**Faculdade de Tecnologia de Franca "Dr. Thomaz Novelino"**
**Curso Tecnológico Superior em Análise e Desenvolvimento de Sistemas**

# (ELETIVA II) PROGRAMAÇÃO DE *SCRIPTS* – 2023/1

Prof. Me. Fausto Gonçalves Cintra – professor@faustocintra.com.br

**LEIA COM ATENÇÃO *TODAS* AS INSTRUÇÕES *ANTES* DE COMEÇAR A FAZER O TRABALHO**

## TRABALHO 1 (T1)

## 1 INSTRUÇÕES GERAIS

1. O trabalho é ***estritamente individual***.
2. A trabalhos idênticos, ou com alto grau de semelhança, será atribuída a nota ZERO.
3. O valor do trabalho é 10,0 (dez), conforme explicado no documento *[ILP502-00] Apresentação*.

## 2 INSTRUÇÕES ESPECÍFICAS

O aluno deverá completar o CRUD das demais tabelas do Diagrama Entidade-Relacionamento (DE-R) atualizado do projeto que está sendo desenvolvido em sala de aula, observando as diretrizes a seguir.

Siga a tabela a seguir para nomear as diferentes entidades que aparecem no código.

| **Nome dos arquivos de *model* e controller (snake_case, singular)** | **Nome do *model* (PascalCase)** | **Nome da tabela e do *route* (snake_case, plural)** | **Constante em `app.js` (camelCase, plural)** |
|---|---|---|---|
| carrier | Carrier | carriers | carriers |
| channel | Channel | channels | channels |
| city | City | cities | cities |
| customer | Customer | customers | customers |
| customer_tag | CustomerTag | customer_tags | customerTags |
| order | Order | orders | orders |
| order_rel_status | OrderRelStatus | order_rel_statuses | orderRelStatuses |
| order_status | OrderStatus | order_statuses | orderStatuses |
| order_tag | OrderTag | order_tags | orderTags |
| payment_method | PaymentMethod | payment_methods | paymentMethods |
| shipment_priority | ShipmentPriority | shipment_priorities | shipmentPriorities |
| tag | Tag | tags | tags |
| user | User | users | users |

1. Gere a *migration* e o *model* para a tabela no terminal com o comando

   `npx sequelize-cli npx sequelize-cli model:generate --name` **`NomeDoModel`** `--attributes` **`<lista de atributos>`**

   Note que:

   a) o nome do *model* deve ser fornecido em `PascalCase`;

   b) na lista de atributos, ***não coloque*** o `id`.

2. Na *migration*, efetue os seguintes alterações:

   a) nas funções up() e down(), modifique o nome da tabela para `snake_case`, plural;

   b) ajuste os tipos de dados, conforme informações do diagrama atualizado. **Campos que aparecem com * no diagrama devem ter a propriedade `allowNull: false`.**

3. Execute a *migration* no terminal com o comando

   `npx sequelize-cli db:migrate`

4. Copie as definições dos campos (***exceto*** dos campos `createdAt` e `updatedAt`) da *migration* e cole no arquivo de *model*, substituindo as definições ali existentes.
5. No arquivo do *model*:
   a) substitua (`Ctrl+H`) todas as ocorrências do termo `Sequelize` por `DataTypes`, ***respeitando a distinção entre letras maiúsculas e minúsculas***;
   b) abaixo do da propriedade `modelName`, adicione a propriedade `tableName` com o nome da tabela entre aspas e em `snake_case`, plural;
   c) se o nome do *model* for composto por mais de uma palavra, renomeie o arquivo do *model* seguindo a convenção `snake_case`.
6. Duplique um *controller* existente e renomeie o arquivo com o mesmo nome do arquivo de *model* que lhe será correspondente.
7. No arquivo do *controller*, substitua (`Ctrl+H`) todas as ocorrências do nome do *model* (que está em `PascalCase`), ***respeitando a distinção entre letras maiúsculas e minúsculas***.
8. Duplique um arquivo de *route* existente, e o renomeie com o nome do *model* correspondente em `snake_case`, no plural.
9. No arquivo de *route*, substitua o caminho do *controller*.
10. No arquivo `app.js`, acrescente a rota correspondente. Observe que:
    a) caso o nome do *route* seja composto por mais de uma palavra, a constante deve ser declarada usando a convenção `camelCase`;
    b) a mesma constante declarada no passo anterior corresponde ao segundo parâmetro da função `app.use()`.
11. Efetue os testes de CRUD com chamadas de API no ThunderClient.
12. Caso você detecte erros na estrutura das tabelas, desfaça a *migration* com o comando

    ```
    npx sequelize-cli db:migrate:undo
    ```

    Corrija a *migration* e depois execute-a novamente com o comando

    ```
    npx sequelize-cli db:migrate
    ```

13. Registre seu progresso, comitando seu trabalho no seu repositório à medida que ele for sendo feito.

## 3 ENTREGA

1. Após concluir o CRUD de todas as tabelas, vá ao seu repositório no GitHub, clique sobre o botão **Pull request** e depois em **New pull request**. Na área de texto, ***escreva seu nome completo*** e termine a criação do *pull request*. Como o seu repositório é um *fork* do repositório do professor, este será notificado de que você pretende enviar alterações para o repositório de origem, podendo ver as adições e modificações de código que você fez.
2. A ***data final*** para fazer o pull request é o dia ***26 de março*** (domingo). Trabalhos enviados com até 24h de atraso terão 25% de desconto na nota; até 48h horas de atraso, 50% de desconto; até 72h, 75% de desconto. Após 72 horas, não serão aceitos *pull requests*.