DML e Primeira Conexão em Python

Aula 12 — Colocando dados dentro do banco

Aula 12 · 23/10/2026 · Indicadores 3, 5

UniSenac Centro Universitário RS
Senac
Fecomércio
IFEP
Sesc
Sindicatos Empresariais

O mapa de hoje

- **Bloco 1** — `INSERT`: colocar dados
- **Bloco 2** — `UPDATE` e `DELETE`: mudar e apagar (com muito cuidado)
- **Bloco 3** — Python conversando com o banco, pelo notebook
- **Bloco 4** — mão na massa: popular o banco pelos **dois** caminhos

> Hoje a gente sai do **esquema** e entra na **instância** (Aula 02). DDL fica para trás; a partir de agora é **DML**.

Retomando a Aula 11

- O que `REFERENCES` garante?
- Qual a diferença entre `PRIMARY KEY` e `UNIQUE`?
- Por que a tabela do meio do N:N é criada por último?

> Abram o pgAdmin e confirmem que **todas** as tabelas do grupo existem. Quem estiver com tabela faltando, me chame agora — hoje a gente enche elas.

Bloco 1

INSERT — colocando dados

O comando, traduzido

```
INSERT INTO plataforma (nome, fabricante)
VALUES ('PlayStation 5', 'Sony');
```

GLOSSÁRIO

`INSERT INTO`	inserir dentro de
`VALUES`	valores

Lê-se: *"insira dentro da tabela plataforma, nas colunas nome e fabricante, os valores 'PlayStation 5' e 'Sony'"*.

⚠ Aspas: a regra que mais dá erro

Tipo do valor	Como escrever
Texto (`VARCHAR`, `TEXT`)	`'PlayStation 5'` — **aspas simples**
Data (`DATE`)	`'2026-10-21'` — aspas simples, formato **ano-mês-dia**
Número (`INTEGER`, `NUMERIC`)	`2018` — **sem aspas**
Booleano	`TRUE` / `FALSE` — sem aspas
Vazio	`NULL` — sem aspas

Aspas simples (`'`), nunca aspas duplas. No PostgreSQL, aspas duplas significam *nome de coluna* — usar `"Sony"` faz o banco procurar uma coluna chamada Sony e dar erro.

Inserindo várias linhas de uma vez

```sql
INSERT INTO plataforma (nome, fabricante) VALUES
    ('PlayStation 5',   'Sony'),
    ('Nintendo Switch', 'Nintendo'),
    ('Xbox Series X',   'Microsoft');
```

Uma linha por parênteses, separados por vírgula. **Ponto e vírgula só no fim.**

A ordem de inserção também importa

> Não dá para inserir um `emprestimo` com `id_cliente = 1` se o cliente 1 ainda não existe. É a **mesma regra de ontem**, agora valendo para os dados.

1. Primeiro as tabelas **sem** chave estrangeira: `plataforma`, `cliente`, `jogo`
2. Depois as que dependem de uma só: `endereco`, `emprestimo`
3. Por último, as tabelas do meio: `jogos_plataforma` e `item_emprestimo`

Checagem do Bloco 1

O que está errado?

```
INSERT INTO jogo (titulo, genero, ano_lancamento, preco_diaria)
VALUES ("God of War", 'Ação', '2018', 12.00)
```

forms.gle/ivoRJB1gRWxSANLMA

Intervalo

10 minutos

Bloco 2

UPDATE e DELETE — com muito cuidado

UPDATE — mudar o que já existe

```sql
UPDATE jogo
SET    genero = 'RPG'
WHERE  id_jogo = 3;
```

GLOSSÁRIO

UPDATE	atualizar
SET	definir — atribua este valor
WHERE	onde — a condição que escolhe as linhas

⚠ O erro que apaga o banco inteiro

`UPDATE` **ou** `DELETE` **sem** `WHERE` **vale para TODAS as linhas da tabela.** Sem aviso, sem confirmação, sem desfazer.

```
UPDATE jogo SET genero = 'RPG';    -- TODOS os jogos viram RPG
DELETE FROM jogo;                  -- TODOS os jogos somem
```

O hábito que protege

1. Escreva **primeiro** um `SELECT` com o mesmo `WHERE`
2. Confira **quantas linhas** apareceram — é esse tanto que vai mudar
3. Só então troque `SELECT *` por `UPDATE ... SET` ou `DELETE`

```
SELECT * FROM jogo WHERE id_jogo = 3;    -- 1 linha? ótimo
DELETE  FROM jogo WHERE id_jogo = 3;     -- agora sim
```

`DELETE` e a integridade referencial

```
DELETE FROM cliente WHERE id_cliente = 1;
```

Se existem empréstimos apontando para esse cliente, o banco **recusa**:

```
ERROR: update or delete on table "cliente" violates foreign key
constraint on table "emprestimo"
```

É o `ON DELETE RESTRICT` da Aula 11 protegendo vocês. O banco está dizendo: *"não posso apagar isso, tem coisa dependendo"*.

Checagem do Bloco 2

Você vai corrigir o gênero do jogo de `id_jogo = 7`.

- Escreva, **na ordem**, os dois comandos que você roda
- Quantas linhas o primeiro deve trazer?
- Se ele trouxer 40, o que você faz?



`forms.gle/HburG1NLEmAqSaJP6`

Intervalo

10 minutos

Bloco 3

Python conversando com o banco

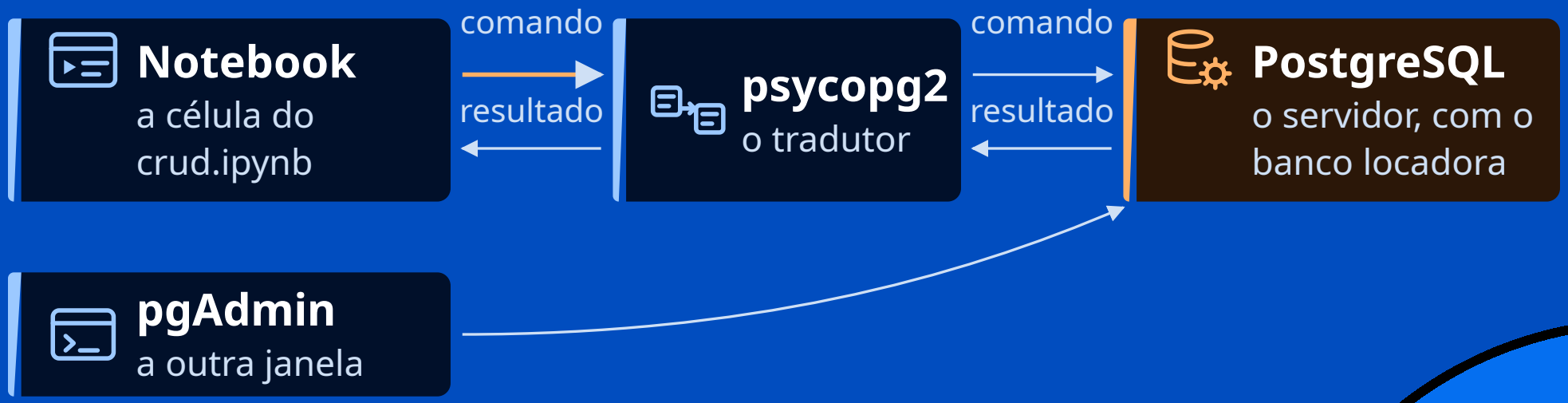
Por que sair do pgAdmin

O pgAdmin é ótimo para **você** trabalhar. Mas o cliente da locadora não vai abrir o pgAdmin para alugar um jogo.

- Um **programa** precisa conversar com o banco
- Esse programa é o que vocês vão construir até a Aula 18 — o **CRUD**
- A linguagem é **Python**, a mesma da UC1

O caminho do comando, e o da resposta

O CAMINHO DE UM COMANDO, E O DA RESPOSTA



Passo a passo — criar o notebook

1. No VS Code, abra a pasta do projeto do grupo
2. Menu **File → New File...** e escolha **Jupyter Notebook**
3. Salve como `crud.ipynb` (**Ctrl+S**)
4. Se o VS Code pedir para escolher um **kernel**, escolha o **Python 3** instalado
5. Clique na primeira célula e escreva `print("oi")`. Rode com o botão ▶

> Apareceu `oi` embaixo da célula? Então o notebook está funcionando. **Nada de terminal** — tudo pelo botão ▶.

Passo a passo — instalar a biblioteca

Numa célula do notebook (não no terminal):

```
%pip install psycopg2-binary
```

O `%` na frente é o que faz o comando rodar **dentro do notebook**, no Python certo. Sem ele, o comando pode instalar em outro lugar e depois "não achar" a biblioteca.

Espere aparecer `Successfully installed psycopg2-binary`.

Passo a passo — conectar

```python
import psycopg2

conexao = psycopg2.connect(
    host="localhost",
    dbname="locadora",
    user="postgres",
    password="SUA_SENHA"
)
cursor = conexao.cursor()
print("conectado!")
```

O que cada palavra da conexão quer dizer

GLOSSÁRIO

`host`	hospedeiro — onde o servidor está (`localhost` = esta máquina)
`dbname`	database name — nome do banco
`cursor`	cursor — executa comandos e lê resultados

Passo a passo — inserir pelo Python

```
cursor.execute(
    "INSERT INTO plataforma (nome, fabricante) VALUES (%s, %s);",
    ("Xbox Series X", "Microsoft")
)
conexao.commit()
print("inserido!")
```

Sem o `commit()`, nada é salvo. O Python abre uma **transação** (Aula 01!) e ela só vira permanente quando você confirma.

Checagem do Bloco 3

O notebook rodou sem erro nenhum, e o dado **não** apareceu no pgAdmin.

- O que faltou no código?
- Escreva a linha que resolve
- Como você confirma, pelo próprio Python, que a inserção pegou?

forms.gle/sqDyxzRQGAFN5Wpx8

Intervalo

10 minutos

MÃO NA MASSA

Popular o banco do projeto

Pelos **dois** caminhos: pgAdmin e notebook.

1. No pgAdmin, insiram 3 a 5 linhas nas tabelas **sem** chave estrangeira
2. Confiram os `id` gerados pelo `SERIAL`
3. Insiram as tabelas que **dependem** dessas, usando os id corretos
4. Por último, a tabela do meio do N:N
5. No notebook, repitam a conexão com o banco do **seu** grupo
6. Façam pelo menos **um** `INSERT` **pelo Python**, com `commit()`
7. Confiram no pgAdmin (com **Refresh**) que a linha do Python apareceu

MÃO NA MASSA

O combinado desta entrega

GLOSSÁRIO

`grupoNN-aulaNN-artefato.<ext>` `grupo03-aula12-inserts.sql`

O que eu confiro: Os dados são **realistas**, do tema do grupo · A ordem de inserção respeitou as **chaves estrangeiras** · Alguém já tentou um `DELETE` **sem** `WHERE`? (Espero que não!) · O notebook conecta e o `INSERT` pelo Python **apareceu** no pgAdmin · O `commit()` está lá

> Terminou? Insira mais dados, variados — jogos de gêneros diferentes, clientes com datas diferentes. A Aula 14 vai fazer perguntas sobre esses dados, e dados parecidos demais dão respostas chatas.

PARA CASA · OPCIONAL

Antes da próxima aula

Testar em casa: abra o notebook no seu computador e confirme que a conexão funciona fora do laboratório. Se a senha for outra, ajuste no código.

Tarefa opcional: insira mais 3 registros no banco do projeto usando o notebook — e confira no pgAdmin que apareceram.

Para treinar: `locadora-exercicios-12-dados.sql`, na página da aula — tem uma bancada própria, para você rodar o `UPDATE` sem `WHERE` e ver o estrago sem estragar nada.

Para ver acontecer: demonstrações da Aula 12 e oito perguntas: o UPDATE sem WHERE mudando tudo, e as aspas de cada valor.

> Guardem o `crud.ipynb`. Ele cresce em todas as aulas até a 18.

Para a próxima aula

Os dados estão lá dentro. **Como perguntar coisas para o banco** — "quais jogos existem?", "quem alugou o quê?"

UniSenac Centro Universitário RS
Senac
Fecomércio
IFEP
Sesc
Sindicatos Empresariais