

FIAP



AULA 09

SQL/DQL – SELECT GROUP

BY

Welcome to the next evolution in higher education.

DISCIPLINA
BUILDING RELATIONAL DATABASE

📄 PROFº DRº FRANCISCO D. L.
✉ ABREU
PROFFRANCISCO@FIAP.COM.BR

SUMÁRIO

□ DQL

□ Sobre Selecionar Dados com Agregação

□ Tipos de Função (SQL Worksheet)

□ Exercício Prático

| OBJETIVO

Introduzir conceitos iniciais do SQL/DQL

Aplicar os conceitos no Oracle SQL Developer

SOBRE SELECT GROUP BY

- ***Data Query Language (DQL)*** – Linguagem de Consulta de Dados , expressa o comando que especifica a:
 - **CONSULTAR** 1 ou vários dados (SELECT)
- Os comandos da **DQL** viabiliza o acesso aos dados de forma compatível ao modelo de dados projetado.
- Em algumas literaturas colocam o **SELECT** dentro da **DML**

- O **SELECT GROUP BY** permite o agrupamento de dados em tabelas separada por meio de uma ou mais colunas.
- Isso é útil quando você deseja resumir dados e realizar operações agregadas, como contar, somar ou encontrar médias em grupos de registros que compartilham valores semelhantes em uma ou mais colunas.
- Para usar o **SELECT GROUP BY** é necessário utilizar as funções disponíveis no SGBDR

I SINTAXE SELECT GROUP BY

- **SELECT** lista as colunas que serão selecionadas além das funções de agregação
- **FROM** especifica o nome das tabelas que serão selecionadas
- **WHERE** condição para consulta (impor uma filtragem)
- **GROUP BY** [coluna] lista as colunas pelas quais deseja agrupar os dados.
- **HAVING** limita os dados a serem mostrados no agrupamento, isto é, funciona similarmente ao WHERE porém é restrito ao GROUP BY.
- **ORDER BY** especifica em ordem os resultados da consulta é exibido [asc] ascendente ou descendente [desc]

```
SELECT coluna1, coluna2, funcao_agregada(coluna)
FROM nome_da_tabela
WHERE condicao_logica
GROUP BY coluna1, coluna2
HAVING condicao_logica_agregada
ORDER BY coluna1;
```


TIPOS DE FUNÇÃO

FUNÇÃO	DESCRIÇÃO
AVG ()	RETORNA A MÉDIA OBTIDA ENTRE OS VALORES
COUNT ()	RETORNA A QUANTIDADE DE OCORRÊNCIAS (LINHAS)
MAX ()	RETORNA O MAIOR VALOR DO CONJUNTO
MIN ()	RETORNA O MENOR VALOR DO CONJUNTO
SUM ()	RETORNA A SOMATÓRIA DOS VALORES DE UM CONJUNTO
STDDEV ()	RETORNA O DESVIO PADRÃO DO CONJUNTO
VARIANCE ()	RETORNA A VARIÂNCIA DO CONJUNTO

EXEMPLO DE TABELA

```
-- Cria a tabela "Clientes"

CREATE TABLE Clientes (
    ID NUMBER(5) PRIMARY KEY,
    Nome VARCHAR2(255),
    Cidade VARCHAR2(255),
    Sexo VARCHAR2(1),
    idade NUMBER
);

-- Insercao de dados na tabela "Clientes"

INSERT INTO Clientes VALUES (1, 'João', 'São Paulo', 'M', 20);
INSERT INTO Clientes VALUES (2, 'Maria', 'Rio de Janeiro', 'F', 30);
INSERT INTO Clientes VALUES (3, 'Carlos', 'Belo Horizonte', 'M', 20);
INSERT INTO Clientes VALUES (4, 'Ana', 'São Paulo', 'F', 25);
INSERT INTO Clientes VALUES (5, 'Rafael', 'Rio de Janeiro', 'M', 50);
```

```
-- Cria a tabela "Pedidos"

CREATE TABLE Pedidos (
    id NUMBER(5) PRIMARY KEY,
    cliente_id NUMBER(5),
    ds_produto VARCHAR2(255),
    vl_pedido Number (5,2),
    FOREIGN KEY (cliente_id) REFERENCES
        Clientes (id)
);

-- Insercao de dados na tabela "Pedidos"

INSERT INTO Pedidos VALUES(101, 1, 'Celular', 1100.00);
INSERT INTO Pedidos VALUES(102, 2, 'Laptop', 4000.00);
INSERT INTO Pedidos VALUES(103, 3, 'Tablet', 3500.00);
INSERT INTO Pedidos VALUES(104, 1, 'TV', 5000.00);
INSERT INTO Pedidos VALUES(105, 4, 'Geladeira', 3000.00);
```

I EXEMPLO DE ORDER BY

FIAP

COMANDO

RESULTADO

ASC

	ID	NOME	CIDADE	SEXO	IDADE
1	4	Ana	São Paulo	F	25
2	3	Carlos	Belo Horizonte	M	20
3	1	João	São Paulo	M	20
4	2	Maria	Rio de Janeiro	F	30
5	5	Rafael	Rio de Janeiro	M	50

DESC

	ID	NOME	CIDADE	SEXO	IDADE
1	5	Rafael	Rio de Janeiro	M	50
2	2	Maria	Rio de Janeiro	F	30
3	1	João	São Paulo	M	20
4	3	Carlos	Belo Horizonte	M	20
5	4	Ana	São Paulo	F	25

```
SELECT * FROM Clientes ORDER BY nome ASC;
```

```
SELECT * FROM Clientes ORDER BY nome DESC;
```

I EXEMPLO DE GROUP BY

FIAP

COMANDO

RESULTADO

	CIDADE
1	São Paulo
2	Rio de Janeiro
3	Belo Horizonte




```
SELECT Cidade  
FROM Clientes  
GROUP BY Cidade;
```

I COMANDO: **AVG ()**

SINTAXE

- O comando **AVG ()** é utilizado para retorna a média do conjunto
- Veja o exemplo da Sintaxe:



```
SELECT AVG(coluna1)  
FROM nome_tabela;
```

I COMANDO: **MIN ()**

SINTAXE

- O comando **MIN ()** é utilizado para retorna o menor valor do conjunto
- Veja o exemplo da Sintaxe:



```
SELECT MIN(coluna1)  
FROM nome_tabela;
```

I COMANDO: **MAX ()**

SINTAXE

- O comando **MAX ()** é utilizado para retorna o maior valor do conjunto
- Veja o exemplo da Sintaxe:




```
SELECT MAX(coluna1)  
FROM nome_tabela;
```


I COMANDO: **SUM ()**

SINTAXE

- O comando **SUM ()** é utilizado para retorna a somatório de valores do conjunto
- Veja o exemplo da Sintaxe:



```
SELECT SUM(coluna1)  
FROM nome_tabela;
```

I COMANDO: **COUNT ()**

SINTAXE

- O comando **COUNT ()** é utilizado para retorna a quantidade de ocorrências do conjunto
- Veja o exemplo da Sintaxe:



```
SELECT COUNT(coluna1)  
FROM nome_tabela;
```

I COMANDO: **STDDEV ()**

SINTAXE

- O comando **STDDEV ()** é utilizado para retorna o desvio padrão do conjunto
- Veja o exemplo da Sintaxe:



```
SELECT STDDEV(coluna1)  
FROM nome_tabela;
```

I COMANDO: **VARIANCE ()**

SINTAXE

- O comando **VARIANCE ()** é utilizado para retorna a variância do conjunto
- Veja o exemplo da Sintaxe:



```
SELECT VARIANCE(coluna1)  
FROM nome_tabela;
```

I EXEMPLO DE FUNCTIONS

FIAP

COMANDO

RESULTADO

AVG(IDADE)	COUNT(NOME)	MIN(IDADE)	MAX(IDADE)	STDDEV(IDADE)	VARIANCE(IDADE)
29	5	20	50	12,44989959798873237483597066678220624365	155

```
SELECT
    AVG(idade),
    COUNT(nome),
    MIN(idade),
    MAX(idade),
    STDDEV(idade),
    VARIANCE(idade)
FROM Clientes;
```

FUNÇÃO COM AGREGAÇÃO

- Muitas vezes, é necessário estabelecer condições para conseguir um resultado específico no **SELECT**
- Portanto, pode ser comum utilizar as funções do SQL em agregação (**GROUP BY**).
- Mas Atenção! Recomenda-se utilizar a instrução **HAVING** para englobar essas funções, uma vez que estaremos utilizando o **GROUP BY**

I EXEMPLO DE FUNCTIONS

FIAP

COMANDO

RESULTADO

encontrar clientes que têm a média de idade maior que 25 anos

	NOME	MEDIA_IDADE
1	Maria	30
2	Rafael	50

```
SELECT C.nome, AVG(C.idade) as media_idade
FROM Clientes C
GROUP BY C.nome
HAVING AVG(C.idade) > 25;
```


I EXEMPLO DE FUNCTIONS

FIAP

COMANDO

RESULTADO

encontrar produtos que têm um preço médio superior a 3000:

	DS_PRODUTO	MEDIA_PRECO
1	TV	5000
2	Laptop	4000
3	Tablet	3500

```
SELECT P.ds_produto, AVG(P.vl_pedido) as  
media_preco  
FROM Pedidos P  
GROUP BY P.ds_produto  
HAVING AVG(P.vl_pedido) > 3000.00;
```

I EXEMPLO DE FUNCTIONS

FIAP

COMANDO

RESULTADO

encontrar clientes que fizeram mais de 2 pedidos

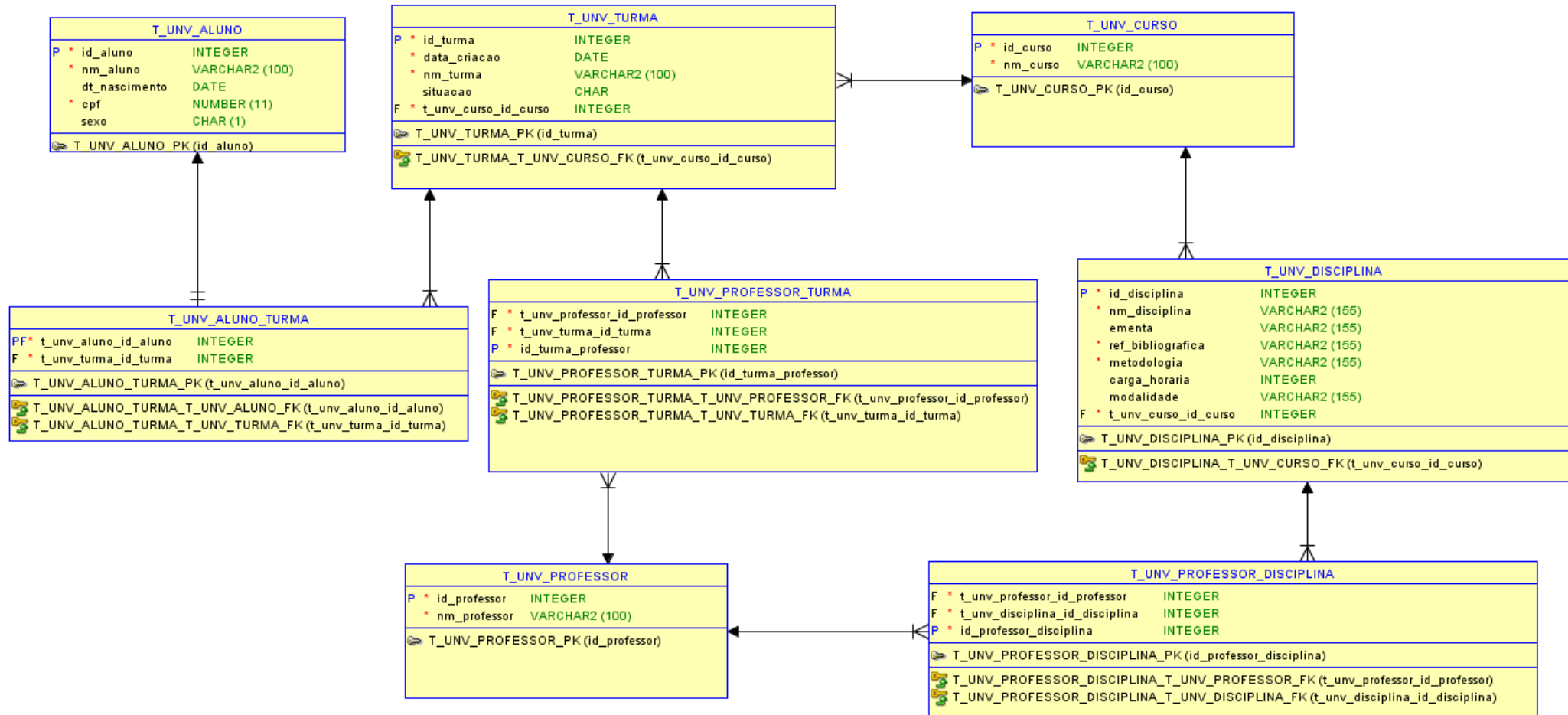
	NOME	TOTAL_PEDIDOS
1	João	2



```
SELECT C.nome, COUNT(P.id) as total_pedidos
FROM Clientes C
LEFT JOIN Pedidos P ON C.id = P.cliente_id
GROUP BY C.nome
HAVING COUNT(P.id) > 1;
```

EXERCÍCIO PRÁTICO

EXERCÍCIO PRÁTICO



- Altere as estruturas das tabelas no SQL Developer:
 - Adicione o campo SITUACAO (CHAR(1)) na tabela ALUNO e PROFESSOR;
 - Altere o nome do campo CARGA HORARIA para CH da tabela DISCIPLINA ;
 - Altere o nome da tabela ALUNO para DISCENTE e PROFESSOR para DOCENTE;
 - Altere o tipo das colunas EMENTA, REF BIBLIOGRAFICA E METODOLOGIA para LONG VARCHAR;
 - Remova todas as tabelas existentes (Observe que há uma ordem correta de exclusão);

Copyright © 2023 Profº Drº Francisco Douglas Lima Abreu

Todos direitos reservados. Reprodução ou divulgação total ou parcial deste documento é expressamente proibido sem o consentimento formal, por escrito ao autor

FIAP

THE WAY WE ARE