INFO0304

Constitution d'une Base de données pour la gestion de Bateaux

THEBAULT Antoine
BOISSIER Sébastien
Dirigé par :
BEAUJET Béatrice

Sommaire

Introduction

Problématique

Afin de gérer les bateaux de ses adhérents, l'association « OMONBATÔÔ » souhaite créer une base de données permettant à ses adhérents de consulter facilement les entretiens à effectuer sur un véhicule et de pouvoir planifier des sorties en mer entre membres.

Celle-ci serait liée à un site Internet permettant de leur procurer facilement l'état de leurs bateaux et les entretiens à réaliser. Leur inscription serait gérée par l'Hôtesse de l'association, tandis que les vérifications et la gestion des notifications seraient appliquées par un Administrateur.

De nombreuses informations doivent être manipulées concernant des bateaux. Que ce soit au niveau mécanique ou de la sécurité, il est essentiel d'être vigilant car la moindre erreur peut avoir des répercussions directes (usure des pièces, pannes, accidents), d'autant que certaines maintenances doivent être effectuées périodiquement et obligatoirement.

Nous utiliserons pour la constitution de la base la méthode Merise, permettant de plus facilement récupérer un ensemble cohérent, non-redondant et avec des relations en $3^{ème}$ forme normale.

Recherches

La création d'une base de données autour de bateaux nécessite de renseigner différents éléments (pièces, équipements) et caractéristiques. La première étape de la constitution du dossier fut donc de rassembler un maximum d'éléments pouvant nous aider dans la constitution de la base de bateaux.

Outre les documents fournis par l'université (Voir « Bibliographie »), nous avons visité divers sites d'achat/vente de pièces et équipements pour bateaux. Ces derniers comportent des listes complètes d'éléments présents, à entretenir ou optionnels, ainsi que des FAQ permettant de mieux comprendre leur fonctionnement.

Cahier des charges

La base de donnée devra répondre aux conditions établies par l'association, ainsi que de permettre son utilisation future.

Pour ses non-utilisateurs :

Chaque non-utilisateur pourra, suite à une demande d'inscription validée à l'association, être inscrit sur la base de donnée par l'hôtesse d'accueil.

Pour ses utilisateurs :

Chaque utilisateur pourra proposer son ou ses bateau(x), ainsi que les sorties liées à ses véhicules. C'est l'Administrateur de la base de données qui se chargera ensuite, ou non, d'inclure les modifications.

L'utilisateur, adhérent par définition, aura accès à la liste des sorties et pourra demander à s'inscrire à une ou plusieurs sortie(s).

La base de donnée permettra le suivi de la maintenance des bateaux (Equipements, Pièces) de l'adhérent, les alertant sur leur état et les entretiens à effectuer.

Pour faciliter la commande de pièces, l'utilisateur pourra rentrer en contact avec des fournisseurs inscrits par l'Administrateur sur la base de données. Il commandera lui-même, selon ses besoins, les pièces auprès de ces derniers.

Pour l'Administrateur :

En cas d'entretien d'un bateau, l'Administrateur pourra suite à la demande de l'adhérent mettre à jour la base de données.

L'Administrateur validera aussi les demandes d'inscription de bateaux et de sorties par les utilisateurs, puis les ajoutera dans la base de données.

Pour l'Hôtesse d'Accueil :

L'Hôtesse pourra éditer la table pour ajouter, modifier ou supprimer des adhérents (Il faut payer tous les ans pour rester dans l'association ?)

Pour les bateaux :

Un bateau est constitué de nombreuses pièces et équipements, ainsi que d'un moteur. L'utilisateur pourra visualiser l'état de tous les composants et accéder à toutes les informations nécessaires à leur entretien.

Un bateau pourra être partagé par plusieurs utilisateurs (Hors du cadre des sorties).

Matrice de Flux

	Adhérent	Administrateur	Secrétaire	Visiteur	Fournisseur
Adhérent	S'inscrit à des expéditions	Envoie le formulaire de son bateau Propose une sortie Met à jour les pièces de son bateau			Contacte
Administrateur	Vérifie les bateaux Gère la mise en ligne				Ajoute à la liste
Secrétaire	Valide inscription			Inscrit	
Visiteur			S'inscrit auprès de		
Fournisseur	Livre les pièces				

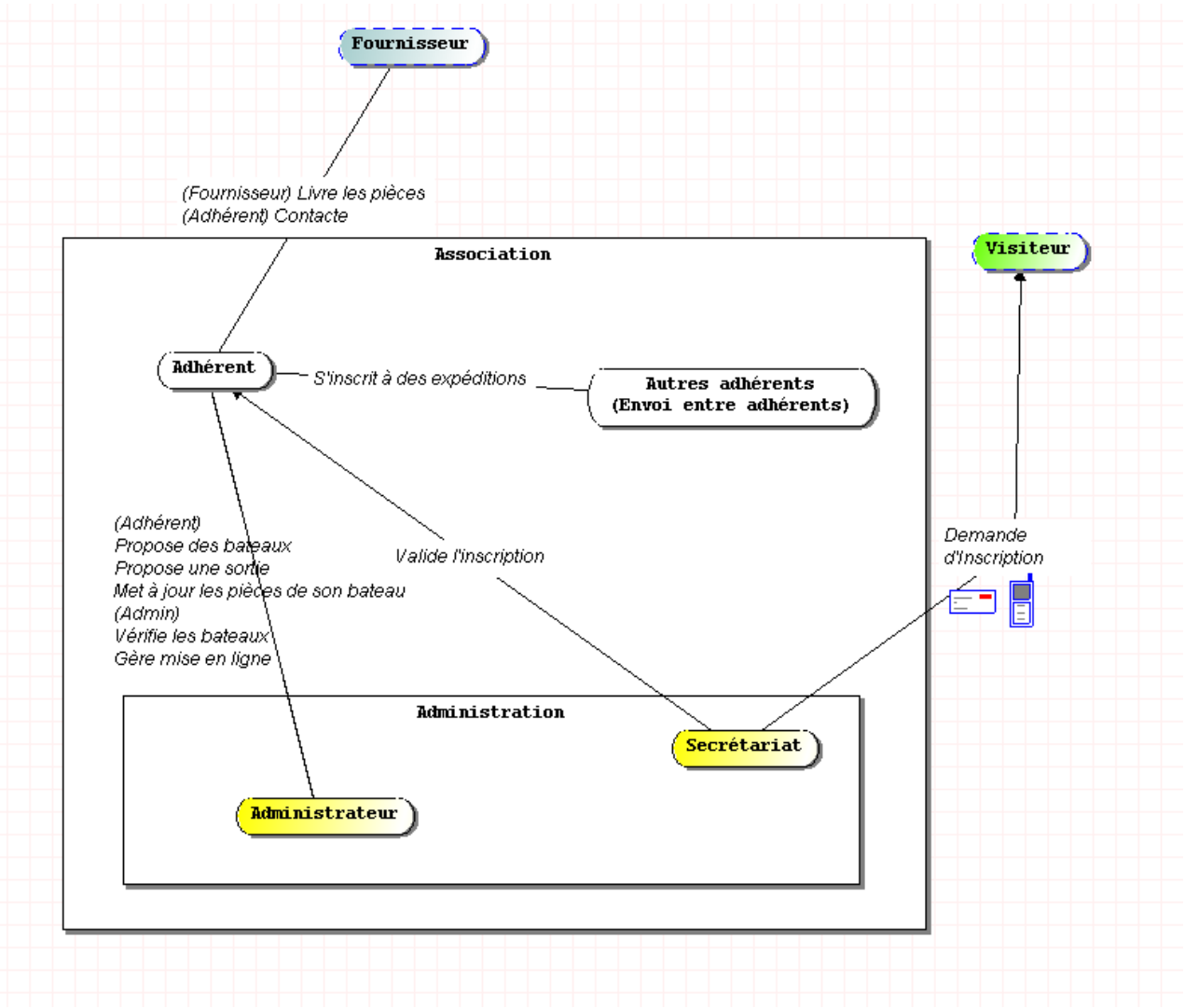
Une Matrice de Flux permet de visualiser les échanges entre les différents acteurs pouvant agir sur la base de données.

Nous avons ici *5* **acteurs** :

- L'Adhérent (Une fois inscrit sur la base de données)
- L'Administrateur
- Le/La Secrétaire (« L'Hôtesse d'accueil » réduit en un seul mot)
- Le Visiteur : Simple consultant externe, pas (encore) inscrit
- Fournisseur : Acteur externe, listé dans la base de données pour faciliter la liaison fournisseur-client

Diagramme de Flux

Comme la Matrice de Flux, le Diagramme de Flux illustre les relations entre les acteurs d'une base de données. Celui-ci peut donner une première idée d'éventuelles relations, attributs et éléments du site Web liés aux informations (notamment les formulaires)



Dictionnaire de données

Légende

Entités

- Attribut (ID)
- Attribut – Type - Exemple de valeur

Dictionnaire

Accident

- Id_accident – BIGINT (auto increment)
- Nb_morts – BIGINT - *5*
- Nb_bless – BIGINT - *5*
- Date_accident - TIME - 01/01/2001

Adresse

- Id_adresse – BIGINT (auto increment)
- Numero_adresse - BIGINT - 17
- Voieire - String - Place de l'esplanade

Bateau

- Id_bateau – BIGINT (auto increment)
- distance_eloignement – BIGINT – 10
    - Permet de déterminer la nouvelle catégorie du bateau
- nb_places – BIGINT - 2
- auto_videur – Boolean – true
    - Le bateau est-il un bateau auto-videur ?
- francise – Boolean – true
    - Le bateau est-il francisé ?
- hors_bord – Boolean – false
    - Le bateau est-il un hors-bord ?
- nb_mat – BIGINT - 1
- ancienne_cat – Char - 'C'
- surface_voilure – BIGINT - 10
- masse_navire – FLOAT - *500.5*
- dimension_x – FLOAT - *5*
- dimension_y – FLOAT - *5*
- dimension_z – FLOAT - *15*
- volume_coque – FLOAT - *25*
- force_vent_max – FLOAT - *25*
- hauteur_max_vagues – FLOAT - *25*
- niveau_reserve – FLOAT - *60.5*
- consommation – FLOAT - *6.23*
- niveau_carburant_max – FLOAT - *50*
- niveau_performance – FLOAT - *50*
- jauge_brut – FLOAT - *25*
- date_construction – TIME - 01/01/2011

- nom_bateau – String - Insurmontable
- niveau_huile – FLOAT - 10
- niveau_liquide_refroidissement - 5

Entretien

- id_entretien – BIGINT (auto-increment)
- date_entretien - TIME - 01/01/2018

Equipement

- id_equipement – BIGINT (auto-increment)
- q_equip_rechange – BIGINT - 0
- equip_origine – Boolean - true
- revision_periodique_equip – TIME - 0
- duree_vie_equip - TIME - 5 Years

Etat

- id_etat – BIGINT (auto-increment)
- desc_etat – String - "Bon"

Fournisseur

- id_fourni – BIGINT (auto-increment)
- nom_fourni – String - "BatiBato"
- tel_fourni - String - "06.xx.xx.xx.xx"
- mail_fourni – String - "batibato@gmail.com"

Immatriculation

- id_immatr – String - "AF-482-FF"
- date_immatr - TIME - "01/01/2015"

Marque

- id_marque – BIGINT (auto-increment)
- nom_marque - String - "Honda"

Modèle

- id_modele – BIGINT (auto-increment)
- nom_modele - String - "CV-zi-514"

Moteur

- id_moteur – BIGINT (auto-increment)
- puissance_moteur - FLOAT - 55
- horametre_moteur - FLOAT - 5621
- kilometrage - FLOAT - 100000

Pays

- id_pays – BIGINT (auto-increment)
- planete – String - Terre
- nom_pays – String - France

Permis

- id_permis – BIGINT (auto-increment)
- nom_permis - String - “C”

Piece

- id_piece – BIGINT (auto-increment)
- quantite_piece – BIGINT - 5
- q_rechange_piece – BIGINT - 0
- piece_origine – BOOLEAN - true
- revisions_periodiques_piece – TIME – 0
- Duree_vie_piece – TIME – 0

Port

- Id_port – BIGINT (auto-increment)
- Latt_port -FLOAT - 45
- Long_port - FLOAT - 58

Type Equipement

- Id_type_equip - BIGINT (auto-increment)
- Nom_type_equip - FLOAT - “Tableau de bord”

Type Pièce

- Id_type_piece - BIGINT (auto-increment)
- Nom_type_piece - FLOAT - “Jointure des cables”

Utilisateur

- Id_utilisateur – BIGINT (auto_increment)
- Tel_utilisateur – String - “03.xx.xx.xx.xx”
- Type_utilisateur - String - “admin”
- Login – String - “charlou”
- Password – String “* * * * * * *”
- Nom_utilisateur - String - “Bomier”
- Prenom_utilisateur - String - “Charles”
- Mail_utilisateur - String - “bomier.charles@gmail.com”

Ville

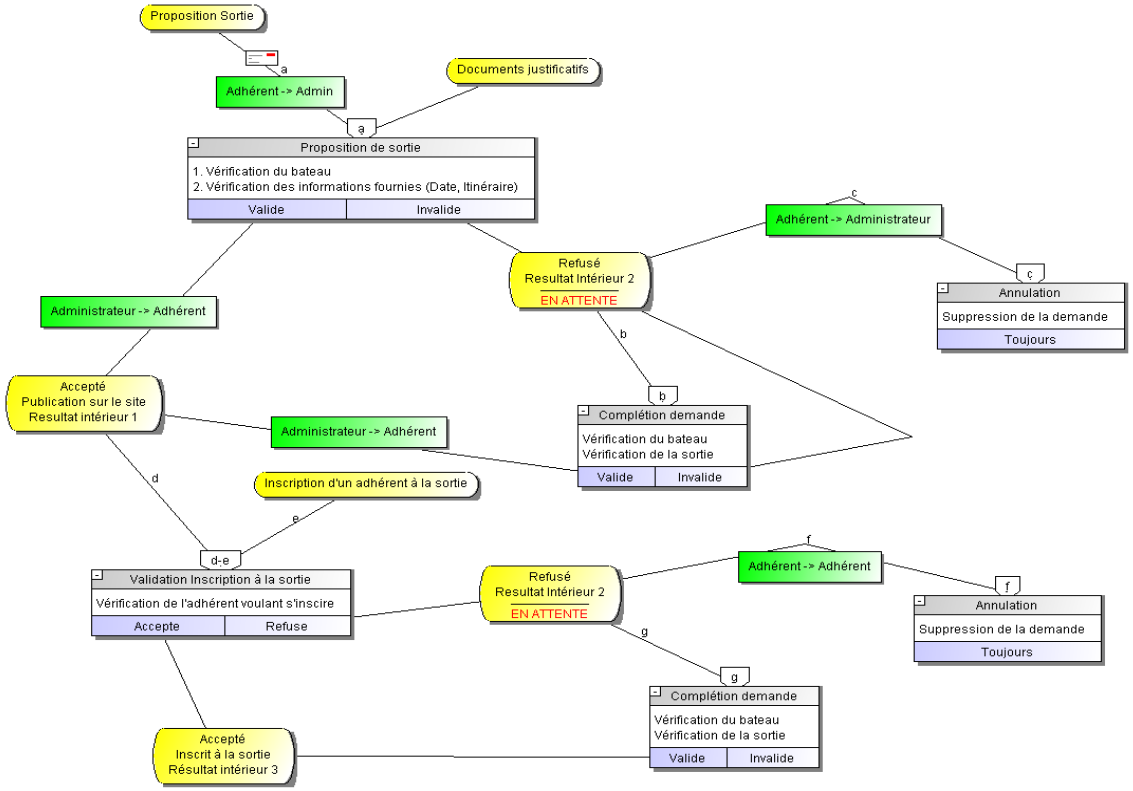
- Id_ville – BIGINT (auto_increment)
- Nom_ville - String - “Reims”
- Code_postal - BIGINT - “51100”

Voyage

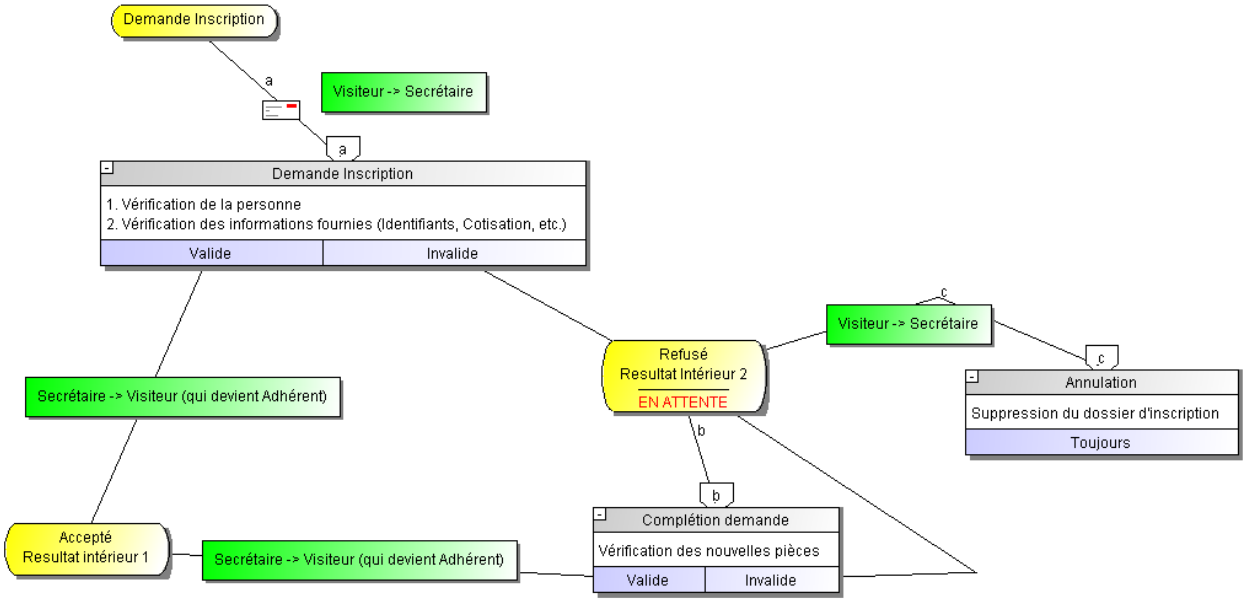
- Id_voyage – BIGINT (auto-increment)
- Cout_voyage - FLOAT - “-1”
- Date_départ - TIME - “01/02/2019:12h56”
- Date_retour – TIME - “01/02/2019:18h46”
- Participants_min - BIGINT - 2
- Participants_max - BIGINT- 5

Modèle Conceptuel de Traitement

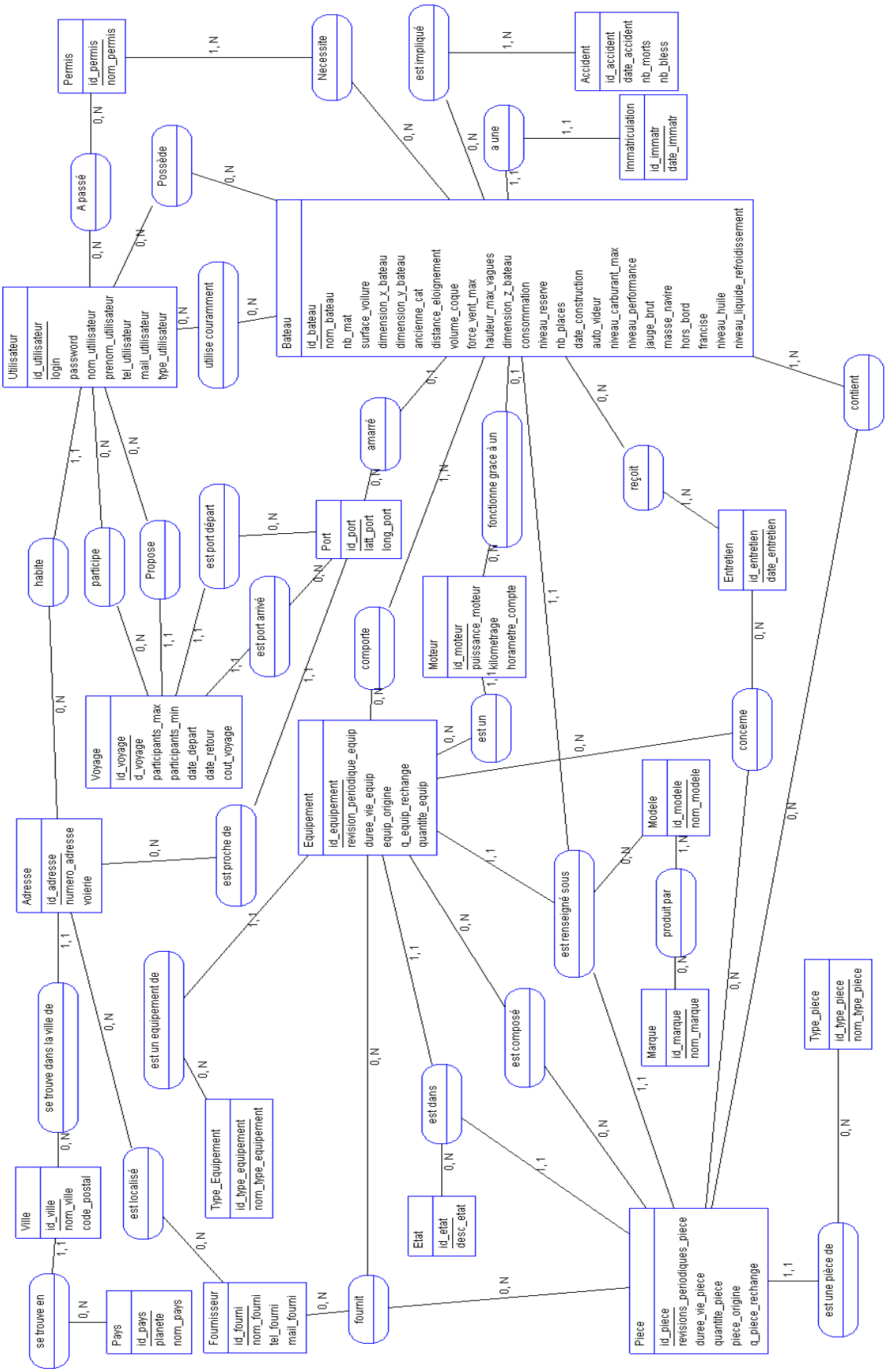
Relations Adhérent-Adhérent et Admin-Adhérent



Relations Visiteur-Adhérent-Secrétaire



Modèle Conceptuel de Données



Le modèle Conceptuel de Données permet de visualiser efficacement les différentes relations, attributs et entités d'une base de données. Les CIF (Contrainte d'Intégrité Fonctionnelle) sont ainsi plus facilement déterminées.

Le MCD, déterminé grâce au Dictionnaire de Données, donne automatiquement via le logiciel « AnalyseSI » le Modèle Relationnel.

Celui-ci s'obtient en rassemblant les Entités avec leurs attributs et les relations. Dans le cas d'une CIF (cardinalités 1-1), on dit que la relation est porteuse d'une Dépendance Fonctionnelle, important dans la normalisation de la base. L'entité porteuse de la CIF reçoit alors une clé étrangère du côté du 1 en cardinalité maximale et qui correspond à la clé primaire de l'entité de l'autre côté de la relation.

<u>Précisions sur les cardinalités :</u>

Une adresse se situe au minimum dans une seule ville, se situant dans au minimum un seul pays.

Un Fournisseur peut avoir plusieurs adresses (Franchise).

Un Port est proche d'une seule adresse de référence.

Un Utilisateur habite à une seule adresse et ne peut être sans habitation.

Un Permis permet de conduire au minimum un type de bateau.

Un Voyage doit être proposé par une et une seule personne.

Une Voyage doit avoir un et un seul port d'arrivée et un port de départ, même s'ils pointent vers le même port.

Un Bateau est amarré, ou non, dans un port.

Les pièces et équipements ont un et un seul état, ainsi qu'un seul type et un seul modèle.

Un bateau a un et un seul modèle.

Un modèle peut être une collaboration entre plusieurs marques.

Un Entretien concerne obligatoirement un et un seul bateau.

Un moteur est un, et un seul, équipement.

Une Immatriculation est associée à un seul bateau, qui ne possède qu'une seule immatriculation.

Un Accident implique obligatoirement au minimum un bateau.

Modèle Relationnel (ou Modèle Logique des Données)

Le Modèle relationnel est une liste des différentes entités et relations, ainsi que les différents attributs. Il est important car les CIF sont très rapidement mis en avant (« # » à la fin d'entités). Nous avons dans la base de données de l'association plusieurs CIF, il faudra donc faire attention lors des vérifications de normalisation de celle-ci.

Entités

Piece (id_piece, revisions_periodiques_piece, duree_vie_piece, quantite_piece, piece_origine, q_piece_rechange, #id_type_piece)

Bateau (id_bateau, nom_bateau, nb_mat, surface_voilure, dimension_x_bateau, dimension_y_bateau, ancienne_cat, distance_eloignement, volume_coque, force_vent_max, hauteur_max_vagues, dimension_z_bateau, consommation, niveau_reserve, nb_places, date_construction, auto_videur, niveau_carburant_max, niveau_performance, url_photo_Bateau, jauge_brut, masse_navire, hors_bord, francise, niveau_huile, niveau_liquide_refroidissement, #immatriculation_id_immatr, #moteur_id_moteur, #port_id_port)

Voyage (id_voyage, d_voyage, participants_max, participants_min, date_depart, date_retour, cout_voyage, #id_utilisateur, #id_port, #id_port_est_port_arrivé)

Utilisateur (id_utilisateur, login, password, type_utilisateur, nom_utilisateur, prenom_utilisateur, tel_utilisateur, mail_utilisateur, #id_adresse)

Entretien (id_entretien, date_entretien)

Adresse (id_adresse, numero_adresse, voierie, #id_ville)

Equipement (id_equipement, revision_periodique_equip, duree_vie_equip, equip_origine, q_equip_rechange, quantite_equip, #id_type_equipement)

Permis (id_permis, nom_permis)

Fournisseur (id_fourni, nom_fourni, tel_fourni, mail_fourni)

Etat (id_etat, desc_etat)

Modele (id_modele, nom_modele)

Marque (id_marque, nom_marque)

Type_piece (id_type_piece, nom_type_piece)

Type_Equipement (id_type_equipement, nom_type_equipement)

Immatriculation (id_immatr, date_immatr, #bateau_id_bateau)

Moteur (id_moteur, puissance_moteur, kilometrage, horametre_compte, #id_equipement)

Port (id_port, latt_port, long_port, #id_adresse)

Accident (id_accident, date_accident, nb_morts, nb_bless)

Ville (id_ville, nom_ville, code_postal, #id_pays)

Pays (id_pays, planete, nom_pays)

Relations

Possède (#id_utilisateur, #id_bateau)

A_passé (#id_utilisateur,#id_permis)

participe (#id_utilisateur, #id_voyage)

utilise_couramment (#id_utilisateur, #id_bateau)

Necessite (#id_bateau, #id_permis)

est_impliqué (#id_bateau, #id_accident)

contient (#id_bateau, #id_piece)

comporte (#id_bateau, #id_equipement)

reçoit (#id_bateau, #id_entretien)

concerne (#id_equipement, #id_piece, #id_entretien)

est_composé (#id_equipement, #id_piece)

produit_par (#id_modele, #id_marque)

est_renseigné_sous (#id_equipement,#id_piece, #id_bateau, #id_modele)

fournit (#id_fourni, #id_equipement, #id_piece)

est_localisé (#id_fourni, #id_adresse)

est_dans (#id_equipement, #id_piece, #id_etat)

Normalisation

Déterminer si une base de données contient des relations d'une certaine forme normale permet de vérifier sa cohérence ainsi que d'améliorer sa conception. Dans ce cas, nous étudierons notre base de données dans l'objectif d'atteindre une troisième forme normale pour toutes ses relations.

Ayant suivi la méthode Merise, notre base est forcément en troisième forme normale. Nous utiliserons donc 4 relations d'exemples : Bateau, Moteur, Equipement et Utilisateur.

Soit les relations suivantes : **Bateau**, **Utilisateur**

Avec les dépendances fonctionnelles :

DF (Bateau) {

Id_bateau => nom_bateau, nb_mat, surface_voilure, dimension_x_bateau, dimension_y,_bateau, ancienne_cat, distance_eloignement, volume_coque, force_vent_max, hauteur_max_vagues, dimension_z_bateau, consommation, niveau_reserve, nb_places, date_construction, jauge_brut, masse_navire, hors_bord, francise, niveau_huile, niveau_liquide_refroidissement, immatriculation_id_immatr, moteur_id_moteur, port_id_port

}

DF (Utilisateur) {

Id_utilisateur => login, password, type_utilisateur, nom_utilisateur, prenom_utilisateur, tel_utilisateur, mail_utilisateur, id_adresse

}

Première forme normale

Cette étape est une étape centrale de la conception. Il s'agit de vérifier si tous les attributs des relations sont « atomiques », c'est-à-dire qu'ils correspondent bien à des types et pas des tableaux ou ensembles de données typées. Par exemple, « rue » est un attribut « Varchar », alors que « adresses » est un tableau d'adresses ce qui est par la même occasion totalement faux dans notre base de données, puisque « Adresse » est considéré comme une entité.

Deuxième forme normale

Une base de données en deuxième forme normale ne possède que des relations dont les attributs non-clés ne dépendent pas d'une partie de la clé.

Bateau : Par la fermeture transitive $(id\_bateau)^+$, on obtient bien tous les attributs de la relation. Id_bateau est donc bien la clé primaire de la relation.

Aucun attribut non-clef ne dépend d'une seule partie de la clef (C'est-à-dire, A => C avec AB la clef et C non-membre de la clef). Donc la relation, possédant des attributs atomiques, est en deuxième forme normale.

Utilisateur : Par la fermeture transitive $(id\_utilisateur)^+$, on obtient bien tous les attributs de la relation. Id_utilisateur est donc bien la clé primaire de la relation.

Aucun attribut non-clef ne dépend d'une seule partie de la clef (C'est-à-dire, A => C avec AB la clef et C non-membre de la clef). Donc la relation, possédant des attributs atomiques, est en deuxième forme normale.

Une fois que cette méthode a été appliquée à toutes les relations, on remarque que la base de données est bien en deuxième forme normale, puisque toutes ses relations le sont.

Troisième forme normale

Est en troisième forme normale une base de données possédant des relations dont les attributs non-clefs ne dépendent pas d'autres attributs non-clefs.

Toutes nos relations sont plutôt unilatérales : Les dépendances fonctionnelles ont tous la clef en tant que partie gauche. Il n'y a aucune dépendance utilisant en membre de gauche un attribut non-clef, donc les relations sont en troisième forme normale.

On notera toutefois qu'Utilisateur ne serait pas en 3NF s'il était possible d'obtenir un attribut à partir de l'adresse mail ou du numéro de téléphone.

Ce n'est cependant pas le cas, donc **notre base de données est en troisième forme normale**.

Script de création de la base de données

Le code fourni ci-joint est généré automatiquement. Il permet de définir les différentes tables de la base de données, qu'il faudra ensuite remplir.
Il a cependant été corrigé. En effet, AnalyseSi ne gère pas la taille des variables de type « Varchar » et, lorsqu'une clé est définie en tant que clé externe, n'arrive pas à trouver son type (ex : BIGINT).

Le fichier « script.sql » fourni contient le script de génération. Celui-ci est aussi consultable sur le github suivant :

https://github.com/Angom8/ohmonbateau/blob/master/script.sql

Exemples de requête

1. Obtenir tous les fournisseurs

⇨ Pour lister les fournisseurs

SQL : SELECT * FROM Fournisseur ;

AR : Fournisseur

2. Obtenir les logins des utilisateurs

⇨ Vérifier si un login n'existe pas déjà

SQL : SELECT login FROM Utilisateur ;

AR : $\prod login(Utilisateur)$

3. Obtenir les noms des bateaux

⇨ Pour les lister

SQL : SELECT nom_bateau FROM Bateau ;

AR : $\prod nom\_bateau(Bateau)$

4. Obtenir les caractéristiques des bateaux d'un utilisateur à partir de son id « 007 »

⇨ Pour afficher la page d'un ou des bateaux d'un utilisateur, à partir de son id

AR :

REP = caractéristiques = "nb_mat, surface_voilure, dimension_x_bateau, dimension_y_bateau, distance_eloignement, volume_coque, force_vent_max, hauteur_max_vagues, dimension_z_bateau, consommation, niveau_reserve, nb_places, date_construction, auto_videur, niveau_carburant_max, niveau_performance, url_photo_Bateau, jauge_brut, masse_navire, hors_bord, francise, niveau_huile, niveau_liquide_refroidissement"

$\prod REP(\prod id\_utilisateur(\sigma_{id\_user}(Utilisateur))) \blacksquare_{id\_utilisateur(} \prod id\_utilisateur(Possède)) \blacksquare_{id\_bateau} (\prod id\_bateau(Possède))$

SQL :

```
SELECT REP FROM Bateau WHERE id_bateau IN (SELECT id_bateau FROM Possède
WHERE id_user IN (SELECT id_utilisateur FROM Utilisateur WHERE
Utilisateur.id_utilisateur LIKE '007')) ;
```

5. Obtenir tous les id des bateaux amarres à un port

AR :

$$\prod_{\cdot id_{bateau}} (\prod id\_port(Port) \blacksquare_{id\_port(} \prod_{\cdot id_{port}, id_{bateau}} (Bateau)))$$

SQL :

```
SELECT id_bateau FROM Bateau NATURAL JOIN Port USING id_port;
```

6. Déterminer l'état du moteur du bateau de l'utilisateur d'id 'abc'

AR :

Cette requête sera divisée en plusieurs parties.

P1 =

$$\prod_{\cdot id_{\_utilisateur}} (\sigma_{id_{\_utilisateur}="abc"} (Utilisateur))$$

P2 =

$$Possède$$

P3 =

$$\prod_{\cdot id_{\_bateau}, id\_moteur} (Bateau)$$

P4 =

$$\prod_{\cdot id_{\_equipement}, id\_moteur} (Moteur)$$

P5 =

$$\prod_{\cdot id_{\_equipement}, id\_etat} (Equipement)$$

P6 = 6 =

$$\prod_{desc\_etat} (P1 \blacksquare_{id_{ustilisateur}} P2 \blacksquare_{id_{bateau}} P3 \blacksquare_{id_{moteur}} P4 \blacksquare_{id_{equipement}} P5 \blacksquare_{id\_etat} Etat)$$

SQL :

```
SELECT desc_etat FROM Etat WHERE id_equipement IN (SELECT id_equip FROM
Possède WHERE id_moteur IN (SELECT id_moteur FROM Moteur WHERE id_bateau
IN (SELECT id_bateau FROM Bateau WHERE id_bateau IN (SELECT id_bateau IN
Possède WHERE id_user IN(SELECT id_utilisateur FROM Utilisateur WHERE
id_utilisateur LIKE 'abc')))));
```

7. **Déterminer la date de prochaine révision du moteur du bateau de l'utilisateur d'id 'abc'**

AR :

P1 =

$$\prod_{revision\_periodique\_equip} \left(\prod_{id\_utilisateur} (\sigma_{id="abc"}(Utilisateur)) \right)$$

P2 = Possède

P3 =

$$\prod_{id_{bateau}, id\_moteur} (Bateau)$$

P4 =

$$\prod_{id\_equipement, id\_moteur} (Moteur)$$

P5 =

$$\prod_{id_{equipement}, revision\_periodique\_equip} (Equipement)$$

P6 =

$$\prod_{id\_utilisateur} (\sigma_{id="abc"}(Utilisateur))$$

P7 = Possède

P8 =

$$\prod_{id_{bateau}, id\_moteur} (Bateau)$$

P9 =

$$\prod_{id\_equipement, id\_moteur} (Moteur)$$

P10 =

$$\prod_{id_{equipement}} (Equipement)$$

P11 = Concerne

P12 = Entretien

Z1 =

$$P1 \blacksquare_{id_{ustilisateur}} P2 \blacksquare_{id_{bateau}} P3 \blacksquare_{id_{moteur}} P4 \blacksquare_{id_{equip}} P5$$

Z2 =

$$P6 \blacksquare_{id_{ustilisateur}} P7 \blacksquare_{id_{bateau}} P8 \blacksquare_{id_{moteur}} P9 \blacksquare_{id_{equipement}} P10 \blacksquare_{id_{equipement}} P11 \blacksquare_{id_{entrtien}} P12$$

Z3 =

$$\prod_{date\_entretien} (Z2)$$

Z4 = 7 = Z3 + Z1

SQL :

SELECT revision_periodique FROM Equipement WHERE id_equipement IN (SELECT id_equipement FROM est_un WHERE id_moteur IN (SELECT id_moteur FROM Moteur WHERE id_bateau IN (SELECT id_bateau FROM Possède WHERE id_utilisateur IN(SELECT id_utilisateur FROM Utilisateur WHERE id_user LIKE 'abc'))))

+

SELECT date_entretien FROM Entretien WHERE id_entretien IN (SELECT id_entretien FROM Concerne WHERE id_equipement IN (SELECT id_equipement FROM est_un WHERE id_moteur IN (SELECT id_moteur FROM Moteur WHERE id_bateau IN (SELECT id_bateau FROM Possède WHERE id_utilisateur IN(SELECT id_utilisateur FROM Utilisateur WHERE id_user LIKE 'abc')))));

8. **Obtenir les composants d'un bateau de l'utilisateur d'id 'abc'**

AR :

P1 =

$$\sigma_{is_{utilisateur}="abc"}(Utilisateur)$$

P2 = Possède

P3 = Bateau

P4 = Contient

P5 = Piece

P6 = Comporte

P7 = Equipement

P8 =

$$P1 \blacksquare_{id_{ustilisateur}} P2 \blacksquare_{id_{bateau}} P3 \blacksquare_{id_{bateau}} P4 \blacksquare_{id_{piece}} P5$$

P9 =

$$P1 \blacksquare_{id_{ustilisateur}} P2 \blacksquare_{id_{bateau}} P3 \blacksquare_{id_{bateau}} P6 \blacksquare_{id_{piece}} P7$$

P10 = 8 = P8 + P9

SQL :

SELECT id_piece FROM Piece WHERE id_piece IN (SELECT id_piece FROM Contient WHERE id_bateau IN (SELECT id_bateau FROM Bateau WHERE id_bateau IN (SELECT id_bateau FROM Possède WHERE id_utilisateur IN (SELECT id_utilisateur FROM Utilisateur WHERE id_utilisateur LIKE 'abc'))))

+

SELECT id_equipement FROM equipement WHERE id_equipement IN (SELECT id_equipement FROM Comporte WHERE id_bateau IN (SELECT id_bateau FROM Bateau WHERE id_bateau IN (SELECT id_bateau FROM Possède WHERE id_utilisateur IN (SELECT id_utilisateur FROM Utilisateur WHERE id_utilisateur LIKE 'abc'))));

Sécurité

A l'exception des mots de passe, la base de données n'hébergera pas de données sensibles. Elle comprendra toutefois des informations personnelles, accessibles à leurs ayants-droits conformément au RGPD et utilisées strictement dans le cadre de l'association, avec consentement des utilisateurs concernés.

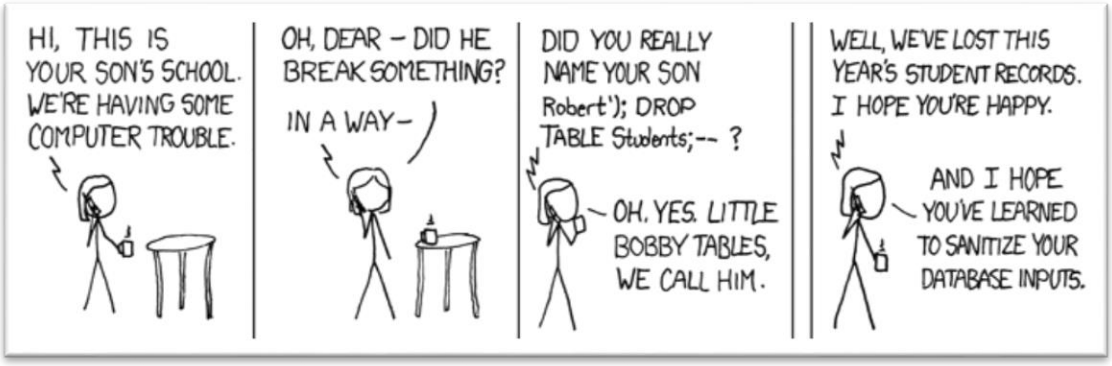
L'accès doit donc être restreint. Un utilisateur doit pouvoir accéder à ses données, et uniquement les siennes. Il ne peut voir les numéros de téléphones, adresses mail ou adresses d'autres adhérents, et vice-versa.

Seule l'administration de l'association (comprenant l'hôtesse d'accueil et l'administrateur de la base de donnée) peut créer, modifier et voir les informations stockées.

Pour des raisons de stabilité, les permissions au sein de l'administration seront divisées. L'hôtesse d'accueil ne pourra ainsi qu'avoir un rôle d'inscription de lignes dans les tables (Ajout d'utilisateurs, éventuelles modifications). L'Administrateur aura donc en charge la gestion globale et toutes les permissions afin d'éviter de mauvaises manipulations.

Ce fonctionnement sera notamment mis en pratique lors de l'utilisation de la base de données par un site WEB.

Les champs d'entrée seront également à vérifier scrupuleusement. Il faut éviter qu'un utilisateur malveillant puisse entrer du code SQL, PHP ou HTML via un form et le stocker sur la base de données.



Source : https://xkcd.com/327/

Conclusion

Grâce à la méthode Merise, nous avons pu constituer une base de données cohérentes et relativement optimisées, prête à être déployée et utilisée pour un site web par exemple, qui pourra alors la peupler.

On notera cependant quelques défauts. En effet, afin d'éviter au maximum les redondances, nous avons un surplus de relations et d'entités, rendant la base de données assez lourde. Nous avons choisi d'être plus précis, tout en permettant une certaine liberté sur la création de bateaux et de leurs pièces et équipements.

Cependant, pour ne pas non plus trop alourdir, l'entité Bateau comporte un grand nombre d'attributs. Ceci peut paraître paradoxal, mais pour traiter ces caractéristiques, il aurait fallu créer une nouvelle entité "Caractéristiques", avec une nouvelle relation pour la lier avec l'entité Bateau. Nous n'aurions également pas pu définir précisément un type (Un attribut peut être un booléen, un Varchar, un entier, un float...).

Bibliographie

Logiciels utilisés :

Suite JMerise pour Etudiant : JMerise, JMCT, JFlux

AnalyseSI (gestion du dictionnaire de données, création de MCD, du modèle relationnel, d'une première version du script)

Documentation :

Documents fournis par Mme.BEAUJET :

- 7748803_FR-VOLVO-PENTA-DB-MOTEURS-DBMOTEURS_0.pdf
- Lien vers le site http://seme.cer.free.fr/plaisance/categories-bateaux.php
- equipement_secu_plaisance_4p_DEF_Web.pdf

Sites Internet utilisés pour lister différents exemples de pièce :

- https://www.piecesbateaux.com/index.cfm
- https://www.ruedelamer.com/pieces-moteurs-bateaux/
- https://www.pescaro.fr/