O.cust_id = B.cust_id
JOIN customer C ON
B.cust_id = C.cust_id
WHERE
O.title IS NOT NULL
```

Nome_Administrador	Sobrenome_Administrador	Nome_Empresa	Cidade
John	Chilton	Chilton Engineering	Salem
Paul	Hardy	Northeast Cooling Inc.	Wilmington
Carl	Lutz	Superior Auto Body	Salem
Stanley	Cheswick	AAA Insurance Inc.	Quincy

4 row(s) fetched.

BRT | pt_BR | Gravável | Inserção Inteligente | 15:27:288 | Sel: 0 | 0

Figura 1: Junções Internas

The screenshot shows a SQL IDE window titled "Script-1" with a query that uses internal joins and a case statement. The query is as follows:

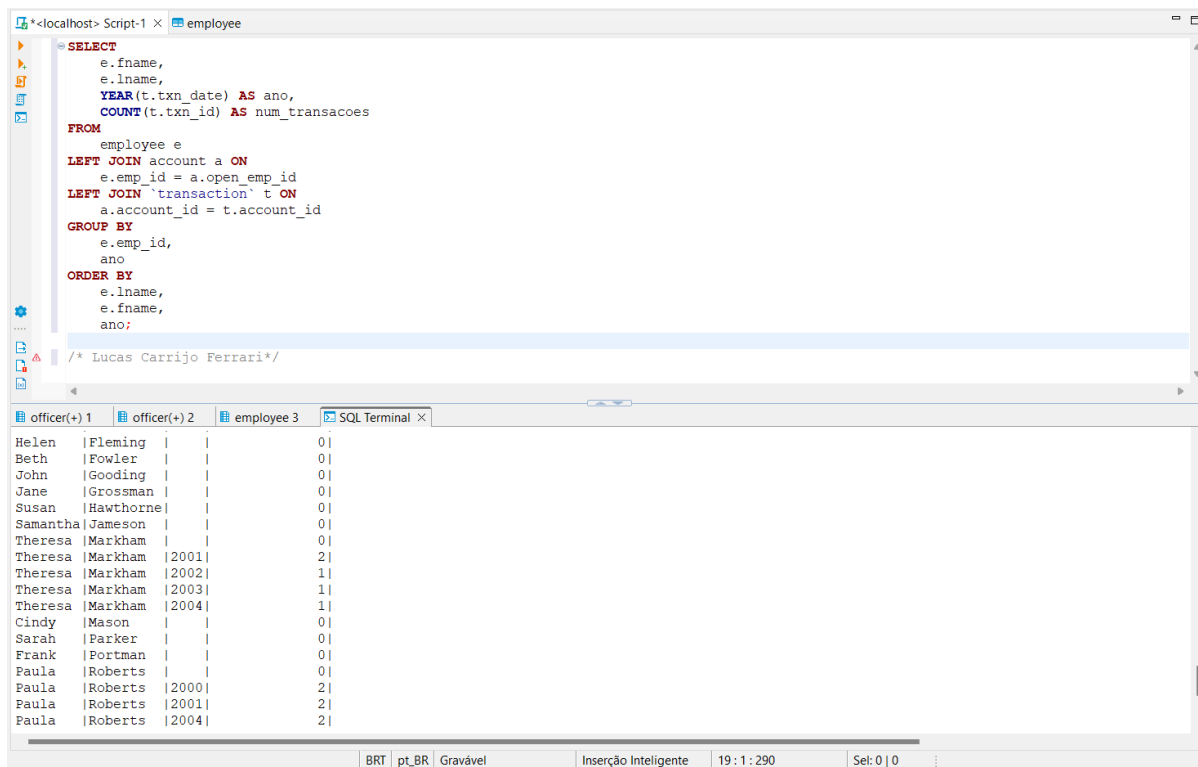
```
SELECT
  CASE
    WHEN c.cust_type_cd = 'I' THEN CONCAT(i.fname, ' ', i.lname)
    WHEN c.cust_type_cd = 'B' THEN b.name
  END AS customer_name
FROM
  account a
  JOIN customer c ON
    a.cust_id = c.cust_id
  LEFT JOIN individual i ON
    c.cust_id = i.cust_id
    AND c.cust_type_cd = 'I'
  LEFT JOIN business b ON
    c.cust_id = b.cust_id
    AND c.cust_type_cd = 'B'
  JOIN branch br ON
    a.open_branch_id = br.branch_id
WHERE
  br.city <> c.city;
```

Below the query editor, the results are displayed in a table with the column "customer_name". The results are:

customer_name
James Hadley
James Hadley
James Hadley
Margaret Young
Richard Farley
Richard Farley
Richard Farley
Northeast Cooling Inc.

At the bottom of the results pane, it says "8 row(s) fetched." The status bar at the very bottom of the IDE shows "BRT | pt_BR | Gravável | Inserção Inteligente | 21:27:468 | Set: 0 | 0".

Figura 2: Junções Internas, União e Seleção



The screenshot shows a SQL IDE window with a query editor and a results pane. The query editor contains the following SQL code:

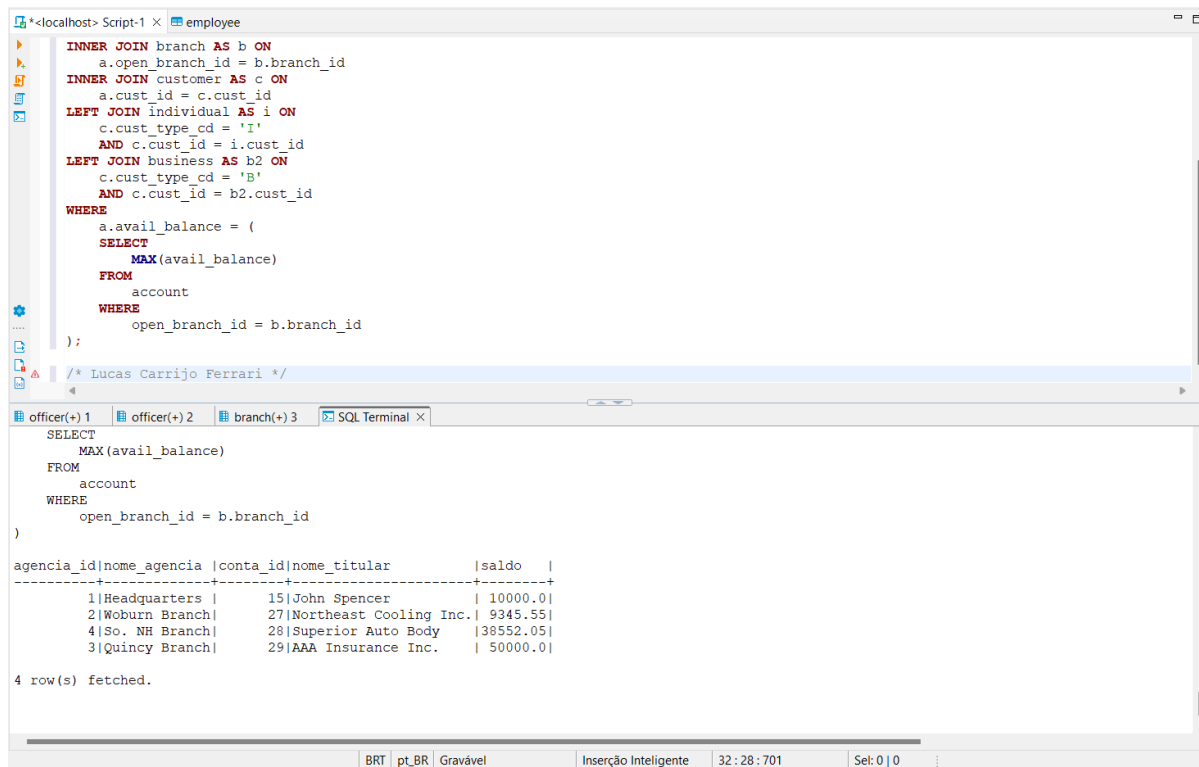
```
SELECT
    e.fname,
    e.lname,
    YEAR(t.txn_date) AS ano,
    COUNT(t.txn_id) AS num_transacoes
FROM
    employee e
LEFT JOIN account a ON
    e.emp_id = a.open_emp_id
LEFT JOIN `transaction` t ON
    a.account_id = t.account_id
GROUP BY
    e.emp_id,
    ano
ORDER BY
    e.lname,
    e.fname,
    ano;
```

Below the query editor, the results pane displays a table with the following data:

emp_id	fname	lname	ano	num_transacoes
1	Helen	Fleming	2001	0
2	Beth	Fowler	2001	0
3	John	Gooding	2001	0
4	Jane	Grossman	2001	0
5	Susan	Hawthorne	2001	0
6	Samantha	Jameson	2001	0
7	Theresa	Markham	2001	0
8	Theresa	Markham	2002	2
9	Theresa	Markham	2002	1
10	Theresa	Markham	2003	1
11	Theresa	Markham	2004	1
12	Cindy	Mason	2001	0
13	Sarah	Parker	2001	0
14	Frank	Portman	2001	0
15	Paula	Roberts	2001	0
16	Paula	Roberts	2000	2
17	Paula	Roberts	2001	2
18	Paula	Roberts	2004	2

The IDE interface includes a toolbar on the left with icons for file operations, a status bar at the bottom with information like 'BRT', 'pt_BR', 'Gravável', 'Inserção Inteligente', '19:1:290', and 'Sel: 0 | 0'.

Figura 3: Junção Externa, Agrupamento, Agregação e Ordenação (alternativa)



```
INNER JOIN branch AS b ON
a.open_branch_id = b.branch_id
INNER JOIN customer AS c ON
a.cust_id = c.cust_id
LEFT JOIN individual AS i ON
c.cust_type_cd = 'I'
AND c.cust_id = i.cust_id
LEFT JOIN business AS b2 ON
c.cust_type_cd = 'B'
AND c.cust_id = b2.cust_id
WHERE
a.avail_balance = (
SELECT
MAX(avail_balance)
FROM
account
WHERE
open_branch_id = b.branch_id
);
/* Lucas Carrijo Ferrari */
```

```
SELECT
MAX(avail_balance)
FROM
account
WHERE
open_branch_id = b.branch_id
)
```

agencia_id	nome_agencia	conta_id	nome_titular	saldo
1	Headquarters	15	John Spencer	10000.0
2	Woburn Branch	27	Northeast Cooling Inc.	9345.55
4	So. NH Branch	28	Superior Auto Body	38552.05
3	Quincy Branch	29	AAA Insurance Inc.	50000.0

4 row(s) fetched.

BRT | pt_BR | Gravável | Inserção Inteligente | 32:28:701 | Sel: 0 | 0

Figura 4: Junções Internas, Agrupamento, Agregação, União e Concatenação