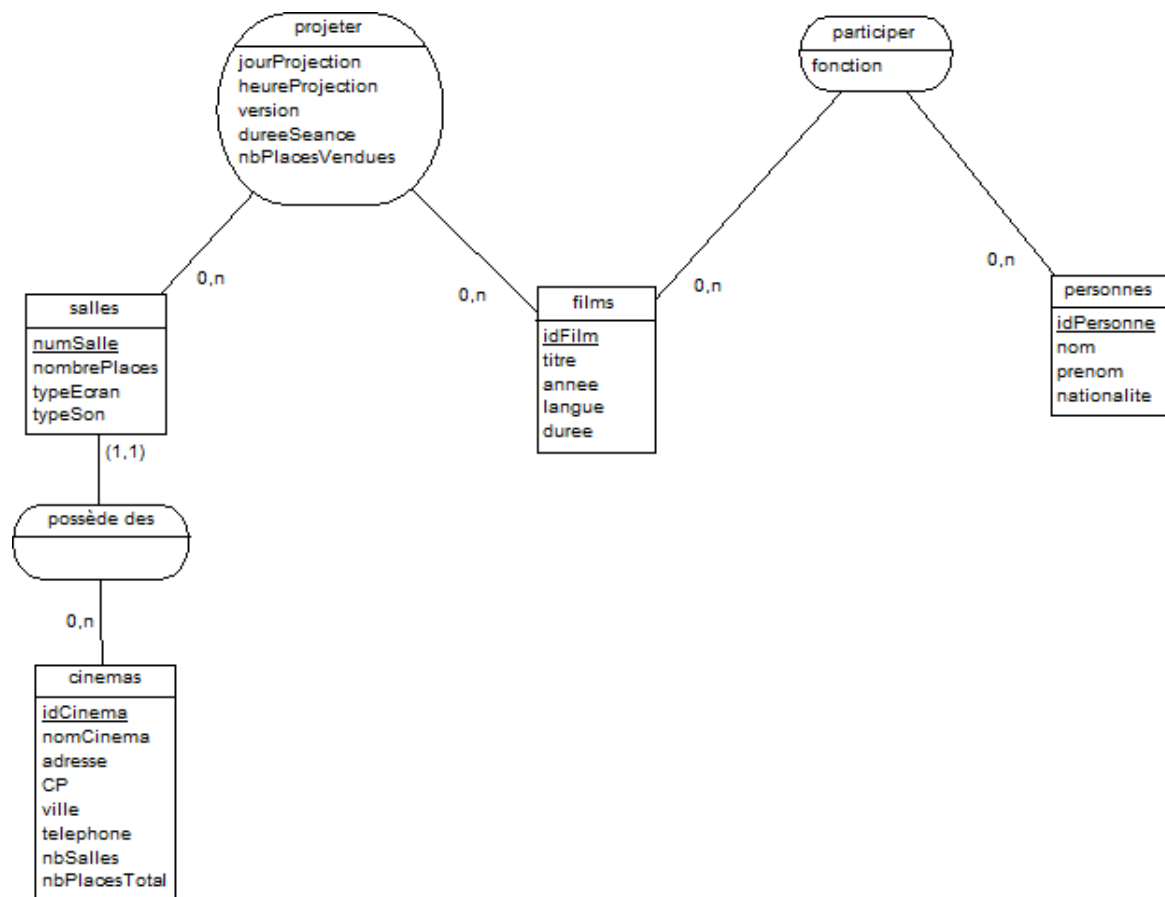
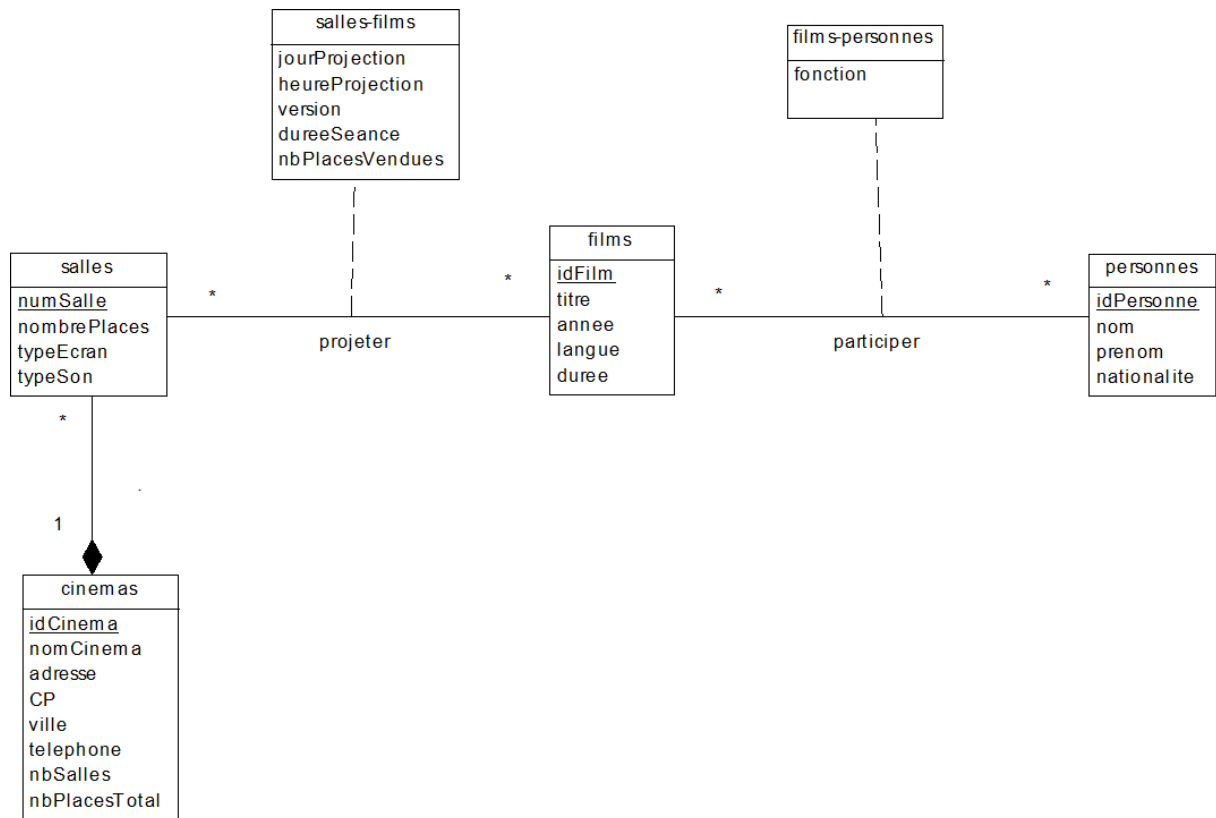


## MCD – Merise



## MCD – UML



## Justification des cardinalités :

### **Cinéma-Salles : Version « avoir » :**

Un cinéma a **plusieurs** salles.

Une salle a **un** cinéma.

⇒ Relation 1 a plusieurs.

Version MOA :

Un cinéma est composé de **plusieurs** salles.

Une salle est un composant d'**un** cinéma.

### **Salles-Films : Version « avoir » :**

Une salle a **plusieurs** films.

Un film a **plusieurs** salles.

⇒ Relation plusieurs à plusieurs - > il peut y avoir des attributs

Version MOA :

Une salle projette **plusieurs** films.

Un film est projeté dans **plusieurs** salles.

Justification des attributs :

Une salle projette un film, à un jour de projection, une heure de projection, avec une version, avec une durée de séance et un certain nombre de place vendues.

La relation plusieurs à plusieurs ce sera la table salles-films ou la table des « projections ».

Une projection « a un » (et pas plusieurs) chaque attributs ajouté.

### **Films-Personnes : Version « avoir » :**

Une personne a **plusieurs** films (auxquels elle participe)

Un film a **plusieurs** personnes.

⇒ Relation plusieurs a plusieurs.

Version MOA :

Une personne participe à **plusieurs** films.

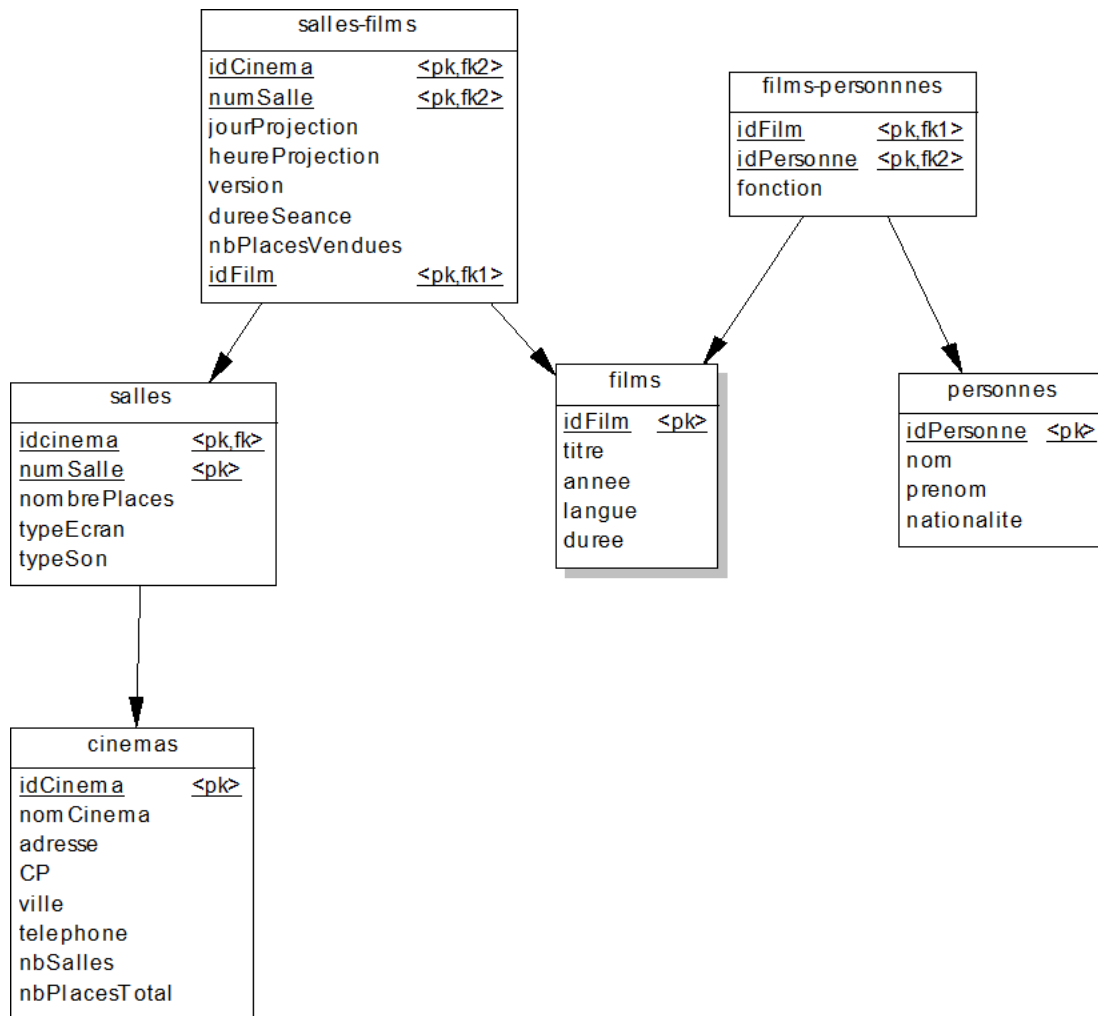
Un film a **plusieurs** personnes participantes.

Justification des attributs :

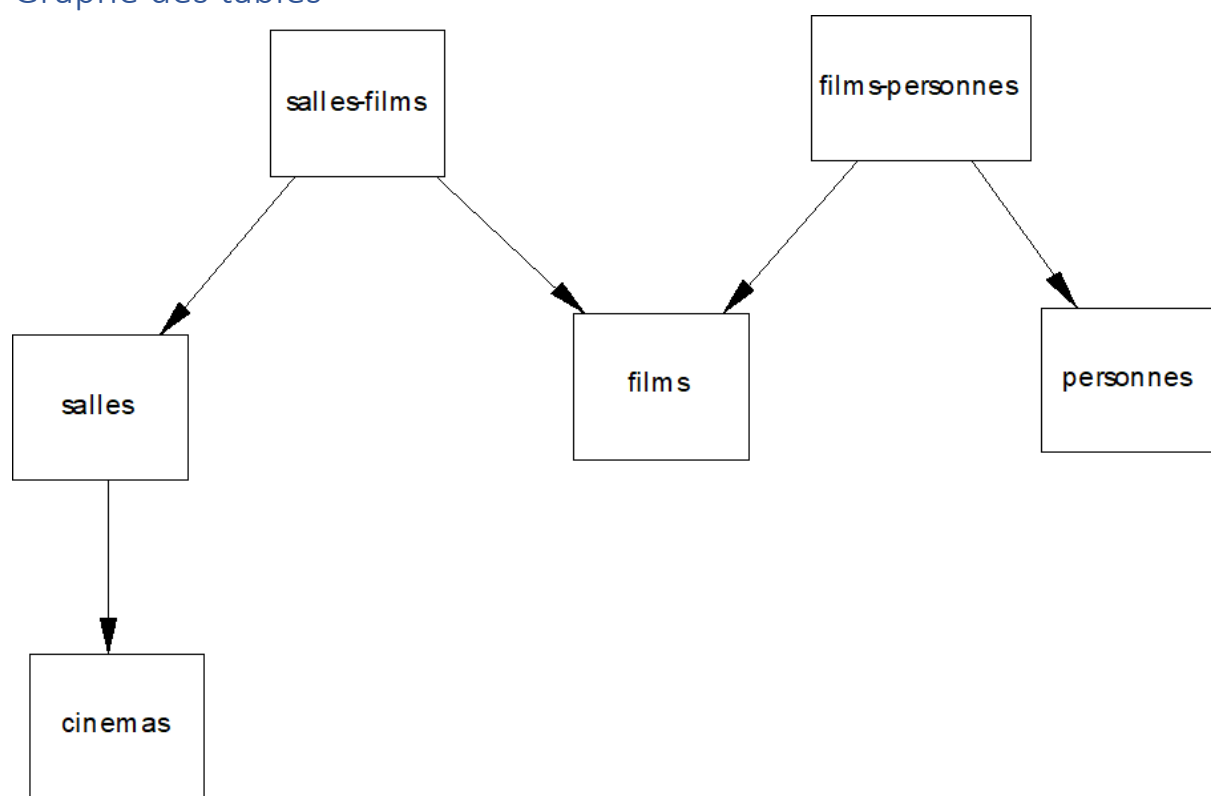
Une personne participe à un film, avec une certaine fonction. Si la personne a plusieurs fonctions, on aura plusieurs participations.

La relation plusieurs à plusieurs ce sera la table films-personnes ou la table des « participations » ou la table des « génériques ». Une participation « a une » (et pas plusieurs) fonction.

## MLD



## Graphe des tables



## SQL-DDL

```
CREATE TABLE films (  
  idFilm    int AUTO_INCREMENT,  
  titre     varchar(50) not null,  
  annee     int not null,  
  langue    varchar(10) not null,  
  duree     int not null,  
  PRIMARY KEY(idFilm)  
);
```

```
CREATE TABLE personnes (  
  idPersonne int AUTO_INCREMENT,  
  nom        varchar(30) not null,  
  prenom     varchar(30) not null,  
  nationalite varchar(30) not null,  
  PRIMARY KEY(idPersonne)  
);
```

```
CREATE TABLE Films-Personnes (  
  idFilm    int not null,  
  idPersonne int not null,  
  fonction  varchar(30) not null,  
  PRIMARY KEY(idFilm, idPersonne, fonction)  
);
```

```
CREATE TABLE cinemas (  
  idCinema  int AUTO_INCREMENT,  
  nomCinema varchar(30) not null,  
  adresse   varchar(30) not null,  
  CP        varchar(10) not null,  
  ville     varchar(30) not null,  
  telephone varchar(10) not null,  
  nbSalles  int not null,  
  nbPlacesTotal int not null,  
  PRIMARY KEY(idCinema)  
);
```

```
CREATE TABLE salles (  
  idcinema  int not null,  
  numSalle  int not null,  
  nombrePlaces int not null,  
  typeEcran varchar(30) not null,  
  typeSon   varchar(30) not null,  
  PRIMARY KEY(idCinema, numSalle)  
);
```

```
CREATE TABLE Salles-Films (  
  idCinema      int not null,  
  numSalle      int not null,  
  jourProjection date not null,  
  heureProjection time not null,  
  version       char(2) not null,  
  dureeSeance   int not null,  
  nbPlacesVendues int,  
  idFilm        int not null,  
  PRIMARY KEY(idCinema, numSalle, jourProjection, heureProjection)  
);
```

```
ALTER TABLE Films-Personnes ADD foreign key(idFilm) references films(idFilm);  
ALTER TABLE Films-Personnes ADD foreign key(idPersonne) references  
personnes(idPersonne);
```

```
ALTER TABLE salles ADD foreign key(idCinema) references cinemas(idCinema);
```

```
ALTER TABLE Salles-Films ADD foreign key(idFilm) references films(idFilm);  
ALTER TABLE Salles-Films ADD foreign key(idCinema, numSalle) references salles(idCinema,  
numSalle);
```