Conclusiones del trabajo práctico de Persistencia

Para realizar este trabajo, utilicé el programa Intellij Idea y SpringBoot con las respectivas dependencias vistas en clase.

A partir del diagrama de clases presentado en el trabajo práctico, cree las clases de Cliente, Domicilio, Pedido, Factura, DetallePedido, Producto y Rubro.

Cliente tiene una relación OneToMany unidireccional con Domicilio y Pedido. Pedido tiene una relación OneToOne con Factura y una relación OneToMany con DetallePedido.

Detalle Pedido tiene una relación ManyToOne con Producto y Rubro tiene una relación OneToMany con Producto.

Luego en el main cree objetos con el método build a modo ejemplo para poder persistirlos en la base de datos H2:

```
Rubro rubro1 = Rubro.builder()
      .denominacion("pizzas")
      .build();


Rubro rubro2 = Rubro.builder()
      .denominacion("bebidas")
      .build();

Producto producto1 = Producto.builder()
      .tipo(Producto.Tipo.manufacturado)
      .tiempoEstimadoCocina(25)
      .denominacion("Pizza a la piedra")
      .precioVenta(1500)
      .precioCompra(900)
      .stockActual(10)
      .stockMinimo(5)
      .unidadMedida("unidad1")
      .receta("a")
      .build();


Producto producto2 = Producto.builder()
      .tipo(Producto.Tipo.insumo)
      .tiempoEstimadoCocina(0)
      .denominacion("Coca Cola")
      .precioVenta(800)
      .precioCompra(500)
      .stockActual(30)
      .stockMinimo(15)
```

```
        .unidadMedida("unidad2")
        .receta("ninguna")
        .build();

Producto producto3 = Producto.builder()
        .tipo(Producto.Tipo.manufacturado)
        .tiempoEstimadoCocina(30)
        .denominacion("Pizza cuatro quesos")
        .precioVenta(2000)
        .precioCompra(1300)
        .stockActual(20)
        .stockMinimo(12)
        .unidadMedida("unidad3")
        .receta("b")
        .build();

rubro1.agregarProductos(producto1);
rubro1.agregarProductos(producto3);
rubro2.agregarProductos(producto2);

rubroRepository.save(rubro1);
rubroRepository.save(rubro2);

Cliente cliente = Cliente.builder()
        .nombre("Juan")
        .apellido("Pérez")
        .telefono("2614567867")
        .email("juan@gmail.com")
        .build();

Domicilio domicilio1 = Domicilio.builder()
        .calle("Calle1")
        .numero("30")
        .localidad("Guaymallén")
        .build();


Domicilio domicilio2 = Domicilio.builder()
        .calle("Calle2")
        .numero("20")
        .localidad("Godoy Cruz")
        .build();

cliente.agregarDomicilio(domicilio1);
cliente.agregarDomicilio(domicilio2);

SimpleDateFormat  formatoFecha = new
SimpleDateFormat("yyyy-MM-dd");
```

```
String fecha1String = "2022-04-02";
String fecha2String = "2022-11-30";
String fecha3String = "2022-12-01";
String fecha4String = "2022-12-03";
Date fecha1 = formatoFecha.parse(fecha1String);
Date fecha2 = formatoFecha.parse(fecha2String);
Date fecha3 = formatoFecha.parse(fecha3String);
Date fecha4 = formatoFecha.parse(fecha4String);

DetallePedido detallePedido1 = DetallePedido.builder()
      .cantidad(2)
      .subtotal(3000)
      .build();


DetallePedido detallePedido2 = DetallePedido.builder()
      .cantidad(1)
      .subtotal(800)
      .build();

DetallePedido detallePedido3 = DetallePedido.builder()
      .cantidad(1)
      .subtotal(2000)
      .build();

detallePedido1.setProducto(producto1);
detallePedido2.setProducto(producto2);
detallePedido3.setProducto(producto3);

Pedido pedido1 = Pedido.builder()
      .estado(Pedido.Estado.iniciado)
      .fecha(fecha1)
      .tipoEnvio(Pedido.TipoEnvio.retira)
      .total(3800)
      .build();

pedido1.agregarDetallesPedido(detallePedido1);
pedido1.agregarDetallesPedido(detallePedido2);

Pedido pedido2 = Pedido.builder()
      .estado(Pedido.Estado.entregado)
      .fecha(fecha2)
      .tipoEnvio(Pedido.TipoEnvio.delivery)
      .total(2000)
      .build();

pedido2.agregarDetallesPedido(detallePedido3);
```

```
Factura factura1 = Factura.builder()
      .numero(1)
      .fecha(fecha1)
      .descuento(20)
      .formaPago(Factura.FormaPago.efectivo)
      .total(3400)
      .build();


Factura factura2 = Factura.builder()
      .numero(2)
      .fecha(fecha2)
      .descuento(0)
      .formaPago(Factura.FormaPago.etc)
      .total(2000)
      .build();

pedido1.setFactura(factura1);
pedido2.setFactura(factura2);

cliente.agregarPedidos(pedido1);
cliente.agregarPedidos(pedido2);

clienteRepository.save(cliente);
```

Por último, decidí probar algunos métodos de las consultas JPA:

Mostrar los productos del rubro1:

```
/* Mostrar los productos del rubro1 */

Rubro rubroRecuperado =
rubroRepository.findById(rubro1.getId()).orElse(null);

if (rubroRecuperado != null) {
   rubroRecuperado.mostrarProductos();
}
```

Productos del rubro: pizzas:
Nombre: Pizza a la piedra
Nombre: Pizza cuatro quesos

Mostrar los pedidos y domicilios del cliente 1:

```
/*Mostrar los pedidos y domicilios del cliente */
Cliente clienteRecuperado =
clienteRepository.findById(cliente.getId()).orElse(null);
```

```
if (clienteRecuperado != null) {
   clienteRecuperado.mostrarPedidos();
   clienteRecuperado.mostrarDomicilios();
}
```

Pedidos de JuanPérez:
ID: 1Estado: iniciado, Fecha: 2022-04-02, Tipo de envío: retira, Total: 3800.0
Producto 1: Pizza a la piedra cantidad: 2 subtotal: 3000.0
Producto 2: Coca Cola cantidad: 1 subtotal: 800.0
ID: 2Estado: entregado, Fecha: 2022-11-30, Tipo de envío: delivery, Total: 2000.0
Producto 1: Pizza cuatro quesos cantidad: 1 subtotal: 2000.0
Domicilios de JuanPérez:
Calle: Calle1, Número: 30, Localidad: Guaymallén
Calle: Calle2, Número: 20, Localidad: Godoy Cruz

Cambiar el valor de un atributo de producto2 y mostrarlo en pantalla:

```
/* Cambiar el tipo de receta en producto2 */

producto2.setReceta("c");
productoRepository.save(producto2);

Producto productoRecuperado =
productoRepository.findById(producto2.getId()).orElse(null);
if (productoRecuperado != null) {
   System.out.println("receta " + productoRecuperado.getReceta());
}
```

receta c

Borrar un domicilio de cliente:

```
/* Eliminar un domicilio de cliente */

domicilioRepository.delete(domicilio2);

Cliente clienteRecuperado2 =
clienteRepository.findById(cliente.getId()).orElse(null);


if (clienteRecuperado2 != null) {
   clienteRecuperado2.mostrarDomicilios();
}
```

Domicilios de JuanPérez:
Calle: Calle1, Número: 30, Localidad: Guaymallén

Es de destacar que tuve un problema a la hora de mapear este modelo:

En la relación OneToMany de rubro con producto, no utilicé `fetch = FetchType.EAGER,` por lo que me generaba un error de carga perezosa.

Para finalizar el informe, es de mencionar que se puede levantar la base de datos H2 con http://localhost:8080/h2-console/ y verificar que los datos ingresados están correctamente mapeados.