Ecole Polytechnique de Louvain

INGI 1271 - Fichiers et bases de données

Rapport de projet

" Gestion d'un aéroport "

Groupe 13
DE GROOTE Charles
LAMOULINE Laurent
NUTTIN Vincent

Q6 - 2009

TABLE DES MATIÈRES

Introduction **2**

1 Schéma entité-association **3**
1.1 Etude du schéma 4
1.2 Contraintes d'intégrité 4

2 Schéma relationnel **7**
2.1 Concept de clé 7

3 Requêtes SQL **10**

4 Création et remplissage de la base de données **13**

5 Interface php du site **14**

Conclusion **16**

INTRODUCTION

Dans le cadre du cours "INGI 1271 - Fichiers et bases de données", il nous a été de demandé de concevoir une base de données reprenant des informations relatives à la gestion d'un aéroport telles que les horaires des membres du personnel, l'utilisation des pistes de l'aéroport ou encore le nombre d'heures de vol des pilotes. Cette base de donnée sera utile et nécessaire pour les applications souhaitant gérer les différents aspects de l'aéroport. De plus, il nous est demandé de concevoir une interface graphique simple et efficace afin de pouvoir effectuer rapidement des requêtes sur la base de données par des membres du personnel de la compagnie.

Ce rapport s'intéressera donc à la conception et la réalisation de la base de donnée. Les différents points de ce rapport seront abordés dans l'ordre suivants :

1. Schéma entité-association et contraintes d'intégrité
2. Schéma relationnel
3. Requêtes SQL
4. Création et remplissage de la base de données
5. Interface en php

CHAPITRE 1

SCHÉMA ENTITÉ-ASSOCIATION

Le modèle entité-relation fournit une description graphique pour représenter des modèles conceptuels de données sous la forme de diagrammes contenant des entités et des associations (les relations).

Grâce à l'organisation de l'aéroport présentée dans l'énoncé, il est possible de construire un schéma entité-association (voir fig. 1.1). Celui-ci synthétisera la manière dont les informations seront stockées dans la base de données.

Ce schéma repose sur le modèle *"entité - association"*. Il permet d'identifier et de caractériser les objets du domaine d'application et d'établir leurs liens ; les cardinalités donnent des renseignements sur le minimum et le maximum d'occurrences d'une association liant une entité à une autre.

Au niveau conceptuel, le modèle entité-association distingue les objets et les relations. Les objets sont représentés par des rectangles, les relations par des ellipses. Les entités, objets ou relations, ont des propriétés ou attributs. Une relation (m, n) se traduit par un segment logique.

1.1 Etude du schéma

L'héritage

L'héritage permet de décrire un même ensemble d'entités à différents niveaux d'abstraction. Etant donné des entités partageants des attributs communs, l'utilisation de l'héritage permet d'éviter la duplication.

Dans notre cas, nous aurons recours à l'héritage pour les entités suivantes :
- les entités `Worker` et `Traveller` héritant de l'entité `Person`
- les entités `Crew member, Air-traffic controller, Luggage handler, Check-in steward` hérite de l'entité `Worker`

1.2 Contraintes d'intégrité

Les contraintes d'intégrité permettent de garantir l'intégrité des données. La plupart de celles-ci sont données par les cardinalités qui donnent le nombre d'occurrences d'une association liant une entité à une autre.

Les autres ne peuvent être explicitées sur le schéma, et doivent donc être citées. Elles sont triées par entitées.

Entité `Person`

- La valeur de l'attribut `Sex` ne peut prendre que les valeurs *'M'* ou *'F'*.
- La valeur de l'attribut `FGS No` doit comprendre 8 nombres entiers.
- La valeur de l'attribut `Card No` doit comprendre 12 nombres entiers.

Entité `Staff member`

- La valeur des attributs `Salary` et `Schedule` doivent être des entiers positif.
- La valeur de l'attribut `Account No` doit comprendre 12 nombres entiers.

Entité `Fly`

- Le format de l'attribut `Day` doit être de type `AnnéeMoisJour`.
- Le jour du vol doit être plus petit que le jour actuel. De même pour l'heure.
- L'heure d'embarquement (`Boarding hour`) doit être plus petit que l'heure de décollage (`hour`)
- La destination (`Destination`) doit être différente du lieu de départ (`Airport name`).
- La valeur de l'attribut `Capacity` doit être un entier positif et plus petit que la capacité de l'avion qui assure le vol.
- Pour tous les vols d'une piste donnée, aucun ne peuvent avoir la même heure de décollage pour le même jour.

Entité `Plane`

- La valeur des attributs `Capacity, WC number` et `Emergency exit number` doivent être des entiers positif.
- La distance de décollage d'un avion assigné à un vol doit être plus petit que la longueur de la piste assignée à ce vol.

Entité `Lane`

- La valeur des attributs `Wind force` et `Length` doivent être des entiers positif.
- Le format de l'attribut `Wind direction` est celui des points cardinaux : *N, NE, E, SE, S, SO, O, NO*.

Entité `Pilot`

- La valeur de l'attribut `Flying hours` doit être un entier positif et plus petit que la limite autorisée.
- Le pilote doit avoir l'aptitude nécessaire pour piloter l'avion qui lui est assigné.

Entité `Traveller`

- Les voyageurs sur un même vol doivent tous avoir un siège distinct.
- L'attribut `Paid` peut prendre deux valeurs : *'yes'* ou *'no'*.
- Un voyageur qui n'a pas payé n'est pas enregistré sur un vol.

Pour finir, tous les attributs doivent être remplis sauf pour les attributs `Passport No` et `Card No` de l'entité `Person`, pour qui un des deux champs minimum sont obligatoires.

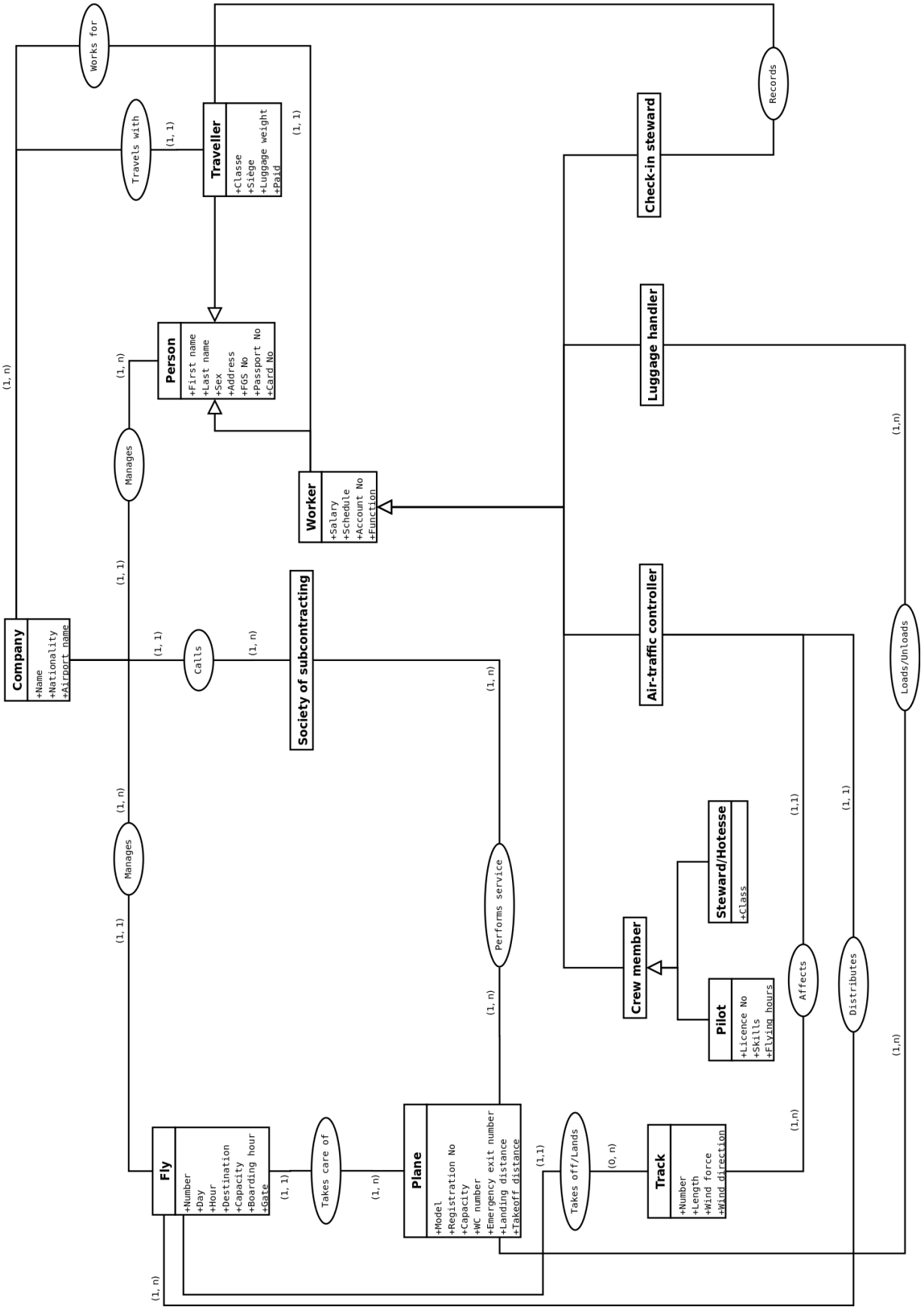

FIGURE 1.1 – Schéma entité-association

CHAPITRE 2

SCHÉMA RELATIONNEL

Le modèle relationnel est une manière de modéliser les informations contenues dans une base de données qui repose sur des principes mathématiques inventés par E.F. Codd.

Le passage du modèle Entité-Association vers le modèle relationnel a nécessité certaines adaptations. En effet, un champ `id` a été ajouté en début de chaque table. En plus, une dixième table, intitulée `general`, a été ajoutée. Celle-ci reprend les `id` de chaque table afin de pouvoir comparer les `id` des tables entre-eux.

2.1 Concept de clé

Par définition, une super-clé est définie comme étant un groupe d'attributs, qui tous ensemble permettent d'identifier une entité de manière unique. Une super-clé peut contenir des attributs qui sont superflus pour une identification unique. Par exemple, dans le cas de l'entité `person`, une super clé pourrait être : `Last Name + Card No`, l'attribut `Last Name` est superflu car le numéro de carte d'identité possède déjà la propriété de l'identification unique. Il est dès lors possible de donner une définition d'une clé sur base de la super clé : une clé correspond à la super clé minimale, c'est-à-dire à un groupe d'attributs tels que la suppression de l'un de ceux-ci ne permet plus d'identifier de manière unique une entité.

La clé primaire correspond à la clé la plus utile dans la relation, en général, elle est suggérée par la conception de la base de données.

Nous allons maintenant analyser la façon dont nous avons réalisé le schéma relationnel de notre application et les critères sur lesquels nous nous sommes basé afin de choisir les clés primaires pour chaque relation car en SQL il est nécessaire de les définir pour que cela fonctionne.

General

Table général permettant de faire le lien entre toutes les autres tables via leurs `id`.

Company

Dans un premier temps, nous avons repris les attributs du schéma entité-association de l'entité `Company` pour réalisé le schéma relationnel.
Nous avons ensuite choisi `Name` comme clé primaire de cette entité car par définition, une compagnie ne peut avoir le même nom qu'une autre. Nous sommes dès lors certain qu'une seule compagnie correspondra au nom `Name`.

Person

Pour l'entité `Person`, nous avons choisi l'attribut `FGS No` comme clé primaire car elle permet d'identifier de manière unique un individu. Les attributs `Passport No` et `Card No` permettent également d'identifier une personne de manière unique mais ces attributs n'étant pas tout deux obligatoires, il est dès lors plus facile d'utiliser le numéro FGS.

Traveller

Dans le cas des voyageurs, nous avons choisi de les identifier via un numéro d'identification unique correspondant à un champ auto-incrément que l'on nomme `id_person`. Dans le schéma ci-dessous, on remarque que l'attribut `id_person` correspond à l'attribut `id` de l'entité `person`.

Worker

Nous avons choisi `id_person` comme clé primaire pour les mêmes raisons que celles évoquées précédemment pour `Traveller`.

Fly

Dans le cas des vols, nous avons choisi d'attribuer un numéro de vol `No` unique à chacun de ceux-ci, c'est pourquoi nous pouvons utiliser `No` comme clé primaire, car celle-ci identifie effectivement un seul vol sans risque de doublon.

Plane

Nous avons attribué une immatriculation unique à chaque avion, celle-ci correspond à l'attribut `Registration No`. Il parait dès lors logique et pratique de choisir cet attribut comme clé primaire afin d'identifier un appareil.

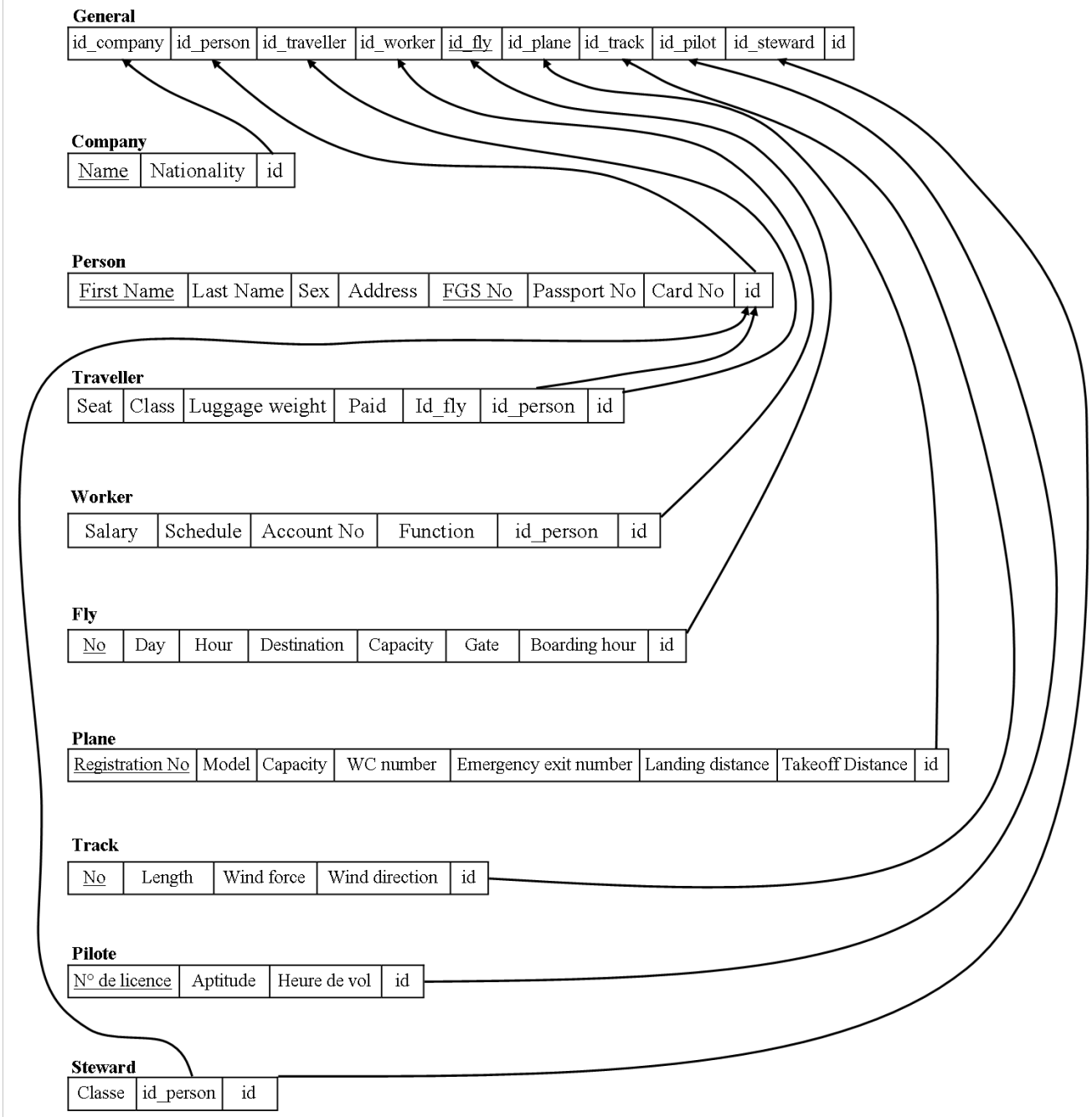

FIGURE 2.1 – Schéma relationnel

CHAPITRE 3

REQUÊTES SQL

Il nous a été demandé de créer 5 requêtes SQL intéressantes. Celles-ci ont été choisies car elles utilisent différents aspects du langage SQL :

- Requêtes multi-tables
- Requêtes imbriquées
- Calcul de nouvelle valeur

1- Demander tous les avions ayant décollés avec le vent de nord à une certaine date.

```
SELECT *
FROM plane
WHERE id IN
(SELECT DISTINCT G.id_plane
FROM general G, plane P, fly F, tracks T
WHERE P.id=G.id_plane AND F.id=G.id_fly AND T.id=G.id_tracks
AND T.windDirection='N' AND F.day='DATE')
```

2- Demander quelle piste fut la plus en activité à une certaine date.

```
SELECT number,COUNT(*)
FROM tracks
WHERE id IN
(SELECT G.id_tracks
FROM general G, tracks T, fly F
WHERE T.id = G.id_tracks AND  F.id = G.id_fly
AND F.day = 'DATE')
GROUP BY number
ORDER BY COUNT(*) DESC
LIMIT 0,1
```

3- Demander de lister les avions disponibles ayant au minimum X places

```
SELECT *
FROM plane
WHERE id IN
(SELECT G.id_plane
FROM general G, plane P
WHERE G.id_plane=P.id AND P.capacity > 'X')
```

4- Demander quel bagagiste s'est occupé des bagages de Monsieur X.

```
SELECT id, lastname, firstname
FROM person
WHERE id IN
(SELECT G.id_luggage
FROM general G, workers W, traveller T, person P
WHERE G.id_luggage = W.id_person AND G.id_fly = T.id_fly
AND T.id_person = P.id AND P.lastname = 'Mr. X' )
```

5- Demander de tirer les fiches de salaire du personnel ayant travaillé durant le mois.

```
SELECT id, lastname, firstname, salary*schedule
FROM person, workers
WHERE person.id=workers.id_person
ORDER by salary*schedule DESC
```

Ces 5 requêtes se trouve dans la page `INGI1271` du site.

En plus de ces 5 requêtes, nous avons ajouté de multiples autres requêtes. Celles-ci sont nombreuses, les plus importantes étant :

- l'ajout de vol
- l'ajout de personnels
- l'ajout de voyageur
- l'ajout d'avion

Pour prendre connaissance de celle-ci, nous vous invitons à parcourir le site ainsi que le code php relatif à celui-ci.

CHAPITRE 4

CRÉATION ET REMPLISSAGE DE LA BASE DE DONNÉES

Lors de la création de la base de données, nous avons créé les 10 tables reprises dans le schéma relationnel ainsi que leurs champs.

Etant donné qu'il nous était impossible de compléter de façon exhaustive notre base de données, nous l'avons remplie en analysant d'abord le type de requêtes intéressantes que nous désirions effectuer. Une fois qu'une ébauche de ces requêtes fut établie, nous avons analysé nos besoins et rempli la base de données en conséquence.

Nous avions besoin notamment de :

- Plusieurs avions ayant décollés à une certaine date. (1^{ere} `Requête`)
- Plusieurs avions ayant décollés sur différentes pistes à une certaine date. (2^{e} `Requête`)
- Plusieurs avions de types différents. (3^{e} `Requête`)
- Plusieurs bagagistes et des voyageurs ayant pris des vols. (4^{e} `Requête`)
- Plusieurs membres du personnel. (5^{e} `Requête`)

Les tables ont de plus été remplies avec quelques données supplémentaires afin que toutes les requêtes que l'on a choisi d'implémenter puisse revoyer un résultat non vide.

CHAPITRE 5

INTERFACE PHP DU SITE

Nous avons créé une interface en PHP pour rendre notre requêtes visibles et agréables à visionner. Nous avons essayé de la rendre la plus complète et intuitive possible.
Dans le menu "Company", il est possible de trouver tout d'habord un listing de toutes les personnes enregistrées dans notre base de données. Ensuite, un lien permet de trouver de plus amples informations sur chaque personne.
Le menu "Flights" présente une liste des vols effectués par nos avions. Il est alors possible d'obtenir des informations plus précises relatives à ces derniers en cliquant sur le numéro de vol.
Enfin, le menu "Planes" présente notre flotte.
Ces menus ont été mis en place pour faciliter les ajouts, modifications dans la base de données. Les cinq requêtes que nous avions proposées sont quant à elles accessibles dans le menu "INGI 1271". Chaque requête est exécutable séparément grâce à un petit formulaire HTML disponnible sur cette page.

FIGURE 5.1 – Visualisation de l'interface graphique

CONCLUSION

Ce projet avait pour but de créer une base de données, de la remplir et ensuite de pouvoir effectuer des requêtes sur celle-ci. Pour arriver à la conception de la base de données, une phase théorique fut nécessaire : le modèle entité-association et le modèle relationnel. Cette première phase nous a permis de mettre en oeuvre la partie théorique du cours `INGI1271 - Fichiers et bases de données`. Ensuite est arrivé la création et le remplissage de la base de données qui nous a permis d'apprendre le langage SQL. En effet, pour la plupart d'entre nous, ce langage nous était inconnu. Cette partie met en oeuvre la partie pratique du cours. Ce projet nous a ausssi permis de dépasser le cadre du cours pour apprendre le langage PHP ou encore utiliser les logiciels tels que `PHP/Myadmin` et `MySQL`.

Ensuite, la gestion d'un aéroport nous a permis de réfléchir sur de nombreux aspects différents. Que ce soit la gestion des aspects techniques, du personnel ou encore celle des clients, chacune d'entre elles imposent des contraintes qui compliquent rapidement la gestion de l'aéroport. Nous avons du faire des choix entre les points importants et les détails. Nous nous sommes fixés une limite car il est possible d'améliorer les fonctionnalités du site ainsi que le nombre de requêtes. Nous pensons être arrivé à un bon compromis entre réalisme et créativité.