Sumário

[Entregáveis 2](#_Toc527053014)

[Situação Problema 2](#_Toc527053015)

[Nome do projeto e configurações 5](#_Toc527053016)

[Alterando o Logo 8](#_Toc527053017)

[Alterando o estilo da aplicação 9](#_Toc527053018)

[Modelos 9](#_Toc527053019)

[DAO 12](#_Toc527053020)

[JPA 13](#_Toc527053021)

[Sessão 15](#_Toc527053022)

[Interceptor 16](#_Toc527053023)

[Controllers 17](#_Toc527053024)

[Importando os dados 22](#_Toc527053025)

[MER 25](#_Toc527053026)

[DUMP 31](#_Toc527053027)

# Entregáveis

MER

Dump do bd

Diagrama de casos de uso

Módulo para login e controle de itens de computadores produzidos

# Situação Problema

*Nome do sistema*

NotOnControl

Primeira tela do sistema:

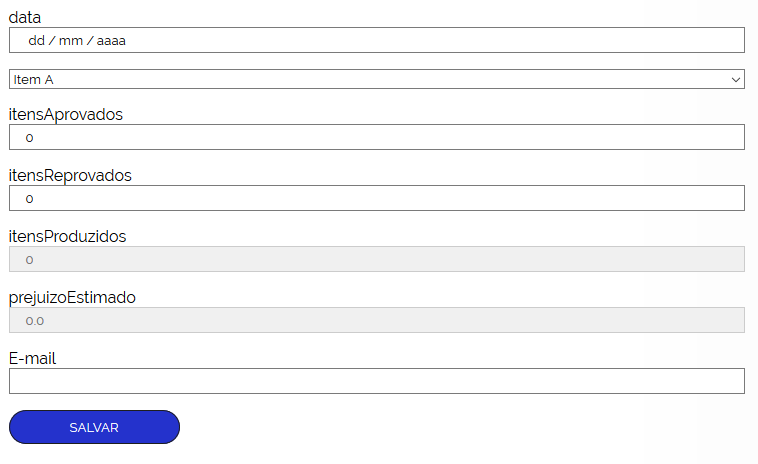
*Controle de acesso*

. somente para usuários cadastrados no banco (script – usuários.txt)

. se o email (login) e a senha não coincidirem com os dados do banco, deve-se mostrar uma mensagem de erro: ‘senha incorreta’

. caso o e-mail não exista na base de dados, mostrar uma mensagem de confirmação, para que o usuário decida se quer ou não criar uma nova conta

*Módulo de produção*



. campos na mesma ordem da imagem

. peças aprovadas, peças reprovadas e prejuízo estimado ficam desabilitados até que uma data seja inserida e uma peça seja escolhida

. peças produzidas irá aparecer automaticamente após a inserção de itens aprovados e reprovados

. prejuízo estimado – quantidade de itens reprovados \* prejuízo\_estimado\_por\_peca (da tabela de itens)

. quantidade de itens reprovados não pode ser maior do que a quantidade de itens aprovados

. campo de e-mail só permite edição se a quantidade de itens reprovados for maior do que 20

. o e-mail deverá ter 4 letras antes do @ e um . após o @

. o cadastro só será realizado caso os campos sejam preenchidos corretamente

. salvar – uma mensagem deve ser exibida

. o logo deve ser o fornecido na pasta

. fonte – Raleway 18pt para títulos, Raleway 10pt para os demais textos

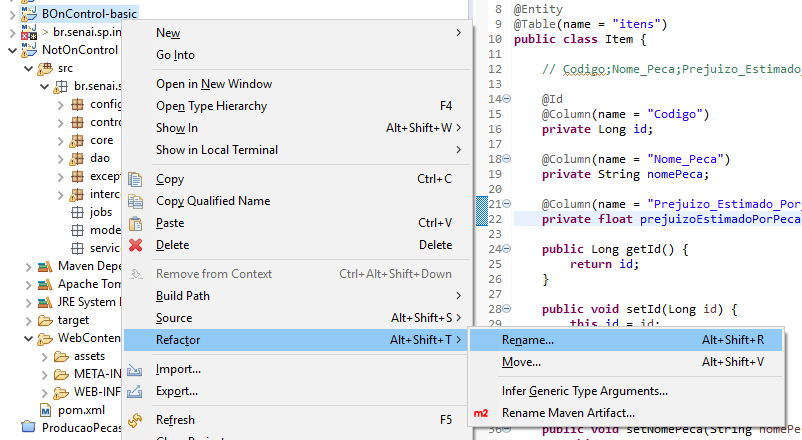
*Começando o treinamento*

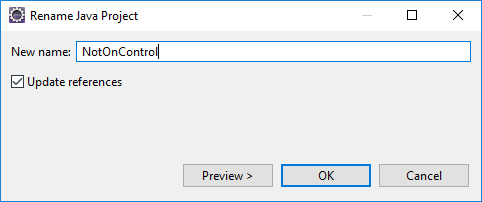
Apache Tomcat 8.5 já deve estar configurado

Importar o projeto senai-boncontrol-basic para o eclipse como maven project

# Nome do projeto e configurações

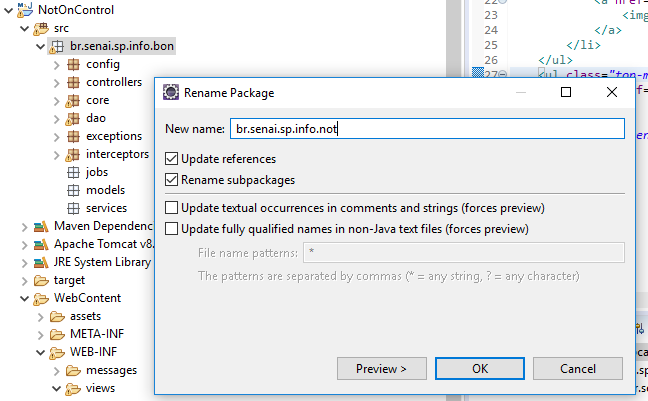
Alterando o nome do projeto – o projeto define que o nome do projeto deve ser NotOnControl



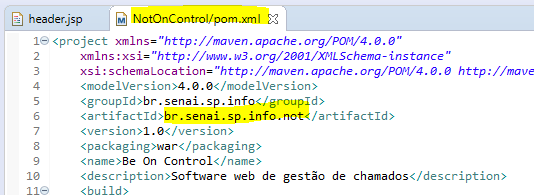


Ok

Renomear o nome dos pacotes

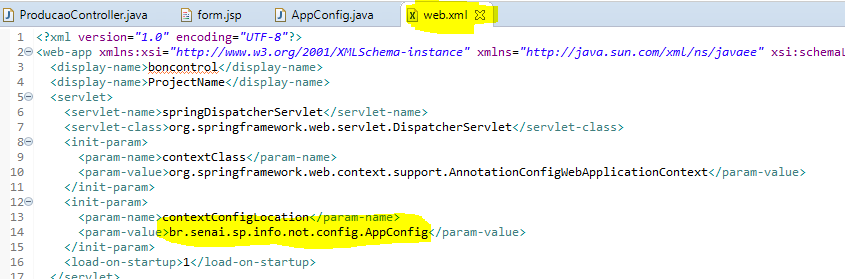


Ok

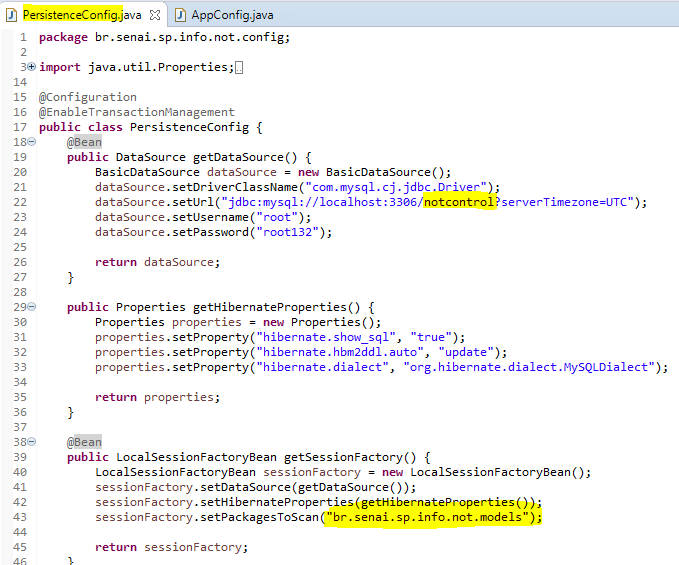


pom.xml -> pacote

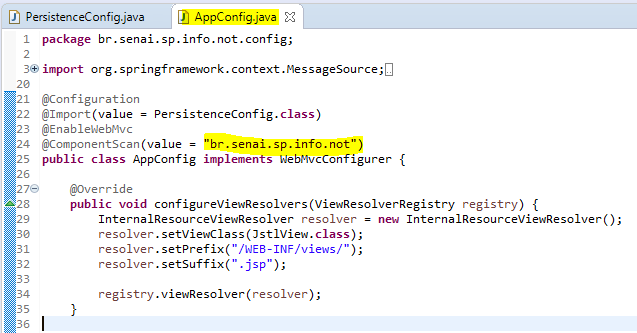
web.xml



PersistenceConfig.java -> o que for necessário dentro do seu projeto

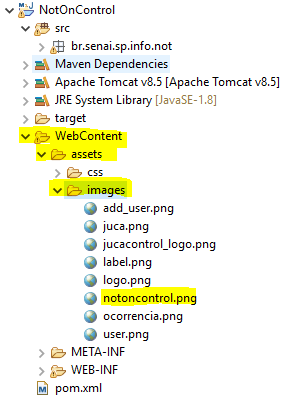


AppConfig.java



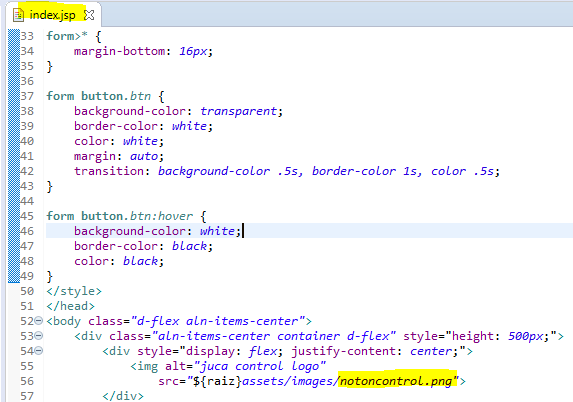
# Alterando o Logo

Alterar o logo da aplicação para o logo fornecido.



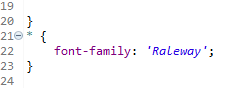
Copiar e colar o correspondente logo dentro da pasta WebContent -> assets -> images -> notoncontrol.png.

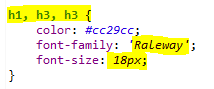
Abrir o index.jsp e alterar o logo atual da aplicação para o logo disponibilizado na pasta.



# Alterando o estilo da aplicação

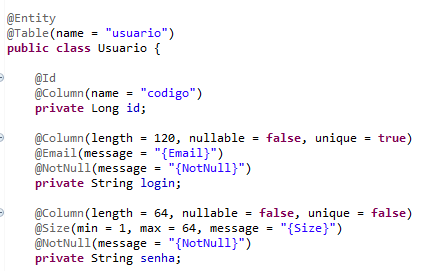
assets -> estilo.css



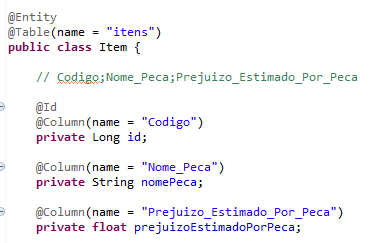


# Modelos

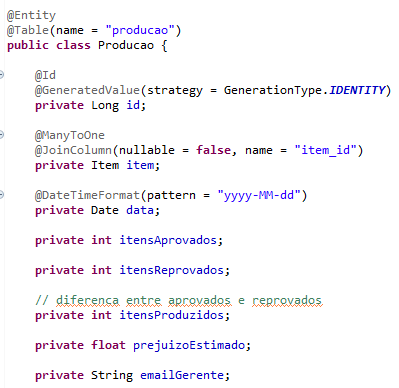
Usuario.java



Item.java

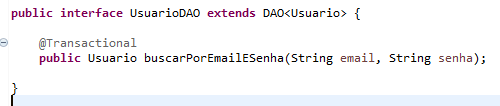


Producao.java

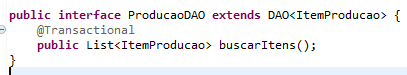


# DAO

UsuarioDAO.java



ProducaoDAO.java

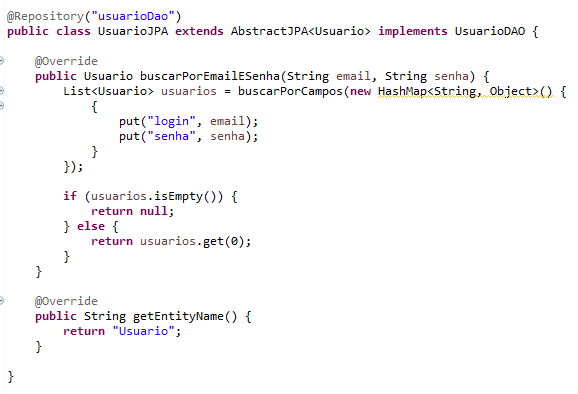


ItemDAO

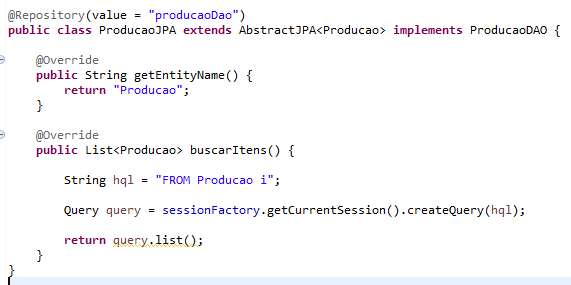


# JPA

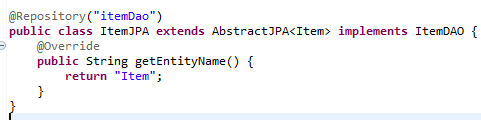
UsuarioJPA.java



ProducaoJPA.java

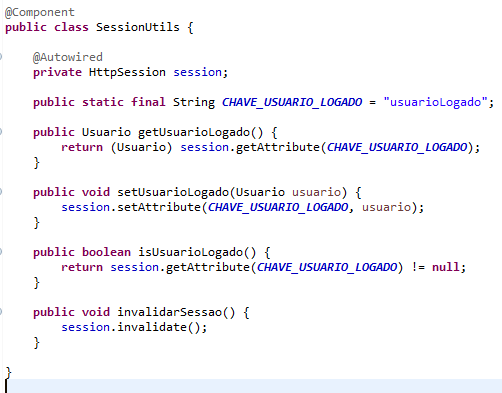


ItemJPA.java



# Sessão

SessionUtils.java



# Interceptor

AutenticacaoPorSessaoInterceptor.java



# Controllers

UsuarioController.java

package br.senai.sp.info.bon.controllers;

import java.io.IOException;

import java.util.List;

import javax.servlet.ServletContext;

import javax.servlet.http.HttpServletRequest;

import javax.servlet.http.HttpServletResponse;

import javax.validation.Valid;

import org.springframework.beans.BeanUtils;

import org.springframework.beans.factory.annotation.Autowired;

import org.springframework.stereotype.Controller;

import org.springframework.ui.Model;

import org.springframework.validation.BindingResult;

import org.springframework.validation.FieldError;

import org.springframework.web.bind.annotation.GetMapping;

import org.springframework.web.bind.annotation.PostMapping;

import org.springframework.web.bind.annotation.RequestParam;

import org.springframework.web.bind.annotation.RequestPart;

import org.springframework.web.multipart.MultipartFile;

import br.senai.sp.info.bon.core.SessionUtils;

import br.senai.sp.info.bon.dao.UsuarioDAO;

import br.senai.sp.info.bon.models.Usuario;

@Controller

public class UsuarioController {

@Autowired

private UsuarioDAO usuarioDao;

@Autowired

private SessionUtils sessionUtils;

@Autowired

private ServletContext context;

@GetMapping(value = { "/", "" })

public String abrirLogin(Model model) {

model.addAttribute("usuario", new Usuario());

return "index";

}

@PostMapping(value = { "/app/adm/usuario/salvar" }, consumes = { "multipart/form-data" })

public String salvar(@Valid Usuario usuario, BindingResult brUsuario,

@RequestParam(name = "isAdministrador", required = false) Boolean administrador,

@RequestPart(name = "foto", required = false) MultipartFile foto, HttpServletRequest request) {

// Se for um cadastro, valida qualquer campo...

if (usuario.getId() == null && brUsuario.hasFieldErrors()) {

return "usuario/form";

} else if (brUsuario.hasFieldErrors("nome") || brUsuario.hasFieldErrors("sobrenome")

|| brUsuario.hasFieldErrors("email")) {

return "usuario/form";

}

if (usuario.getId() == null) {

usuarioDao.inserir(usuario);

} else {

Usuario usuarioBuscado = usuarioDao.buscar(usuario.getId());

BeanUtils.copyProperties(usuario, usuarioBuscado, "id", "senha");

System.out.println(usuarioBuscado.getSenha());

System.out.println(usuarioBuscado.getId());

usuarioDao.alterar(usuarioBuscado);

}

return "redirect:/app/adm/usuario";

}

@PostMapping({ "/usuario/autenticar" })

public String autenticar(@Valid Usuario usuario, BindingResult brUsuario) {

// Se houver erros no usuario, reabre o index

if (brUsuario.hasFieldErrors("email") || brUsuario.hasFieldErrors("senha")) {

return "index";

}

Usuario usuarioBuscado = usuarioDao.buscarPorEmailESenha(usuario.getLogin(), usuario.getSenha());

if (usuarioBuscado == null) {

brUsuario.addError(new FieldError("usuario", "email", "Email ou senha inválidos"));

return "index";

}

// Aplica o usuário na sessão e abre a página inicial

sessionUtils.setUsuarioLogado(usuarioBuscado);

return "redirect:/app/";

}

@GetMapping({ "/sair" })

public String logout() {

sessionUtils.invalidarSessao();

return "redirect:/";

}

}

ProducaoController.java

**package** br.senai.sp.info.not.controllers;

**import** javax.validation.Valid;

**import** org.springframework.beans.factory.annotation.Autowired;

**import** org.springframework.stereotype.Controller;

**import** org.springframework.ui.Model;

**import** org.springframework.validation.BindingResult;

**import** org.springframework.web.bind.annotation.GetMapping;

**import** org.springframework.web.bind.annotation.PostMapping;

**import** org.springframework.web.bind.annotation.RequestMapping;

**import** br.senai.sp.info.not.dao.ItemDAO;

**import** br.senai.sp.info.not.dao.ProducaoDAO;

**import** br.senai.sp.info.not.models.Producao;

@Controller

@RequestMapping("/app")

**public** **class** ProducaoController {

@Autowired

**private** ItemDAO itemDao;

@Autowired

**private** ProducaoDAO producaoDao;

@GetMapping({ "", "/" })

**public** String abrirListaOcorrencia(Model model) {

model.addAttribute("itens", itemDao.buscarTodos());

model.addAttribute("itemProducao", **new** Producao());

**return** "producao/form";

}

@PostMapping("/producao/salvar")

**public** String salvar(@Valid Producao item, BindingResult brItemProducao, Model model) {

**if** (item.getId() == **null**) {

item.setPrejuizoEstimado(itemDao.buscar(item.getItem().getId()).getPrejuizoEstimadoPorPeca() \* item.getItensReprovados());

item.setItensProduzidos(item.getItensAprovados() - item.getItensReprovados());

producaoDao.inserir(item);

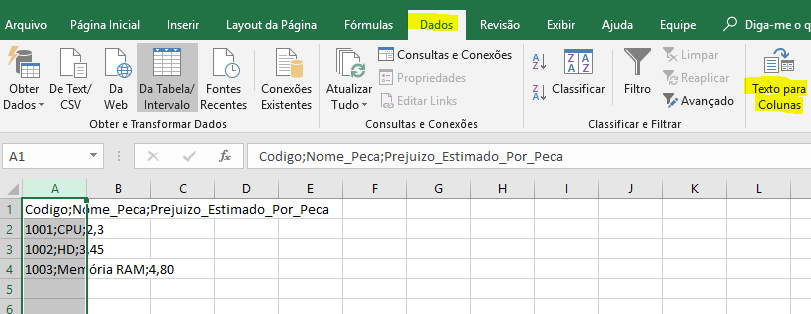
}

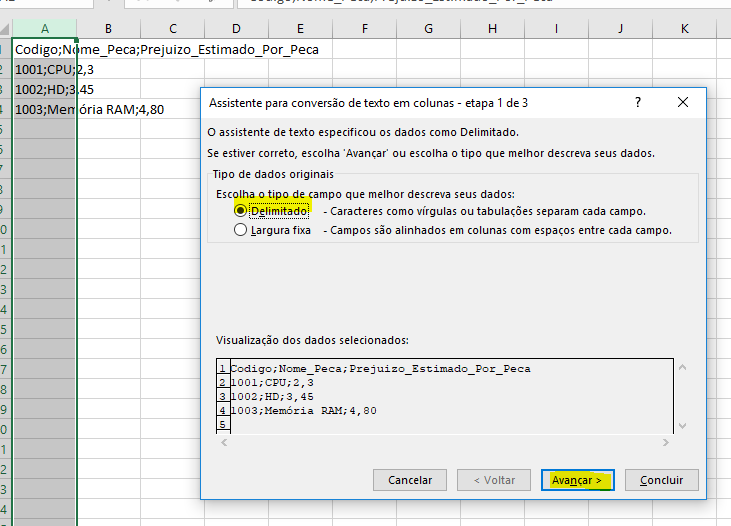
**return** "redirect:/app";

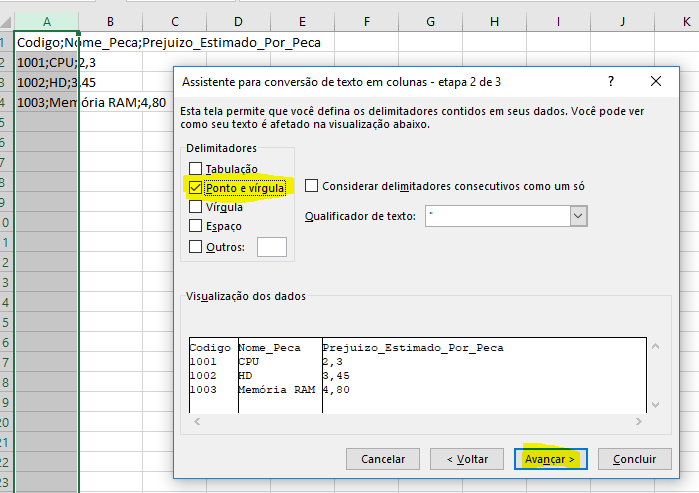
}

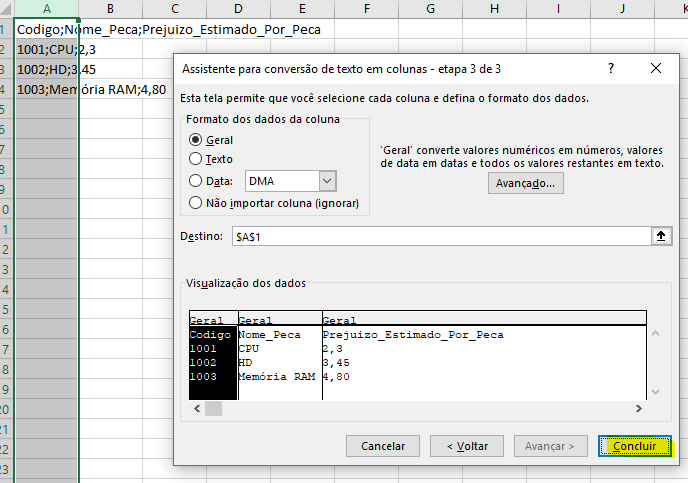
}

# Importando os dados



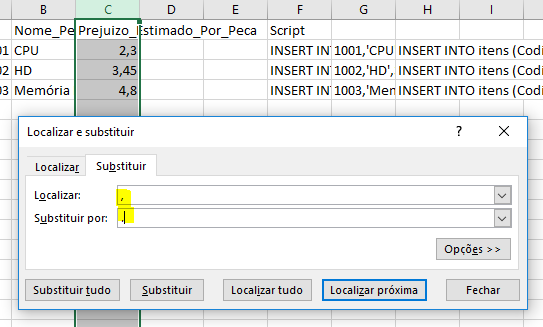




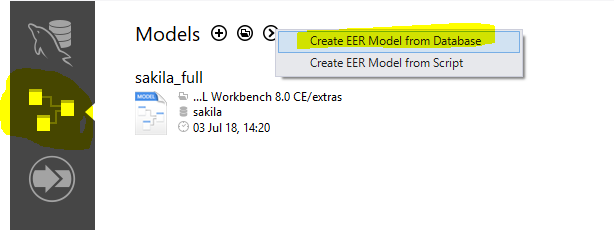


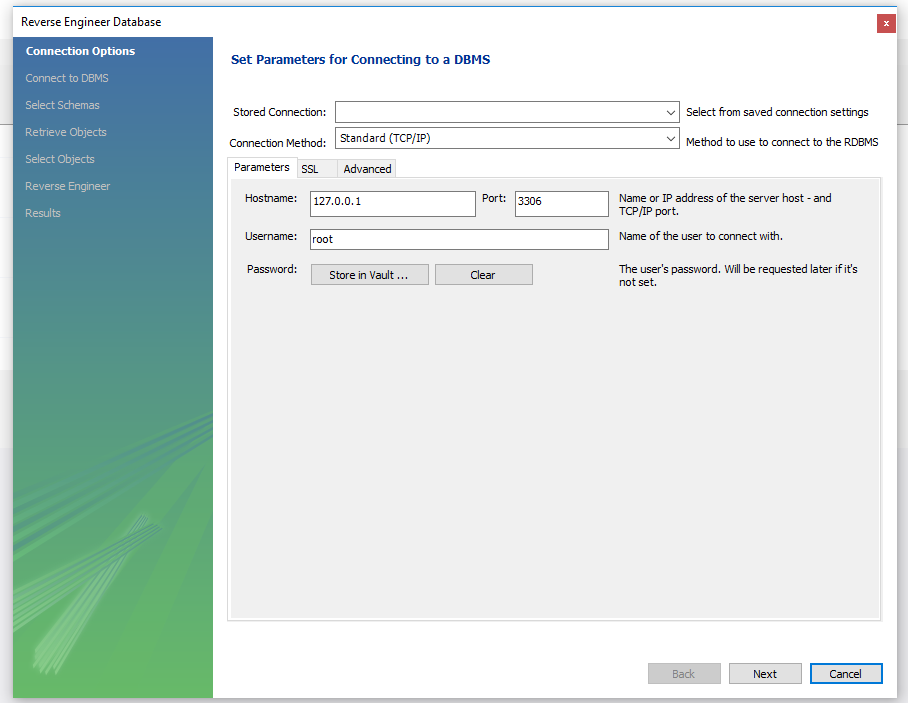
Localizar e substituir

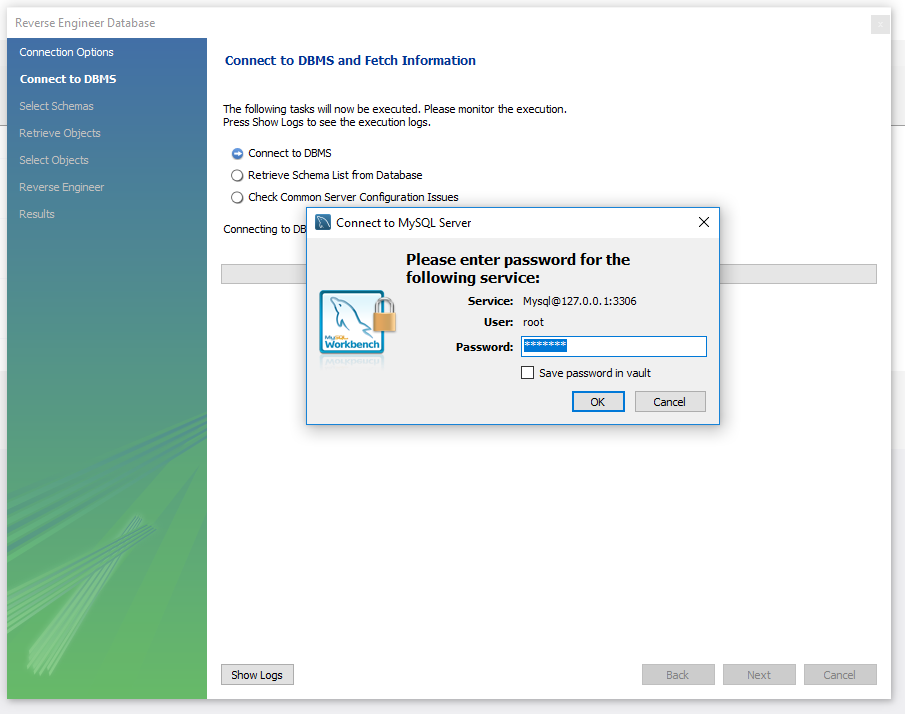
Trocar tudo o que for vírgula para ponto

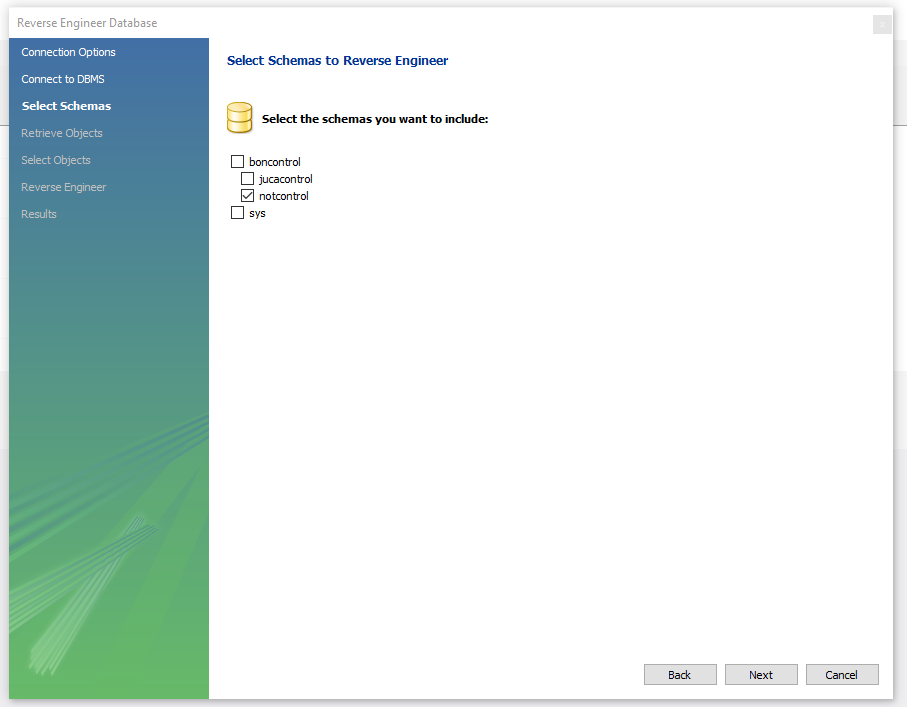


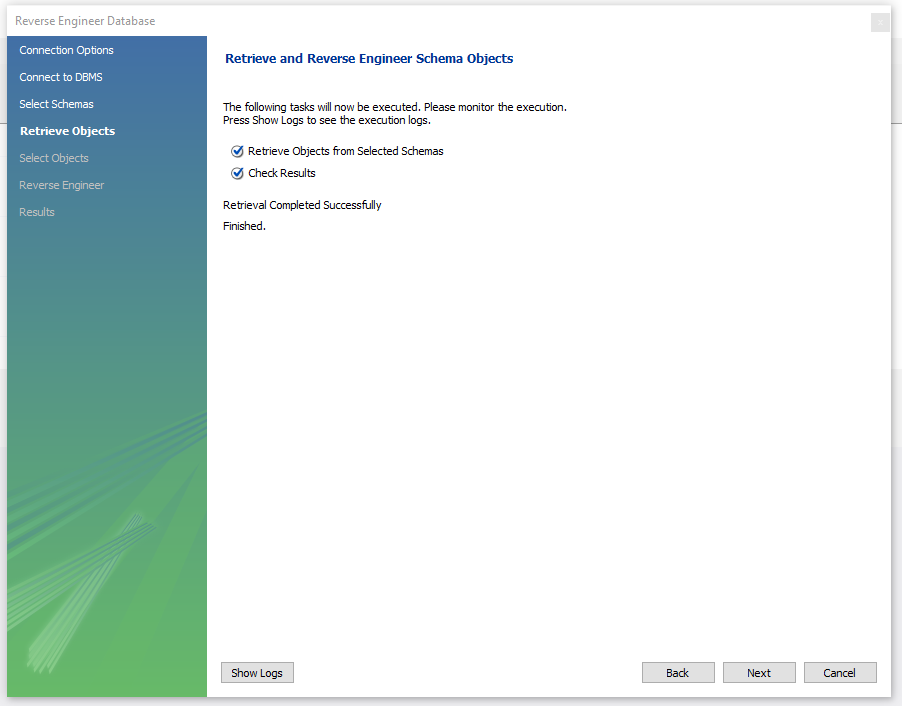
# MER

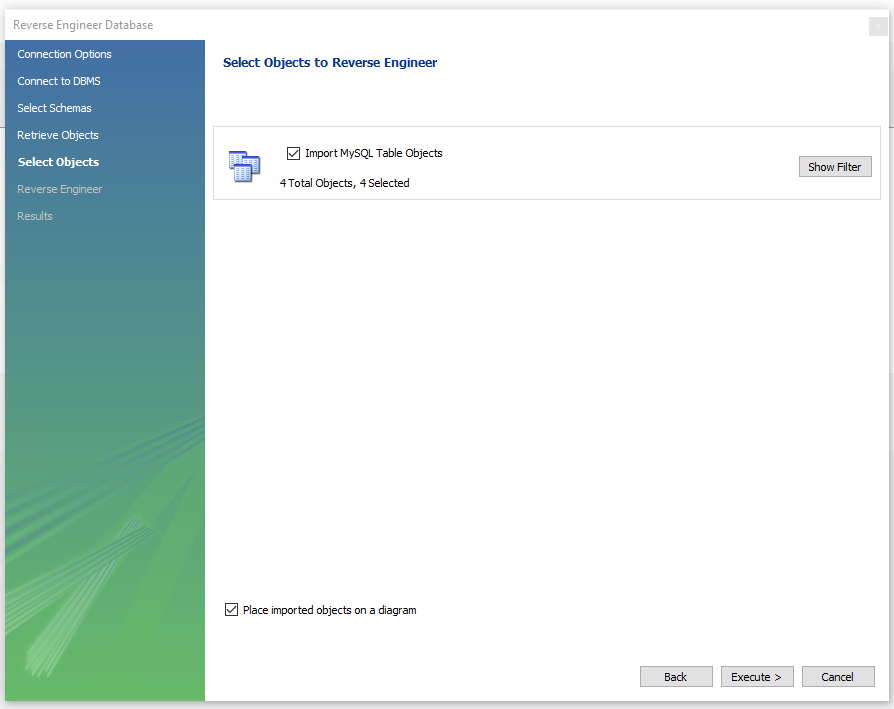


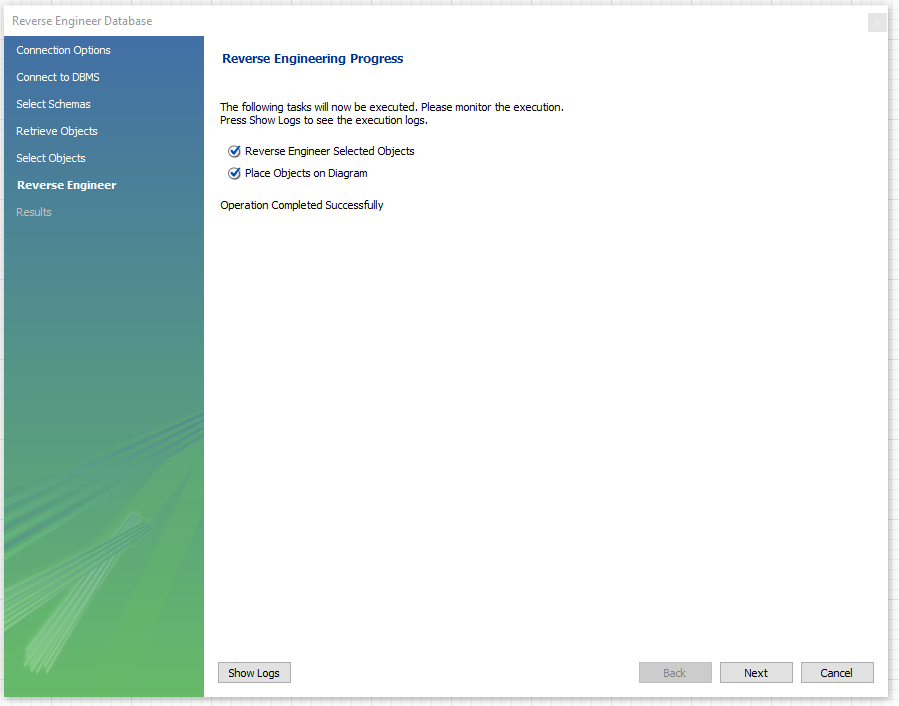


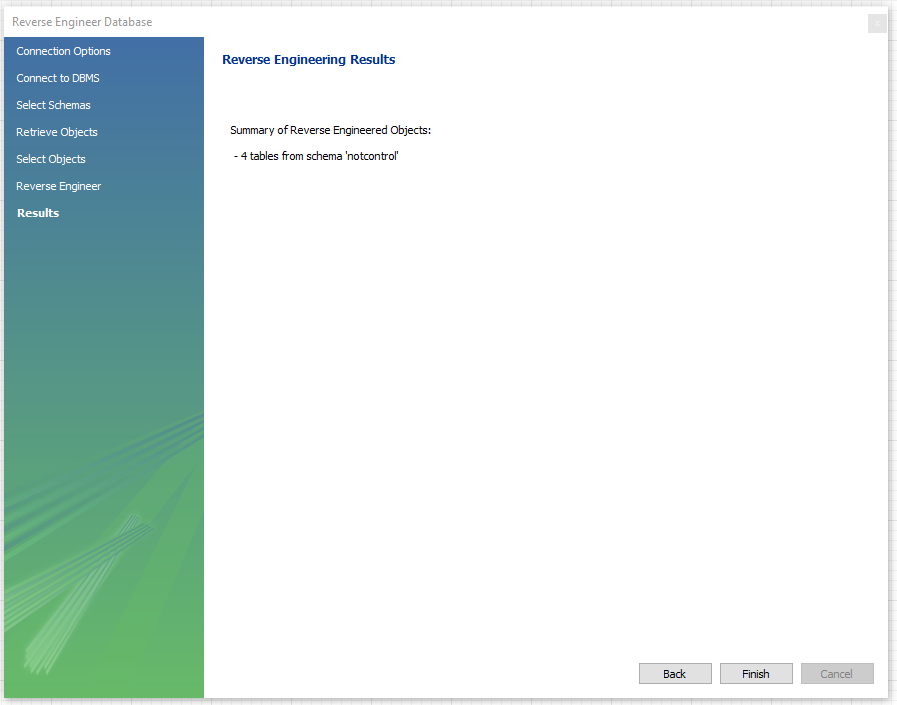


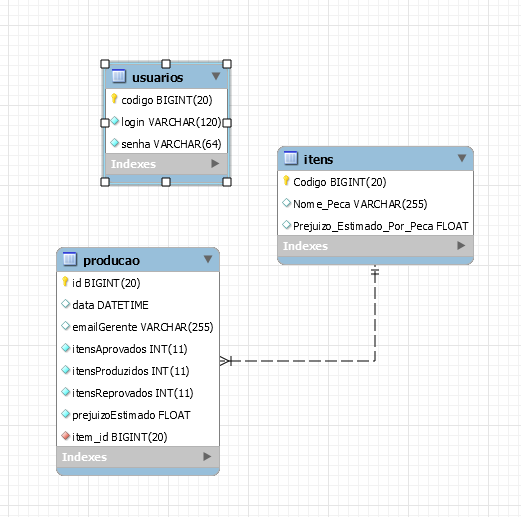












# DUMP

