





Funções de Grupo **GROUP BY**

Objetivos

FIAP

Ao concluir esta lição, você será capaz de:

- Identificar as functions de grupo disponíveis
- Descrever o uso de functions de grupo
- Agrupar dados com a cláusula `GROUP BY`
- Incluir ou excluir linhas agrupadas com a cláusula `HAVING`

1-3

Objetivos

Esta lição aborda o uso de functions com mais detalhes. Ela concentra-se na obtenção de informações sumariadas (como médias) relativas a grupos de linhas. A lição mostra como agrupar as linhas de uma tabela em conjuntos menores e como especificar critérios de pesquisa para grupos de linhas.

O Que São Functions de Grupo?

As functions de grupo operam em conjuntos de linhas para fornecer um resultado por grupo.

EMPLOYEES

DEPARTMENT_ID	SALARY
90	24000
90	17000
90	17000
60	9000
60	6000
60	4200
50	5800
50	3500
50	3100
50	2600
50	2500
80	10500
80	11000
80	8600
	7000
10	4400

20 rows selected.

Salário máximo na
tabela EMPLOYEES

MAX(SALARY)
24000

Functions de Grupo

Diferentemente das functions de uma única linha, as functions de grupo operam em conjuntos de linhas para fornecer um resultado por grupo. Esses conjuntos podem abranger a tabela inteira ou a tabela dividida em grupos.

Tipos de Functions de Grupo

FIAP

• • • • •
• • • • •
• + •
+ •

- **AVG**
- **COUNT**
- **MAX**
- **MIN**
- **STDDEV**
- **SUM**
- **VARIANCE**



|
+

1-5

•
• +
■ □ •
• •
•
• + • •
• • •

Tipos de Functions de Grupo

Cada function aceita um argumento. Esta tabela identifica as opções que você pode usar na sintaxe:

Function	Descrição
AVG ([DISTINCT <u>ALL</u>] <i>n</i>)	Valor médio de <i>n</i> ; ignora valores nulos
COUNT ({ * [DISTINCT <u>ALL</u>] <i>expr</i> })	Número de linhas, em que <i>expr</i> é avaliado como um valor diferente de nulo (conta todas as linhas selecionadas usando *, inclusive valores duplicados e linhas com valores nulos)
MAX ([DISTINCT <u>ALL</u>] <i>expr</i>)	Valor máximo de <i>expr</i> ; ignora valores nulos
MIN ([DISTINCT <u>ALL</u>] <i>expr</i>)	Valor mínimo de <i>expr</i> ; ignora valores nulos
STDDEV ([DISTINCT <u>ALL</u>] <i>x</i>)	Desvio padrão de <i>n</i> ; ignora valores nulos
SUM ([DISTINCT <u>ALL</u>] <i>n</i>)	Valores somados de <i>n</i> ; ignora valores nulos
VARIANCE ([DISTINCT <u>ALL</u>] <i>x</i>)	Variação de <i>n</i> ; ignora valores nulos

Functions de Grupo: Sintaxe

FIAP

```
SELECT    [column,] group_function(column), ...
FROM      table
[WHERE    condition]
[GROUP BY column]
[ORDER BY column];
```

1-6

Diretrizes para a Utilização de Functions de Grupo

Com `DISTINCT`, a function considera apenas valores não duplicados; com `ALL`, ela considera todos os valores, inclusive os duplicados.

Como é o default, `ALL` não precisa ser especificado.

Os tipos de dados das functions com um argumento `expr` podem ser `CHAR`, `VARCHAR2`, `NUMBER` ou `DATE`.

Todas as functions de grupo ignoram valores nulos. Para substituir um valor por valores nulos, use as functions `NVL`, `NVL2` ou `COALESCE`.

Usando as Functions AVG e SUM

FIAP

É possível usar as functions AVG e SUM para dados numéricos.

```
SELECT AVG(salary), MAX(salary),  
       MIN(salary), SUM(salary)  
FROM   employees  
WHERE  job_id LIKE '%REP%';
```

AVG(SALARY)	MAX(SALARY)	MIN(SALARY)	SUM(SALARY)
8150	11000	6000	32600

1-7

Usando as Functions de Grupo

É possível usar as functions AVG, SUM, MIN e MAX em colunas que podem armazenar dados numéricos. O exemplo do slide mostra os salários médio, máximo e mínimo, bem como a soma dos salários mensais, de todos os representantes de vendas.

Usando as Functions MIN e MAX

FIAP

É possível usar MIN e MAX para tipos de dados numéricos, de caractere e de data.

```
SELECT MIN(hire_date), MAX(hire_date)
FROM employees;
```

MIN(HIRE_	MAX(HIRE_
17-JUN-87	29-JAN-00

1-8

Usando as Functions de Grupo (continuação)

É possível usar as functions MAX e MIN para os tipos de dados numéricos, de caractere e de data. O exemplo do slide mostra o funcionário mais recente e o mais antigo.

O exemplo a seguir mostra o primeiro e o último sobrenome de funcionário em uma lista em ordem alfabética de todos os funcionários:

```
SELECT MIN(last_name), MAX(last_name)
FROM employees;
```

Observação: Só é possível usar as functions AVG, SUM, VARIANCE e

MIN(LAST_NAME)	MAX(LAST_NAME)
Abel	Zlotkey

Usando a Function COUNT

FIAP

COUNT (*) retorna o número de linhas de uma tabela:

1

```
SELECT COUNT(*)
FROM employees
WHERE department_id = 50;
```

COUNT(*)
5

COUNT (expr) retorna o número de linhas com valores não nulos para *expr*:

2

```
SELECT COUNT(commission_pct)
FROM employees
WHERE department_id = 80;
```

COUNT(COMMISSION_PCT)
3

1-9

Function COUNT

A function COUNT tem três formatos:

COUNT (*)

COUNT (expr)

COUNT (DISTINCT expr)

COUNT (*) retorna o número de linhas de uma tabela que atendem aos critérios da instrução SELECT, incluindo as linhas duplicadas e as linhas com valores nulos de qualquer uma das colunas. Se uma cláusula WHERE estiver incluída na instrução SELECT, COUNT (*) retornará o número de linhas que atendem à condição especificada nessa cláusula.

Por outro lado, COUNT (expr) retorna o número de valores não nulos na coluna identificada por *expr*.

COUNT (DISTINCT expr) retorna o número de valores exclusivos e não nulos na coluna identificada por *expr*.

Exemplos

1. O exemplo do slide mostra o número de funcionários do departamento 50.
2. O exemplo do slide mostra o número de funcionários do departamento 80 que podem receber comissão.

Usando a Palavra-Chave DISTINCT

FIAP

- `COUNT (DISTINCT expr)` retorna o número de valores não nulos e distintos de *expr*.
- Para exibir os valores de números de departamentos distintos da tabela `EMPLOYEES`:

```
SELECT COUNT(DISTINCT department id)
FROM employees;
```

COUNT(DISTINCTDEPARTMENT_ID)
7

1-10

Palavra-chave DISTINCT

Use a palavra-chave `DISTINCT` para suprimir a contagem de valores duplicados em uma coluna.

O exemplo do slide mostra os valores de números de departamentos distintos da tabela `EMPLOYEES`.

Functions de Grupo e Valores Nulos

FIAP

As functions de grupo ignoram valores nulos na coluna:

1	<pre>SELECT AVG(commission_pct) FROM employees;</pre>		
	<table><tr><th>AVG(COMMISSION_PCT)</th></tr><tr><td>.2125</td></tr></table>	AVG(COMMISSION_PCT)	.2125
AVG(COMMISSION_PCT)			
.2125			

A function NVL determina que as functions de grupo incluam valores nulos:

2	<pre>SELECT AVG(NVL(commission_pct, 0)) FROM employees;</pre>		
	<table><tr><th>AVG(NVL(COMMISSION_PCT,0))</th></tr><tr><td>.0425</td></tr></table>	AVG(NVL(COMMISSION_PCT,0))	.0425
AVG(NVL(COMMISSION_PCT,0))			
.0425			

1-11

Functions de Grupo e Valores Nulos

Todas as functions de grupo ignoram valores nulos na coluna.

A function NVL determina que as functions de grupo incluam valores nulos.

Exemplos

1. A média é calculada com base *apenas* nas linhas da tabela com valores válidos armazenados na coluna `COMMISSION_PCT`. A média é calculada como a comissão total paga a todos os funcionários dividida pelo número de funcionários que recebem comissão (quatro).
2. A média é calculada com base em *todas* as linhas da tabela, mesmo que valores nulos estejam armazenados na coluna `COMMISSION_PCT`. A média é calculada como a comissão total paga a todos os funcionários dividida pelo número total de funcionários da empresa (20).

Criando Grupos de Dados

FIAP

EMPLOYEES

DEPARTMENT_ID	SALARY
10	4400
20	13000
20	6000
50	5800
50	3500
50	3100
50	2500
50	2600
60	9000
60	6000
60	4200
80	10500
80	8600
80	11000
90	24000
90	17000

...
20 rows selected.

4400

9500

3500

6400

10033

Salário médio
na tabela
EMPLOYEES
para cada
departamento

DEPARTMENT_ID	AVG(SALARY)
10	4400
20	9500
50	3500
60	6400
80	10033.3333
90	19333.3333
110	10150
	7000

1-12

Criando Grupos de Dados

Até esta etapa do nosso estudo, todas as functions de grupo trataram a tabela como um grande grupo de informações.

Entretanto, às vezes é necessário dividir a tabela de informações em grupos menores. É possível efetuar essa divisão com a cláusula `GROUP BY`.

Criando Grupos de Dados: Sintaxe da Cláusula GROUP BY

FIAP

```
SELECT    column, group_function(column)
FROM      table
[WHERE    condition]
[GROUP BY group_by_expression]
[ORDER BY column];
```

É possível dividir as linhas de uma tabela em grupos menores com a cláusula GROUP BY.

1-13

Cláusula GROUP BY

Você pode usar a cláusula GROUP BY para dividir as linhas de uma tabela em grupos. Depois, pode usar as functions de grupo para retornar informações sumariadas de cada grupo.

Na sintaxe:

group_by_expression especifica colunas cujos valores determinam a base

do agrupamento de linhas

Diretrizes

Se incluir uma function de grupo em uma cláusula SELECT, você não poderá selecionar resultados individuais, *a menos que* a coluna individual apareça na cláusula GROUP BY. Se não conseguir incluir a lista de colunas na cláusula GROUP BY, você receberá uma mensagem de erro.

Com uma cláusula WHERE, você poderá excluir linhas antes de dividi-las em grupos.

Inclua as *colunas* na cláusula GROUP BY.

Não é possível usar um apelido de coluna na cláusula GROUP BY.

Usando a Cláusula GROUP BY

FIAP

Todas as colunas da lista **SELECT** que não são functions de grupo devem estar incluídas na cláusula **GROUP BY**.

```
SELECT department_id, AVG(salary)
FROM employees
GROUP BY department_id ;
```

DEPARTMENT_ID	AVG(SALARY)
10	4400
20	9500
50	3500
60	6400
80	10033.3333
90	19333.3333
110	10150
	7000

8 rows selected.

1-14

Usando a Cláusula GROUP BY

Ao usar a cláusula **GROUP BY**, verifique se todas as colunas da lista **SELECT** que não são functions de grupo estão incluídas nessa cláusula. O exemplo do slide mostra o número e o salário médio de cada departamento. A instrução **SELECT** com uma cláusula **GROUP BY** é avaliada da seguinte maneira:

A cláusula **SELECT** especifica as colunas a serem recuperadas da seguinte forma:

A coluna de número de departamento da tabela **EMPLOYEES**

A média de todos os salários no grupo especificado na cláusula **GROUP BY**

A cláusula **FROM** especifica as tabelas que o banco de dados deve acessar: a tabela **EMPLOYEES**.

A cláusula **WHERE** especifica as linhas a serem recuperadas. Como não há cláusula **WHERE**, todas as linhas são recuperadas por default.

A cláusula **GROUP BY** especifica como as linhas devem ser agrupadas.

As linhas são agrupadas por número de departamento, portanto, a function **AVG** aplicada à coluna de salário calculará o *salário médio de cada departamento*.

Usando a Cláusula GROUP BY

FIAP

A cláusula GROUP BY seguida pelo nome da coluna não precisa estar na lista SELECT.

```
SELECT  AVG(salary)
FROM    employees
GROUP BY department_id ;
```

AVG(SALARY)
4400
9500
3500
6400
10033.3333
19333.3333
10150
7000

1-15

Usando a Cláusula GROUP BY (continuação)

A cláusula GROUP BY seguida pelo nome da coluna não precisa estar na cláusula SELECT. Por exemplo, a instrução SELECT do slide mostra os salários médios de cada departamento sem exibir os respectivos números de departamento. No entanto, sem os números de departamento, os resultados parecem não fazer sentido.

É possível usar a function de grupo na cláusula ORDER BY:

```
SELECT department_id, AVG(salary)
FROM employees
GROUP BY department_id
ORDER BY AVG(salary) ;
```

DEPARTMENT_ID	AVG(SALARY)
50	3500
10	4400
60	6400
...	
90	19333.3333

8 rows selected.

Agrupando por Mais de Uma Coluna

FIAP

EMPLOYEES

DEPARTMENT_ID	JOB_ID	SALARY
90	AD_PRES	24000
90	AD_VP	17000
90	AD_VP	17000
60	IT_PROG	9000
60	IT_PROG	6000
60	IT_PROG	4200
50	ST_MAN	5800
50	ST_CLERK	3500
50	ST_CLERK	3100
50	ST_CLERK	2600
50	ST_CLERK	2500
80	SA_MAN	10500
80	SA_REP	11000
80	SA_REP	8600
...		
20	MK_REP	6000
110	AC_MGR	12000
110	AC_ACCOUNT	8300

20 rows selected.

Adicione os
salários à tabela
EMPLOYEES
para cada cargo,
agrupados por
departamento

DEPARTMENT_ID	JOB_ID	SUM(SALARY)
10	AD_ASST	4400
20	MK_MAN	13000
20	MK_REP	6000
50	ST_CLERK	11700
50	ST_MAN	5800
60	IT_PROG	19200
80	SA_MAN	10500
80	SA_REP	19600
90	AD_PRES	24000
90	AD_VP	34000
110	AC_ACCOUNT	8300
110	AC_MGR	12000
	SA_REP	7000

13 rows selected.

1-16

Grupos Contidos em Outros Grupos

Às vezes, você precisa ver os resultados de grupos contidos em outros grupos.

O slide mostra um relatório que exibe o salário total pago a cada cargo, em cada departamento.

A tabela **EMPLOYEES** é agrupada primeiro por número de departamento e, nesse agrupamento, por cargo. Por exemplo, os quatro estoquistas do departamento 50 são agrupados e um único resultado (salário total) é fornecido para todos os estoquistas do grupo.

Usando a Cláusula GROUP BY em Várias Colunas

FIAP

```
SELECT department_id dept_id, job_id, SUM(salary)
FROM employees
GROUP BY department id, job id ;
```

DEPT_ID	JOB_ID	SUM(SALARY)
10	AD_ASST	4400
20	MK_MAN	13000
20	MK_REP	6000
50	ST_CLERK	11700
50	ST_MAN	5800
60	IT_PROG	19200
80	SA_MAN	10500
80	SA_REP	19600
90	AD_PRES	24000
90	AD_VP	34000
110	AC_ACCOUNT	8300
110	AC_MGR	12000
	SA_REP	7000

13 rows selected.

1-17

Grupos Contidos em Outros Grupos (continuação)

É possível obter resultados sumariados de grupos e subgrupos listando mais de uma coluna na cláusula GROUP BY. Você pode determinar a ordem de classificação default dos resultados pela ordem das colunas na cláusula GROUP BY. No exemplo do slide, a instrução SELECT com uma cláusula GROUP BY é avaliada da seguinte maneira:

A cláusula SELECT especifica a coluna a ser recuperada:

O número do departamento na tabela EMPLOYEES

O ID do cargo na tabela EMPLOYEES

A soma de todos os salários no grupo especificado na cláusula GROUP BY

A cláusula FROM especifica as tabelas que o banco de dados deve acessar: a tabela EMPLOYEES.

A cláusula GROUP BY especifica como você deve agrupar as linhas:

Primeiro, as linhas são agrupadas por número de departamento.

Depois, as linhas são agrupadas por ID de cargo nos grupos de números de departamento.

Portanto, a function SUM é aplicada à coluna de salário para todos os IDs de cargo em cada grupo de números de departamento.

Consultas Inválidas Usando Functions de Grupo

FIAP

Qualquer coluna ou expressão da lista **SELECT** que não seja uma function agregada deverá estar na cláusula **GROUP BY**:

```
SELECT department_id, COUNT(last_name)
FROM employees;
```

```
SELECT department_id, COUNT(last_name)
*
ERROR at line 1:
ORA-00937: not a single-group group function
```

Coluna ausente na cláusula **GROUP BY**

1-18

Consultas Inválidas Usando Functions de Grupo

Sempre que usar uma combinação de itens individuais (**DEPARTMENT_ID**) e functions de grupo (**COUNT**) na mesma instrução **SELECT**, inclua uma cláusula **GROUP BY** que especifique os itens individuais (neste caso, **DEPARTMENT_ID**). Se a cláusula **GROUP BY** não for incluída, a mensagem de erro "not a single-group group function" será exibida e um asterisco (*) indicará a coluna afetada. Para corrigir o erro no slide, adicione a cláusula **GROUP BY**:

```
SELECT department_id, count(last_name)
FROM employees
GROUP BY department_id;
```

DEPARTMENT_ID	COUNT(LAST_NAME)
10	1
20	2
30	1

8 rows selected.

Consultas Inválidas Usando Functions de Grupo

- Não é possível usar a cláusula **WHERE** para restringir grupos.
- Use a cláusula **HAVING** para restringir grupos.
- Não é possível usar functions de grupo na cláusula **WHERE**.

```
SELECT department_id, AVG(salary)
FROM employees
WHERE AVG(salary) > 8000
GROUP BY department_id;
```

```
WHERE AVG(salary) > 8000
*
ERROR at line 3:
ORA-00934: group function is not allowed here
```

Não é possível usar a cláusula **WHERE para restringir grupos**

1-19

Consultas Inválidas Usando Functions de Grupo (continuação)

Não é possível usar a cláusula **WHERE** para restringir grupos. A instrução **SELECT** do exemplo do slide resulta em erro, pois utiliza a cláusula **WHERE** para restringir a exibição dos salários médios dos departamentos cujo salário médio é superior a US\$ 8.000.

Para corrigir o erro do exemplo, use a cláusula **HAVING** para restringir grupos:

```
SELECT department_id, AVG(salary)
FROM employees
HAVING AVG(salary) > 8000
GROUP BY department_id;
```

DEPARTMENT_ID	AVG(SALARY)
20	9500
80	10033.3333
90	19333.3333
110	10150

Restringindo Resultados de Grupos

FIAP

EMPLOYEES

DEPARTMENT_ID	SALARY
90	24000
90	17000
90	17000
60	9000
60	6000
60	4200
50	5800
50	3500
50	3100
50	2600
50	2500
80	10500
80	11000
80	8500
...	...
20	6000
110	12000
110	8300

20 rows selected.

O salário máximo por departamento quando for maior que US\$ 10.000

DEPARTMENT_ID	MAX(SALARY)
20	13000
80	11000
90	24000
110	12000

1-20

Restringindo Resultados de Grupos

Da mesma maneira que você utiliza a cláusula **WHERE** para restringir as linhas selecionadas, use a cláusula **HAVING** para restringir grupos. Para obter o salário máximo de cada departamento cujo salário máximo é superior a US\$ 10.000, você precisa fazer o seguinte:

1. Obtenha o salário médio de cada departamento agrupando por número de departamento.
2. Restrinja os grupos aos departamentos com um salário máximo maior que US\$10.000.

Restringindo Resultados de Grupos com a Cláusula HAVING

Quando a cláusula `HAVING` é utilizada, o servidor Oracle restringe os grupos da seguinte forma:

1. As linhas são agrupadas.
2. A function de grupo é aplicada.
3. Os grupos que correspondem à cláusula `HAVING` são exibidos.

```
SELECT    column, group_function
FROM      table
[WHERE    condition]
[GROUP BY group by expression]
[HAVING   group_condition]
[ORDER BY column];
```

1-21

Restringindo Resultados de Grupos com a Cláusula `HAVING`

Use a cláusula `HAVING` para especificar quais grupos devem ser exibidos e, dessa forma, restringir ainda mais os grupos com base nas informações agregadas.

Na sintaxe, *group_condition* restringe os grupos de linhas retornados aos grupos cuja condição especificada é verdadeira.

Quando você usa a cláusula `HAVING`, o servidor Oracle executa as seguintes etapas:

1. As linhas são agrupadas.
2. A function de grupo é aplicada ao grupo.
3. Os grupos que correspondem aos critérios na cláusula `HAVING` são exibidos.

A cláusula `HAVING` pode anteceder a cláusula `GROUP BY`, mas é recomendável usar a cláusula `GROUP BY` primeiro por razões lógicas. Os grupos são formados e as functions de grupo são calculadas antes de a cláusula `HAVING` ser aplicada aos grupos na lista `SELECT`.

Usando a Cláusula HAVING

FIAP

```
SELECT department_id, MAX(salary)
FROM employees
GROUP BY department_id
HAVING MAX(salary) > 10000 ;
```

DEPARTMENT_ID	MAX(SALARY)
20	13000
80	11000
90	24000
110	12000

1-22

Usando a Cláusula HAVING

O exemplo do slide mostra os números e os salários máximos dos departamentos cujo salário máximo é maior que US\$ 10.000.

É possível usar a cláusula `GROUP BY` sem uma function de grupo na lista `SELECT`.

Se você restringir as linhas com base no resultado de uma function de grupo, especifique as cláusulas `GROUP BY` e `HAVING`.

Este exemplo mostra os números e os salários médios dos departamentos cujo salário máximo ultrapassa US\$ 10.000:

```
SELECT department_id, AVG(salary)
FROM employees
GROUP BY department_id
HAVING max(salary) > 10000;
```

DEPARTMENT_ID	AVG(SALARY)
20	9500
80	10033.3333
90	19333.3333
110	10150

Usando a Cláusula HAVING

FIAP

```
SELECT job_id, SUM(salary) PAYROLL
FROM employees
WHERE job_id NOT LIKE '%REP%'
GROUP BY job_id
HAVING SUM(salary) > 13000
ORDER BY SUM(salary);
```

JOB_ID	PAYROLL
IT_PROG	19200
AD_PRES	24000
AD_VP	34000

1-23

Usando a Cláusula HAVING (continuação)

O exemplo do slide mostra o ID do cargo e o salário mensal total de cada cargo com uma folha de pagamento total que ultrapassa US\$ 13.000. O exemplo exclui representantes de vendas e classifica a lista pelo salário mensal total.

Aninhando Functions de Grupo

FIAP

Exiba a média de salário máximo:

```
SELECT MAX(AVG(salary))  
FROM employees  
GROUP BY department_id;
```

MAX(AVG(SALARY))
19333.3333

1-24

Aninhando Functions de Grupo

É possível aninhar até duas functions de grupo. O exemplo do slide mostra a média de salário máximo.

Sumário

FIAP

Nesta lição, você aprendeu a:

- Usar as functions de grupo COUNT, MAX, MIN e AVG
- Criar consultas que utilizam a cláusula GROUP BY
- Criar consultas que utilizam a cláusula HAVING

```
SELECT    column, group_function
FROM      table
[WHERE    condition]
[GROUP BY group_by_expression]
[HAVING   group_condition]
[ORDER BY column];
```

1-25

Sumário

Várias functions de grupo estão disponíveis em SQL, como estas:

AVG, COUNT, MAX, MIN, SUM, STDDEV e VARIANCE

Para criar subgrupos, use a cláusula GROUP BY. É possível restringir grupos com a cláusula HAVING.

Especifique as cláusulas HAVING e GROUP BY após a cláusula WHERE em uma instrução. A ordem na qual você especifica essas cláusulas após a cláusula WHERE não é importante. Informe a cláusula ORDER BY por último.

O servidor Oracle avalia as cláusulas na seguinte ordem:

1. Se a instrução contiver uma cláusula WHERE, o servidor estabelecerá as linhas candidatas.
2. O servidor identifica os grupos especificados na cláusula GROUP BY.
3. A cláusula HAVING restringe os grupos de resultados que não atendem aos critérios de grupo especificados na cláusula HAVING.

Observação: Para obter uma lista completa das functions de grupo, consulte o manual *Oracle SQL Reference*.

Bibliografia Utilizada

FIAP

Database SQL Language Reference: <http://docs.oracle.com/database/121/SQLRF/toc.htm>

Manuais Oracle – Oracle Database 12c: SQL Workshop I/II

*Esta apresentação possui material de referência com propriedade da Oracle.
Copyright © 2004, Oracle. Todos os direitos reservados.*



OBRIGADO



profalexandre.barcelos@fiap.com.br



<https://www.linkedin.com/in/alexandrebarcelos>

FIAP

Copyright © 2023 | Professor Me. Alexandre Barcelos
Todos os direitos reservados. Reprodução ou divulgação total ou parcial deste documento, é expressamente
proibido sem consentimento formal, por escrito, do professor/autor.