POLYTECHNIQUE MONTRÉAL

LE GÉNIE EN PREMIÈRE CLASSE

Proposition- Solution
Contrôle Périodique

INF3710
Automne 2018

Exercice 1 – Conception d'une Base de Données relationnelle : Gestion du Système d'Information des sections universitaires dans une université (13 points)

Le Système d'Information des sections universitaires dans une université répond aux exigences suivantes :

1. Il existe de nombreuses sections universitaires à l'université. Chaque section académique est enregistrée avec l'identifiant de section académique, le nom et le nom long.

2. Chaque section académique offre de nombreuses unités; chaque unité est enregistrée avec l'identifiant de l'unité, le code de l'unité, le nom, le crédit et la section académique qui offre l'unité.

3. Une unité peut être offerte dans de nombreuses périodes d'enseignement. Chaque période d'enseignement est enregistrée avec l'identifiant de la période d'enseignement, l'année, la période d'enseignement de l'année, la date de début, la date de fin, la date de début de pause et la date de fin de cours.

4. Chaque personne est enregistrée avec son identité, son mot de passe, son nom de famille, son prénom, son titre, son adresse électronique et sa date de naissance.

5. Chaque étudiant est enregistré avec son identifiant d'étudiant, son type d'étudiant (international ou local).

6. Un membre du personnel travaille pour une section universitaire. Nous enregistrons l'ID de leur personnel, leur bureau, leur téléphone et la section académique pour laquelle ils travaillent.

7. La base de données enregistre également les unités dans lesquelles l'étudiant est inscrit, ainsi que la note et la note qu'un étudiant gagne dans chaque unité.

L'explication suivante vous aidera à comprendre la conception du modèle E-A :

- Une unité universitaire peut être une école, un département.
- Un universitaire offre de nombreuses unités.

- Un membre du personnel travaille pour une seule unité scolaire.
- Une unité peut être offerte dans de nombreuses périodes d'enseignement.
  Il n'y a qu'un seul LUC (coordinateur d'unité locale) d'une unité offerte.

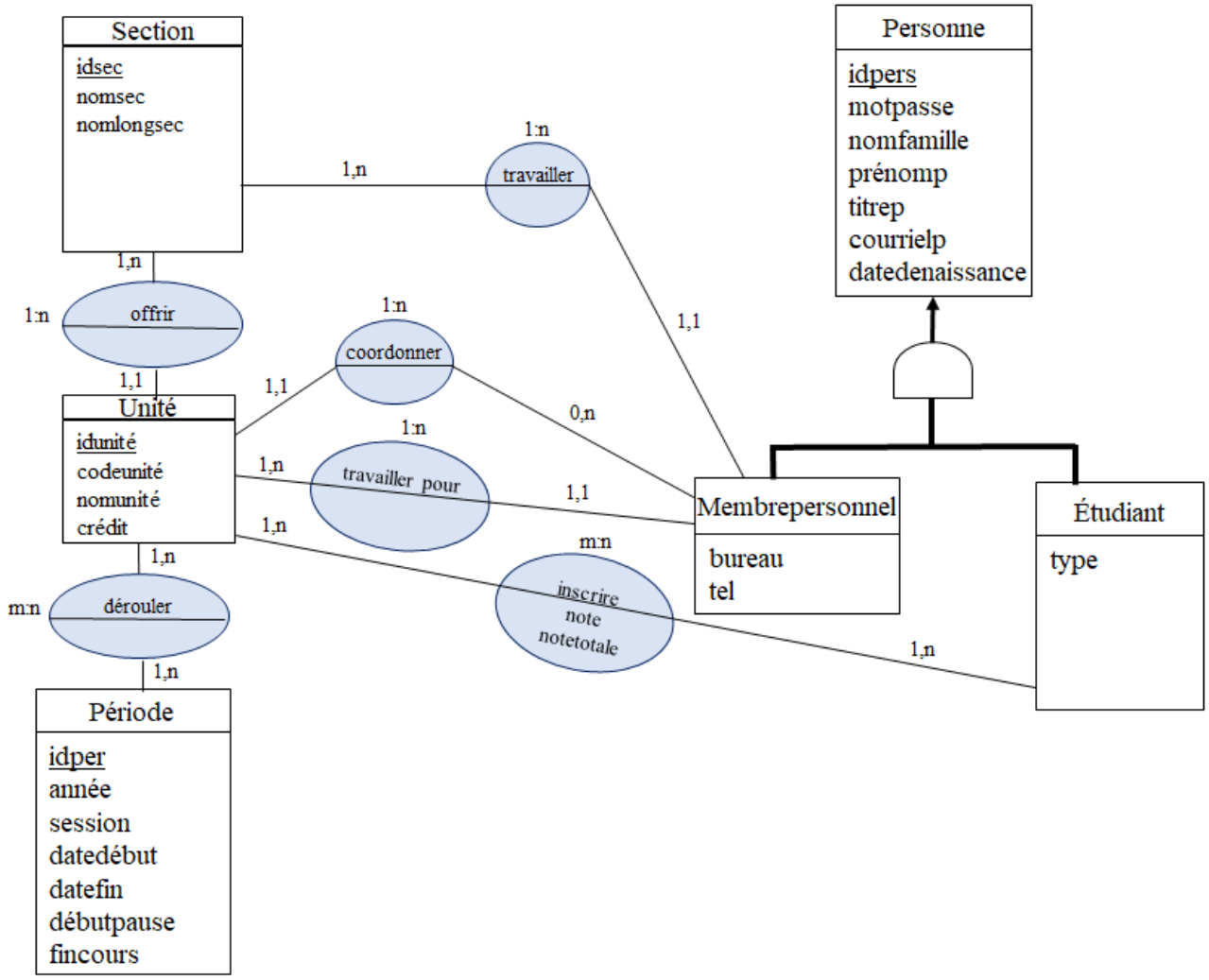
1. **Nommez les Types d'Entités de l'application. Spécifiez le(s) type(s) faible(s) s'il(s) en existe(nt). (2 point)**

2. **Pour chaque TE, spécifiez les attributs et soulignez l'identifiant. (2 points)**

3. **Proposez un schéma conceptuel dans le modèle Entité-Association en spécifiant les types des différents Types d'Associations existants, ainsi que les cardinalités des différents TE participant aux différents TA. (4 points)**

4. **Établissez la Conversion du modèle E-A obtenu en modèle logique de données, en identifiant aussi les contraintes d'intégrités. (5 points)**

Réponse de l'exercice 1 : Diagramme E/A



Réponse de l'exercice 1 (suite) - Transformation en schéma relationnel

Section (idsec, nomsec, nomlongsec)

PK idsec

Unité (idunité, codeunité, nomunité, crédit, idsec, idpers)

PK idunité

FK idsec ref Section, NN idsec

FK idpers ref Membrepersonnel, NN idpers

Période (idper, année, session, datedébut, datefin, débutpause, fincours)

PK idper

Personne (idpers, motpasse, nomfamille, prénomp, titrep, courrielp, datedenaissance)

PK idpers

Membrepersonnel (idpers, bureau, tel, idsec, idunité)

PK idpers

FK idpers ref Personne

FK idsec ref Section, NN idsec

FK idunité ref Unité, NN idunité

Étudiant (idpers, type)

PK idpers

FK idpers ref Personne

Réponse de l'exercice 1 (suite) - Transformation en schéma relationnel

Inscrire (idpers, idunité, note, notetotale)

PK (idpers, idunité)

FK idpers ref Étudiant

FK idunité ref Unité

NN note, notetotale

Dérouler (idunité, idper)

PK (idunité, idper)

FK idper ref Période

FK idunité ref Unité

Exercice 2 – Algèbre relationnel (6 points)

Supposons que nous avons le schéma relationnel suivant :

Employee (idemp, nomemp, adremp, iddepart, idposte)

Department (iddepart, nomdepart, étage)

Project (numproj, idemp, heures)

Poste (idposte, descriptionposte)

Écrivez les requêtes suivantes en algèbre relationnel.

a- Affichez l'identifiant des employés et le nom des employés pour tous les employés qui travaillent dans le département 'Génie'. **(2 points)**

$\pi_{[idemp, nomemp]} (\sigma_{[nomdepart = \text{'Génie'}]} (Employee \bowtie Departement))$

b- Affichez l'identifiant des employés et le nom des employés pour tous les employés qui ont travaillé plus de 5 heures sur un projet. **(2 points)**

$\pi_{[idemp, nomemp]} (\sigma_{[heures > 5]} (Employee \bowtie Projet))$

c- Affichez tous les départements qui contiennent des employés effectuant 12 heures de travail sur un projet. **(2 points)**

$\pi_{[iddepart, nomdepart, étage]} (\sigma_{[heures = 12]} (Departement \bowtie Employee \bowtie Projet))$

Exercice 3 – Langage SQL (11 points)

Considérons le schéma relationnel qui suit. Un employé peut travailler dans plus qu'un département; le champ **pcttemps** de la relation TRAVAIL représente le pourcentage de temps qu'un employé donné travaille dans un département donné; le champ **iddirecteur** de la relation DEPARTEMENT représente l'identifiant de l'employé qui a le poste directeur dans un département donné.

EMPLOYE (idemp, nomemp, age, salaire, adresse)

TRAVAIL (idemp, iddept, pcttemps)

DEPARTEMENT (iddept, nomdept, budget, iddirecteur)

1. **Utilisez le langage SQL pour créer les tables de la base en spécifiant les différentes contraintes. Marquer et justifier l'ordre de création de ces tables. (2 points)**

```
Create table EMPLOYE (idemp int primary key,
                      nomemp varchar(25) not null,
                      age int,
                      salaire in check (salaire > 0),
                      adresse varchar(45) );

Create table DEPARTEMENT (iddept int primary key,
                      nomdept varchar(35) not null,
                      budget int,
                      iddirecteur int not null references EMPLOYE);

Create table TRAVAIL (idemp int,
        iddept int,
        pcttemps int check (pcttemps > 0),
        constraint travail_pk primary key (idemp, iddept),
        constraint idemp_fk foreign key idemp references EMPLOYEE,
        constraint iddept_fk foreign key iddept references DEPARTEMENT);
```

La table EMPLOYE doit être créée avant la table DEPARTEMENT, parce que la table DEPARTEMENT contient une clé étrangère de la table EMPLOYE.
La table TRAVAIL doit être créée en troisième lieu après la création de la table EMPLOYE puis la table DEPARTEMENT, parce qu'elle dépend de ces 2 tables (idemp pour EMPLOYE et iddept pour DEPARTEMENT).

2. Écrivez en SQL les requêtes et manipulations suivantes.

a. Affichez les employés par ordre décroissant de leur nom et leur salaire. **(1 point)**

```
Select *
From  EMPLOYE
Order by nomemp, salaire desc;
```

b. Augmentez de 10 % le salaire des employés qui sont directeurs de département. **(1.5 points)**

```
Update EMPLOYE
Set salaire = salaire * 1.10
Where idemp in (select iddirecteur from DEPARTEMENT);
```

c. Affichez par nom du département le nombre d'employés qui y travaillent. **(1.5 points)**

```
Select D.nomdept, count(T.idemp)
From TRAVAIL T, DEPARTEMENT D
Where T.iddept = D. iddept
Group by D.nomdep;
```

d. Affichez les identifiants du département, les noms des employés et les noms des départements pour les employés qui travaillent à temps plein (100%) dans un département. **(2 points)**

```
Select D.iddept, E.nomemp, D.nomdept
From DEPARTEMENT D, TRAVAIL T, EMPLOYE E
Where D.iddept = T.iddept
And T.idemp = E.idemp
And T.pcttemps = 100;
```

e. Si un directeur dirige plus qu'un département, il contrôle la somme des budgets de tous les départements qu'il dirige. **Trouvez les identifiants des directeurs qui contrôlent plus que 5,000,000**. **(3 points)**

```
Select iddirecteur, sum(budget)
From DEPARTEMENT
Group by iddirecteur
Having sum(budget) > 5000000;
```

Bon travail!