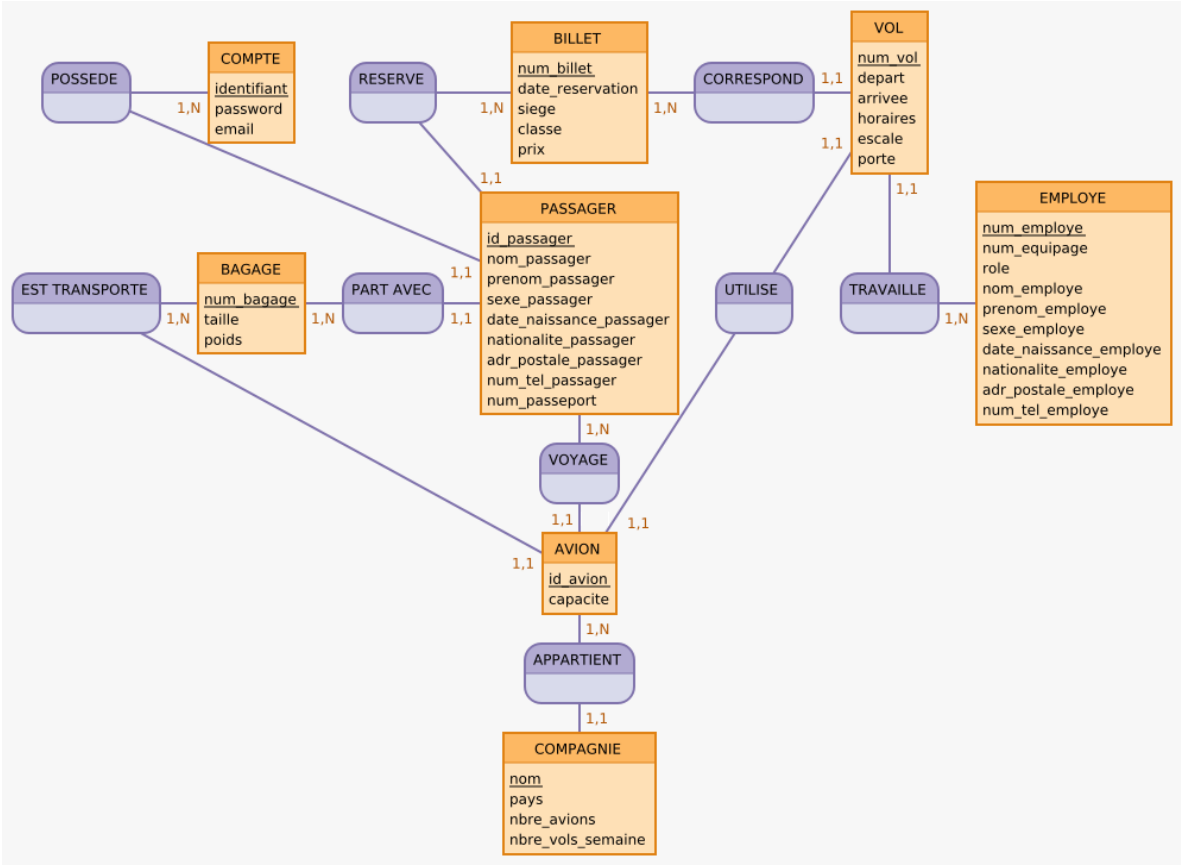
Rapport projet BD

BONNET Ludivine – LAMOUR Lauriane

Dans le cadre de notre projet, nous devions choisir un thème pour faire une base de données. Nous avions alors choisi de créer une base de données d'un aéroport donné. En premier lieu, nous allons représenter les enregistrements de la base de données sous forme de modèles logiques que sont le modèle entité-association et le modèle relationnel qui en dérive.

Modèle entité-association



Légende :
Souligné : clé primaire

Pour constituer notre base de donnée, nous avons distingué les huit classes d'entités (objets,événements, personnes,etc.. identifiables sans ambiguïté) : VOL, BILLET, COMPTE, BAGAGE, PASSAGER, EMPLOYE , AVION et COMPAGNIE. Ainsi, dans le cas de la création du site web, il sera possible à l'utilisateur de se créer un compte afin de réserver un billet pour un vol. Il lui sera également possible de consulter les vols disponibles et à venir. Dans le cas d'une utilisation de la base de données par des employés de l'aéroport qualifiés pour cela, ils leur sera possible de gérer les données privées des employés et des passagers, ainsi que l'ensemble des données importantes à la bonne gestion des vols.

Par la suite, on désignera par entité une classe d'entités.

Chacune de ces entités a ses propriétés spécifiques que sont les attributs de l'entité. Pour les employés et les passagers, nous avons fait le choix d'en faire deux entités distinctes. En effet, nous aurions pu créer une entité PERSONNE dans laquelle nous aurions mis les attributs en commun entre passager et employé, sauf e passager n'a pas d'attribut particulier qui se distingue d'une personne. Or une entité comporte au minimum un attribut selon le modèle E/A

donc cela n'était pas cohérent de créer une table personne et une table passager sans attributs. Nous pouvions stocker dans l'entité AVION le nom de la compagnie comme étant un attribut mais nous devions stocker également d'autres propriétés de la compagnie comme son pays ou encore son nombre d'avions. C'est pour cette raison que nous avons décidé de créer une entité COMPAGNIE qui contient comme attributs les informations concernant la compagnie qui sont nécessaires de garder en mémoire.

Chaque occurrence de la base de données doit pouvoir être repérée de manière unique et sans ambiguïté, pour être distinguée de toutes les autres. C'est pour cela que l'on a attribué une clé primaire pour chaque entité qui identifie sans ambiguïté l'entité. Par exemple, le numéro de billet est unique pour chaque billet et permet d'identifier le billet, c'est alors une clé primaire que l'on a représenté en soulignant.

À chaque fois qu'il existe un lien entre deux entités, nous l'avons représenté par une association : par exemple un passager possède un compte et réserve un billet. Par contre, lorsqu'il n'existe aucun lien entre deux entités, cela est traduit par l'absence d'association entre les deux entités correspondantes : entre un employé et un billet il n'y a aucun lien c'est donc pour cela qu'il n'existe pas d'association entre l'entité EMPLOYE et BILLET.
La cardinalité indiquée sur chaque association se lit de la manière suivante :

• min du côté A correspond à la réponse à la question « combien de fois au moins une entité de A est relié à une entité de B »
• max du côté A correspond à la réponse à la question « combien de fois au plus une entité de A est relié à une entité de B »

On doit également se poser ces questions dans l'autre sens du côté de B pour obtenir min et max du côté de B.

Ceci est un schéma conceptuel, on va donc le transformer en un modèle relationnel.

Modèle relationnel

COMPTE (identifiant, password, email, id_passager)
- COMPTE.id_passager → PASSAGER.id_passager

BILLET (num_billet, date_reservation, siege, classe, prix, num_vol, id_passager)
- BILLET.num_vol → VOL.num_vol
- BILLET.id_passager → PASSAGER.id_passager

VOL (num_vol, depart, arrivee, horaires, escale, porte, id_avion)
- VOL.id_avion → AVION.id_avion

BAGAGE (num_bagage, taille, poids, id_passager, id_avion)
- BAGAGE.id_passager → PASSAGER.id_passager
- BAGAGE.id_avion → AVION.id_passager

PASSAGER (id_passager, nom_passager, prenom_passager, sexe_passager, date_naissance_passager, nationalite_passager, adr_postale_passager, num_tel_passager, num_passeport, id_avion)
- PASSAGER.id_avion → AVION.id_avion

EMPLOYE (num_employe, num_equipage, role, nom_employe, prenom_employe, sexe_employe, date_naissance_employe, nationalite_employe, adr_postale_employe, num_tel_employe, num_vol)
- EMPLOYE.num_vol → VOL.num_vol

AVION (id_avion, capacite,nom)
- AVION.nom → COMPAGNIE.nom

COMPAGNIE (nom, pays, nbre_avions, nbre_vols_semaine)

Légende
En gras : clé primaire
Souligné : clé étrangère

Le modèle relationnel est composé de huit tables d'entités. Nous avons fait le choix de fusionner les tables d'associations dans les tables des entités participant à l'association. La fusion est visible par la présence de la clé primaire d'une entité dans la table de l'autre entité associée.
Ainsi, la table COMPTE contient id_passager référençant la classe d'entité PASSAGER afin de représenter l'association POSSEDE.
De même, id_avion référence la classe d'entité AVION dans la table VOL pour représenter l'association UTILISE.
La table BILLET contient num_vol et id_passager pour référencer les classes d'entité VOL et PASSAGER pour les associations respectives CORRESPOND et RESERVE.
La table BAGAGE contient les clés primaires de PASSAGER et AVION, respectivement id_passager et id_avion pour représenter les associations respectives PART AVEC et EST TRANSPORTE.
La table PASSAGER contient la clé primaire de AVION, id_avion pour représenter l'association VOYAGE.
La table EMPLOYE contient num_vol de la classe d'entité VOL pour représenter l'association TRAVAILLE.
Dans la table AVION, nom référence la classe d'entité COMPAGNIE pour l'association APPARTIENT.

Schéma relationnel (corrigé)

- **Compagnie** (nom, pays, nbre_avions, nbre_vols_semaine)
- **Avion** (id_avion, capacite, nom)

  Avion.nom référence la clé primaire Compagnie.nom
- **Vol** (num_vol, depart, arrivee, date_vol, horaires, escale, porte, id_avion)

  Vol.id_avion référence la clé primaire Avion.id_avion
- **Passager** (id_passager, nom_passager, prenom_passager, sexe_passager, date_naissance_passager, nationalite_passager, adr_postale_passager, num_tel_passager, num_passeport)
- **Employe** (num_employe, num_equipage, role, nom_employe, prenom_employe, sexe_employe, date_naissance_employe, nationalite_employe, adr_postale_employe , num_tel_employe)
- **Compte** (identifiant, password, email, id_passager)

  Compte.id_passager référence la clé primaire Passager.id_passager
- **Billet** (num_billet, date_reservation, siege, classe, prix, num_vol, id_passager)

  Billet.num_vol réfèrence la clé primaire Vol.num_vol

  Billet.id_passager référence la clé primaire Passager.id_passager
- **Bagage** (num_bagage, taille, poids, num_billet)

  Bagage.num_billet référence la clé primaire Billet.num_billet
- **Travaille** (num_vol, num_employe)

  Travaille.num_vol référence Vol.num_vol

  Travaille.num_employe référence Employe.num_employe

Création des tables et insertions

La base de données est composé de 9 tables correspondantes à celles décrites dans le schéma relationnel rappelé ci-dessus.

Pour remplir notre base de données, nous sommes partis de la table Compagnie et nous avons décidé d'en insérer 28. En découle la table Avion dans laquelle a été inséré 100 instances d'avions. Nous avons également fait le choix d'insérer 122 instances de vol dans la table Vol. Sur chaque vol, on décompte 7 employés soit 98 employés au total. Le nombre de passagers est variable en fonction des vols mais le total est de 1000 passagers. Ainsi la table Bagage contient 1000 instances de bagages. De même pour la table Compte. Enfin la table Billet contient 10000 insertions. Pour pouvoir insérer autant de données dans la table Billet, nous avons utilisé un site générateur de données *generatedate.com* dans lequel on indique le type de données à insérer. Celui-ci génére automatiquement 100 données, nous avons donc répéter cette opération 100 fois pour obtenir 10000 billets dans la table Billet.

Création des vues

Nous avons créé les vues suivantes :

- une vue sur les vols *gestion_vols* : on récupère les attributs de la table Vol ainsi que les attributs *id_avion* et *capacite* de l'avion associé au vol et enfin l'attribut *nom* de la compagnie associé à l'avion.
- une vue *nb_passager_vol* comptant le nombre de passagers par vol.
- une vue *gestion_billet* contenant les attributs de Billet, ainsi que les attributs *num_vol*, *depart*, *arrivee*, *date_vol* et *horaires* de Vol et l'identifiant du Passager.

Création des utilisateurs

Nous avons créé les utilisateurs suivants :

- un utilisateur gérant les vols : il a accès à la vue *gestion_vols* avec le droit de sélection pour lui donner une vue globale des vols. Il a également accès à la vue *nb_passager_vol* pour vérifier que le nombre de passagers ne dépassent pas la capacité de l'avion avec le droit de sélection. Enfin nous lui avons donné les droits SELECT, INSERT, UPDATE et DELETE sur les tables Vol, Avion et Compagnie afin de pouvoir les modifier selon sa convenance.
- un utilisateur gérant les billets : nous lui avons donné accès à la vue *gestion_billet* avec le droit SELECT pour avoir une vue globale des billets. Il a également un droit de sélection sur la table Vol afin de pouvoir connaître les vols en cours. Enfin nous lui avons donné les droits SELECT, INSERT, UPDATE et DELETE sur la table Billet afin de modifier les instances si besoin.
- un utilisateur gérant les passagers et les employes : nous lui avons donné les droits SELECT, INSERT, UPDATE et DELETE sur les tables Employe, Travaille et le droit de sélection sur la table Vol pour gérer les allers-retours de chaque employé. Il a également les droits SELECT, INSERT, UPDATE et DELETEsur les tables Passager et Bagage.
- un utilisateur *administrateur* ayant tous les droits sur les tables de la base de données avec la possibilité de donner ses droits.

Requêtes

Requêtes d'interrogation

Tous les vols à une date précise

Nous demandons à afficher toutes les lignes de la table Vol dont la date de vol correspond à celle indiquée.

Tous les billets correspondant à un compte

Nous demandons d'afficher tous les billets correspondant au numéro de passager d'un compte.

Tous les vols au départ de Paris

Nous demandons d'afficher toutes les lignes de la table Vol où depart correspond à Paris.

Tous les billets au départ de Paris

Nous demandons d'afficher les informations ainsi que le numéro de vol et le nom de la compagnie de tous les billets au départ de Paris ordonnés par arrivée, date de vol et prix.

Le nombre d'avions utilisés par compagnie

Nous demandons, pour chaque compagnie, de compter les lignes correspondant à son nom dans la table Avion afin de connaître le nombre d'avion utilisé.

Le vol et le nombre de bagages dont le poids est supérieur à un seuil donné dans chaque vol

Nous demandons pour chaque vol, d'afficher son numéro et de compter chaque ligne dans la table Bagage correspondant au numéro de vol où la condition suivante est respecter : le poids doit être supérieur à 25 kg. On regroupe le tout par vol.

La moyenne de prix d'un billet pour une destination donnée

Nous demandons pour une destination donnée, de trouver toutes les lignes dont le numéro de vol correspond à l'arrivée donnée et de faire la moyenne de prix.

Tous les vols pour une destination avec un prix en dessous de la moyenne

Nous demandons pour une destination donnée, de trouver tous les vols en direction de cette arrivée et dont le prix du billet est inférieur à la moyenne des prix de tous les vols pour cette destination (requête précédente).

Tous les vols où il y a au moins un passager vietnamien

Nous demandons d'afficher toutes les lignes de la table Vol où il existe au moins un passager vietnamien.

Le vol ayant le plus bas prix pour une destination donnée

Nous demandons d'afficher la ligne correspondant au vol où le prix du billet est le moins chère pour une destination donnée.

Le classement des destinations selon leur popularité

Nous demandons d'afficher les vols du plus populaire au moins populaire selon le nombre de billets vendus.

La destination rapportant le plus d'argent

Nous demandons d'afficher la destination où la somme totale du prix des billets correspond au maximum d'argent sur toutes les destinations.

La destination la plus prisé

Nous demandons d'afficher la destination où le nombre de billets vendus pour cette destination correspond au maximum de billets vendus.

Le pourcentage d'employés femmes par vol

Nous demandons de calculer pour chaque vol, le pourcentage d'employés femmes en comptant le nombre d'employés femmes et le nombre total d'employés par vol et en faisant le rapport.

Le nombre de mineurs par vol

Nous demandons de calculer pour chaque vol, le nombre de mineurs en calculant l'âge de chaque passager grâce à la récupération de leur année de naissance puis à la soustraction de cette année 2019 et de leur année de naissance.

Les vols où le nombre de billets vendus dépasse le nombre de places dans l'avion

Nous demandons d'afficher le numéro de vol, sa capacité et le nombre de billets vendus des vols en testant grâce à la vue *nb_passager_vol* si le nombre de passagers est supérieur au nombre de places dans l'avion correspondant au vol.

Requêtes de modifications

La suppression d'un vol

Nous demandons de supprimer un vol en supprimant les lignes des tables contenant le numéro du vol en clé étrangère puis l'on supprime le vol.

La mise à jour d'un mot de passe

Nous demandons de remplacer le mot de passe d'un compte donné par celui précisé dans la requête de modification.

Site internet et fonctionnalités

Consultations des horaires de vols

Il y a 2 possibiltés de vol : un vol au départ de Paris, ou un vol qui a pour destination Paris. Par conséquent, nous avons séparés ceci en 2 pages distincts : une page pour rechercher un vol ayant pour destination Paris ainsi qu'une autre page pour rechercher un vol ayant pour départ Paris. Ces 2 pages se présentent sous forme de formulaire : pour choisir la destination ou le départ, il y a un menu déroulant pour choisir le départ ou la destination parmi les vols disponibles enregistrés dans la base de données. Dans le code html du formulaire, nous avons insérer une balise php dans laquelle nous réalisons une requête SQL qui indique tous les départs ou destinations dans la liste déroulante (par exemple **SELECT depart FROM Vol WHERE arrivee LIKE 'Paris' GROUP BY depart ORDER BY depart** dans le cas où l'on souhaite choisir un vol ayant pour destination Paris).

Ainsi, une fois que l'utilisateur a cliqué sur le bouton "Trouver", il atterrit sur une page dans laquelle s'affiche tous les résultats de vols correspondants au départ et à la destination recherchés, avec leur date de vol et leurs horaires de vol.

Annuaire des compagnies et des destinations

- Annuaire des compagnies

Dans cette page, l'utilisateur peut visualiser la liste des compagnies aériennes dans l'aéroport. Celles-ci sont rangées par ordre alphabétique et correspondent à la requête suivante :

```
SELECT nom FROM Compagnie ORDER BY nom
```

- Annuaire des destinations

Dans cette page, l'utilisateur peut visualiser l'ensemble des destinations disponibles au départ de l'aéroport de Paris. Celles-ci sont également rangées par ordre alphabétique et correspondent à la requête suivante :

```
SELECT arrivee FROM Vol WHERE depart LIKE 'Paris' GROUP BY arrivee ORDER BY
arrivee
```

Classement des destinations

Dans cette page, l'utilisateur a la possibilité de savoir les destinations les plus touristiques, c'est à dire celles pour lesquelles le plus grand nombre de billets a été acheté. Les destinations sont classées du plus populaire au moins populaire, ce qui correspond à la requête suivante :

```
SELECT arrivee
FROM(
  SELECT arrivee,COUNT(num_billet) as Nb_billets
  FROM Vol,Billet
  WHERE Vol.num_vol = Billet.num_vol
  AND Vol.arrivee NOT LIKE "Paris"
  GROUP BY arrivee
) as NPPV
ORDER BY Nb_billets DESC;
```

Consultation des billets les moins chers

L'utilisateur a la possibilité de consulter les billets les moins chers parmi ceux qui ne sont pas encore vendus (c'est-à-dire où l'attribut id_passager est égale à NULL). La requête recherche le billet le moins cher pour chaque vol étant donné un départ et une destination.

Connexion et inscription des passagers

Les fichiers correspondant à cette partie se trouvent dans le dossier **connexion_passager**. Pour la connexion d'un passager, nous avons créer un formulaire afin que l'utilisateur puisse entrer les informations propres à son compte (identifiant et mot de passe). Ensuite, il y a une vérification des données en interrogeant la base de données avec un select sur la table Compte avec une restriction. Si les données entrées sont erronées, l'utilisateur est renvoyé vers la page de connexion contenant une erreur. Sinon, il arrive sur une page indiquant que les vérifications ont réussi. Après il arrive sur une page correspondant à ses données dans la table Billet et la table Bagage. Si l'utilisateur a oublié son mot de passe, il a la possibilité de le modifier en cliquant sur le bouton "Mot de passe oublié". Cela le mène vers un formulaire où il doit entrer son identifiant et son email. Puis nous interrogions la base de donnée afin de vérifier les informations grâce à la table Compte. Si il y a une erreur, nous renvoyons l'utilisateur vers le formulaire avec une erreur sinon, nous l'envoyons vers un formulaire lui permettant de changer son mot de passe.

Pour l'inscription, l'utilisateur arrive sur un formulaire qu'il doit remplir avec ses informations personnelles. Des erreurs sont levés si l'identifiant, l'email ou le numéro de passeport entré existe déjà. Sinon le nouveau passager est inséré dans la table Passager et Compte à l'aide de requêtes d'insertion.

Connexion et interface des employés

Les fichiers correspondant à cette partie se trouvent dans le dossier **connexion_employe**, **ing_vol**, **gestionnaire_billet** et **gestionnaire_humains**. Les employés sont représentés par les utilisateurs de la base de données : ing.op.vol, gestionnaire.billets et gestionnaire.humains. La connexion des employés se fait à l'aide d'un formulaire. Les vérifications se font grâce à *new mysqli* qui renvoie une erreur si les données sont erronées. L'erreur renvoie sur le page de connexion.

Pour chaque employé, nous avons créer une interface d'utilisation de la base de données leur permettant d'insérer, de supprimer, de mettre à jour et de sélectionner des données dans celle-ci. Tout d'abord, ils sont confrontés à un menu de choix d'action : insertion, supression, sélection ou mise à jour. Ensuite, selon les droits qu'ils ont sur les tables de la base de données, la page sur laquelle ils sont envoyés varie. Par exemple, si un employé possède un droit d'insertion sur plusieurs tables, il va être emmenés vers une page de choix de la table dans laquelle il veut faire une insertion, sinon il sera emmené directement vers un formulaire d'insertion vers la seule table dans laquelle il peut insérer. Pour l'action de sélection, selon le nombre de tables auquel l'employé avait accès avec ce droit, nous avons créer mécanisme de filtrage afin de permettre d'affiner la recherche sur ces tables là.

Réserver un vol

Les fichiers correspondant à cette partie se trouve dans le dossier **connexion_passager**. Sur l'accueil, un bouton de réservation permet d'accéder au formulaire de connexion du passager. Arrivé sur la page de son compte, il peut réserver un billet pour un vol en appuyant sur un bouton qui va le rediriger vers une page de choix de la destination. Puis, il est emmené sur une page donnant tous les billets dont la date de réservation est à null selon la destination grâce à une requête d'interrogation. Un menu de choix se trouve sur cette même page avec un menu déroulant contenant les numéros de billet affichés à l'écran. Le passager remplit ce formulaire et grâce à une requête de mise à jour de la date de réservation et de l'id_passager du billet, il retrouve son billet réservé dans les billets affichés sur sa page de compte.

Feuille de style

Nous avons insérer une image en fond pour chaque page, pour cela nous avons pris des images qui sont libres de droit sur le site *https://pixabay.com/fr/images/*. Sur la page accueil, pour avoir un effet de pression de boutons, nous avons utilisé la pseudo-classe :active qui indique les caractéristiques une fois que le bouton est pressé, dans ces caractéristiques, nous avons notamment changé la couleur de fond du bouton une fois que celui-ci est pressé. Pour les pages correspondant aux actions des passagers et à ceux des employés, nous avons utilisé une image d'un bouton pour les envoi de formulaires et les liens hypertextes.