**Nome: Marcos Paulo de Souza**

O código está relacionado à criação de quatro tabelas em um banco de dados, tipicamente usado para gerenciar usuários, certificados e questões de avaliação.

CREATE TABLE tbUsuario (

id\_usuario INT NOT NULL,

nome VARCHAR(255) NULL,

email VARCHAR(255) NULL,

PRIMARY KEY(id\_usuario)

);

CREATE TABLE tbCertificado (

id\_usuario INT NOT NULL,

data\_avaliacao TIMESTAMP NULL,

score DECIMAL(5,2) NULL,

PRIMARY KEY(id\_usuario),

CONSTRAINT fk\_usuario

FOREIGN KEY(id\_usuario)

REFERENCES tbUsuario(id\_usuario)

);

CREATE TABLE tbQuestao (

id\_questao INT NOT NULL,

enunciado TEXT NULL,

resposta\_correta BOOL NULL,

PRIMARY KEY(id\_questao)

);

CREATE TABLE tbCertificadoQuestao (

id\_usuario INT NOT NULL,

id\_questao INT NOT NULL,

resposta\_usuario BOOL NULL,

PRIMARY KEY(id\_usuario, id\_questao),

CONSTRAINT fk\_certificado

FOREIGN KEY(id\_usuario)

REFERENCES tbCertificado(id\_usuario),

CONSTRAINT fk\_questao

FOREIGN KEY(id\_questao)

REFERENCES tbQuestao(id\_questao)

);

Interface gráfica do usuário, Aplicativo

Descrição gerada automaticamente

Interface gráfica do usuário, Texto, Aplicativo

Descrição gerada automaticamente

**pgAdmin 4**

Uma imagem contendo Gráfico

Descrição gerada automaticamente

Texto

Descrição gerada automaticamente

**Chave primaria = Essa chave é utilizada como identificador único da tabela**

* **tbUsuario**: **Esta é a tabela que armazena informações sobre os usuários.**
  + **id\_usuario**: Uma coluna do tipo inteiro que não pode ser nula (**NOT NULL**). Ela armazena um identificador único para cada usuário. Definida como chave primária (**PRIMARY KEY**) da tabela, o que significa que cada valor nesta coluna deve ser único e não pode ser nulo.
  + **nome**: Uma coluna do tipo **VARCHAR(255)** que pode armazenar strings com até 255 caracteres. É permitido que essa coluna seja nula (**NULL**).
  + **email**: Semelhante à coluna **nome**, armazena strings de até 255 caracteres e também pode ser nula.

Tabela ‘tbCertificado’

Texto

Descrição gerada automaticamente

* **tbCertificado**: Esta tabela guarda informações sobre os certificados dos usuários.
  + **id\_usuario**: Coluna do tipo inteiro, chave estrangeira (**FOREIGN KEY**) que referencia **id\_usuario** da tabela **tbUsuario**. Essa configuração garante que cada **id\_usuario** nesta tabela deve existir na tabela **tbUsuario**. O **id\_usuario** também é definido como chave primária (**PRIMARY KEY**) **aqui, implicando que cada usuário tem no máximo um certificado.**
  + **data\_avaliacao**: Coluna do tipo **TIMESTAMP** que pode ser nula e é usada para armazenar a data e a hora da avaliação do usuário.
  + **score**: Coluna do tipo **DECIMAL(5,2)**, que pode ser nula. Isso significa que pode armazenar números com até 5 dígitos no total, dos quais 2 dígitos podem ser após a vírgula decimal, para registrar a pontuação do usuário na avaliação.

**Obs:** 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.

A finalidade da chave estrangeira é garantir a integridade dos dados referenciais, pois apenas serão permitidos valores que supostamente vão aparecer na base de dados.

Tabela ‘tbQuestao’

Uma imagem contendo Padrão do plano de fundo

Descrição gerada automaticamente

* **tbQuestao**: **Armazena questões usadas nas avaliações.**
  + **id\_questao**: Um inteiro que serve como um identificador único para cada questão, não pode ser nulo e é a chave primária da tabela.
  + **enunciado**: Coluna do tipo **TEXT** que pode ser nula, usada para armazenar o enunciado da questão.
  + **resposta\_correta**: Coluna do tipo **BOOL** (booleano) que pode ser nula, usada para indicar a **resposta correta da questão (verdadeiro ou falso).**

Tabela ‘tbCertificadoQuestão’

Tela de computador com texto preto sobre fundo branco

Descrição gerada automaticamente

* **tbCertificadoQuestao**: **Tabela de relação entre usuários e questões, armazenando as respostas dos usuários.**
  + **id\_usuario** e **id\_questao**: Estas colunas formam uma **chave primária composta** (**PRIMARY KEY**), significando que a combinação de **id\_usuario** e **id\_questao** deve ser única. Ambas são chaves estrangeiras que referenciam **id\_usuario** da tabela **tbCertificado** e **id\_questao** da tabela **tbQuestao**, respectivamente.
  + **resposta\_usuario**: Coluna do tipo **BOOL** que pode ser nula, usada para registrar a resposta do usuário para uma questão específica.

Este conjunto de tabelas permite que o sistema gerencie informações de usuários, suas avaliações, questões de avaliações e as respostas dadas por cada usuário a cada questão, tudo isso mantendo a integridade dos dados por meio de chaves primárias e estrangeiras.

**1. Relacionamento entre tbUsuario e tbCertificado**

* **Relacionamento**: Um-para-um
* **Direção**: De **tbUsuario** para **tbCertificado**
* **Implementação**: **id\_usuario** em **tbCertificado** é uma chave estrangeira que referencia **id\_usuario** em **tbUsuario**.

**Descrição**: Cada linha na tabela **tbUsuario** está associada a no máximo uma linha na tabela **tbCertificado** através do **id\_usuario**. Isso é garantido pela configuração de **id\_usuario** como chave primária em **tbCertificado**. Esse relacionamento é útil para **vincular cada usuário a um único certificado que ele possa obter após uma avaliação.**

**2. Relacionamento entre tbCertificado e tbCertificadoQuestao**

* **Relacionamento**: Um-para-muitos
* **Direção**: De **tbCertificado** para **tbCertificadoQuestao**
* **Implementação**: **id\_usuario** em **tbCertificadoQuestao** é uma chave estrangeira que referencia **id\_usuario** em **tbCertificado**.

**Descrição**: Cada certificado em **tbCertificado** pode estar relacionado a múltiplas entradas em **tbCertificadoQuestao**. Isso permite registrar as **respostas do usuário** para várias questões de uma avaliação **associada a um certificado**. Note que este design implica que, para cada entrada em **tbCertificadoQuestao**, deve existir uma entrada correspondente em **tbCertificado**.

**3. Relacionamento entre tbQuestao e tbCertificadoQuestao**

* **Relacionamento**: Um-para-muitos
* **Direção**: De **tbQuestao** para **tbCertificadoQuestao**
* **Implementação**: **id\_questao** em **tbCertificadoQuestao** é uma chave estrangeira que referencia **id\_questao** em **tbQuestao**.

**Descrição**: Este relacionamento permite que **uma questão** em **tbQuestao** seja **associada a várias respostas** em **tbCertificadoQuestao**, correspondendo a **diferentes usuários que respondem à mesma questão** em suas respectivas avaliações. Este design facilita a análise de respostas a uma determinada questão por múltiplos usuários, permitindo avaliar a eficácia ou dificuldade da questão.

**Comentários gerais sobre o design**

O design das tabelas e seus relacionamentos é robusto para um sistema de gerenciamento de avaliações, com a capacidade de registrar usuários, suas avaliações (e detalhes associados como pontuação e datas), questões das avaliações e as respostas dadas. O uso de chaves estrangeiras não apenas impõe a integridade referencial, garantindo que não existam registros órfãos, mas também facilita junções eficientes entre as tabelas para consultas relacionadas ao desempenho dos usuários nas avaliações, análise das respostas, entre outros.

Este design de base de dados suporta bem as operações típicas em um ambiente educacional ou de certificação, como inscrever usuários, registrar suas avaliações e respostas, e gerar relatórios de desempenho baseados em dados concretos.

**Constraints –** Constraints são objetos usados com a finalidade de estabelecer regras referentes à integridade e à consistência nas colunas das tabelas pertencentes a um sistema de banco de dados. Isso é importante porque para planejar e criar tabelas devemos garantir a integridade dos dados presentes nas colunas e identificar os valores válidos para tais dados.

Com o objetivo de assegurar a integridade dos dados de uma tabela, o SQL Server nos oferece cinco tipos diferentes de constraints, os quais estão relacionados a cinco tipos de integridades. Veja na tabela abaixo os cinco tipos de constraints e os respectivos tipos de integridade:

Tabela

Descrição gerada automaticamente

**2. Constraints**

Constraints são restrições criadas no banco de dados para:

* Impedir que dados inválidos sejam cadastrados no banco de dados
* Garantir a qualidade dos dados
* Garantir que campos obrigatórios sejam preenchidos
* Garantir integridade referencial definida no modelo relacional
* Em geral fazem parte da especificação do projeto conceitual do banco de dados e estão descritas no modelo relacional

**A adição de constraints em uma tabela pode ser feita de duas maneiras:**

1. Como parte do CREATE TABLE ao criar a tabela
2. Como parte do ALTER TABLE quando se adiciona as constraints  
   depois de ter criado a tabela (é o mais comum)

Interface gráfica do usuário, Aplicativo

Descrição gerada automaticamente com confiança média

**Tipos de Constraints**

Diagrama, Texto

Descrição gerada automaticamente

**CONSTRAINT NOT NULL**

Garante que a coluna não aceite valores nulos

Texto

Descrição gerada automaticamente

**CONSTRAINT UNIQUE**

Garante que não existe valores não nulos repetidos

Texto

Descrição gerada automaticamente com confiança baixa

**CONSTRAINT PRIMARY KEY**

Identifica unicamente cada registro. Não pode ter valores nulos nem repetidos

Texto

Descrição gerada automaticamente

**CONSTRAINT FOREIGN KEY**

Estabelece o relacionamento entre duas tabelas

* É sempre criada no lado “MUITO” do relacionamento
* É criada na coluna que é chave estrangeira
* Deve sempre referenciar uma coluna em uma tabela que obrigatoriamente é UNIQUE ou PRIMARY KEY

Texto

Descrição gerada automaticamente

**Constraint FOREIGN KEY:**

Palavras-chave

* FOREIGN KEY: Define a coluna da tabela filha no nível de constraint da tabela
* REFERENCES: Identifica a tabela e a coluna da tabela mãe
* ON DELETE CASCADE: Deleta as linhas dependentes da tabela filha quando uma linha da tabela mãe é deletada
* ON DELETE SET NULL: Converte os valores da chave estrangeira dependente em nulos