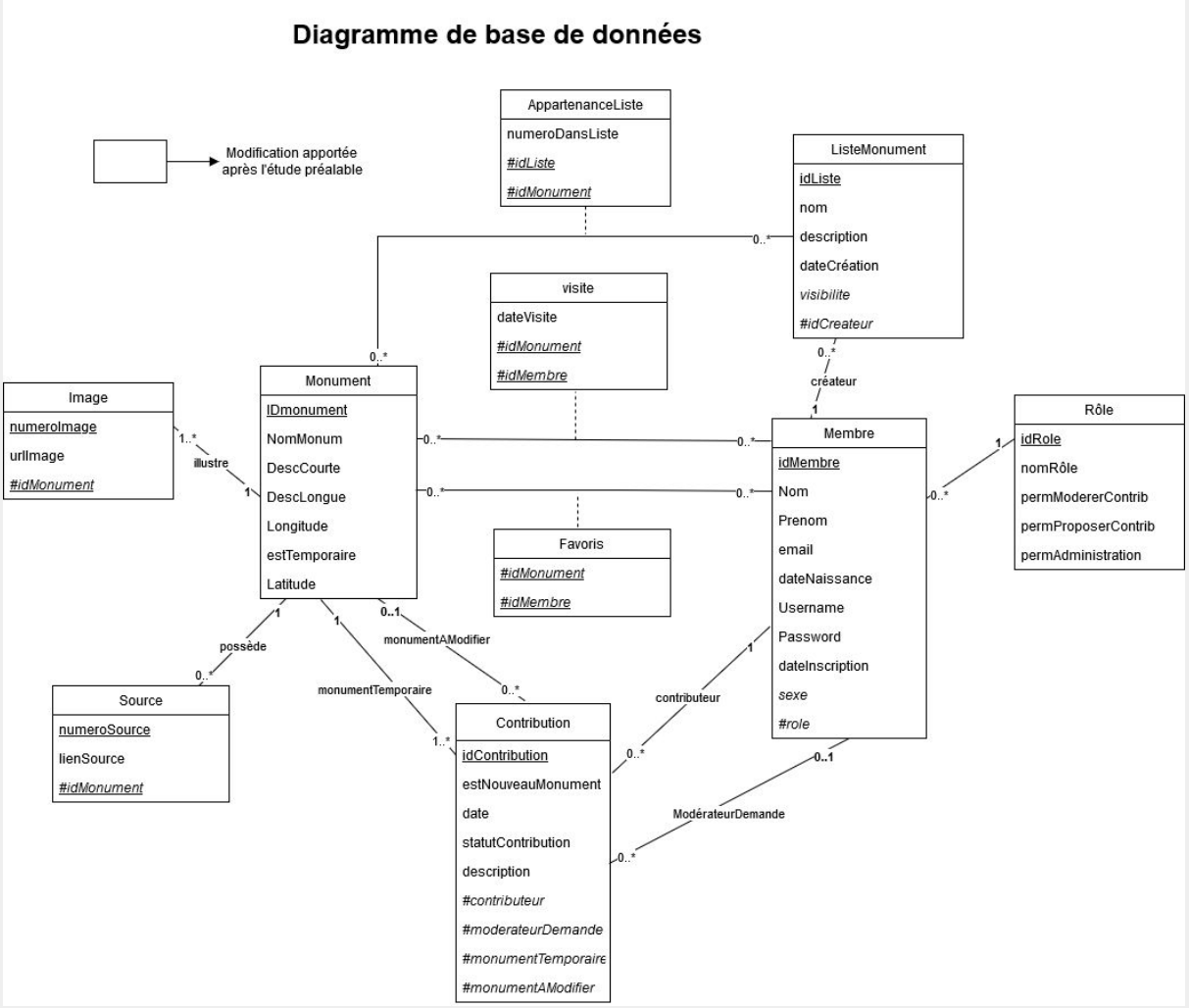
Travail sur la base de données itération 2

Diagramme de synthèse à l’issue de l'itération 1 :



Tokens pour les urls

1 - Conception / Explication

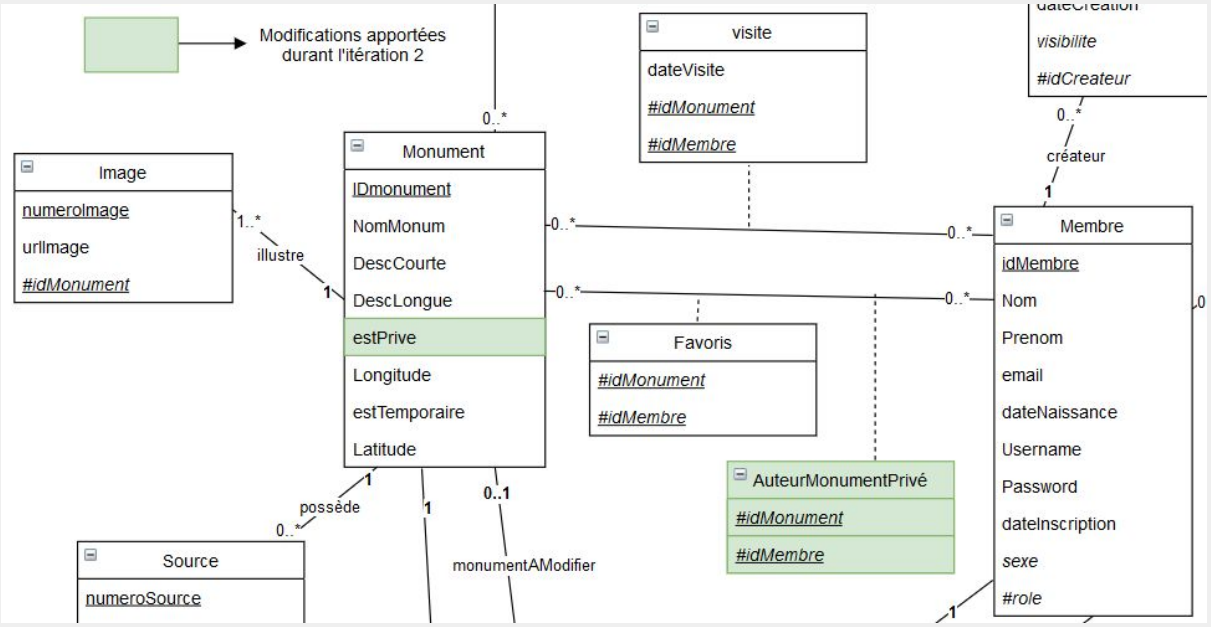
Avec la base actuelle il est impossible de générer des urls “non devinables" pour les monuments, les listes de monuments et les membres.
Pour cette raison nous avons décidé d’ajouter une colonne TOKEN de type varchar(20) aux trois tables susmentionnées qui serviront à générer des urls et liens de partage indépendamment de leur clé primaire dans la table.

Système de monument privé

1 - Conception / Explication

Il est ressorti de la soutenance de l’itération 1 la nécessité de concevoir un système de monument privé.
Un utilisateur pourra donc au moment de l’ajout d’un monument choisir de mettre celui-ci en privé. Le monument sera alors uniquement accessible par son créateur ou tout autre personne possédant un lien vers le monument.
Un monument privé est automatiquement validé : il ne rentre pas dans le système de contribution.

D’un point de vue conception il faut ajouter un attribut booléen “estPrive” à la table Monument est ajouter une table AuteurMonumentPrive pour enregistrer l’auteur de chaque monument privé (nécessaire car on ne passe pas par le système de contribution classique) :



2 - SQL

```sql
CREATE TABLE `monument` (
-- ...
-- ...
 `estPrive` BOOLEAN NOT NULL DEFAULT FALSE ,
 PRIMARY KEY (`idMonument`)
) ENGINE = InnoDB;
```

```sql
CREATE TABLE `auteurMonumentPrive` (
 `idMonument` INT UNSIGNED NOT NULL ,
 `idMembre` INT UNSIGNED NOT NULL ,
 PRIMARY KEY (`idMonument`, `idMembre`)
) ENGINE = InnoDB;
```

Système d’amis

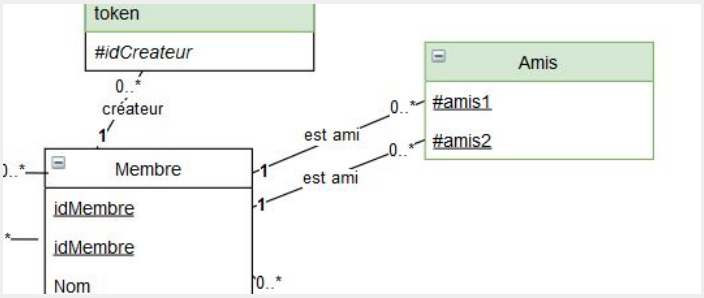
1 - Conception / Explication

Après discussions avec monsieur Guénégo il a été convenu que l’ajout d’un système d’amis dans notre projet est pertinent et présente des aspects intéressants d’un point de vue conception.
Un des objectifs est de donner la possibilité de partager une liste privée de monument à un ami et que celui-ci soit automatiquement notifié lors de l’ajout d’un nouveau monument.

D’un point de vue conception on a la table :

Amis(#Ami1, #Ami2)

On convient arbitrairement que lors d’une nouvelle relation d’amitié ami1 sera celui qui aura l’id le plus petit.



2 - SQL

```
CREATE TABLE `amis` (
 `amis1` INT UNSIGNED NOT NULL COMMENT 'on convient arbitrairement que amis1 est celui qui a l id le plus petit',
 `amis2` INT UNSIGNED NOT NULL ,
 PRIMARY KEY (`ami1`, `ami2`)
) ENGINE = InnoDB;
```

Et pour les clés étrangères :

```
ALTER TABLE amis
       ADD CONSTRAINT fk_amis_membre1
       FOREIGN KEY (ami1)
       REFERENCES membre(idMembre);

ALTER TABLE amis
```

```
        ADD CONSTRAINT fk_amis_membre2
        FOREIGN KEY (ami2)
        REFERENCES membre(idMembre);
```

3 - Trigger

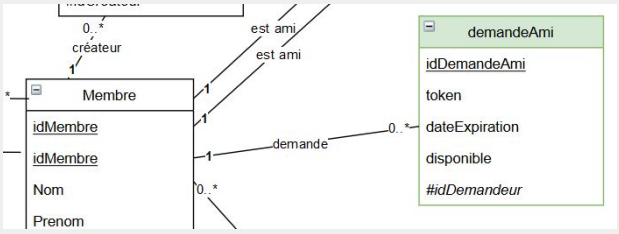
Dans le cas d'un ajout dans le mauvais ordre des amis le trigger rétablit l'ordre croissant. En assurant que l'ordre est respecté cela garantit aussi qu'il n'y a pas deux lignes qui indiqueraient la même relation d'amis mais dans deux sens.

```
DELIMITER |
CREATE OR REPLACE TRIGGER `trigger_amis_ajoutOrdre`
BEFORE INSERT ON `amis`
FOR EACH ROW
BEGIN
    DECLARE tmp INTEGER;
    IF NEW.amis1 > NEW.ami2
    THEN
        SET tmp := NEW.ami1;
        SET NEW.ami1 = NEW.ami2;
        SET NEW.ami2 = tmp;
    END IF;
    IF NEW.ami1 = NEW.ami2
    THEN
        signal sqlstate '45000' set message_text = 'On ne peut être amis avec soit-même
(ce serait triste)';
    END IF;

END |
DELIMITER ;
```

Demande d'amis

Pour ajouter un nouvel ami on génère un lien qui permet à celui qui l'ouvrira de devenir ami avec celui qui a généré le lien. Dans la base de données cela se traduit par une table : DemandeAmi(idDemandeAmi, #idDemandeur, disponible, dateExpiration, token)



2 - SQL

```sql
CREATE TABLE `demandeAmi` (
`idDemandeAmi` INT UNSIGNED NOT NULL AUTO_INCREMENT ,
`idDemandeur`  INT UNSIGNED NOT NULL ,
`dateExpiration` DATE NOT NULL DEFAULT CURRENT_TIMESTAMP ,
`token` VARCHAR(20) DEFAULT NULL,
`disponible` BOOLEAN NOT NULL DEFAULT TRUE ,
PRIMARY KEY (`idDemandeAmi`)
) ENGINE = InnoDB;
```

```sql
ALTER TABLE demandeAmi
   ADD CONSTRAINT fk_demandeAmi_demandeur
   FOREIGN KEY (idDemandeur)
   REFERENCES membre(idMembre);
```

Triggers et procédures divers :

1 - Contribution

Le trigger suivant permet, lorsque le statut d'une contribution passe à "acceptée", de vérifier qu'il y a un modérateur et que celui-ci possède la permission appropriée.

```
DELIMITER |
CREATE OR REPLACE TRIGGER `trigger_contrib_moderateur`
BEFORE UPDATE ON `contribution`
FOR EACH ROW
BEGIN

    DECLARE V_ROLE_MODO tinyint(3) ;
    DECLARE V_PERM_Moderer tinyint(1) ;

    IF ( NEW.statutContribution = 'acceptée' ) THEN
        IF NEW.moderateurDemande IS NULL THEN
                signal sqlstate '45000' set message_text = 'La contribution ne peut-être acceptée
avec le champ moderateurDemande  à null';

        ELSE

                SELECT ROLE
                        INTO V_ROLE_MODO
                        FROM MEMBRE
                        WHERE IdMembre = NEW.ModerateurDemande;

                SELECT permModererContrib
                        INTO V_PERM_Moderer
                        FROM ROLE
                        WHERE idRole = V_ROLE_MODO;

                IF (  V_PERM_Moderer = FALSE )
                THEN
                        signal sqlstate '45000' set message_text = 'L attribut moderateurDemande
correspond à un membre qui n a pas la permission de moderer les contributions';
                END IF;

        END IF;
    END IF;
END |
DELIMITER ;
```

2 - Images

Le trigger suivant donne automatiquement une valeur à numeroImage lors de l'ajout d'une image :

```
DELIMITER |
```

```
CREATE OR REPLACE TRIGGER `trigger_images_ajoutNumero`
BEFORE INSERT ON `image`
FOR EACH ROW
BEGIN
    DECLARE maxNumero INTEGER;

    IF NEW.numeroImage IS NOT NULL
    THEN
        signal sqlstate '45000' set message_text = 'Le numero est determiné
automatiquement, merci de ne rien renseigner';
    END IF;

    SELECT max(numeroImage)+1 INTO maxNumero
    from Image
    where idMonument = NEW.idMonument;

    SET NEW.numeroImage := maxNumero;

END |
DELIMITER ;
```

Après la suppression d'une image si d'autre image ont un numéro supérieur alors cela laisse un "trou". La procédure suivante permet donc de décaler les images pour régler ce problème:

```
DELIMITER |
CREATE OR REPLACE PROCEDURE p_Image_decalageNumero(IN old_idMonument INT, IN
old_numeroImage TINYINT )
BEGIN
    -- éxecuter la ligne suivante
    -- CALL p_Image_decalageNumero(deleted_monumentID, deleted_numeroImage);
       UPDATE Image
    SET numeroImage = numeroImage -1
    WHERE idMonument = old_idMonument AND numeroImage > old_numeroImage;
END |
DELIMITER ;
```

3 - Sources

La table source est très similaire à la table image. On reprend donc presque à l'identique le premier trigger de cette seconde table afin d'attribuer automatiquement un numéro à la source lors de l'ajout.

```
DELIMITER |
CREATE OR REPLACE TRIGGER `trigger_Source_ajoutNumero`
BEFORE INSERT ON `source`
FOR EACH ROW
BEGIN
    DECLARE maxNumero INTEGER;

    IF NEW.numeroSource IS NOT NULL
    THEN
        signal sqlstate '45000' set message_text = 'Le numero est determiné
automatiquement, merci de ne rien renseigner';
```

```
    END IF;

    SELECT max(numeroSource)+1 INTO maxNumero
    from Source
    where idMonument = NEW.idMonument;

    SET NEW.numeroSource := maxNumero;

END |
DELIMITER ;
```

Synthèse :

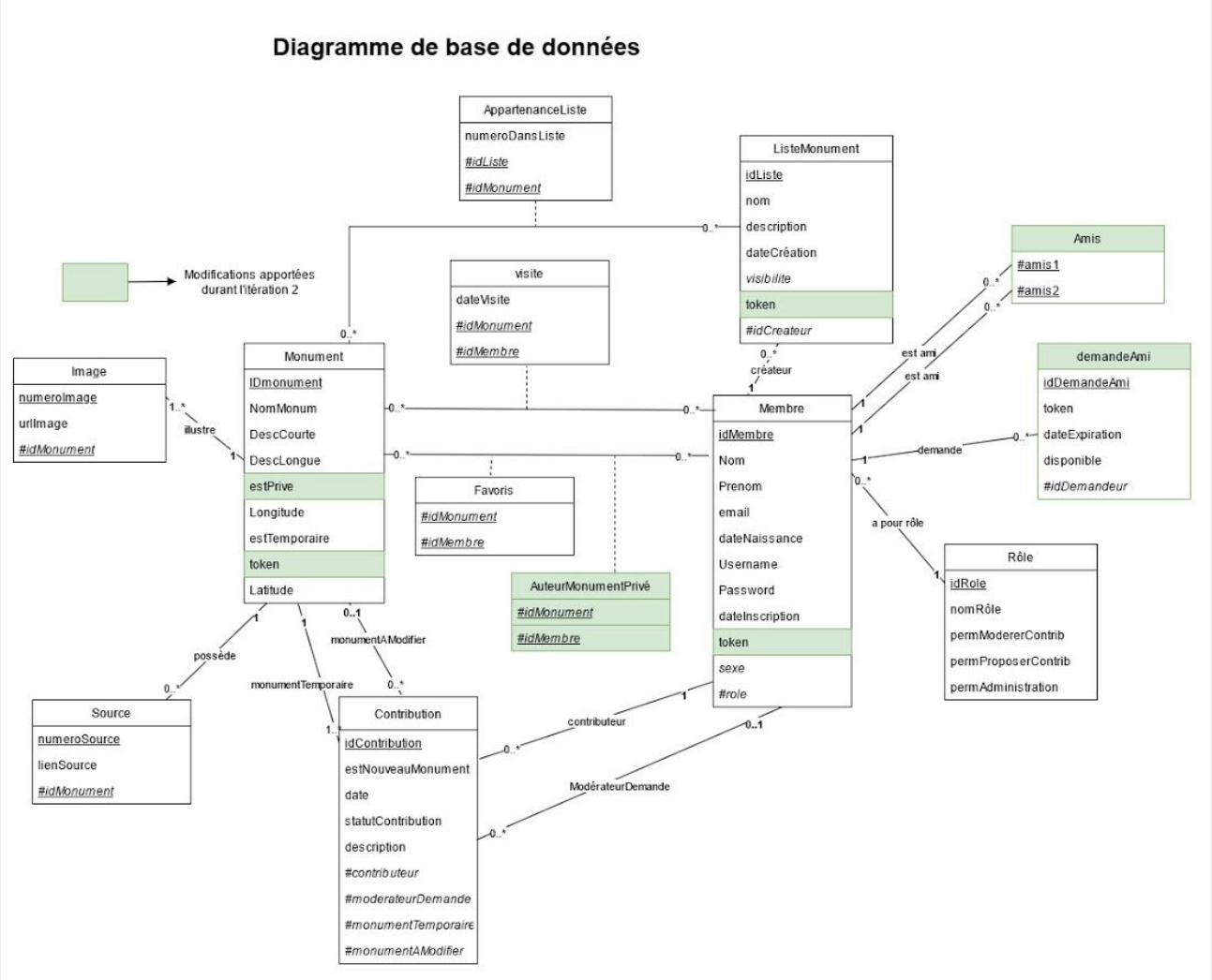


Diagramme de base de données
AppartenanceListe
numeroDansListe
#idListe
#idMonument
ListeMonument
idListe
nom
description
dateCréation
visibilite
token
#idCreateur
Amis
#amis1
#amis2
Modifications apportées durant l'itération 2
visite
dateVisite
#idMonument
#idMembre
Image
numeroImage
urlImage
#idMonument
Monument
IDmonument
NomMonum
DescCourte
DescLongue
estPrive
Longitude
estTemporaire
token
Latitude
Favoris
#idMonument
#idMembre
Membre
idMembre
Nom
Prenom
email
dateNaissance
Username
Password
dateInscription
token
sexe
#role
demandeAmi
idDemandeAmi
token
dateExpiration
disponible
#idDemandeur
AuteurMonumentPrivé
#idMonument
#idMembre
Rôle
idRole
nomRôle
permModererContrib
permProposerContrib
permAdministration
Source
numeroSource
lienSource
#idMonument
Contribution
idContribution
estNouveauMonument
date
statutContribution
description
#contributeur
#moderateurDemande
#monumentTemporaire
#monumentAModifier
illustre
possède
monumentTemporaire
monumentAModifier
créateur
est ami
est ami
demande
a pour rôle
contributeur
ModérateurDemande