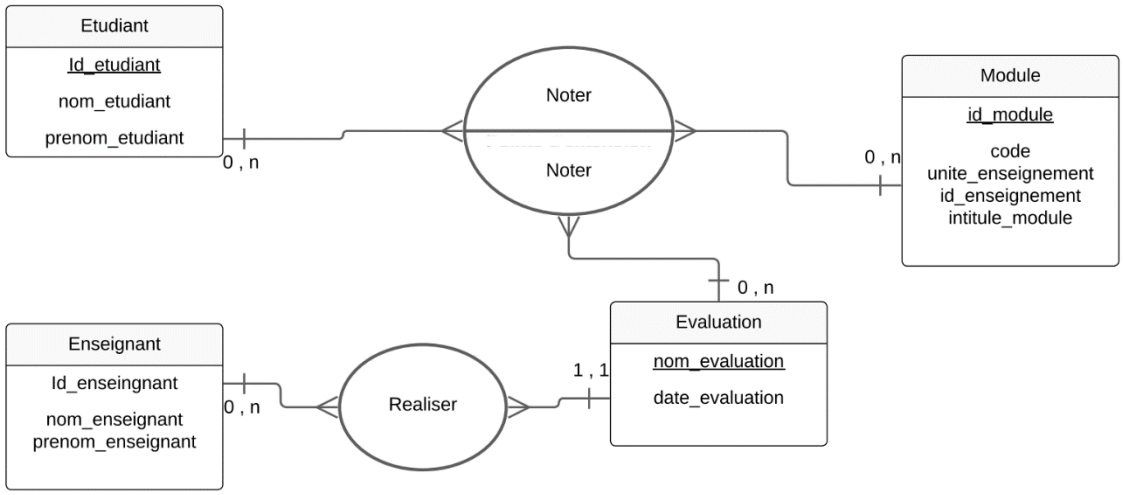
RATTINASSABABADY
Bharani
INDRA

SAE Bases de données et langage SQL

2.1/ Modélisation et script de création sans AGL :

1)



2)

Etudiant (id_etudiant, nom_etudiant, prenom_etudiant)
Module (id_module, #id_enseignant, code, unite_enseignement, intitule_module)
Enseignant (id_enseignant, nom_enseignant, prenom_enseignant)
Evaluation (nom_evalution, date_evaluation)
Note (nom_evaluation, id_module, id_etudiant, note)

3)

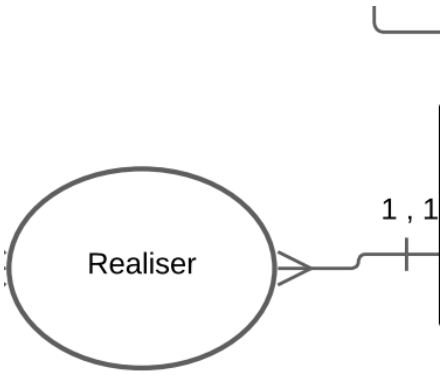
```
CEATE TABLE Enseignant (
id_enseignant INTEGER PRIMARY KEY,
nom_enseignant VARCHAR NOT NULL,
prenom_enseignant VARCHAR NOT NULL
) ;

CEATE TABLE Module (
id_module INTEGER PRIMARY KEY,
```

```
id_enseignant INTEGER REFERENCES Enseignant ON DELETE SET NULL,
intitule_module VARCHAR NOT NULL,
code VARCHAR NOT NULL,
unite_enseignement VARCHAR NOT NULL
 ) ;

CEATE TABLE Etudiant (
id_etudiant INTEGER PRIMARY KEY,
nom_etudiant VARCHAR NOT NULL,
prenom_etudiant VARCHAR NOT NULL
) ;

CEATE TABLE Evaluation (
nom_evaluation VARCHAR PRIMARY KEY,
id_enseignant INTEGER REFERENCES Enseignant ON DELETE SET NULL,
date_evaluation DATE NOTE NULL
) ;

CEATE TABLE Note (
nom_evaluation VARCHAR PRIMARY KEY,
id_module INTEGER NOT NULL,
id_etudiant INTEGER NOT NULL,
note FLOAT NOT NULL,
nom_evaluation VARCHAR NOT NULL
) ;
```

2.2/ Modèlisation et script de creation avec AGL

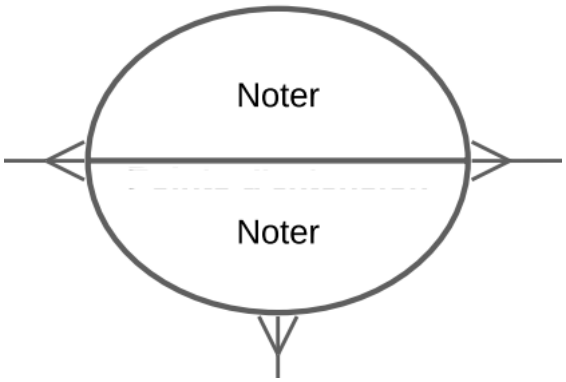
1)

Association de fonction, il suffit d'ajouter une clé étrangère lorsque la cardinalité maximale est de 1, Donc 1.1. (voir ci-dessous)

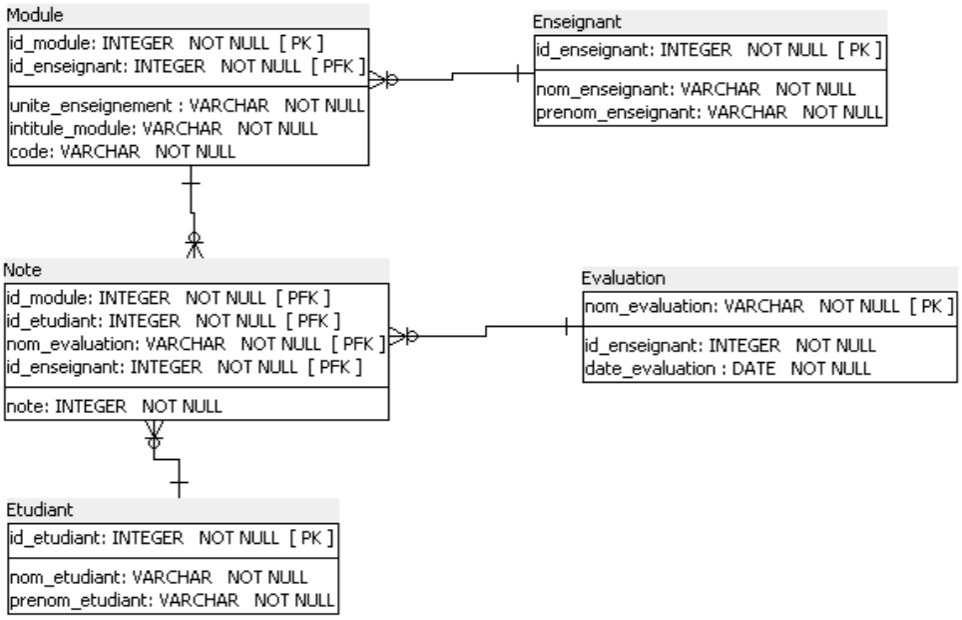


2)

Une association maillée de type est une jointure de deux tables avec plusieurs clés étrangères, de cardinalité 0, n. (voir ci-dessous)



3)



4)

```
CREATE TABLE Evaluation (
nom_evaluation VARCHAR NOT NULL,
 id_enseignant INTEGER NOT NULL,
date_evaluation_ DATE NOT NULL,
CONSTRAINT nom_evaluation PRIMARY KEY (nom_evaluation)
);

CREATE TABLE Etudiant (
id_etudiant INTEGER NOT NULL,
nom_etudiant VARCHAR NOT NULL,
prenom_etudiant VARCHAR NOT NULL,
CONSTRAINT id_etudiant PRIMARY KEY (id_etudiant)
);

CREATE TABLE Enseignant (
id_enseignant INTEGER NOT NULL,
nom_enseignant VARCHAR NOT NULL,
prenom_enseignant VARCHAR NOT NULL,
CONSTRAINT id_enseignant PRIMARY KEY (id_enseignant)
);

CREATE TABLE Module (
id_module INTEGER NOT NULL,
id_enseignant INTEGER NOT NULL,
unite_enseignement_ VARCHAR NOT NULL,
intitule_module VARCHAR NOT NULL,
Code VARCHAR NOT NULL,
CONSTRAINT id_module PRIMARY KEY (id_module, id_enseignant)
);

CREATE TABLE Note (
id_module INTEGER NOT NULL,
id_etudiant INTEGER NOT NULL,
nom_evaluation VARCHAR NOT NULL,
id_enseignant INTEGER NOT NULL,
note INTEGER NOT NULL,
```

```
CONSTRAINT id_note PRIMARY KEY (id_module, id_etudiant, nom_evaluation,
id_enseignant)
);

ALTER TABLE Note ADD CONSTRAINT Evaluation_Note_fk
FOREIGN KEY (nom_evaluation)
REFERENCES Evaluation (nom_evaluation)
ON DELETE NO ACTION
ON UPDATE NO ACTION
NOT DEFERRABLE;

ALTER TABLE Note ADD CONSTRAINT Etudiant_Note_fk
FOREIGN KEY (id_etudiant)
REFERENCES Etudiant (id_etudiant)
ON DELETE NO ACTION
ON UPDATE NO ACTION NOT DEFERRABLE;

ALTER TABLE Module ADD CONSTRAINT Enseignant_Module_fk
FOREIGN KEY (id_enseignant)
REFERENCES Enseignant (id_enseignant)
ON DELETE NO ACTION
ON UPDATE NO ACTION
NOT DEFERRABLE;

ALTER TABLE Note ADD CONSTRAINT Module_Note_fk
FOREIGN KEY (id_module, id_enseignant)
REFERENCES Module (id_module, id_enseignant)
ON DELETE NO ACTION
ON UPDATE NO ACTION NOT
DEFERRABLE;
```

5)

Le script fait manuellement est assez long, mais s'il est fait correctement, nous aurons moins d'erreur.
Et AGL ajoute ALTER TABLE à la fin. (donc cela modifie les contraintes des attributs à la fin)

2.3 / Peuplement des tables et requêtes

1)

```
COPY ENSEIGNANT (id_enseignant, prenom_enseignant, nom_enseignant)
FROM'C:\Users\bharani\Downloads\SAE104_data.csv' WITH
(FORMATCSV,HEADER,DELIMITER';');
```

La commande COPY, sert à copier depuis la table ENSEIGNANT seulement les colonnes que nous avons besoin c'est-à-dire id_enseignant, prenom_enseignant, nom_enseignant.
Le WITH ouvre le fichier en format CSV.

```
COPY ETUDIANT (id_etudiant, prenom_etudiant, nom_etudiant) FROM
'C:\Users\bharani\Downloads\SAE104_data.csv' WITH (FORMAT
CSV,HEADER,DELIMITER';');
```

La commande COPY, sert à copier depuis la table ETUDIANT seulement les colonnes que nous avons besoin c'est-à-dire id_etudiant, prenom_etudiant, nom_etudiant.
Le WITH ouvre le fichier en format CSV.

```
COPY MODULE (id_module, id_enseignant, intitule_module, code,
unite_enseignement)FROM'C:\Users\bharani\Downloads\SAE104_data.csv' WITH
(FORMAT CSV,HEADER,DELIMITER';');
```

La commande COPY, sert à copier depuis la table MODULE seulement les colonnes que nous avons besoin c'est-à-dire id_enseignant, intitule_module, code, unite_enseignement.
Le WITH ouvre le fichier en format CSV.

```
COPY EVALUATION (nom_evaluation, date_evaluation, id_enseignant) FROM
'C:\Users\bharani\Downloads\SAE104_data.csv' WITH (FORMAT
CSV,HEADER,DELIMITER';');
```

La commande COPY, sert à copier depuis la table EVALUATION seulement les colonnes que nous avons besoin c'est-à-dire nom_evaluation, date_evaluation, id_enseignement
Le WITH ouvre le fichier en format CSV.

```
COPY NOTER (id_module, id_etudiant, nom_evaluation, note) FROM
'C:\Users\bharani\Downloads\SAE104_data.csv' WITH (FORMAT
CSV,HEADER,DELIMITER';');
```

La commande COPY, sert à copier depuis la table NOTER seulement les colonnes que nous avons besoin c'est-à-dire id_module, id_etudiant, nom_evaluation, note
Le WITH ouvre le fichier en format CSV.

2)

```
SELECT nom,prenom FROM etudiants NATURAL JOIN module USING (id_etudiants) ;
```

Cette requête va afficher les noms des étudiants.

SELECT nom_evaluation FROM EVALUATION;
Cette requete va verifier les clés primaires.