INSTITUTO INFNET

ESCOLA SUPERIOR DE TECNOLOGIA

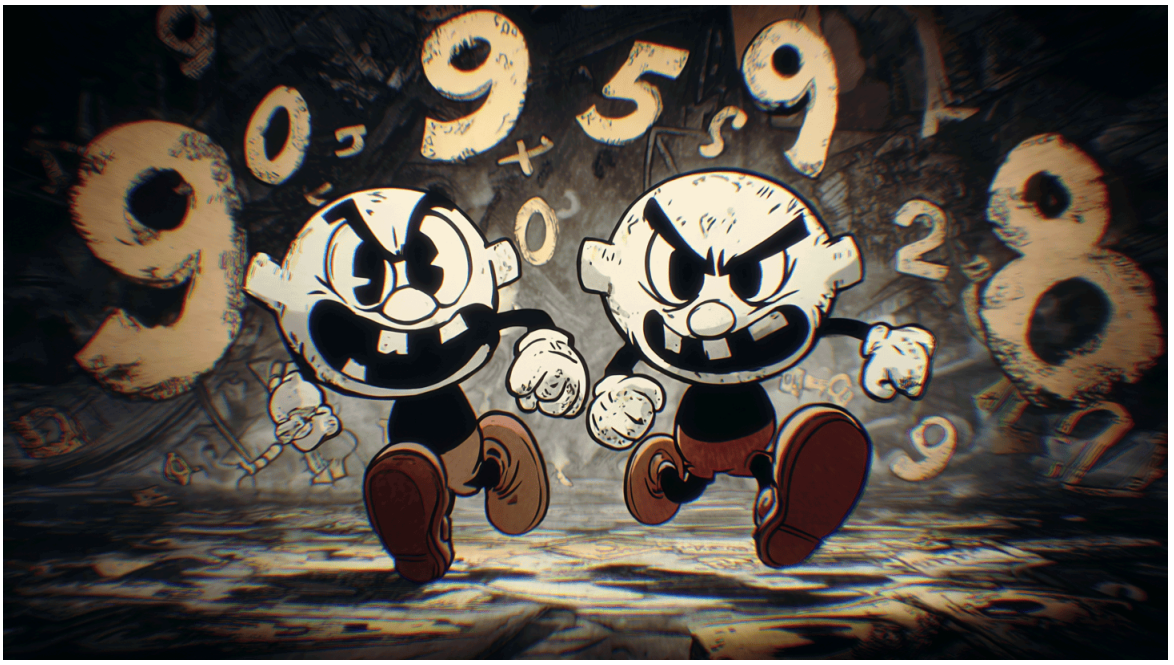
GRADUAÇÃO EM ENGENHARIA DE SOFTWARE

Lista de Exercícios TP3 SQL

Prof: Leonardo Glória

Aluno: Richard de Jesus Cabral Alves.

Tema: Plataforma IndieLearn



Como rodar o ambiente (Docker)

Imagem pronta: `leogloriainfnet/indielearn:1.0`
O banco já vem inicializado com a tabela `usuario_antigo`
e os CSVs estão dentro do container em `/import/`:

- `/import/usuarios.csv`
- `/import/cursos.csv`
- `/import/pagamentos.csv`

1) Subir o container

docker run -d --name indielearn_pg -e POSTGRES_DB=postgres -e POSTGRES_USER=postgres -e POSTGRES_PASSWORD=postgres -p 5432:5432 leogloriainfnet/indielearn:1.0

Entrando no container:

docker exec -it indielearn_pg /bin/bash

⚠️ Atenção

Nesse ponto use:

echo "SEU NOME COMPLETO"

Obs: Substitua seu nome pelo seu nome, rs.
OBs2: Tire print.

Dentro do container executar:

psql -U postgres

Nesse ponto você tem acesso ao banco já populado.

Uma dica: o database criado se chama: ""indielearn""

Ato 0 – Banco herdado (Ajustes Iniciais)

A tabela `usuario_antigo` já existe. Corrija:

Q0.

1. Remova a constraint de unicidade em `email`.
2. Renomeie `nome_completo` para `nome`.
3. Altere `criado_em` de `DATE` para `TIMESTAMPTZ`.

Ato 1 – O Alicerce (Modelagem e DDL)

Q1. Crie a tabela `usuario` com:

- `id` (PK, identidade)
- `nome` (NOT NULL)
- `email` (NOT NULL, UNIQUE)
- `tipo` (NOT NULL, `'ALUNO' | 'INSTRUTOR'`)
- `criado_em` (TIMESTAMPTZ, DEFAULT now())

Q2. Crie a tabela `curso` com:

- `id` (PK, identidade)
- `titulo` (NOT NULL)
- `categoria` (NOT NULL)
- `nivel` (NOT NULL, `'inic' | 'inter' | 'avanc'`)
- `preco` (NUMERIC >= 0)
- `publicado_em` (DATE)
- `instrutor_id` (BIGINT, sem FK)

Q3. Crie a tabela `pagamento` com:

- `id` (PK, identidade)
- `matricula_id` (BIGINT, sem FK)
- `valor_pago` (NUMERIC >= 0)
- `realizado_em` (TIMESTAMPTZ)
- `meio` (`'PIX' | 'CARTAO' | 'BOLETO'`)
- `status` (`'OK' | 'FALHA'`)

Q4. Crie ao menos um índice em cada tabela.

Ato 2 – Inserts manuais

Q5. Insira 5 usuários.
Q6. Insira 5 cursos.
Q7. Insira 8 pagamentos, incluindo pelo menos um com `status='FALHA'`.

Ato 3 – A Enxurrada de Dados (Bulk Load)

Q8. Carregue os arquivos que estão em :

- `/import/usuarios.csv`
- `/import/cursos.csv` →
- `/import/pagamentos.csv`

Atenção:
Não apague registros. Não use o id do CSV.
Descreva qual solução que você utilizou.

Q9. Explique a diferença entre `GENERATED ALWAYS AS IDENTITY` e `GENERATED BY DEFAULT AS IDENTITY` em cenários de importação com id no CSV.

Q10. Valide as contagens com `SELECT COUNT(*)` nas três tabelas.

Ato 4 – Consultas (SQL)

Bloco A

Q11. Liste 20 cursos publicados em **2024**, `nivel='inter'`, `categoria IN ('Data','DBA','DevOps')`, `titulo ILIKE '%sql%'`. Ordene por `publicado_em DESC` com `OFFSET 20`.
Q12. Traga `id, nome, email` de usuários cujo e-mail termina em `.edu.br`, resolvendo uma vez com `LIKE` e outra com `RIGHT(email,7)`.
Q13. Mostre `id, titulo, REPLACE(titulo,'SQL','SQL (2025)') AS titulo_formatado`.
Q14. Pagamentos entre **2025-02-01** e **2025-03-15**.
Q15. Cursos `nivel='avanc'` com `preco BETWEEN 300 AND 800` e `categoria <> 'ML'`.

Bloco B

Q16. Por categoria (apenas cursos de 2025): `categoria, cursos_publicados, preco_medio`.
Q17. Por mês de 2025: `mes, total_pagamentos_ok = SUM(valor_pago)` com `status='OK'`.
Q18. Receita mensal com `status='OK'` com `HAVING SUM(valor_pago) >= 10000`.
Q19. Compare `COUNT(*)` vs `COUNT(meio)` em `pagamento` e explique a diferença (NULLs).
Q20. Por `nivel`: `qtd_cursos, preco_min, preco_max, preco_avg` ordenado por `preco_avg DESC`.

Bloco C

Q21. Cursos com `preco` acima da **média global**.
Q22. `NOT EXISTS`: usuários `tipo='ALUNO'` que **não** aparecem como `instrutor_id` em `curso`.

Bloco E

Q25. Para cada **instrutor**, `STRING_AGG` dos títulos dos cursos (ordenados por `publicado_em`), somente quem tem `COUNT(*) >= 3`.
Q26. Extraia `dominio_email` (parte após `@`) e conte usuários por domínio, ordenando desc.

Bloco F — Qualidade de dados

Q27. Pagamentos `status='OK'` com `valor_pago <= 0`.
Q29. E-mails duplicados em `usuario` (`GROUP BY email HAVING COUNT(*)>1`).