

# Banco de dados para a Oficina de Perna-de-Pau

---

O projeto contempla o desenvolvimento de um banco de dados relacionais para a Oficina Perna-de-Pau.

## Sobre

A Oficina Perna de Pau é a atual maior escola, que oferece aulas de perna de pau no Rio de Janeiro.

Com o crescimento de seu número de alunos, intensificou-se a necessidade de uma preparação mais adequada dos dados gerados a partir dessas aulas para um crescimento mais saudável utilizando inteligência de negócio para melhorar a gestão do negócio.

O grande volume de informações descentralizadas foi se tornando um obstáculo no crescimento do negócio. Uma vez que o tempo utilizado na organização do próprio negócio poderia ser melhor administrado para questões relacionadas a negociação e fechamento das vendas.

## Scripts

1. [Criação de tabelas](#)
2. [Inserção de dados](#)
3. [Views](#)
4. [Funções](#)
5. [Stored Procedures](#)
6. [Triggers](#)
7. [Usuários](#)

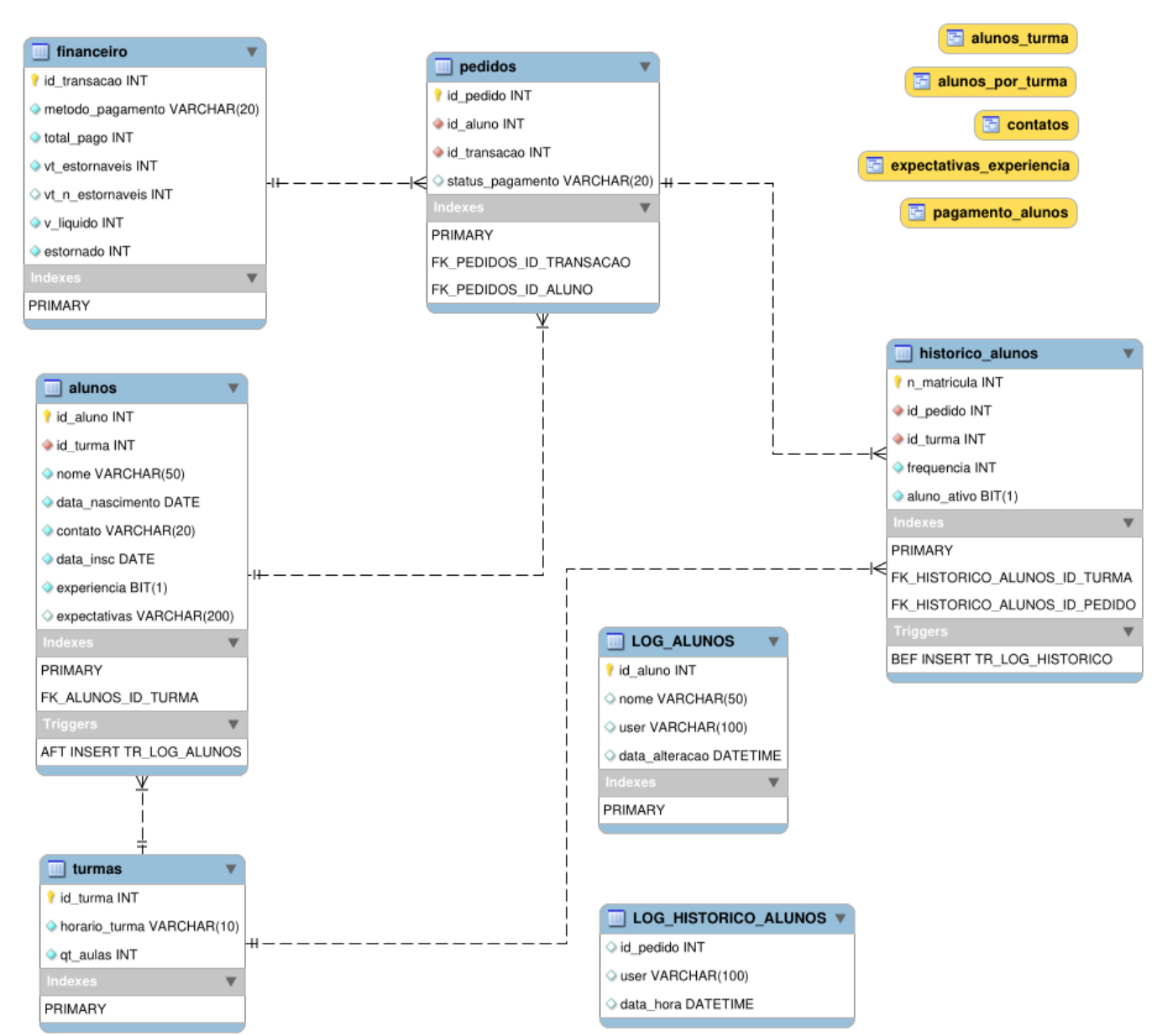
Observação: É importante que os scripts sejam rodados na ordem indicada pela numeração.

## Entidades, atributos e relacionamentos

Para a composição do banco de dados, foram definidas **cinco** entidades

- Turmas;
- Alunos;
- Financeiro;
- Pedidos e
- Histórico de alunos.

Os atributos e relacionamentos entre as entidades definidas podem ser vistos no diagrama a seguir:



Tabelas

Seguindo a lógica das entidades definidas anteriormente, foram estabelecidas cinco tabelas, sendo elas respectivamente turmas, alunos, financeiro, pedidos e historico\_alunos.

turmas

A tabela turmas possui informações de turma, quantidade de aulas e horários.

Nome do atributo	Tipo do dado	Chave	Índice	Descrição
id_turma	INT	PK	PRIMARY	ID da turma
horario_turma	VARCHAR(10)	-	-	Horário da aula da turma
qt_aulas	INT	-	-	Quantidade de aulas por turma

alunos

A tabela alunos contém os dados pessoais dos alunos, como nome e data de nascimento, além da turma que frequenta.

Nome do atributo	Tipo do dado	Chave	Índice	Descrição
id_aluno	INT	PK	PRIMARY	ID do aluno
id_turma	INT	FK	FK_ALUNOS_ID_TURMA	ID da turma
nome	VARCHAR(50)	-	-	Nome do aluno
data_nascimento	DATE	-	-	Data de nascimento
contato	VARCHAR(20)	-	-	Contato do aluno
data_insc	DATE	-	-	Data de inscrição
experiencia	BIT(1)	-	-	Experiência com perna-de-pau
expectativas	VARCHAR(200)	-	-	Expectativas com a oficina

financeiro

A tabela financeiro contém dados financeiros alunos matriculados. É a partir dela que é gerado o id\_transação que identifica as informações financeiras dos alunos. A tabela contém dados sobre o método de pagamento, tabela e valor total pago.

Nome do atributo	Tipo do dado	Chave	Índice	Descrição
id_transacao	INT	PK	PRIMARY	ID da transacao
metodo_pagamento	VARCHAR(20)	-	-	Método de pagamento
total_pago	INT	-	-	Valor total pago
vt_estornaveis	INT	-	-	Valor de taxas estornáveis
vt_n_estornaveis	INT	-	-	Valor de taxas não estornáveis
estornado	INT	-	-	Valor estornado

pedidos

A tabela pedidos possui o registro dos pedidos realizados por cada aluno, o id da transação e o status de pagamento da oficina.

Nome do atributo	Tipo do dado	Chave	Índice	Descrição
------------------	--------------	-------	--------	-----------

Nome do atributo	Tipo do dado	Chave	Índice	Descrição
id_pedido	INT	PK	PRIMARY	ID do pedido
id_aluno	INT	FK	FK_PEDIDOS_ID_ALUNO	ID do aluno
id_transacao	INT	FK	FK_PEDIDOS_ID_TRANSACAO	ID da transação
status_pagamento	VARCHAR(20)	-	-	Status de pagamento

historico\_alunos

A tabela historico\_alunos possui dados dos alunos matriculados na oficina perna-de-pau. São dados gerados inicialmente no ato da matrícula a partir de um formulário, juntamente com os ids de pedido, turma e aluno.

Nome do atributo	Tipo do dado	Chave	Índice	Descrição
n_matricula	INT	PK	PRIMARY	Número de matricula
id_pedido	INT	FK	FK_HISTORICO_ALUNOS_ID_PEDIDO	ID do pedido
id_turma	INT	FK	FK_HISTORICO_ALUNOS_ID_TURMA	ID da turma
frequencia	INT	-	-	Frequência nas aulas
aluno_ativo	INT	-	-	Aluno ativo ou se já finalizou a oficina.

Tabelas de auditoria

LOG\_ALUNOS

A tabela LOG\_ALUNOS apresenta o histórico de inserção de dados na tabela alunos.

Nome do atributo	Tipo do dado	Chave	Índice	Descrição
id_aluno	INT	PK	PRIMARY	ID do aluno
nome	VARCHAR(50)	-	-	Nome do aluno
user	VARCHAR(100)	-	-	Usuário que inseriu no banco de dados
data_alteracao	DATETIME	-	-	Data de inserção

LOG\_HISTORICO\_ALUNOS

A tabela LOG\_HISTORICO\_ALUNOS apresenta o histórico de tentativas de inserção de dados (sucedidas ou não) na tabela historico\_aluno.

Nome do atributo	Tipo do dado	Chave	Índice	Descrição
id_pedido	INT	-	-	ID do pedido
user	VARCHAR(100)	-	-	Usuário que inseriu/tentou inserir no banco de dados
data_hora	DATETIME	-	-	Data de inserção e/ou tentativa

## Views

Para otimizar consultas na tabela, foram criadas 5 views, sendo elas pagamento\_alunos, alunos\_turma, expectativas\_experiencia, contatos e alunos\_por\_turma.

### pagamento\_alunos

A view pagamento\_alunos possui os dados pessoais de cada aluno juntamente com as informações financeiras, para verificar se algum aluno pagou ou não a mensalidade e quanto falta para ser pago.

```
2 • SELECT * FROM pagamento_alunos;
```

id_pedido	nome	contato	metodo_pagamento	status_pagamento	total_pago	falta
1	João Silva	(11) 98765-4321	Cartão de Crédito	Pago	500	0
2	Maria Souza	(21) 99876-5432	Boleto	Pendente	300	200
3	Pedro Santos	(31) 87654-3210	Transferência	Pago	200	300
4	Ana Oliveira	(51) 91234-5678	Pix	Pendente	500	0
5	Carlos Mendes	(11) 93245-6789	Dinheiro	Pago	150	350
6	Larissa Oliveira	(11) 91234-5678	Cartão de Débito	Recusado	500	0
7	Fernando Men...	(21) 93245-6789	Boleto	Pendente	350	150
8	Camila Rodrigues	(31) 91234-5678	Transferência	Pago	500	0

### alunos\_turma

A view alunos\_turma é composta pela matricula e nome dos alunos, a turma em que estão matriculados e a frequência nas aulas, sendo este último o principal objetivo da view.

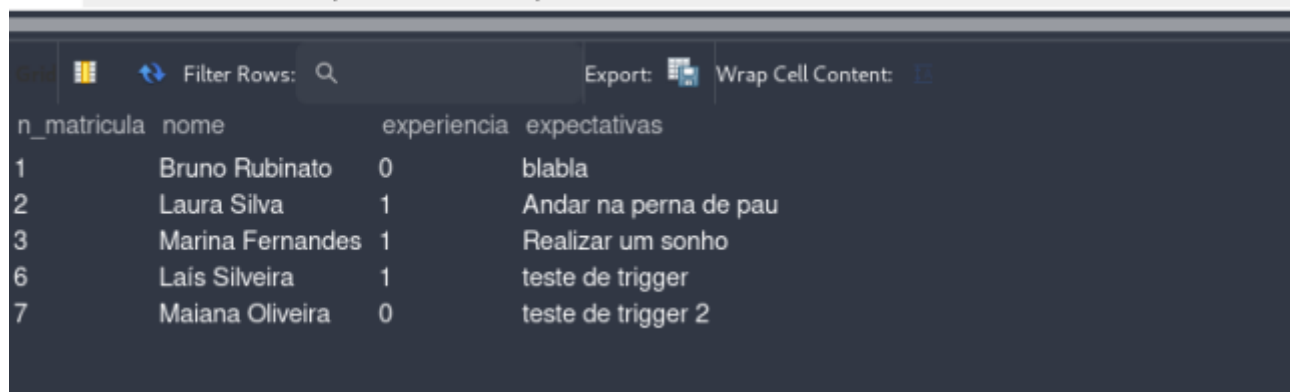
```
5 • SELECT * FROM alunos_turma;
```

n_matricula	nome	id_turma	horario_turma	frequencia
1	Bruno Rubinato	1	08h	4
2	Laura Silva	2	11h	1
3	Marina Fernandes	1	08h	6
6	Laís Silveira	2	11h	1
7	Maiana Oliveira	1	08h	2

## expectativas\_experiencia

A view de expectativas\_experiencia tem como objetivo mostrar os dados de expectativas do curso e experiência de cada aluno.

```
5 • SELECT * FROM expectativas_experiencia;
```

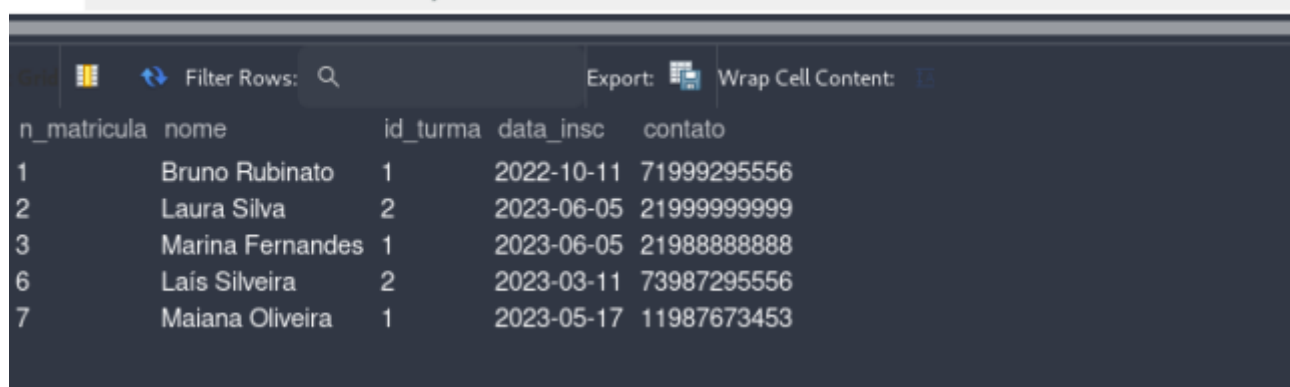


n_matricula	nome	experiencia	expectativas
1	Bruno Rubinato	0	blabla
2	Laura Silva	1	Andar na perna de pau
3	Marina Fernandes	1	Realizar um sonho
6	Laís Silveira	1	teste de trigger
7	Maiana Oliveira	0	teste de trigger 2

## contatos

A view contatos tem como objetivo mostrar o contato de cada aluno, juntamente com a data de ingresso.

```
5 • SELECT * FROM contatos;
```

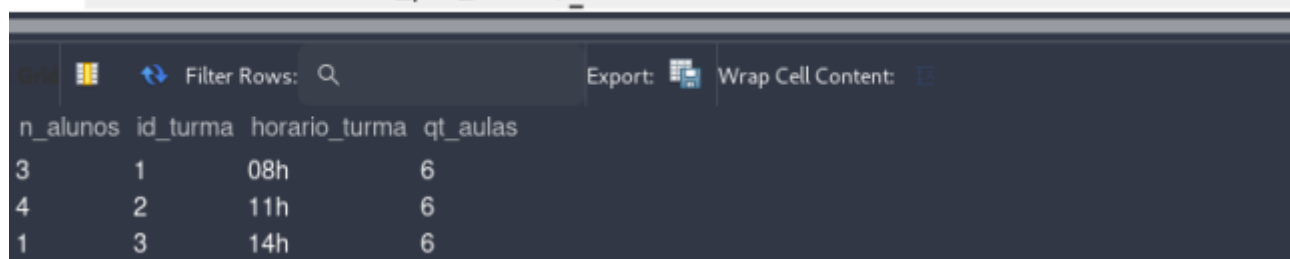


n_matricula	nome	id_turma	data_insc	contato
1	Bruno Rubinato	1	2022-10-11	71999295556
2	Laura Silva	2	2023-06-05	21999999999
3	Marina Fernandes	1	2023-06-05	21988888888
6	Laís Silveira	2	2023-03-11	73987295556
7	Maiana Oliveira	1	2023-05-17	11987673453

## alunos\_por\_turma

A view alunos\_por\_turma possui a quantidade de alunos por turma, horário das aulas e a quantidade de aulas.

```
5 • SELECT * FROM alunos_por_turma;
```



n_alunos	id_turma	horario_turma	qt_aulas
3	1	08h	6
4	2	11h	6
1	3	14h	6

## Funções

As funções criadas para esse banco de dados foram a receita\_total e a total\_taxas, com o objetivo de calcular valores da tabela financeiro e facilitar a análise da receita gerada pelo curso.

## receita\_total

É a diferença entre o valor líquido (parâmetro 1) e o valor das taxas estornáveis (parâmetro 2).

```
24 • SELECT receita_total(100, 45);
```

#	receita_total(100, 45)
1	55

## total\_texas

É o somatório de todas as taxas, estornáveis (parâmetro 1) e não estornáveis (parâmetro 2).

```
25 SELECT total_texas(3, 9);
```

#	total_texas(3, 9)
1	12

## Stored Procedures

Foram elaborados dois stored procedures, um de ordenamento de tabelas e outro para inserção de novos alunos na tabela alunos.

### ordem\_tabela

Ordena qualquer tabela do banco de dados a partir de três parâmetros: t\_nome (nome da tabela), col\_campo (campo de referência para o ordenamento) e ordem (ascendente ou descendente).

```
5 • call ordem_tabela('historico_alunos', 'frequencia', 'ASC');
```

n_matricula	id_pedido	id_turma	id_aluno	data_insc	contato	experiencia	expectativas	frequencia
2	2	2	2	2023-06-05	21999999999	1	Andar na perna de pau	1
6	4	2	6	2023-03-11	73987295556	1	teste de trigger	1
7	5	1	7	2023-05-17	11987673453	0	teste de trigger 2	2
1	1	2	1	2022-10-11	71999295556	0	blabla	4
3	3	1	3	2023-06-05	21988888888	1	Realizar um sonho	6

### novo\_aluno

Insere um novo aluno na tabela alunos, recebendo como parâmetro: a\_nome (nome do aluno), a\_dnasc (data de nascimento) e turma (número da turma).

```
1 • CALL novo_aluno('Joana Silva', '1998-08-25', 3, '(41) 98765-4321', '2023-07-26', 1, 'Tenho interesse em aprender novas técnicas.');
```

id_aluno	id_turma	nome	data_nascimento	contato	data_insc	experiencia	expectativas
10	3	Amanda Lima	2002-07-12	(11) 93245-6789	2023-07-30	0	Dedicada em aprender tudo que puder.
11	1	Rafaela Santos	1996-04-22	(21) 98765-4321	2023-07-25	1	Estou empolgada para aprender novas técnicas.
12	2	Gustavo Silva	1999-12-05	(31) 99876-5432	2023-07-26	1	Quero me especializar na área.
13	3	Fernanda Souza	1991-08-17	(51) 91234-5678	2023-07-27	1	Espero que seja uma experiência enriquecedora.
14	1	Marcelo Oliveira	1997-11-28	(11) 99876-5432	2023-07-28	0	Estou começando agora e tenho muito a aprender.
15	2	Carolina Mendes	1995-03-10	(21) 98765-4321	2023-07-30	0	Ansiosa para desenvolver minhas habilidades.
16	1	Julia Ferreira	1992-12-05	(11) 98765-1111	2023-08-01	1	Tenho interesse em programação e quero aprofundar meus conhe...
17	2	Lucas Santos	1992-03-20	(31) 12345-6789	2023-07-26	1	Espero aprender muito sobre o assunto.
18	3	Joana Silva	1998-08-25	(41) 98765-4321	2023-07-26	1	Tenho interesse em aprender novas técnicas.

## Triggers

Os triggers foram elaborados juntamente com as tabelas de log/auditoria (LOG\_ALUNOS e LOG\_HISTORICO\_ALUNOS), e tem a funcionalidade de registrar e verificar dados no banco antes de inseri-los.

### TR\_LOG\_ALUNOS

É acionado depois que um novo aluno é inserido no banco de dados. O registro de inserção é salvo na tabela LOG\_ALUNOS e conta com os dados do aluno inserido, o usuário que fez o procedimento e o momento (data e hora).

```
1 • CALL novo_aluno('Joana Silva', '1998-08-25', 3, '(41) 98765-4321', '2023-07-26', 1, 'Tenho interesse em aprender novas técnicas.');
```

```
2 • SELECT * FROM LOG_ALUNOS;
```

id_aluno	nome	user	data_alteracao
16	Julia Ferreira	root@localhost	2023-07-27 13:32:22
17	Lucas Santos	administrativo@localhost	2023-07-27 15:31:18
18	Joana Silva	administrativo@localhost	2023-07-27 15:35:58

### TR\_LOG\_HISTORICO\_ALUNOS

É feita a verificação do status de pagamento do pedido do aluno antes de inserir os dados dele na tabela historico\_alunos. Caso o pagamento não tenha sido realizado, aparecerá uma mensagem de erro. O acompanhamento da inserção desses dados é feita na tabela LOG\_HISTORICO\_ALUNOS, que conta com o número do pedido, usuário e a data e hora do procedimento.

```
5 • SELECT * FROM LOG_HISTORICO_ALUNOS;
```

id_pedido	user	data_hora
4	root@localhost	2023-07-06 22:10:09
5	root@localhost	2023-07-06 22:16:43



# Usuários e permissões

Foram criados três perfis de usuários diferentes para manipulação dos dados do banco, são eles:

- RH;
- Financeiro;
- Administrativo;

## RH

O perfil RH é destinado para o setor de Recursos Humanos e só está permitido a acessar os dados de todas as tabelas pelo modo de leitura (SELECT).

Descrição	DML	Grants	Tabelas
Leitura	SELECT	Sim	turmas, alunos, financeiro, pedidos, historico_alunos, Views
Inserção	INSERT	Não	-
Atualização	UPDATE	Não	-
Remoção	DELETE	Não	-

## Financeiro

O perfil Financeiro é destinado para o setor de contabilidade. O usuário poderá ler os dados de todas as tabelas (SELECT) mas apenas pode inserir ou modificar (INSERT, UPDATE) os dados das tabelas pedidos e financeiro.

Descrição	DML	Grants	Tabelas
Leitura	SELECT	Sim	turmas, alunos, financeiro, pedidos, historico_alunos, View pagamento_alunos
Inserção	INSERT	Sim	financeiro e pedidos
Atualização	UPDATE	Sim	financeiro e pedidos
Remoção	DELETE	Não	-

## Administrativo

O perfil Administrativo está destinado ao setor de administração e poderá ler os dados de todas as tabelas (SELECT) mas apenas inserir ou modificar (INSERT, UPDATE) os dados das tabelas turmas, alunos e historico\_alunos.

Descrição	DML	Grants	Tabelas
-----------	-----	--------	---------

Descrição	DML	Grants	Tabelas
Leitura	SELECT	Sim	turmas, alunos, financeiro, pedidos, historico_alunos, LOG_ALUNOS, LOG_HISTORICO_ALUNOS, Views e Stored Procedures
Inserção	INSERT	Sim	turmas, alunos e historico_alunos
Atualização	UPDATE	Sim	turmas, alunos e historico_alunos
Remoção	DELETE	Não	-

Exemplos de utilização

Inserindo um novo aluno na tabela alunos

Podemos tentar inserir novos dados com o seguinte código:

```
INSERT INTO alunos (id_turma, nome, data_nascimento, contato, data_insc,
experiencia, expectativas)
VALUES (2, 'Lucas Santos', '1992-03-20', '(31) 12345-6789', '2023-07-26',
1, 'Espero aprender muito sobre o assunto.');
```

Caso preferir, poderá usar também o stored procedure novo\_aluno:

```
CALL novo_aluno('Joana Silva', '1998-08-25', 3, '(41) 98765-4321', '2023-
07-26', 1, 'Tenho interesse em aprender novas técnicas.');
```

O resultado dos exemplos consultando na tabela alunos :

```
SELECT * FROM alunos;
```

Poderá consultar posteriormente as inserções realizadas na tabela por meio da tabela de auditoria LOG\_ALUNOS.

Inserindo registros na tabela financeiro

```
INSERT INTO financeiro (metodo_pagamento, total_pago, vt_estornaveis,
vt_n_estornaveis, v_liquido, estornado) VALUES ('Boleto', 400, 50, 0, 350,
0);
INSERT INTO financeiro (metodo_pagamento, total_pago, vt_estornaveis,
```

```
vt_n_estornaveis, v_liquido, estornado) VALUES ('Dinheiro', 300, 0, 0, 300, 0);
```

Ao consultar a tabela **financeiro** podemos verificar os dados inseridos:

```
SELECT * FROM financeiro;
```

### Inserindo registros na tabela pedidos

```
INSERT INTO pedidos (id_aluno, id_transacao, status_pagamento) VALUES (16, 'Pendente');  
INSERT INTO pedidos (id_aluno, id_transacao, status_pagamento) VALUES (17, 'Pago');
```

Podemos verificar os dados inseridos na tabela **pedidos**:

```
SELECT * FROM pedidos;
```

### Inserindo registros na tabela historico\_alunos

Antes é preciso saber que, para inserir um aluno na tabela de histórico de alunos, o aluno deverá ter pago a mensalidade ou o valor integral da oficina. Caso não aconteça, sua matrícula não será efetivada. Portanto, tente executar:

```
INSERT INTO historico_alunos (id_pedido, id_turma, frequencia, aluno_ativo)  
VALUES (16, 2, 80, 1);  
INSERT INTO historico_alunos (id_pedido, id_turma, frequencia, aluno_ativo)  
VALUES (17, 3, 95, 1);
```

Provavelmente, por conta do trigger **TR\_LOG\_HISTORICO** você não irá conseguir inserir os dados na tabela, uma vez que o aluno não efetuou o pagamento. Para alterar o dado e assim inserir podemos realizar a seguinte modificação:

```
UPDATE pedidos SET status_pagamento = 'Pago' WHERE id_pedido = 16;
```

Após isso, tente novamente inserir os dados na tabela de histórico de aluno:

```
INSERT INTO historico_alunos (id_pedido, id_turma, frequencia, aluno_ativo)  
VALUES (16, 2, 80, 1);
```

### Logs de auditoria

Para consultar as tabelas de auditoria:

```
SELECT * FROM LOG_ALUNOS;  
SELECT * FROM LOG_HISTORICO_ALUNOS;
```