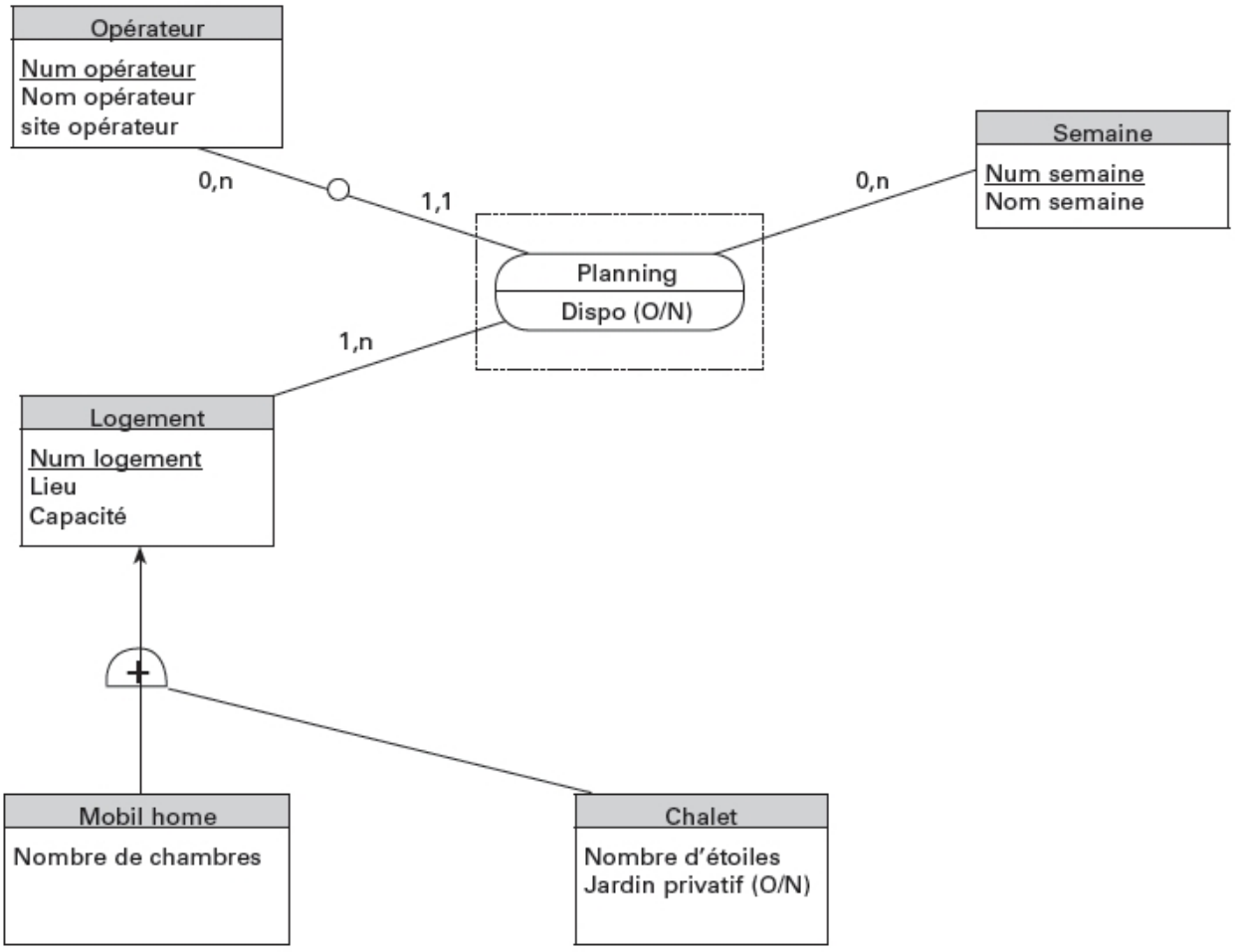
Gestion d'un site de location de vacances

Partie 1 : modèle conceptuel de données et schéma relationnel

Question 1.1 et 1.2 : rédiger le MCD du site de logements de vacances de Ticarol.



Question 1.3 : rédiger le schéma relationnel correspondant.

Opérateur(Num opérateur, Nom opérateur, Site opérateur)
Semaine(Num semaine, Nom semaine)
Logement(Num logement, Lieu, Capacité, Type Logement)
Planning(#Num logement, #Num semaine, #Num opérateur, Dispo (O/N))
Mobil-home(#Num logement, Nombre de chambres)
Chalet(#Num logement, Nombre d'étoiles, Jardin privatif (O/N))

Compléments :
- on remarquera qu'ici, on a également préfixé par un « # » les clefs étrangères qui font partie des clefs primaires des tables.

- le champ « Type Logement » est un champ qu'on qualifie de champ discriminant. Il sert à savoir, sans passer par des jointures, si le logement existe dans la table MobilHome ou (exclusif) dans la table Chalet.
- on rappelle que le symbole « + » (équivalent à XT), pour exclusivité et totalité, correspond à un opérateur OUX (ou exclusif, XOR en anglais). « + » signifie que l'on est forcément l'un ou l'autre (totalité) mais pas les deux (exclusivité). En l'occurrence, un logement est nécessairement un mobil home ou un chalet, mais ne peut être à la fois un mobil home et un chalet. N.B. : en logique booléenne, le « + » (addition booléenne) est l'opérateur « ou ».

Partie 2 : SQL (rappels de 1ère année)

o Question 2.1 : rédiger les requêtes de création des tables (CREATE TABLE).

```
CREATE TABLE Operateur(
        NumOperateur AUTOINCREMENT PRIMARY KEY,
        NomOperateur VARCHAR(50),
        SiteOperateur VARCHAR(50)
);
CREATE TABLE Semaine(
        NumSemaine SHORT PRIMARY KEY,
        NomSemaine VARCHAR(30)
);
CREATE TABLE Logement(
        NumLogement LONG PRIMARY KEY,
        Lieu VARCHAR(50),
        Capacite NUMBER(2),
        TypeLogement VARCHAR(15)
);
CREATE TABLE Planning(
        NumSemaine SHORT REFERENCES Semaine(NumSemaine) ON UPDATE DELETE CASCADE,
        NumLogement LONG REFERENCES Logement(NumLogement) ON UPDATE DELETE CASCADE,
        NumOperateur LONG REFERENCES Operateur(NumOperateur)
        ON UPDATE DELETE CASCADE,
        Dispo BIT,
        PRIMARY KEY (NumSemaine, NumLogement)
);
CREATE TABLE MobilHome(
        NumLogement LONG PRIMARY KEY REFERENCES Logement(NumLogement)
        ON UPDATE DELETE CASCADE,
        NombreDeChambres SHORT
);
```

CREATE TABLE Chalet(
NumLogement **LONG PRIMARY KEY REFERENCES** Logement(NumLogement)
ON UPDATE DELETE CASCADE,
NombreEtoiles **SHORT**,
JardinPrivatif **BIT**
)

Compléments :
- SHORT et LONG sont des types d'entiers (entiers courts et entiers longs) ;
- BIT est un type booléen (vrai/faux) ;
- on rappelle que AUTO_INCREMENT est un type « compteur » (à chaque insertion, le compteur est incrémenté de 1) ;
- on remarquera la présence d'instructions ON UPDATE DELETE CASCADE (explication fournie en cours).

o Question 2.2 : rédiger la requête qui permet de supprimer le mobil home n°4 puis celle permettant de supprimer le logement correspondant.

DELETE FROM MobilHome **WHERE** NumLogement = 4 ;
DELETE FROM Logement **WHERE** NumLogement = 4 ;

o Question 2.3 : quel problème pourrait occasionner le fait d'essayer de supprimer d'abord le logement n°4 et seulement ensuite le mobil home correspondant ? Justifier.

Si l'on tente de supprimer d'abord le mobil home n°4 de la table Logement, en l'absence de suppression en cascade, on obtiendra une erreur. En effet, une contrainte d'intégrité référentielle n'est plus respectée, plus exactement un CIF. A savoir, dans la table MobilHome, il existe une ligne (un tûple) pointant sur le mobil home n°4 de la table Logement (via sa clef étrangère). Toujours en l'absence de suppression en cascade, cette suppression est refusée car la clef étrangère en question ne peut pointer sur un Logement qui n'existe plus.

o Question 2.4 : rédiger la requête qui permet de préciser qu'un logement mis en location est désormais indisponible.

UPDATE Planning
SET Dispo = 1
WHERE NumLogement = [Quel logement ?]
AND NumSemaine = [Quelle semaine ?]

o Question 2.5 : rédiger la requête qui permet d'afficher la liste des opérateurs.

SELETE * **FROM** Operateur

o Question 2.6 : rédiger la requête qui permet d'afficher la liste des logements du lieu « Les Peupliers ».

```
SELECT L.NumLogement, L.TypeLogement, M.NombreDeChambres, null, null
FROM Logement AS L INNER JOIN MobilHome AS M ON L.NumLogement = M.NumLogement
WHERE L.Lieu = « Les Peupliers »
UNION
SELECT L.NumLogement, L.TypeLogement, null, C.NombreEtoiles, C.JardinPrivatif
FROM Logement AS L INNER JOIN CHALET AS C ON L.NumLogement = C.NumLogement
WHERE L.Lieu = « Les Peupliers »
```

N.B. : les … **INNER JOIN** … **ON** … peuvent être remplacés par … **NATURAL JOIN** …
Pour le BTS, on retiendra le **INNER JOIN**. En pratique, le **NATURAL JOIN** est beaucoup plus efficace !
Pour N lignes dans Logement et environ N ligne dans Chalet ou MobilHome, le INNER JOIN affiche en NxN opérations, le NATURAL JOIN en Nxlog(N) opérations.

Autre version :

```
SELECT L.NumLogement, L.TypeLogement, M.NombreDeChambres, null, null
FROM Logement AS L, MobilHome AS M
WHERE L.Lieu = « Les Peupliers »
AND L.NumLogement = M.NumLogement
UNION
SELECT L.NumLogement, L.TypeLogement, null, C.NombreEtoiles, C.JardinPrivatif
FROM Logement AS L, CHALET AS C
WHERE L.Lieu = « Les Peupliers »
AND L.NumLogement = C.NumLogement
```

o Question 2.7 : rédiger la requête qui permet d'afficher la liste des logements disponibles la semaine 14.

```
SELECT L.NumLogement, L.TypeLogement, L.Lieu, O.NomOperateur, P.Dispo
FROM (Planning AS P INNER JOIN Logement AS L ON P.NumLogement = L.NumLogement )
        INNER JOIN Operateur AS O ON P.NumOperateur = O.NumOperateur
WHERE P.NumSemaine = 14 AND P.Dispo = ”O”
```

> Autre version :

```
SELECT L.NumLogement, L.TypeLogement, L.Lieu, O.NomOperateur, P.Dispo
FROM Planning AS P, Logement AS L, Operateur AS O
WHERE P.NumSemaine = 14 AND P.Dispo = “O”
AND P.NumLogement = L.NumLogement
AND P.NumOperateur = O.NumOperateur
```

o Question 2.8 : rédiger la requête qui permet de connaître la quantité de logements disponibles par opérateur entre la semaine 14 et la semaine 52.

```
SELECT O.NomOperateur, count(*) AS « Logements disponibles »
FROM Planning AS P INNER JOIN Operateur  AS O ON P.NumOperateur = O.NumOperateur
WHERE P.NumSemaine BETWEEN 14 AND 52
GROUP BY O.NomOperateur
```

Autre version :

```
SELECT O.NomOperateur, count(*) AS « Logements disponibles »
FROM Planning AS P, Operateur  AS O
WHERE P.NumSemaine BETWEEN 14 AND 52
AND P.NumOperateur = O.NumOperateur
GROUP BY O.NomOperateur
```