escola britânica de artes criativas & tecnologia

Desenvolvedor Full Stack Python

Relacionamentos entre Tabelas

escola britânica de artes criativa & tecnologia

AGENDA

- Chaves Primárias e Chaves Compostas entre Tabelas
- Como criar relacionamento entre tabelas
- Como unir dados de diferentes tabelas utilizando o SQL Join
- Filtrando dados com inner join

Chaves Primárias e Chaves Compostas

FK ou **Chave Estrangeira** nada mais é do que a **Chave Primária** de uma tabela 'colocada' em outra tabela.

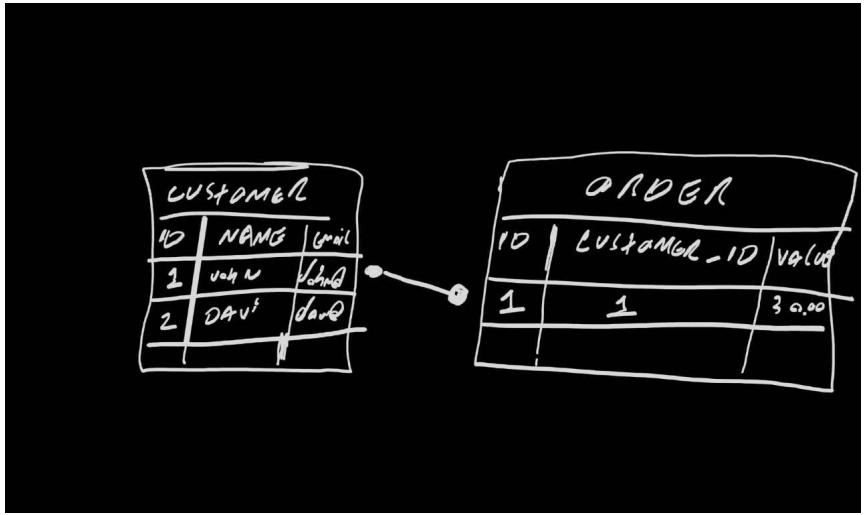
A Chave Estrangeira, além de conectar tabelas, tem mais esses propósitos:

- Ela impede que você adicione um valor inválido no ID de uma tabela
- Ela impede que você exclua um registro caso ele faça referência em outra tabela

Chaves Primárias e Chaves Compostas

PK ou Chave Primária é usada quando precisamos dos seguintes objetivos em uma tabela:

- Ter unicidade de um registro
- Que esse registro NÃO seja nulo



Criando Tabelas com Chaves Primárias

```
CREATE TABLE order (

        id INTEGER PRIMARY KEY,
        value INTEGER,
);
```

Adicionando Chaves Primárias em Tabelas Existentes

```
ALTER TABLE table_name ADD PRIMARY KEY (column_1, column_2);

ALTER TABLE product ADD PRIMARY KEY (product_no);
```

Criando Tabelas com Chaves Estrangeiras

```
CREATE TABLE customer (

        id INTEGER PRIMARY KEY,
        value INTEGER,
);
```

```
CREATE TABLE contact (

        id INTEGER PRIMARY KEY,
        customer_id INTEGER REFERENCES customer(id)
);
```

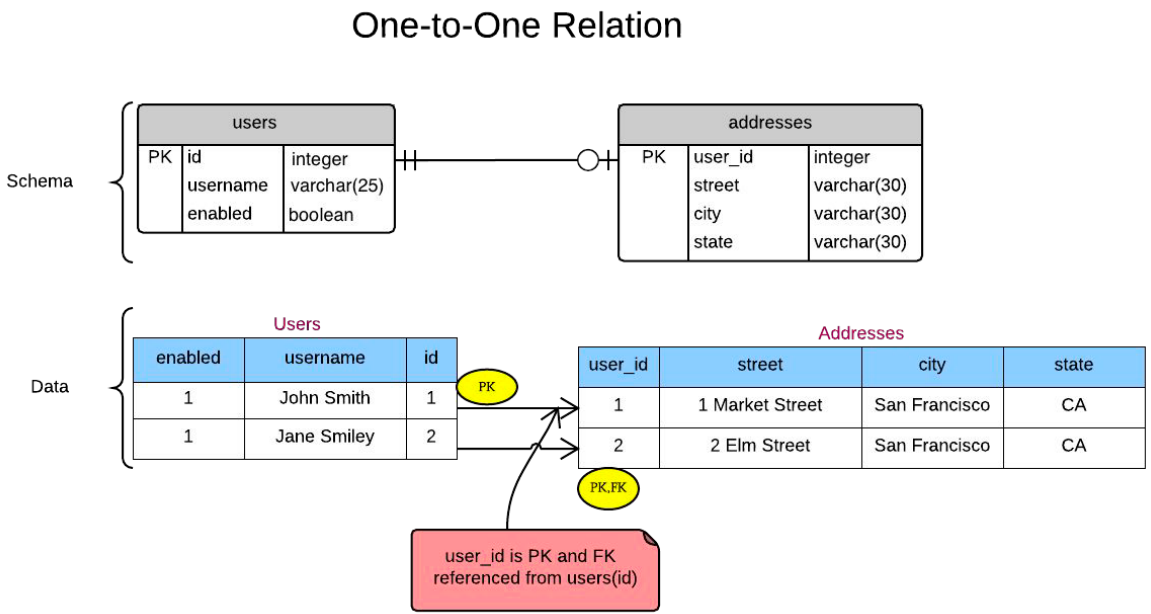
Adicionando Relacionamento (FK) entre tabelas

ALTER TABLE table_name ADD **CONSTRAINT** <nome-da-constraint> **FOREIGN KEY** (column_1) **REFERENCES** customer (ID);

ALTER TABLE customer_order ADD **CONSTRAINT** fk_order_customer **FOREIGN KEY** (id) REFERENCES customer (email);

escola britânica de artes criativas & tecnologia

Tipos de Relacionamento entre tabelas - 1:1 um para um



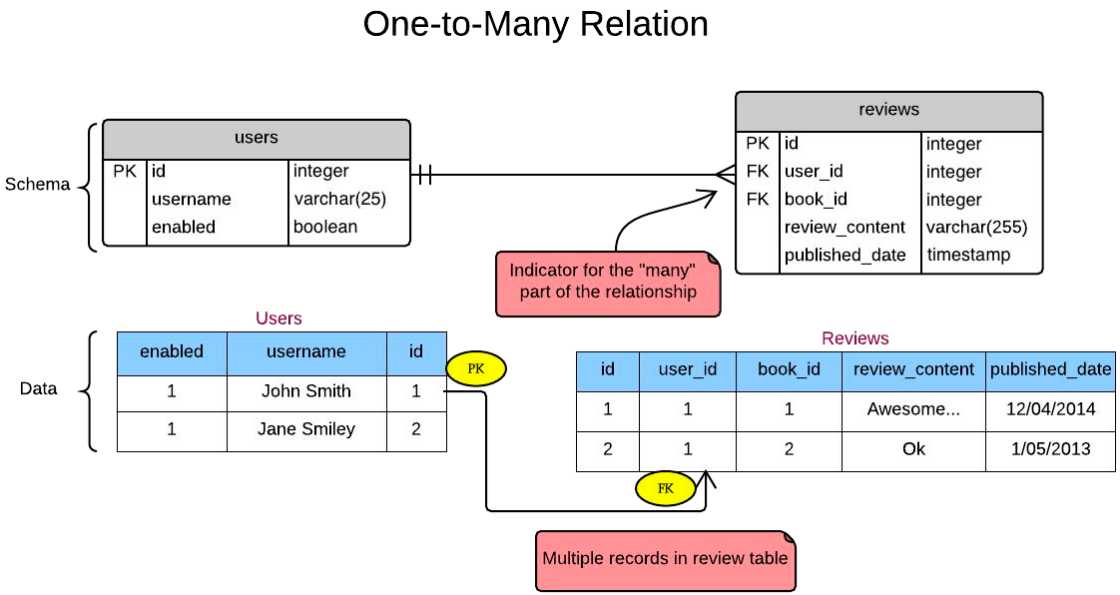
Tipos de Relacionamento entre tabelas - 1:1 um para um

```
CREATE TABLE users (
        id serial,
        username VARCHAR(25) NOT NULL,
        enabled boolean DEFAULT TRUE,
        last_login timestamp NOT NULL
DEFAULT NOW(),
        PRIMARY KEY (id)
);
```

```
CREATE TABLE addresses (
        user_id int NOT NULL,
        street VARCHAR(30) NOT NULL,
        city VARCHAR(30) NOT NULL,
        state VARCHAR(30) NOT NULL,
        PRIMARY KEY (user_id),
        CONSTRAINT fk_user_id FOREIGN KEY
(user_id) REFERENCES users (id)
);
```

escola britânica de artes criativas & tecnologia

Tipos de Relacionamento entre tabelas - 1:N ou 1 para muitos



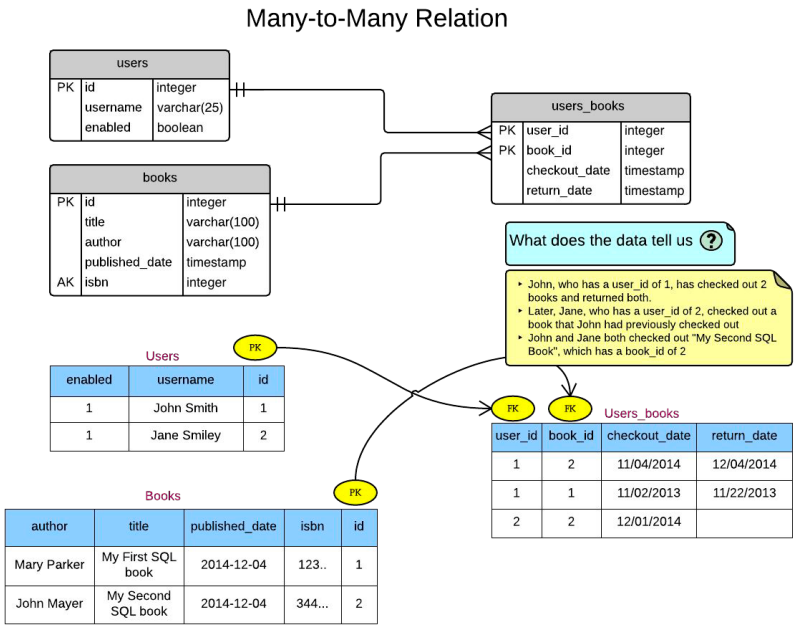
Tipos de Relacionamento entre tabelas - 1:N ou 1 para muitos

```
CREATE TABLE books (
        id serial,
        title VARCHAR(100) NOT NULL,
        author VARCHAR(100) NOT NULL,
        published_date timestamp NOT NULL,
        isbn int,
        PRIMARY KEY (id),
        UNIQUE (isbn)
);
```

```
DROP TABLE IF EXISTS reviews;
CREATE TABLE reviews (
        id serial,
        book_id int NOT NULL,
        user_id int NOT NULL,
        review_content VARCHAR(255),
        rating int,
        published_date timestamp DEFAULT
CURRENT_TIMESTAMP,
        PRIMARY KEY (id),
        FOREIGN KEY (book_id) REFERENCES
books(id) ON DELETE CASCADE,
        FOREIGN KEY (user_id) REFERENCES
users(id) ON DELETE CASCADE
);
```

escola britânica de artes criativas & tecnologia

Tipos de Relacionamento entre tabelas - 1:N ou 1 para muitos



Tipos de Relacionamento entre tabelas - 1:N ou 1 para muitos

```
CREATE TABLE users_books (
        user_id int NOT NULL,
        book_id int NOT NULL,
        checkout_date timestamp,
        return_date timestamp,
        PRIMARY KEY (user_id, book_id),
        FOREIGN KEY (user_id) REFERENCES users(id) ON UPDATE CASCADE,
        FOREIGN KEY (book_id) REFERENCES books(id) ON UPDATE CASCADE
);
```

escola britânica de artes criativas & tecnologia

Como unir dados de diferentes tabelas utilizando o SQL Join

Usamos o comando **Joins** para combinar dados de 2 tabelas

Normalmente usamos como referencia as chaves primárias e secundárias:

```
SELECT
        a,
        fruit_a,
        b,
        fruit_b

FROM
        basket_a
INNER JOIN basket_b
        ON fruit_a = fruit_b;
```

	a integer	fruit_a character varying (100)	b integer	fruit_b character varying (100)
1	1	Apple	2	Apple
2	2	Orange	1	Orange

escola britânica de artes criativas & tecnologia

Como unir dados de diferentes tabelas utilizando o SQL Left Join

Também usamos o **Left Join** para combinar dados de 2 tabelas porém invertemos a tabela de referência:

```
SELECT
        a,
        fruit_a,
        b,
        fruit_b
FROM
        basket_a
LEFT JOIN basket_b
        ON fruit_a = fruit_b;
```

	a integer	fruit_a character varying (100)	b integer	fruit_b character varying (100)
1	1	Apple	2	Apple
2	2	Orange	1	Orange
3	3	Banana	[null]	[null]
4	4	Cucumber	[null]	[null]

Resumo

Aprendemos um pouco como **relacionamentos** entre tabelas funciona e como utilizar JOIN para combinar dados.

Exercício

Nesse exercício, vamos criar a tabela de produto e estoque adicionando um **relacionamento** entre elas.

Você pode compartilhar os arquivos de criação na plataforma da EBAC.