BB

Bases de dades avançades

Pràctica 1

Judit Domènech Miró

Karen Samsó Muñoz

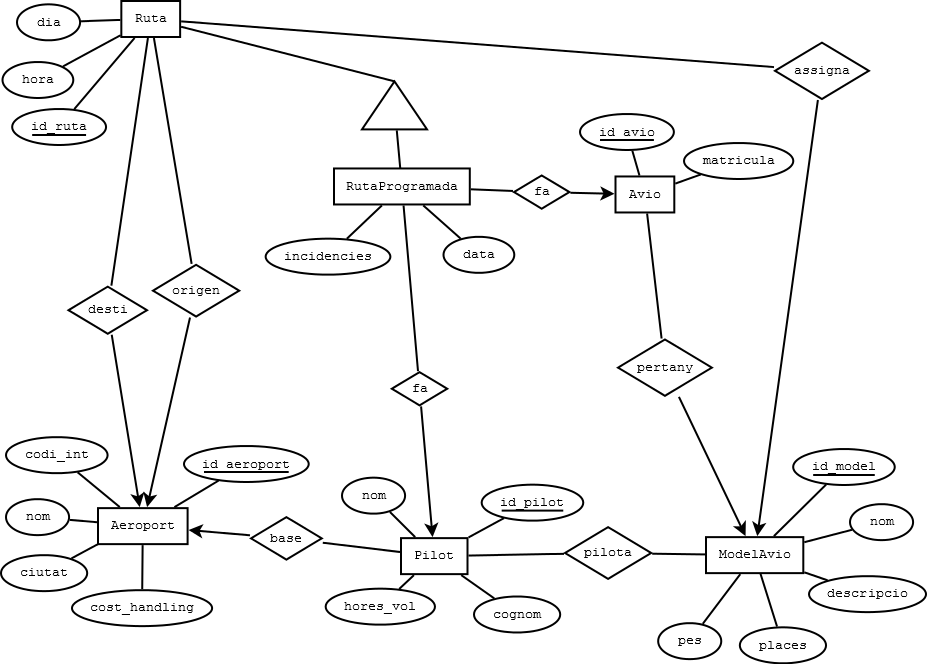
# Introducció

En aquesta pràctica tenim com a objectiu realitzar una aplicació per gestionar una aerolinia de forma simplificada, d’aquesta manera aprendrem a integrar l’ús de bases de dades en aplicacions utilitzant un llenguatge d’alt nivell com és Java. A més aprendrem a utilitzar hibernate, un framework de Java per ORM *(Object Relational mapping).*

# Desenvolupament

## Diagrama ER i pas a taules

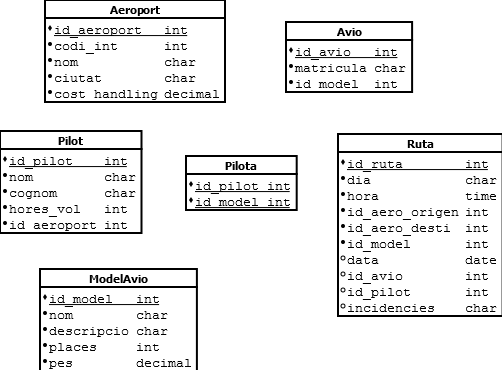
Per començar el desenvolupament, el primer pas ha estat crear el diagrama entitat-relació i després hem fet el pas a taules. El resultat es pot veure a la imatge següent:



Il·lustració : Diagrama E-R

Explicar diagrama........................................................................shit

El més destacat d’aquest diagrama potser és l’herència de Ruta, tot i que la implementació de l’aplicació final i el pas a taules tenim únicament una entitat ruta.

No sé que més podem dir sobre això.

Il·lustració : Pas a taules

Explicar pas a taules ¿Cal?

## Implementació aplicació

Un cop ja hem fet el model i el pas a taules, hem passat a implementar l’aplicació a partir de l’aplicació que se’ns va proporcionar per a la pràctica 0. Hem creat una classe java per a cada entitat i a diferència de l’exemple, el mapejat s’ha fet mitjançant les *annotations*.

L’aplicació funciona de tal manera que un cop l’usuari ha fet el *log in*, apareix un menú amb les opcions CRUD, Afegir (Create), Consultar (Retrieve), Modificar (Update) i Borrar (Delete). Un cop seleccionada alguna d’aquestes opcions, apareix una llista de les entitats sobre les quals podem realitzar alguna de les opcions anteriors.

* Create: en aquest cas, demanem les dades necessàries per a crear un nou objecte i l’afegim a la base de dades. Es recomana no equivocar-se a l’hora d’entrar les dades ja que no hi ha gaire control d’errors per qüestió de temps. Un cop tenim les dades, creem l’objecte i l’incloem a la base de dades.
* Retrieve: Un cop seleccionada l’entitat que volem consultar, fem una query a la base de dades i obtenim els objectes, després imprimim la informació desitjada d’aquests objectes.
* Update: Tal com fem amb la part de consulta, mostrem obtenim la llista d’objectes d’una entitat, llavors demanem a l’usuari que indiqui l’index de la llista que es mostra, i obtenim l’objecte situat en aquest index, demanem les dades de tot a l’usuari i modifiquem l’objecte mitjançant els mètodes *set* i fem un commit a la base de dades.
* Delete: Igual que en el cas de l’*update*, però en comptes de demanar a l’usuari, borrem l’objecte de la base de dades fent un delete i un commit.

# Fonts consultades

* Codi d’exemple del campus virtual.
* <https://docs.jboss.org/hibernate/stable/annotations/reference/en/html_single/>
* <https://examples.javacodegeeks.com/enterprise-java/hibernate/hibernate-annotations-example/>
* [www.stackoverflow.com](http://www.stackoverflow.com)
* <http://www.simplecodestuffs.com/auto-generate-primary-key-in-hibernate/>
* <http://www.java2s.com/Code/Java/Development-Class/Javaprogramtodemonstratemenuselection.htm>
* <http://docs.oracle.com/javase/8/docs/>
* Probablement alguna web més.