AVALIAÇÃO - PROGRAMAÇÃO PARA BANCO DE DADOS

Objetivo

Você foi contratado por uma startup de tecnologia educacional que está desenvolvendo uma **plataforma online de cursos**. Sua missão é **criar e manipular a base de dados** responsável por armazenar as informações de **alunos, cursos e matrículas**.

Essa avaliação prática tem como objetivo aplicar os conhecimentos de SQL em um cenário real de modelagem, manipulação e consulta de dados relacionais.

📌 Instruções

- Utilize **comandos SQL** organizados em **comentários por etapa** (`-- Etapa 1`, `-- Etapa 2`, etc.).
- Ao final, envie um único arquivo `.sql` com todos os comandos.
- Utilize o padrão `SNACK_CASE` em todos os nomes de tabelas, colunas e constraints.
- Nomeie chaves primárias, estrangeiras, unique e check constraints conforme padrão visto em aula.

✅ Etapa 1 – Criação das Tabelas (3 pontos)

Crie as três tabelas abaixo com seus respectivos campos, tipos e regras.

📋 Tabela: ALUNO

Coluna	Tipo	Regra
ALUNO_ID	`INT`	Chave Primária, auto incrementável
NOME	`VARCHAR(255)`	**NOT NULL**
EMAIL	`VARCHAR(150)`	**NOT NULL**, UNIQUE
DATA_NASCIMENTO	`DATE`	**NOT NULL**

Constraints:

- Chave Primária: `PK_ALUNO`
- Unique: `UN_ALUNO_EMAIL`

📋 Tabela: CURSO

Coluna	Tipo	Regra
CURSO_ID	`INT`	Chave Primária, auto incrementável
NOME	`VARCHAR(100)`	**NOT NULL**, UNIQUE
CARGA_HORARIA	`INT`	**NOT NULL**, maior que 0
NIVEL	`VARCHAR(20)`	**NOT NULL**, valores: "BÁSICO", "INTERMEDIÁRIO", "AVANÇADO"

Constraints:

- Chave Primária: `PK_CURSO`
- Check: `CK_CURSO_CARGA_HORARIA`
- Check: `CK_CURSO_NIVEL`
- Check: `CK_CURSO_NOME`

📋 Tabela: MATRICULA

Coluna	Tipo	Regra
MATRICULA_ID	`INT`	Chave Primária, auto incrementável
ALUNO_ID	`INT`	Foreign Key para `ALUNO`, **NOT NULL**
CURSO_ID	`INT`	Foreign Key para `CURSO`, **NOT NULL**
DATA_MATRICULA	`DATETIME`	**NOT NULL**

Constraints:

- Chave Primária: `PK_MATRICULA`
- Foreign Key: `FK_MATRICULA_ALUNO`
- Foreign Key: `FK_MATRICULA_CURSO`

✍️ Etapa 2 – Inserção de Dados (2 pontos)

Crie comandos `INSERT INTO` com os seguintes dados mínimos:

- Pelo menos **5 alunos**
- Pelo menos **4 cursos**

- Pelo menos **8 matrículas** (alunos diferentes em cursos variados)

🔄 Etapa 3 – Atualizações (1 ponto)

Crie os seguintes comandos `UPDATE`:

1. Altere o **nome e o e-mail de um aluno** usando o ALUNO_ID como referência.
2. Atualize o **nível** de todos os cursos do tipo **"BÁSICO"** para **"INTERMEDIÁRIO"**.
3. Altere a **carga horária** de um curso específico usando o CURSO_ID como referência.

❌ Etapa 4 – Remoções (1 ponto)

Crie os seguintes comandos `DELETE`:

1. Remova uma matrícula específica usando o ALUNO_ID e o CURSO_ID como referência.
2. Remova um curso específico. Antes delete todas as matrículas associadas a esse curso (em ordem correta para evitar erro de integridade).

🔍 Etapa 5 – Consultas (3 pontos)

Crie consultas SQL (`SELECT`) para os seguintes cenários:

1. Liste o **nome e o e-mail de todos os alunos**.
2. Liste o **nome dos cursos e suas respectivas cargas horárias**.
3. Liste os **cursos com carga horária superior a 40h**.
4. Liste os **alunos e os cursos nos quais estão matriculados**.
5. Liste os **alunos que ainda não estão matriculados em nenhum curso**. Dica: Para filtrar valores nulos, use `IS NULL` (e não `= NULL`). Ex.: `WHERE TABELA.COLUNA IS NULL`.

📤 Entrega

- Envie um único arquivo `.sql` com todos os comandos.
- Organize por etapas com comentários no próprio código.