

Banco de Dados

Criando Instruções SQL


André Carvalho
andrepacheco@ufpa.br

A series of white diagonal lines of varying lengths and thicknesses, located in the bottom right corner of the slide, creating a modern, abstract graphic element.


SQL

- SQL
 - Apesar de sua padronização pela ANSI e ISO, é comum encontrar grandes fabricantes de banco de dados, criarem especificações da linguagem SQL

DML (Data Manipulation Language)

- É um subconjunto de instruções da linguagem SQL que é utilizado para realizar inclusões, consultas, alterações e exclusões de dados presentes em uma tabela ;
 - As tarefas podem ser executadas sobre diversos registros de várias tabelas ao mesmo tempo.
 - É constituída de :
 - INSERT;
 - SELECT;
 - UPDATE;
 - DELETE.
- 
- A series of white diagonal lines of varying lengths and thicknesses are positioned in the bottom right corner of the slide, creating a modern, abstract graphic element.

DML

- Uma instrução DML é executada quando você:
 - Adiciona novas linhas a uma tabela
 - Modifica linhas existentes em uma tabela
 - Remove linhas existentes de uma tabela
- 
- Several thin, white, parallel diagonal lines are located in the bottom right corner of the slide, extending from the bottom edge towards the right edge.

DDL (Data Definition Language)

- Uma DDL permite ao utilizador definir novas tabelas e novos e elementos associados. A maioria dos bancos de dados que utilizam da linguagem SQL comerciais tem extensões proprietárias no DDL.
- É constituída de:
 - CREATE;
 - DROP

DCL (Data Control Language)

- DCL controla os aspectos de autorização de dados e licenças de usuários para controlar quem tem acesso para ver ou manipular dados dentro do banco de dados.
- É constituída de:
 - GRANT - autoriza ao usuário executar operações.
 - REVOKE - remove ou restringe a capacidade de um usuário de executar operações.

DTL (Data Transaction Language)

- É utilizada para marcar o começo e o fim de uma transação, e em caso de erro é realizada a restauração do registro para o estágio de quando começou a transação.
- É constituída de:
 - BEGAN;
 - ROLLBACK;
 - COMMIT.

UPDATE

- O comando UPDATE é utilizado quando desejamos alterar um registro do banco de dados
- SINTAXE

```
UPDATE nome_tabela  
  SET nome_coluna = novo_valor  
  WHERE campo = valor
```

Obs.: No Where aconselha-se a utilizar a chave da tabela.

DELETE

- O comando DELETE é utilizado quando queremos deletar do banco de dados um registro.

```
DELETE FROM nome_tabela  
WHERE “condição para deletar”
```

WHERE

- WHERE: O comando where é utilizado quando em uma consulta deseja-se restringir o resultado desejado que fora solicitado no SELECT.

```
mysql> select nome, matricula  
-> from aluno  
-> where fk_turma = 1;
```

nome	matricula
Larissa Nobre	201308002
Jose Augusto	201308003
Leandro Silva	201308004
Anderson Silva	201308005
Felipe Barra	201308006
Altemir Pacheco	201308007
Dam Carvalho	201308008
Gabriele Souza	201308009
Gabriel Silva	201308010

```
9 rows in set (0.00 sec)
```

WHERE AND

- Ao utilizar o comando WHERE pode ser utilizando o comando AND para que seja feita a concatenação de 2 ou mais afirmações.

```
mysql> select nome, matricula  
-> from aluno  
-> where fk_turma = 1  
-> AND matricula = '201308004';
```

nome	matricula
Leandro Silva	201308004

1 row in set (0.00 sec)

```
mysql> select nome, matricula  
-> from aluno  
-> where fk_turma = 1  
-> AND matricula = '201308004'  
-> AND dt_nascimento = '1984-02-05'
```

nome	matricula
Leandro Silva	201308004

1 row in set (0.01 sec)

WHERE OR

- Ao utilizar o comando WHERE pode ser utilizando o comando OR para trazer duas condições diferentes em um mesmo campo.

```
mysql> select nome, matricula  
-> from aluno  
-> where matricula = '201308004'  
-> OR matricula = '201308008';
```

nome	matricula
Leandro Silva	201308004
Dam Carvalho	201308008

2 rows in set (0.01 sec)

WHERE LIKE

- Ao utilizar o comando WHERE pode ser utilizando o comando LIKE para trazer partes de uma string, utilizando o caracter '%'.

```
mysql> select nome, matricula  
-> from aluno  
-> where nome like 'andre%';
```

nome	matricula
André Carvalho	201308001

```
1 row in set (0.00 sec)
```

WHERE IN

- Ao utilizar o comando WHERE pode ser utilizando o comando IN para que seja feita a concatenação de 2 ou mais opções no mesmo campo.

```
mysql> select nome, matricula
-> from aluno
-> where matricula in ('201308004','201308008');
```

nome	matricula
Leandro Silva	201308004
Dam Carvalho	201308008

2 rows in set (0.00 sec)

INNER JOIN

- **INNER JOIN:** O Inner Join é utilizado nas situações em que você quer selecionar os registros das duas tabelas, desde que as mesmas possuam informações cruzadas (relacionadas).

Exemplo JOIN

```
mysql> select a.nome, t.turma  
      -> from aluno a  
      -> join turma t on(t.id_turma=a.fk_turma);
```

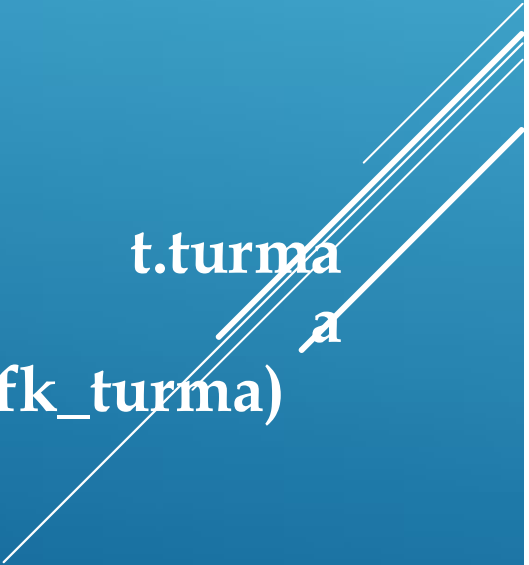
nome	turma
Larissa Nobre	ENG2010
Jose Augusto	ENG2010
Leandro Silva	ENG2010
Anderson Silva	ENG2010
Felipe Barra	ENG2010
Altemir Pacheco	ENG2010
Dam Carvalho	ENG2010
Gabriele Souza	ENG2010
Gabriel Silva	ENG2010

```
9 rows in set (0.00 sec)
```


LEFT JOIN

- LEFT JOIN: O left Join é utilizado nas situações em que você quer selecionar os registro das duas tabelas, desde que as mesmas possuam informações cruzadas (relacionadas), independente se existem ou não dados preenchidos.
- Exemplo:

```
SELECT          a.nome,          t.turma
FROM            alunos
LEFT JOIN turma t ON (t.id_turma = a.fk_turma)
```



Exemplo LEFT JOIN

```
mysql> select a.nome, t.turma  
-> from aluno a  
-> left join turma t on(t.id_turma=a.fk_turma)  
-> ;
```

nome	turma
André Carvalho	NULL
Larissa Nobre	ENG2010
Jose Augusto	ENG2010
Leandro Silva	ENG2010
Anderson Silva	ENG2010
Felipe Barra	ENG2010
Altemir Pacheco	ENG2010
Dam Carvalho	ENG2010
Gabriele Souza	ENG2010
Gabriel Silva	ENG2010
Carmen Carvalho	NULL
Ricardo Macedo	NULL
Fernanda Silva	NULL
Socorro Bandeira	NULL
Alan Costa	NULL
Regina Duarte	NULL

16 rows in set (0.00 sec)

RIGHT JOIN

- **RIGHT JOIN:** O right Join é utilizado nas situações em que você quer selecionar os registros das duas tabelas, desde que as mesmas possuam informações cruzadas (relacionadas), onde serão retornados todos os registros da tabela da direita.

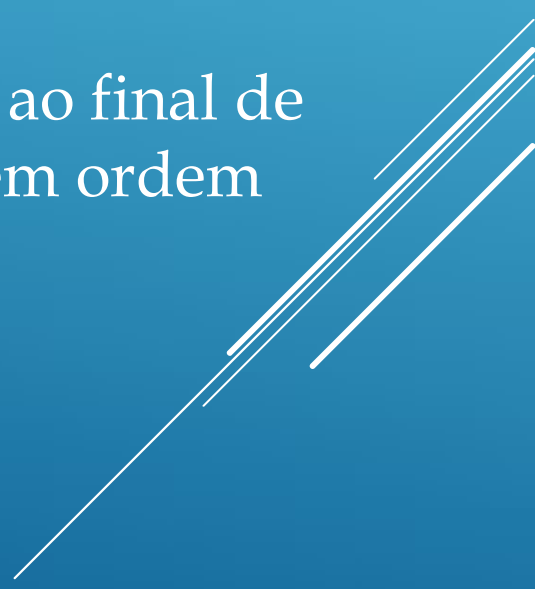
Exemplo RIGHT JOIN

```
mysql> select a.nome, t.turma  
       -> from aluno a  
       -> right join turma t on(t.id_turma=a.fk_turma);
```

nome	turma
Larissa Nobre	ENG2010
Jose Augusto	ENG2010
Leandro Silva	ENG2010
Anderson Silva	ENG2010
Felipe Barra	ENG2010
Altemir Pacheco	ENG2010
Dam Carvalho	ENG2010
Gabriele Souza	ENG2010
Gabriel Silva	ENG2010
NULL	CIE2010

```
10 rows in set (0.00 sec)
```

ORDER BY

- ORDER BY classifica na ordem alfabética de A a Z no caso de string e de 0 a 9 em caso de números.
 - O padrão ordem de classificação é ascendente (A a Z, 0 a 9).
 - Adicionando a palavra reservada DESC ao final de cada campo que você quiser classificar em ordem descendente (Z a A, 9 a 0).
- 
- A series of white diagonal lines of varying lengths and thicknesses are positioned in the bottom right corner of the slide, creating a modern, abstract graphic element.

Exemplo ORDER BY

```
mysql> select nome  
-> from aluno  
-> order by nome;
```

nome
Alan Costa
Altemir Pacheco
Anderson Silva
André Carvalho
Carmen Carvalho
Dam Carvalho
Felipe Barra
Fernanda Silva
Gabriel Silva
Gabriele Souza
Jose Augusto
Larissa Nobre
Leandro Silva
Regina Duarte
Ricardo Macedo
Socorro Bandeira

16 rows in set (0.00 sec)

```
mysql> select nome  
-> from aluno  
-> order by nome desc;
```

nome
Socorro Bandeira
Ricardo Macedo
Regina Duarte
Leandro Silva
Larissa Nobre
Jose Augusto
Gabriele Souza
Gabriel Silva
Fernanda Silva
Felipe Barra
Dam Carvalho
Carmen Carvalho
André Carvalho
Anderson Silva
Altemir Pacheco
Alan Costa

16 rows in set (0.00 sec)

GROUP BY

- A palavra GROUP BY é utilizada para dividir registros da tabela em grupos pequenos.
- As funções de grupo podem ser utilizadas para produzir informação resumida para cada grupo.

```
mysql> select count(1), fk_turma  
->      from aluno  
-> group by fk_turma  
-> ;
```

count(1)	fk_turma
7	NULL
9	1

```
2 rows in set (0.00 sec)
```


HAVING

- A cláusula HAVING vem a complementar a cláusula GROUP BY.
- Ao utilizar o GROUP BY , os registros retornados serão todos os que satisfizerem ao critério informado após o comando WHERE.
- Com HAVING pode ser realizada uma segunda filtragem após termos os resultados dos cálculos das funções agregadas.

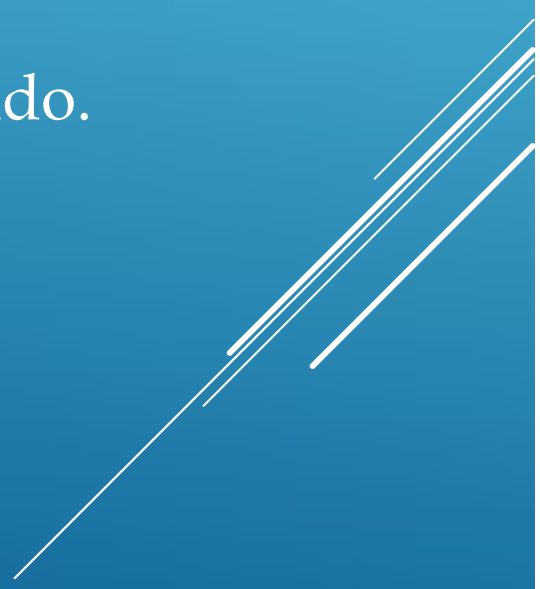
Exemplo HAVING

```
mysql> select count(1), fk_turma  
->      from aluno  
-> group by fk_turma  
-> having count(1) > 8;
```

count(1)	fk_turma
9	1

1 row in set (0.00 sec)

SUBCONSULTA

- A subconsulta é o artifício criado para criar um SELECT dentro de outro SELECT;
 - Em modelagem bem projetadas o subselect pode ser evitado;
 - 90% dos casos o subselect pode ser evitado.
- 
- A series of white diagonal lines of varying lengths and thicknesses, located in the bottom right corner of the slide, creating a modern, abstract graphic element.

Exemplo SUBCONSULTA

```
mysql> select a.nome, a.matricula  
-> ,(select t.turma  
-> from turma t  
-> where t.id_turma = a.fk_turma) turma  
-> from aluno a;
```

nome	matricula	turma
André Carvalho	201308001	NULL
Larissa Nobre	201308002	ENG2010
Jose Augusto	201308003	ENG2010
Leandro Silva	201308004	ENG2010
Anderson Silva	201308005	ENG2010
Felipe Barra	201308006	ENG2010
Altemir Pacheco	201308007	ENG2010
Dam Carvalho	201308008	ENG2010
Gabriele Souza	201308009	ENG2010
Gabriel Silva	201308010	ENG2010
Carmen Carvalho	201308011	NULL
Ricardo Macedo	201308012	NULL
Fernanda Silva	201308013	NULL
Socorro Bandeira	201308014	NULL
Alan Costa	201308015	NULL
Regina Duarte	201308016	NULL

16 rows in set (0.00 sec)

TABELA DINÂMICA

- A tabela dinâmica é a criação de uma tabela em tempo real para que seja manipulada os dados para melhor desempenho;



TABELA DINÂMICA

```
mysql> select *  
      -> from (select nome, matricula, (date_format(now(), '%Y')-year(dt_nascimento)) idade  
      -> from aluno) tab;
```

nome	matricula	idade
André Carvalho	201308001	28
Larissa Nobre	201308002	28
Jose Augusto	201308003	27
Leandro Silva	201308004	29
Anderson Silva	201308005	27
Felipe Barra	201308006	26
Altemir Pacheco	201308007	33
Dam Carvalho	201308008	31
Gabriele Souza	201308009	23
Gabriel Silva	201308010	24
Carmen Carvalho	201308011	43
Ricardo Macedo	201308012	43
Fernanda Silva	201308013	23
Socorro Bandeira	201308014	24
Alan Costa	201308015	29
Regina Duarte	201308016	33

```
16 rows in set (0.00 sec)
```

LIMIT

- Ao utilizar o comando LIMIT será limitado o número de registros que aparecem no resultado da consulta.

```
mysql> select nome  
-> from aluno  
-> limit 0,2;
```

nome
André Carvalho
Larissa Nobre

```
2 rows in set (0.00 sec)
```

CASE

- Utilizado em situações em que várias condições determinam um valor de retorno diferente, mas do mesmo tipo.
- SINTAXE:

SELECT

Case [campo]

When [condição] Then [resultado/calculo]

Else [resultado contrario a condicao]

End

FROM [tabela]

Manipulando String

FUNÇÃO	DESCRIÇÃO
CHAR_LENGTH(<i>str</i>)	Retorna o comprimento da string <i>str</i> , medido em caracteres.
CONCAT(<i>var1</i> , <i>var2</i>)	Retorna o a <i>var1</i> e <i>var2</i> em uma coluna
LEFT(<i>campo</i> , <i>qtd</i>)	Retorna a somente a quantidade de caracteres especificado no <i>qtd</i> , contando a partir da esquerda
FORMAT(<i>campo</i> , <i>numero</i>)	Retorna o valor de <i>campo</i> com o numero de casas especificado no número
LOCATE(<i>texto</i> , <i>campo</i>)	Retorna a posição da primeira ocorrência do texto no campo pesquisado.
LOWER(<i>campo</i>)	Retorna o campo todo em minúsculo
UPPER(<i>campo</i>)	Retorna o campo todo em maiúsculo

Manipulando Data

FUNÇÃO	DESCRIÇÃO
DATE(variavel);	Extraí a parte da data da expressão date ou datetime em “variavel”;
TIME(variavel);	Extraí a parte da hora da expressão time ou datetime em “variavel”;
TIMESTAMP(variavel) Ou TIMESTAMP(variavel, variavel2)	Com um argumento, retorna a expressão date ou datetime em “variavel” como um valor datetime. Com dois argumentos, adiciona a expressão time e “variavel2” à expressão date ou datetime em “variavel” e retorna um valor datetime.
DAYOFWEEK(data)	Retorna o índice do dia da semana para data (1 = Domingo, 2 = Segunda, ... 7 = Sábado). Estes valores de índices correspondem ao padrão ODBC.
PERIOD_DIFF(P1 P2)	Retorna o número de meses entre os períodos P1 e P2. P1 e P2 devem estar no formato AAMM ou

Realizando Cálculos

Função	descrição
COS(numero)	Calcula o cosseno do numero
SUM(campo)	Calcula a soma de todos os valores do campo
ABS(numero)	Retorna o valor absoluto do numero
EXP(X)	Retorna o valor de e (a base dos logaritmos naturais) para o poder de X.
LN(X)	Retorna o logaritmo natural de X, ou seja, X logaritmo de base e.

Definições Básicas

Dado: É qualquer elemento identificado em sua forma bruta que por si só não conduz a uma compreensão de determinado fato ou situação.

A definição mais simples de **metadados** é que eles são dados sobre dados – mais especificamente, informações (dados) sobre um determinado conteúdo. (Guardam informações sobre os dados)