



EducaCiência FastCode

Fala Galera,

Neste artigo, abordaremos um tema bem comum.

- Artigo: 17/2020 Data: Abril/2020
- Público Alvo: Desenvolvedores – Iniciantes
- Tecnologia: Banco de Dados MySql
- Tema: comando Select - Banco de Dados - usabilidade
- Link: <https://github.com/perucello/DevFP>

Desta vez, escolhi um tema decorrente de quem está iniciando os estudos do Banco de Dados. Visto que o comando SELECT é um dos comandos mais poderosos e mais utilizados na linguagem de programação e de acordo com o que vemos, destinamos um artigo somente para este tema onde tem como propósito explicar e tentar simplificar o SELECT e INNER JOIN.

Nesse artigo, vou tratar de trazer um pouco mais sobre o comando SELECT.

Antes de mais nada, vamos entender melhor o que é o Banco de Dados e por que o comando SELECT é tão poderoso.

Banco de Dados - Um banco de dados é uma coleção organizada de informações - ou dados - estruturadas, normalmente armazenadas eletronicamente em um sistema de computador. Um banco de dados é geralmente controlado por um sistema de gerenciamento de banco de dados (DBMS). Juntos, os dados e o DBMS, juntamente com os aplicativos associados a eles, são chamados de sistema de banco de dados, geralmente abreviados para apenas banco de dados. Os dados nos tipos mais comuns de bancos de dados em operação atualmente são modelados em linhas e colunas em uma série de tabelas para tornar o processamento e a consulta de dados eficientes. Os dados podem ser facilmente acessados, gerenciados, modificados, atualizados, controlados e organizados. A maioria dos bancos de dados usa a linguagem de consulta estruturada (SQL) para escrever e consultar dados.

SQL (Structured Query Language, Linguagem de consulta estruturada) - SQL é uma linguagem de programação usada por quase todos os bancos de dados relacionais para consultar, manipular e definir dados e fornecer controle de acesso.

O SQL foi desenvolvido pela primeira vez na IBM nos anos 1970, com a Oracle como principal contribuinte, o que levou à implementação do padrão SQL ANSI; o SQL estimulou muitas extensões de empresas como IBM, Oracle e Microsoft. Embora o SQL ainda seja amplamente usado hoje em dia, novas linguagens de programação estão começando a aparecer.

Fonte: <https://www.oracle.com/br/database/what-is-database.html>

Após darmos uma breve introdução ao que é um Banco de Dados e sua Linguagem, vamos preparar nosso cenário.



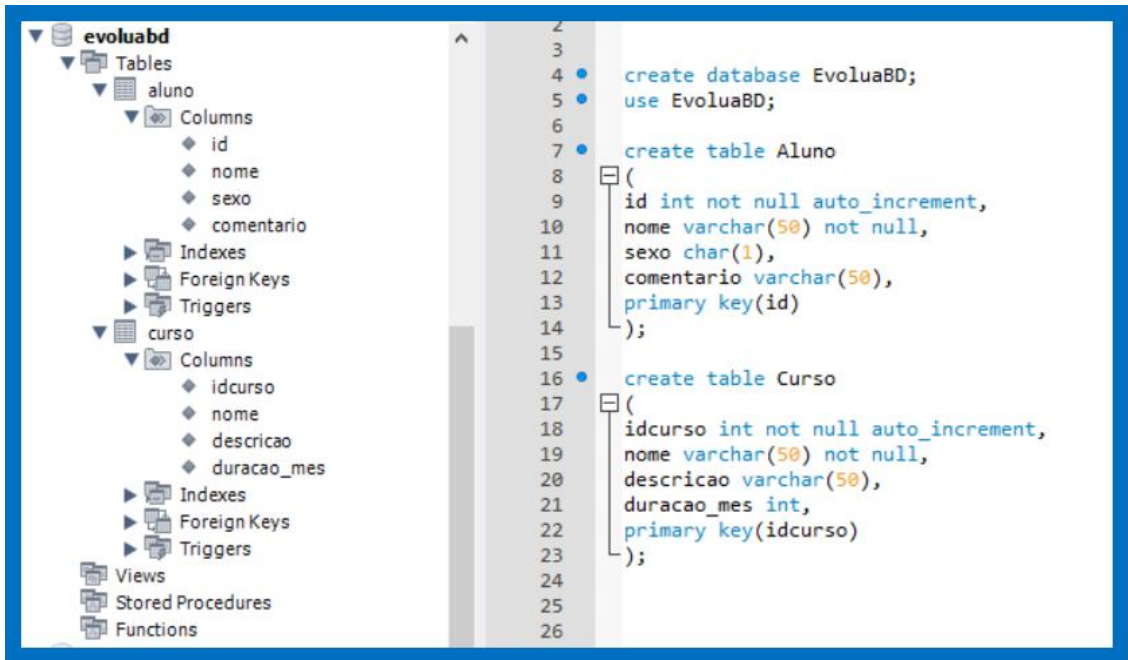
Atualmente estou ministrando aula na Evolua de Sumaré , no entanto, nada mais justo que utilizarmos deste ambiente como ambiente de demonstração e didática neste artigo.

Criaremos um Banco de Dados onde teremos:

Nosso Banco de Dados chamará EvoluaBD

Teremos:

- ⇒ Tabela aluno
- ⇒ Tabela curso

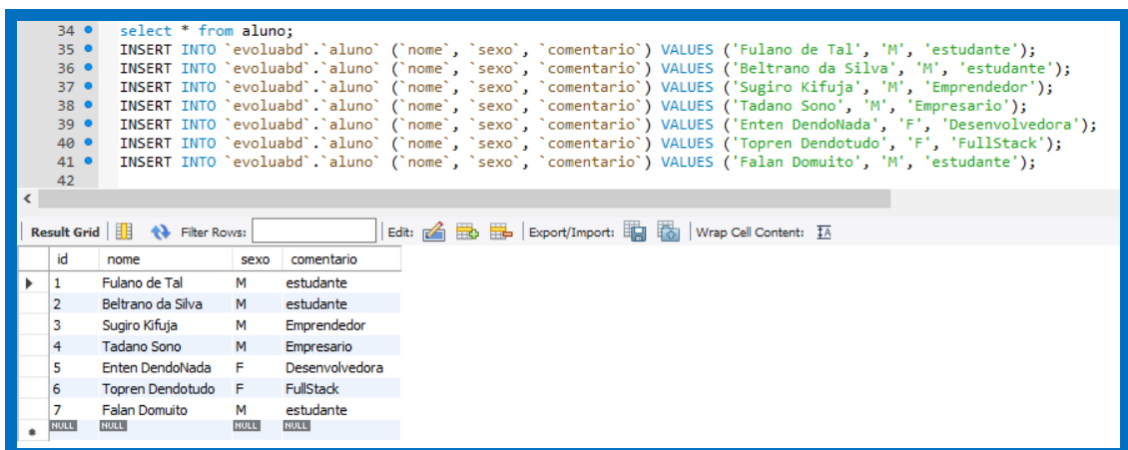


```

1
2
3
4 • create database EvoluaBD;
5 • use EvoluaBD;
6
7 • create table Aluno
8 • (
9 •   id int not null auto_increment,
10 •  nome varchar(50) not null,
11 •  sexo char(1),
12 •  comentario varchar(50),
13 •  primary key(id)
14 • );
15
16 • create table Curso
17 • (
18 •   idcurso int not null auto_increment,
19 •  nome varchar(50) not null,
20 •  descricao varchar(50),
21 •  duracao_mes int,
22 •  primary key(idcurso)
23 • );
24
25
26

```

Para simplificar, vamos popular nossas tabelas !



```

34 • select * from aluno;
35 • INSERT INTO `evoluaabd`.`aluno` (`nome`, `sexo`, `comentario`) VALUES ('Fulano de Tal', 'M', 'estudante');
36 • INSERT INTO `evoluaabd`.`aluno` (`nome`, `sexo`, `comentario`) VALUES ('Beltrano da Silva', 'M', 'estudante');
37 • INSERT INTO `evoluaabd`.`aluno` (`nome`, `sexo`, `comentario`) VALUES ('Sugiro Kifuja', 'M', 'Emprendedor');
38 • INSERT INTO `evoluaabd`.`aluno` (`nome`, `sexo`, `comentario`) VALUES ('Tadano Sono', 'M', 'Empresario');
39 • INSERT INTO `evoluaabd`.`aluno` (`nome`, `sexo`, `comentario`) VALUES ('Enten DendoNada', 'F', 'Desenvolvedora');
40 • INSERT INTO `evoluaabd`.`aluno` (`nome`, `sexo`, `comentario`) VALUES ('Topren Dendotudo', 'F', 'FullStack');
41 • INSERT INTO `evoluaabd`.`aluno` (`nome`, `sexo`, `comentario`) VALUES ('Falan Domuito', 'M', 'estudante');
42

```

id	nome	sexo	comentario
1	Fulano de Tal	M	estudante
2	Beltrano da Silva	M	estudante
3	Sugiro Kifuja	M	Emprendedor
4	Tadano Sono	M	Empresario
5	Enten DendoNada	F	Desenvolvedora
6	Topren Dendotudo	F	FullStack
7	Falan Domuito	M	estudante
*	NULL	NULL	NULL

```

44 • select * from curso;
45 • INSERT INTO `evoluabd`.`curso` (`nome`, `descricao`, `duracao_mes`) VALUES ('Java', 'Essencial', '2');
46 • INSERT INTO `evoluabd`.`curso` (`nome`, `descricao`, `duracao_mes`) VALUES ('Java', 'Intermediario', '3');
47 • INSERT INTO `evoluabd`.`curso` (`nome`, `descricao`, `duracao_mes`) VALUES ('C#', 'Essencial', '2');
48 • INSERT INTO `evoluabd`.`curso` (`nome`, `descricao`, `duracao_mes`) VALUES ('C#', 'Intermediario', '3');
49 • INSERT INTO `evoluabd`.`curso` (`nome`, `descricao`, `duracao_mes`) VALUES ('PHP', 'Intermediario', '3');
50 • INSERT INTO `evoluabd`.`curso` (`nome`, `descricao`, `duracao_mes`) VALUES ('HTML5', 'Intermediario', '3');
51

```

	idcurso	nome	descricao	duracao_mes
▶	1	Java	Essencial	2
	2	Java	Intermediario	3
	3	C#	Essencial	2
	4	C#	Intermediario	3
	5	PHP	Intermediario	3
	6	HTML5	Intermediario	3
*	NULL	NULL	NULL	NULL

Agora com todos os dados inseridos, vamos ao mais importante, conhecer um pouco mais do Comando **SELECT** – que tão poderoso este comando é !

```

54 • select * from aluno;

```

	idcurso	nome	descricao	duracao_mes
▶	1	Java	Essencial	2
	2	Java	Intermediario	3
	3	C#	Essencial	2
	4	C#	Intermediario	3
	5	PHP	Intermediario	3
	6	HTML5	Intermediario	3
*	NULL	NULL	NULL	NULL

Podemos de maneira bem simples, trazer apenas uma coluna de uma determinada tabela.

```

49 • select nome from aluno;
50
51

```

	nome
▶	Fulano de Tal
	Beltrano da Silva
	Sugiro Kifuja
	Tadano Sono
	Enten DendoNada
	Topren Dendotudo
	Falan Domuito

Ou até mesclar duas colunas de duas tabelas como demonstramos a seguir, lembrando no script abaixo pedimos para listar todos os dados da tabela aluno e curso , o retorno virá da seguinte maneira:

⇒ Não temos nenhuma referencia neste caso entre as tabelas, sendo assim os dados se juntarão como demonstrado abaixo:

52 • `select * from aluno, curso;`
53

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

	id	nome	sexo	comentario	idcurso	nome	descricao	duracao_mes
▶	1	Fulano de Tal	M	estudante	1	Java	Essencial	2
	1	Fulano de Tal	M	estudante	2	Java	Intermediario	3
	1	Fulano de Tal	M	estudante	3	C#	Essencial	2
	1	Fulano de Tal	M	estudante	4	C#	Intermediario	3
	1	Fulano de Tal	M	estudante	5	PHP	Intermediario	3
	1	Fulano de Tal	M	estudante	6	HTML5	Intermediario	3
	2	Beltrano da Silva	M	estudante	1	Java	Essencial	2
	2	Beltrano da Silva	M	estudante	2	Java	Intermediario	3
	2	Beltrano da Silva	M	estudante	3	C#	Essencial	2
	2	Beltrano da Silva	M	estudante	4	C#	Intermediario	3
	2	Beltrano da Silva	M	estudante	5	PHP	Intermediario	3
	2	Beltrano da Silva	M	estudante	6	HTML5	Intermediario	3
	3	Sugiro Kifuja	M	Emprende...	1	Java	Essencial	2
	3	Sugiro Kifuja	M	Emprende...	2	Java	Intermediario	3
	3	Sugiro Kifuja	M	Emprende...	3	C#	Essencial	2
	3	Sugiro Kifuja	M	Emprende...	4	C#	Intermediario	3
	3	Sugiro Kifuja	M	Emprende...	5	PHP	Intermediario	3
	3	Sugiro Kifuja	M	Emprende...	6	HTML5	Intermediario	3
	4	Tadano Sono	M	Empresario	1	Java	Essencial	2
	4	Tadano Sono	M	Empresario	2	Java	Intermediario	3
	4	Tadano Sono	M	Empresario	3	C#	Essencial	2
	4	Tadano Sono	M	Empresario	4	C#	Intermediario	3
	4	Tadano Sono	M	Empresario	5	PHP	Intermediario	3
	4	Tadano Sono	M	Empresario	6	HTML5	Intermediario	3
	5	Enten DendoNada	F	Desenvolv...	1	Java	Essencial	2
	5	Enten DendoNada	F	Desenvolv...	2	Java	Intermediario	3
	5	Enten DendoNada	F	Desenvolv...	3	C#	Essencial	2
	5	Enten DendoNada	F	Desenvolv...	4	C#	Intermediario	3

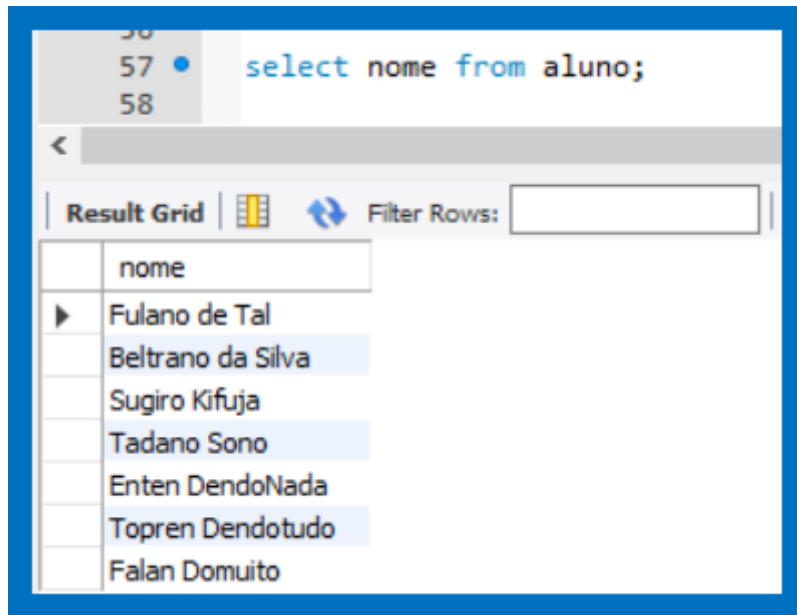
Result 164 x

Porém, vemos que o código está totalmente desordenado e sem fundamento algum, sendo assim, mais uma vez mostramos o poder enorme que o **SELECT** tem, vamos minimizar as falhas e trazer de maneira mais consistente ainda neste artigo, porém vamos ainda mostrar o grande poder do SELECT.

Tendo por base a tabela “aluno” a seguir , vamos manipular algumas queries:

	id	nome	sexo	comentario
▶	1	Fulano de Tal	M	estudante
	2	Beltrano da Silva	M	estudante
	3	Sugiro Kifuja	M	Emprendedor
	4	Tadano Sono	M	Empresario
	5	Enten DendoNada	F	Desenvolvedora
	6	Topren Dendotudo	F	FullStack
	7	Falan Domuito	M	estudante
•	NULL	NULL	NULL	NULL

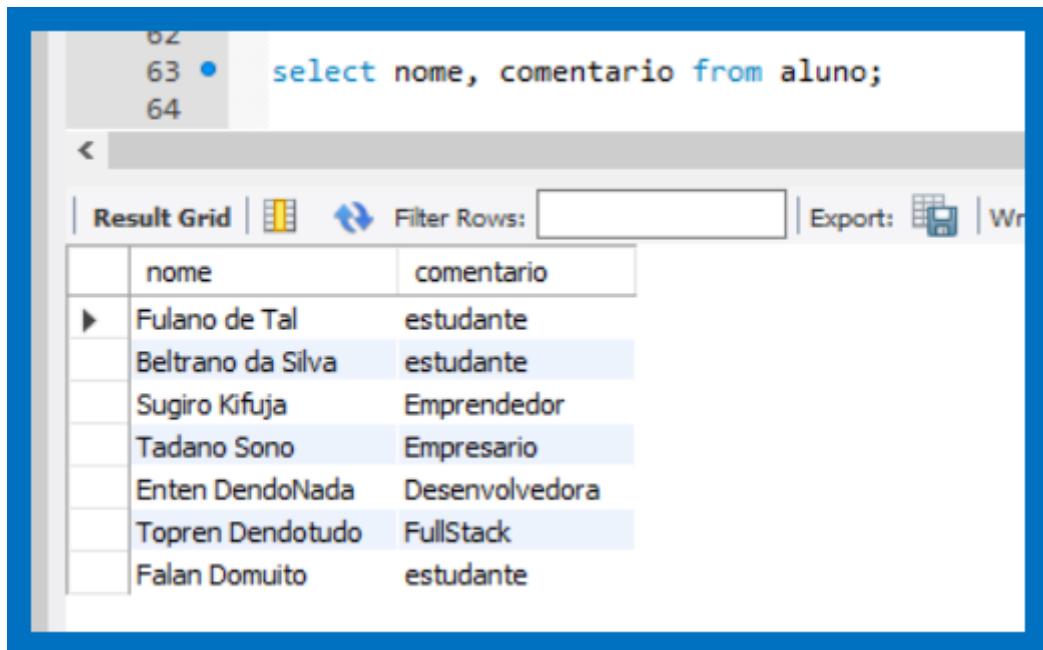
Para retornar apenas os dados da coluna **NOME** podemos facilmente invocar o seguinte código, onde trará o seguinte resultado.



The screenshot shows a SQL query editor with the query `select nome from aluno;` entered. Below the query, the 'Result Grid' is displayed, showing a table with one column named 'nome' and seven rows of names.

nome
Fulano de Tal
Beltrano da Silva
Sugiro Kifuja
Tadano Sono
Enten DendoNada
Topren Dendotudo
Falan Domuito

Para retornar apenas os dados da coluna **NOME** e **COMENTARIO** podemos facilmente invocar o seguinte código, onde trará o seguinte resultado MANIPULANDO a mesma tabela porem somente resultando DUAS colunas.



The screenshot shows a SQL query editor with the query `select nome, comentario from aluno;` entered. Below the query, the 'Result Grid' is displayed, showing a table with two columns: 'nome' and 'comentario'. The table contains seven rows of data.

nome	comentario
Fulano de Tal	estudante
Beltrano da Silva	estudante
Sugiro Kifuja	Emprendedor
Tadano Sono	Empresario
Enten DendoNada	Desenvolvedora
Topren Dendotudo	FullStack
Falan Domuito	estudante

Para se tornar um pouco mais PERFORMÁTICO nossa TABELA, podemos alterar o nome da coluna, deixando de uma maneira ainda mais profissional utilizando de um ALIAS onde ele alterará o nome de uma determinada coluna para um nome que seja mais didático aos olhos de quem irá obter o retorno dos dados.

Consulta normal Banco de Dados

```
65 • select nome, sexo, comentario from aluno;
66
67
```

nome	sexo	comentario
Fulano de Tal	M	estudante
Beltrano da Silva	M	estudante
Sugiro Kifuja	M	Emprendedor
Tadano Sono	M	Empresario
Enten DendoNada	F	Desenvolvedora
Topren Dendotudo	F	FullStack
Falan Domuito	M	estudante

Consulta utilizando ALIAS renomeando colunas da tabela

```
67
68 • select
69   nome as NomeDoAluno,
70   sexo as Sexo,
71   Comentario as InformacaoAdicional
72   from aluno;
73
```

NomeDoAluno	Sexo	InformacaoAdicional
Fulano de Tal	M	estudante
Beltrano da Silva	M	estudante
Sugiro Kifuja	M	Emprendedor
Tadano Sono	M	Empresario
Enten DendoNada	F	Desenvolvedora
Topren Dendotudo	F	FullStack
Falan Domuito	M	estudante

Há diversos recursos interessantes ainda, porém vamos agora manipular as tabelas pelo INNER JOIN.

INNER JOIN => Uma cláusula **join** da SQL - correspondente a uma operação de junção em álgebra relacional - combina colunas de uma ou mais tabelas em um banco de dados relacional. Ela cria um conjunto que pode ser salvo como uma tabela ou usado da forma como está.

[https://pt.wikipedia.org/wiki/Join_\(SQL\)](https://pt.wikipedia.org/wiki/Join_(SQL))

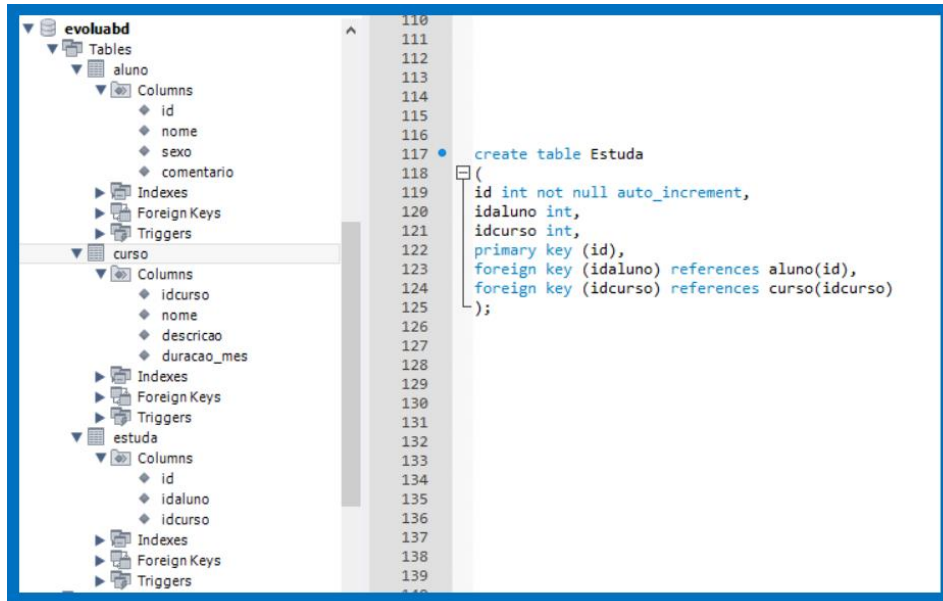
Para que usemos deste nosso Banco de Dados, precisamos relacioná-las, sendo assim, como temos as Tabelas Aluno e Curso, vamos criar agora uma tabela ESTUDA para relacioná-las por chaves estrangeiras.

CHAVE ESTRANGEIRA => No contexto dos banco de dados, o conceito de chave estrangeira ou chave externa se refere ao tipo de relacionamento entre distintas tabelas de dados do banco de dados.

Uma chave estrangeira é chamada quando há o relacionamento entre duas tabelas.

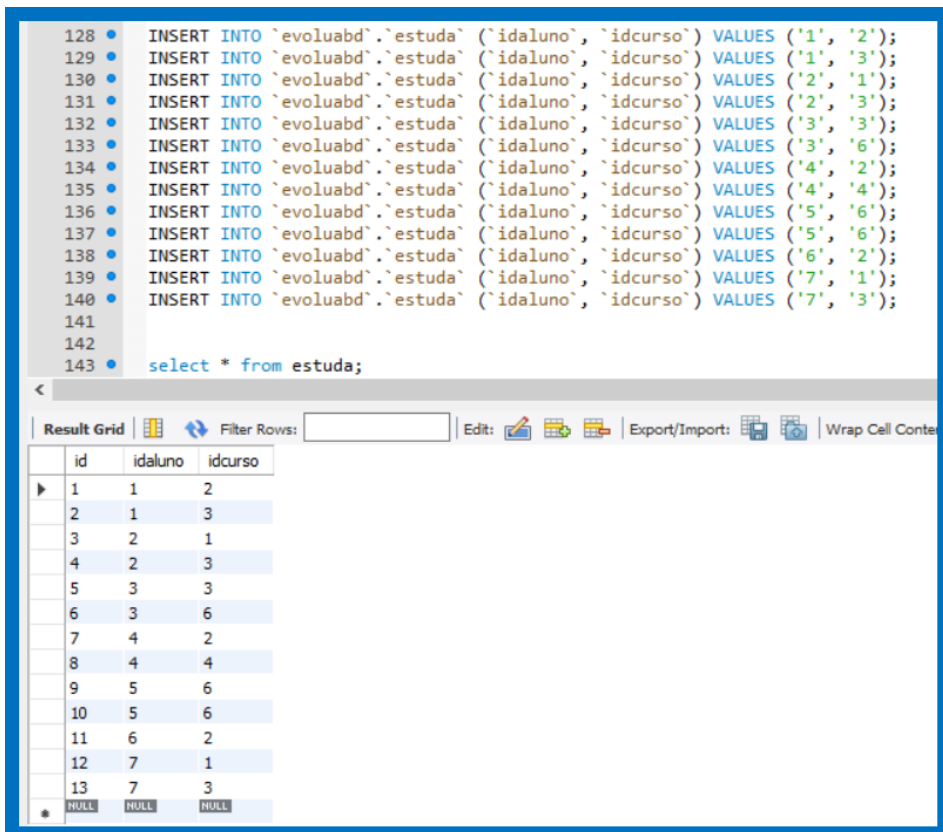
https://pt.wikipedia.org/wiki/Chave_estrangeira

Vamos criar nossa tabela e popular:



The screenshot shows a database management tool interface. On the left, the 'evoluabd' database is expanded, showing tables 'aluno' and 'curso'. The 'aluno' table has columns 'id', 'nome', 'sexo', and 'comentario'. The 'curso' table has columns 'idcurso', 'nome', 'descricao', and 'duracao_mes'. The 'estuda' table is also shown with columns 'id', 'idaluno', and 'idcurso'. On the right, the SQL code for creating the 'Estuda' table is displayed:

```
create table Estuda
(
  id int not null auto_increment,
  idaluno int,
  idcurso int,
  primary key (id),
  foreign key (idaluno) references aluno(id),
  foreign key (idcurso) references curso(idcurso)
);
```



The screenshot shows the same database management tool interface. On the left, the SQL code for inserting data into the 'estuda' table is displayed:

```
INSERT INTO `evoluabd`.`estuda` (`idaluno`, `idcurso`) VALUES ('1', '2');
INSERT INTO `evoluabd`.`estuda` (`idaluno`, `idcurso`) VALUES ('1', '3');
INSERT INTO `evoluabd`.`estuda` (`idaluno`, `idcurso`) VALUES ('2', '1');
INSERT INTO `evoluabd`.`estuda` (`idaluno`, `idcurso`) VALUES ('2', '3');
INSERT INTO `evoluabd`.`estuda` (`idaluno`, `idcurso`) VALUES ('3', '3');
INSERT INTO `evoluabd`.`estuda` (`idaluno`, `idcurso`) VALUES ('3', '6');
INSERT INTO `evoluabd`.`estuda` (`idaluno`, `idcurso`) VALUES ('4', '2');
INSERT INTO `evoluabd`.`estuda` (`idaluno`, `idcurso`) VALUES ('4', '4');
INSERT INTO `evoluabd`.`estuda` (`idaluno`, `idcurso`) VALUES ('5', '6');
INSERT INTO `evoluabd`.`estuda` (`idaluno`, `idcurso`) VALUES ('5', '6');
INSERT INTO `evoluabd`.`estuda` (`idaluno`, `idcurso`) VALUES ('6', '2');
INSERT INTO `evoluabd`.`estuda` (`idaluno`, `idcurso`) VALUES ('7', '1');
INSERT INTO `evoluabd`.`estuda` (`idaluno`, `idcurso`) VALUES ('7', '3');
```

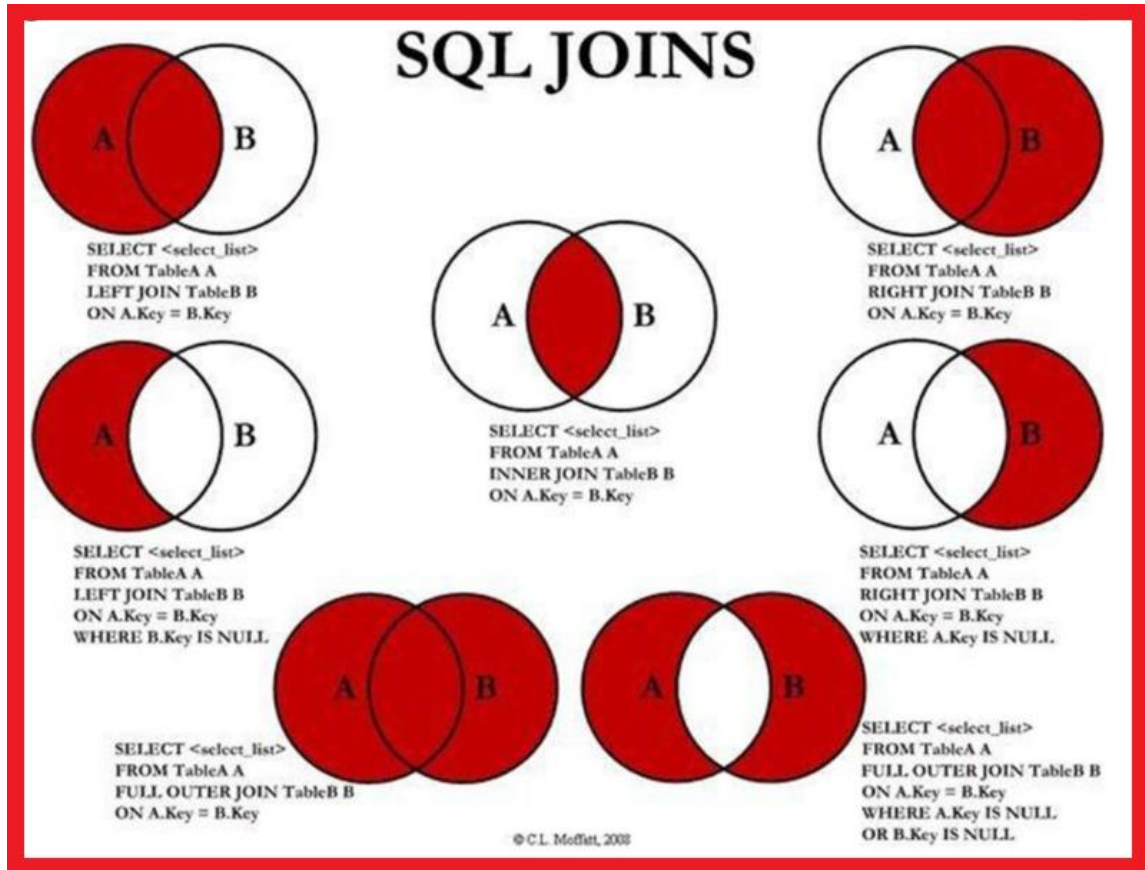
Below the code, the 'select * from estuda;' query is executed, and the results are displayed in a grid:

	id	idaluno	idcurso
1	1	2	
2	1	3	
3	2	1	
4	2	3	
5	3	3	
6	3	6	
7	4	2	
8	4	4	
9	5	6	
10	5	6	
11	6	2	
12	7	1	
13	7	3	
*	NULL	NULL	NULL

Pronto, agora que inserimos os dados vamos à mágica !

Lembrando que existem algumas maneiras de manipularmos a cláusula JOIN porém neste artigo, estarei apenas manipulando o INNER JOIN convencional.

Antes, vamos aos tipos de INNER JOIN:



Vamos à prática de acordo com nosso Banco de Dados criado:

Primeiramente, vamos fazer um **SELECT** da tabela aluno vinculando com a tabela estuda, porém é notório que os dados estão “juntos” e que para fins de apresentação se torna um pouco “poluído”

```

145 • select * from aluno
146 inner join estuda
147 on aluno.id = estuda.idaluno;

```

	id	nome	sexo	comentario	id	idaluno	idcurso
▶	1	Fulano de Tal	M	estudante	1	1	2
	1	Fulano de Tal	M	estudante	2	1	3
	2	Beltrano da Silva	M	estudante	3	2	1
	2	Beltrano da Silva	M	estudante	4	2	3
	3	Sugiro Kifuja	M	Emprendedor	5	3	3
	3	Sugiro Kifuja	M	Emprendedor	6	3	6
	4	Tadano Sono	M	Empresario	7	4	2
	4	Tadano Sono	M	Empresario	8	4	4
	5	Enten DendoNada	F	Desenvolvedora	9	5	6
	5	Enten DendoNada	F	Desenvolvedora	10	5	6
	6	Topren Dendotudo	F	FullStack	11	6	2
	7	Falan Domuito	M	estudante	12	7	1
	7	Falan Domuito	M	estudante	13	7	3

Suponhamos que os dados que nos interessa seja somente Nome e Comentário, neste caso, é simples, onde apenas informaremos quais as colunas queremos que seja relacionada no nosso script SQL mesmo com o INNER JOIN como mostro abaixo:

```

153 • select a.nome, a.comentario from aluno a
154 inner join estuda e
155 on a.id = e.idaluno;
156

```

nome	comentario
Fulano de Tal	estudante
Fulano de Tal	estudante
Beltrano da Silva	estudante
Beltrano da Silva	estudante
Sugiro Kifuja	Empreendedor
Sugiro Kifuja	Empreendedor
Tadano Sono	Empresario
Tadano Sono	Empresario
Enten DendoNada	Desenvolvedora
Enten DendoNada	Desenvolvedora
Topren Dendotudo	FullStack
Falan Domuito	estudante
Falan Domuito	estudante

Neste momento, vamos agora “mesclar” os dados da Tabela aluno e estuda:

```

157 • select a.nome, e.idcurso from aluno a
158 inner join estuda e
159 on a.id = e.idaluno;
160

```

nome	idcurso
Fulano de Tal	2
Fulano de Tal	3
Beltrano da Silva	1
Beltrano da Silva	3
Sugiro Kifuja	3
Sugiro Kifuja	6
Tadano Sono	2
Tadano Sono	4
Enten DendoNada	6
Enten DendoNada	6
Topren Dendotudo	2
Falan Domuito	1
Falan Domuito	3

Nota-se que os dados das duas tabelas retornaram, porem, os dados da tabela estuda está retornando o código, e não faz muito sentido apresentarmos “código” ao cliente, vamos assim, referenciá-lo com a tabela curso e melhorar nossa querie.

```

160
161 • select a.nome, e.idcurso, c.nome from aluno a
162 inner join estuda e
163 on a.id = e.idaluno
164 inner join curso c
165 on e.idcurso = c.idcurso;
166

```

nome	idcurso	nome
Beltrano da Silva	1	Java
Falan Domuito	1	Java
Fulano de Tal	2	Java
Tadano Sono	2	Java
Topren Dendotudo	2	Java
Fulano de Tal	3	C#
Beltrano da Silva	3	C#
Sugiro Kifuja	3	C#
Falan Domuito	3	C#
Tadano Sono	4	C#
Sugiro Kifuja	6	HTML5
Enten DendoNada	6	HTML5
Enten DendoNada	6	HTML5

O resultado está OK, vamos agora apenas melhorar “os nomes” das nossas tabelas para sermos mais “profissionais” :

```

174
175 • select a.nome as Aluno, c.nome as Curso from aluno a
176 inner join estuda e
177 on a.id = e.idaluno
178 inner join curso c
179 on e.idcurso = c.idcurso;
180
181 • select
182 a.nome as Aluno,
183 c.nome as Curso
184 from aluno a
185 inner join estuda e
186 on a.id = e.idaluno
187 inner join curso c
188 on e.idcurso = c.idcurso;
189
190

```

Aluno	Curso
Beltrano da Silva	Java
Falan Domuito	Java
Fulano de Tal	Java
Tadano Sono	Java
Topren Dendotudo	Java
Fulano de Tal	C#
Beltrano da Silva	C#
Sugiro Kifuja	C#
Falan Domuito	C#
Tadano Sono	C#
Sugiro Kifuja	HTML5
Enten DendoNada	HTML5
Enten DendoNada	HTML5

Espero ter trazido um resumo bem prático e didático sobre o PODEROSO comando SELECT !

Os códigos estarão disponíveis no Git.

Até mais !

Espero ter ajudado !