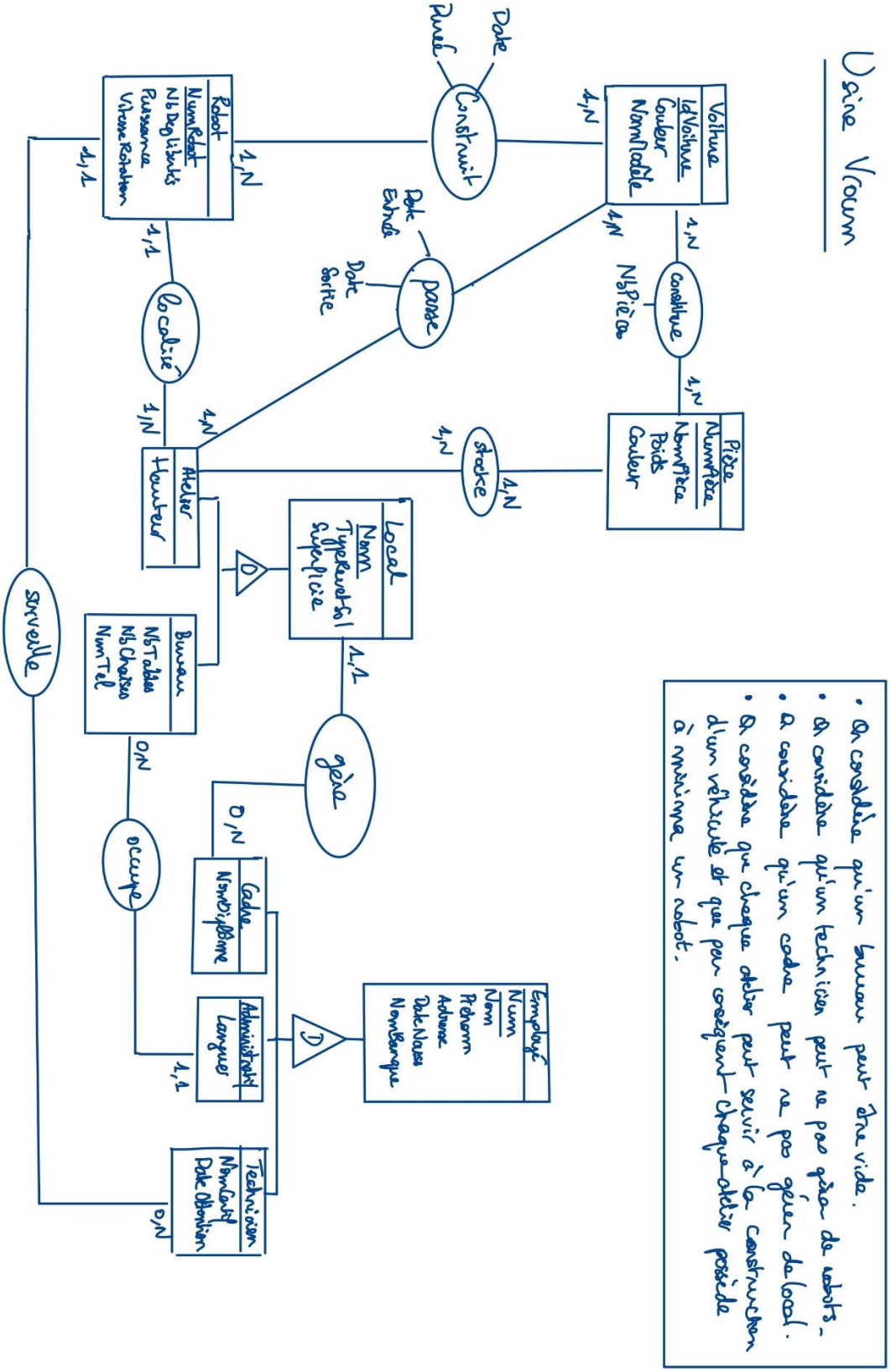
DELEFOSSE Aymeric

Examen – Systèmes d'Informations et Bases de Données Relationnelles

Exercice 1



- On considère qu'un bureau peut être vide.
- On considère qu'un technicien peut ne pas gérer de robots.
- On considère qu'un codeur peut ne pas gérer de local.
- On considère que chaque atelier peut servir à la construction d'un véhicule et que par conséquent chaque atelier possède à minima un robot.

Exercice 2

Modèle relationnel associé au schéma entité-association :

Entité 1 (Id1, Date, # Id5)

Entité 2 (Id2, Prénom, # Id1)

Entité 3 (Id3, Nom, n°SS, # Id1)

Entité 4 (Id4, Sport)

Entité 5 (Id5, Adresse, Nom)

Ass24 (# Id2, # Id4, Heure, Adresse)

Ass4 (# Id4, Date, Lieu)

Requêtes SQL permettant de définir la base de données :

```sql
CREATE TABLE [Entité 4] (
    Id4 INT AUTO_INCREMENT,
    Sport VARCHAR(30),
    PRIMARY KEY (Id4)
) ;

CREATE TABLE [Entité 5] (
    Id5 INT AUTO_INCREMENT,
    Nom VARCHAR(30),
    Adresse VARCHAR(255),
    CodePostal VARCHAR(30),
    Ville VARCHAR(30),
    PRIMARY KEY (Id5)
) ;

CREATE TABLE [Entité 1] (
    Id1 INT AUTO_INCREMENT,
    Date DATE,
    Id5 INT,
    PRIMARY KEY (Id1),
    FOREIGN KEY (Id5) REFERENCES [Entité 5](Id5)
) ;

CREATE TABLE [Entité 2] (
    Id2 INT AUTO_INCREMENT,
    Prénom VARCHAR(30),
    Id1 INT,
    PRIMARY KEY (Id2),
    FOREIGN KEY (Id1) REFERENCES [Entité 1](Id1)
) ;
```

DELEFOSSE Aymeric

```sql
CREATE TABLE [Entité 3] (
    Id3 INT AUTO_INCREMENT,
    Nom VARCHAR(30),
    numSS NUMERIC,
    Id1 INT,
    PRIMARY KEY (Id3),
    FOREIGN KEY (Id1) REFERENCES [Entité 1](Id1)
)

CREATE TABLE Ass24 (
    Id2 INT,
    Id4 INT,
    Heure TIME,
    Adresse VARCHAR(255),
    CodePostal VARCHAR(30),
    Ville VARCHAR(30),
    PRIMARY KEY (Id2, Id4),
    FOREIGN KEY (Id2) REFERENCES [Entité 2](Id2),
    FOREIGN KEY (Id4) REFERENCES [Entité 4](Id4)
) ;

CREATE TABLE Ass4 (
    Id4 INT,
    Date DATE,
    Lieu VARCHAR(30),
    PRIMARY KEY (Id4),
    FOREIGN KEY (Id4) REFERENCES [Entité 4](Id4)
) ;
```

DELEFOSSE Aymeric

Exercice 3

Langage de définition/manipulation des données

1.
```
CREATE TABLE T_Film (
    Id INT AUTO_INCREMENT,
    IdScénariste INT,
    Titre VARCHAR(255),
    Catégorie VARCHAR(255),
    PRIMARY KEY (Id),
    FOREIGN KEY (IdScénariste) REFERENCES T_Scénariste(Id)
) ;
```

2.
```
INSERT INTO T_Acteur (Nom, Prénom, AnnéeNaissance)
VALUES ("Tito", "Julie", "1985") ;
```

On considère que l'Id est auto-incrémenté.

3.
```
ALTER TABLE T_Film RENAME TO T_FilmArchive ;
```

4.
```
ALTER TABLE T_Film
RENAME COLUMN Catégorie TO CatégorieFilm ;
```

5.
```
ALTER TABLE T_Scénariste
DROP COLUMN Nom ;
```

6.
```
ALTER TABLE T_Film
ADD COLUMN Durée TIME ;
```

7.
```
UPDATE T_Scénariste
SET Ville = "Lyon" AND Salaire = 2800 WHERE Id = 3 ;
```

8.
```
GRANT UPDATE
ON T_Film
TO Adrien
WITH GRANT OPTION ;
```

9.
```
REVOKE ALL
ON T_Scénariste, T_JoueDans, T_FilmArchive, T_Acteur ;
FROM Bastien
```

DELEFOSSE Aymeric

Requêtes SQL

```sql
1.
SELECT Id, Ville, Salaire
FROM T_Scenariste
ORDER BY Salaire DESC ;

2.
SELECT IdScénariste, Catégorie
FROM T_Film
WHERE Id = 5 ;

3.
SELECT Titre
FROM T_Film
    INNER JOIN T_Scenariste ON T_Film.IdScénariste = T_Scenariste.Id
WHERE Salaire = 2700
ORDER BY T_Film.Id ;

4.
SELECT DISTINCT Ville
FROM T_Scenariste
WHERE Salaire < (SELECT AVG(Salaire) FROM T_Scenariste) ;

5.
SELECT DISTINCT Ville
FROM T_Film
    INNER JOIN T_Scenariste ON T_Film.IdScénariste = T_Scenariste.Id
WHERE Catégorie IS NULL ;

6.
SELECT Titre
FROM T_Film
    INNER JOIN T_JoueDans ON T_Film.Id = T_JoueDans.IdFilm
WHERE IdActeur = 1 AND IdActeur = 3 ;

7.
SELECT IdFilm, COUNT(IdActeur)
FROM T_JoueDans
GROUP BY IdFilm ;

8.
SELECT IdFilm
FROM T_JoueDans
GROUP BY IdFilm
HAVING COUNT(IdActeur) = (SELECT MAX(nbacteurs)
                          FROM (SELECT IdFilm, COUNT(IdActeur) AS
                          nbacteurs FROM T_JoueDans GROUP BY IdFilm)) ;
```

9.
```sql
SELECT IdScénariste, Salaire
FROM T_Scenariste
    INNER JOIN T_Film ON T_Scenariste.Id = T_Film.IdScénariste
GROUP BY IdScénariste
HAVING COUNT(*) = 4 ;
```

10.
```sql
SELECT IdScénariste, Ville
FROM T_Scenariste
    INNER JOIN T_Film ON T_Scenariste.Id = T_Film.IdScénariste
GROUP BY IdScénariste, Ville
HAVING COUNT(*) = 0 ;
```

Si on considère que l'association se fait sur les scénaristes.

11.
```sql
SELECT Ville
FROM T_Scenariste
    INNER JOIN T_Film ON T_Scenariste.Id = T_Film.IdScénariste
GROUP BY Ville
HAVING COUNT(*) = 0 ;
```

Si on considère que l'association se fait sur les villes.