

Primera consulta: QBE

Aquesta es l'únic tipus de Query que he implementat de manera útil al codi. Bàsicament fa una query de tots els objectes a la BD, filtrant per Conductors/Vehicles depenent de la taula a plenar al java.

Aquests resultats els utilitzo per a plenar la Collection que es la que utilitzo per a carregar la taula.

```
private void carregarBD() {
    //DB40
    modelo.buidarCol();

    //CARREGAR VEHICLES A LA COL·LECCIÓ
    ObjectSet<Vehicle> resultV = modelo.getDB().queryByExample(new Vehicle(null, null, 0, 0));
    for (Vehicle vehicle : resultV) {
        modelo.insertarVehicle(vehicle);
    }

    //CARREGAR CONDUCTOR A LA COL·LECCIÓ
    ObjectSet<Conductor> resultC = modelo.getDB().queryByExample(new Conductor(null, null, 0, 0, 0));
    for (Conductor conductor : resultC) {
        modelo.insertarConductor(conductor);
    }

    carregarTaulaVehicleActual();
    carregarTaulaConductorActual();
}
```

NATIVES

Tant per a natives com per a SODA, he implementat 2 senzilles consultes per tal de justificar l'utilització de les consultes. En aquest cas es una consulta que et mostra un llistat amb tots els conductors que comencen per una lletra específica.

```
//LLISTAT CONDUCTORS PER NOM
view.getLlistarCondButton().addActionListener(e -> {
    view.getLlistarCondTextarea().setText("");
    if (view.getLlistarCondTextfield().getText().length() <= 1) {
        char lletra = view.getLlistarCondTextfield().getText().charAt(0);
        Comparator<Conductor> cmpCond = new Comparator<Conductor>() {
            public int compare(Conductor c1, Conductor c2) {
                return c1.get4_nom_Conductor().compareTo(c2.get4_nom_Conductor());
            }
        };
        ObjectSet<Conductor> llistat = modelo.getDB().query(new Predicate<Conductor>() {
            @Override
            public boolean match(Conductor co) {
                return co.get4_nom_Conductor().charAt(0) == lletra;
            }
        }, cmpCond);

        for(Conductor c: llistat) {
            view.getLlistarCondTextarea().append(c.get4_nom_Conductor()+" "+c.get2_cognom_Conductor()+" ");
        }
    } else {
        view.getLlistarCondTextarea().setText("Només pots introduir una lletra!");
    }
});
```

ID_CONDUCTOR	COGNOM_CONDUCTOR	EDAT_CONDUCTOR	NOM_CONDUCTOR	VEHICLE_CONDUCTOR
1486	Fujiwara	21	Takumi	86
12365	Walker	45	Paul	22
12547	Cooper	37	Aaron	22
12587	Toretto	54	Domenic	22
54879	Heitz	19	Axel	22
65987	Fernandez	28	Pep	66
99999	Speer	58	Alex	66

CONDUCTORS

Eliminar fila!

Ordenar per ID ▾

Editar!

Nom Conductor

Nom Placeholder

Edat Conductor

999

Afegir registre

Cognom Conductor

Cognom Placeholder

ID Conductor

99999

Numero del vehicle

model.Vehicle@14bdbc74 ▾

Quants conductors hi han amb mes de

18

anys?

Consulta!

Llistar els conductors que comencen per la lletra:

A

Llistar!

Aaron Cooper, Alex Speer, Axel H ▾

SODA

Per al soda la consulta es que ens mostrarà quants (numero) conductors hi ha que superen la edat indicada.

```
//CONSULTA EDAT CONDUCTORS
view.getConsultaEdatButton().addActionListener(e -> {
    int i = 0;
    int consta = Integer.parseInt(view.getConsultaEdatText1().getText());

    Query q = modelo.getDB().query();
    q.constrain(model.Conductor.class);
    q.descend("_3_edat_Conductor").constrain(consta).greater();

    ObjectSet<Conductor> result = q.execute();

    for (Conductor c : result) {
        System.out.println(c.toString());
        i++;
    }
    view.getConsultaEdatText2().setText(String.valueOf(i));
});
```

ID_CONDUCTOR	COGNOM_CONDUCTOR	EDAT_CONDUCTOR	NOM
1486	Fujiwara	21	Takumi
12365	Walker	45	Paul
12547	Cooper	37	Aaron
12587	Toretto	54	Domeni
54879	Heitz	19	Axel
65987	Fernandez	28	Pep
99999	Speer	58	Alex

Nom Conductor

Nom Placeholder

Edat Conductor

999

Cognom Conductor

Cognom Placeholder

ID Conductor

99999

Numero del vehicle

model.Vehicle@14bdbc74 ▾

Quants conductors hi han amb mes de

26

anys?

5

Consulta!

Afegir

