# L1 Introduction aux bases de données

## Exercice

On s'intéresse à la gestion de la participation des adhérents aux différentes activités organisées par un centre de remise en forme dans le but d'établir des statistiques sur la pratique des activités, et pouvoir ainsi mieux les planifier et mieux connaître la population qui les pratiquent.

Le centre de remise en forme organise toute l'année différentes activités (gym tonique, fitness, musculation, piscine, jacuzzi, cardio…).
On dispose d'informations concernant les adhérents : leur numéro d'identification, leur nom, prénom, adresse et âge.
Pour chaque adhérent, on enregistre également la liste des séances d'activités auxquelles il participe.
Les activités proposées au sein du centre sont référencées par leur nom, le niveau de difficulté (facile, moyen, difficile).
Sur chaque activité, on affecte le prix fixe pour chaque séance que le client devra débourser.
Pour chaque séance d'activité, on veut pouvoir retrouver ses dates et heures d'organisation, ainsi qu'une note d'appréciation attribuée individuellement par chaque client qui s'y est inscrit.
Enfin il est possible mais non obligatoire qu'une activité soit encadrée par un moniteur du centre dont on connait le nom, le prenom et son diplôme.

## Questions :

Il vous appartient de faire les choix qui peuvent être implicites ou non précisés dans cet énoncé.

1 – Fournissez les scripts de création de tables qui permettent de structurer l'activité décrite ci-dessus. Il vous appartient de définir des clés primaires et des clés étrangères nécessaires.

```
CREATE TABLE ACTIVITE
(
ID_ACTIVITE NUMBER(10),
NOM VARCHAR2(100),
PRIX NUMBER(15,2),
DIFFICULTE VARCHAR2(50)
);

CREATE TABLE ADHERENT
(
ID_ADHERENT NUMBER(10),
NOM VARCHAR2(100),
ADRESSE VARCHAR2(200),
VILLE VARCHAR2(50),
DATE_NAISSANCE DATE
);

CREATE TABLE SEANCE
(
ID_SEANCE NUMBER(10),
ID_ACTIVITE NUMBER(10),
DATE_SEANCE DATE,
HEURE_DEB NUMBER(2),
HEURE_FIN NUMBER(2)
);

CREATE TABLE PARTICIPANT
(
ID_ADHERENT NUMBER(10),
ID_SEANCE NUMBER(10),
NOTE NUMBER(2)
);
```

2 – Donner la séquence d'instructions SQL qui permet d'insérer dans votre schéma les informations décrivant ceci :
« L'adhérant Dupond a pratiqué une séance de musculation encadrée par Tony RINER le 14/01/13 à 18h00. »

```
INSERT INTO ACTIVITE (ID_ACTIVITE, NOM, PRIX, DIFFICULTE )
VALUES (01, 'Musculation', 25, 'Moyen');
```

```
INSERT INTO ADHERENT (ID_ADHERENT, NOM, ADRESSE, VILLE,
DATE_NAISSANCE)
VALUES (01, 'Dupond', '28 rue de Lille', 'Tours', TO_DATE('01/01/79',
'DD/MM/YY'));

INSERT INTO MONITEUR (ID_MONITEUR, NOM, PRENOM,
SPECIALITE)
VALUES (01, 'RINER', 'Teddy', 'Judo');

INSERT INTO SEANCE (ID_SEANCE, ID_ACTIVITE, DATE_SEANCE,
HEURE_DEB, HEURE_FIN, ID_MONITEUR )
VALUES (01, 01, TO_DATE('14/01/13', 'DD/MM/YY'), 18, 19, 01);

INSERT INTO PARTICIPANT (ID_ADHERENT, ID_SEANCE, NOTE )
VALUES (01, 01, 13) ;

INSERT INTO SEANCE (ID_SEANCE, ID_ACTIVITE, DATE_SEANCE,
HEURE_DEB, HEURE_FIN)
VALUES (02, 01, TO_DATE('14/01/13', 'DD/MM/YY'), 18, 19);
```

3 – Compléter les données de votre schéma afin d'obtenir au minimum 5 lignes par table.

```
INSERT....
```

4 – Formulez les demandes suivantes en requêtes SQL :

a - Quel est ou quels sont les noms des activités pour lesquelles aucun adhérent n'a encore participé ?

**Plusieurs façons, en voilà une :**

```
SELECT NOM
FROM  ACTIVITE
WHERE ID_ACTIVITE NOT IN (SELECT ID_ACTIVITE
                          FROM SEANCE S, PARTICIPANT P
                          WHERE S.ID_SEANCE = P.ID_SEANCE
                    );
```

b- Quel est le nom du moniteur générant les meilleures appréciations en moyenne?

```
SELECT NOM
FROM MONITEUR
WHERE ID_MONITEUR IN (
        SELECT ID_MONITEUR
        FROM PARTICIPANT P, SEANCE S
        WHERE P.ID_SEANCE = S.ID_SEANCE
        GROUP BY S.ID_MONITEUR
        HAVING AVG(NOTE) = (SELECT MAX(AVG(NOTE))
                FROM PARTICIPANT P, SEANCE S
                WHERE P.ID_SEANCE = S.ID_SEANCE
                GROUP BY S.ID_MONITEUR)
        );
```

c - Quels sont les participants à la séance de musculation encadrée par Tony RINER le 14/01/13 à 18h00 ?

```
SELECT ID_ADHERENT
FROM PARTICIPANT P, SEANCE S, MONITEUR M
WHERE P.ID_SEANCE = S.ID_SEANCE
AND M.ID_MONITEUR = S.ID_MONITEUR
AND S.DATE_SEANCE = TO_DATE('14/01/2013', 'DD/MM/YYYY')
AND M.NOM = 'RINER' AND M.PRENOM = 'Tony'
AND S.HEURE_DEB = 18;
```

d - On souhaite modifier l'horaire de la séance encadrée par Tony RINER le 12/06/16 initialiment prévue à 17H00 en la reportant à 19H00.

```
UPDATE SEANCE S
SET S.HEURE_DEB = 19
WHERE S.DATE_SEANCE = TO_DATE('14/01/2013', 'DD/MM/YYYY')
AND S.HEURE_DEB = 17
AND S.ID_MONITEUR IN (SELECT ID_MONITEUR
                      FROM MONITEUR M
                      WHERE M.NOM = 'RINER' AND
                      M.PRENOM = 'Tony')
```

e - On souhaite faire une évolution du système ci-dessus afin de stocker désormais la profession des adhérents. Quelle instruction SQL vous permet d'implémenter cela dans votre modèle ?

```
ALTER TABLE ADHERENT ADD (PROFESSION
VARCHAR2(100));
```

f - Quel est le listing complet de tous les adhérents avec leur nombre de séances

associé toute activité confondue ?

**– jointure externe attendue sinon en toute rigueur on n'a pas les adhérents n'ayant pas fait d'activités**

```
SELECT A.NOM, COUNT(*)
FROM ADHERENT A, SEANCE S, PARTICIPANT P
WHERE A.ID_ADHERENT = P.ID_ADHERENT (+)
AND P.ID_SEANCE = S.ID_SEANCE
GROUP BY A.ID_ADHERENT, A.NOM;
```

g – Quel est le nom de l'activité ayant généré le plus de séances en janvier 2014 ?

**Il faut regrouper chaque activité et compter le nombre de séances associées. On compare ce nombre avec le nombre max de séances qu'une activité a généré sur la période.**

```
SELECT A.NOM, COUNT(*)
FROM ACTIVITE A, SEANCE S
WHERE A.ID_ACTIVITE = S.ID_ACTIVITE
AND TRUNC(S.DATE_SEANCE) BETWEEN TO_DATE('01/01/14', 'DD/MM/YY') AND
TO_DATE('31/01/14', 'DD/MM/YY')
GROUP BY A.ID_ACTIVITE, A.NOM
HAVING COUNT(*) = (SELECT MAX(COUNT(*))
        FROM ACTIVITE A, SEANCE S
        WHERE A.ID_ACTIVITE = S.ID_ACTIVITE
        AND TRUNC(S.DATE_SEANCE) BETWEEN TO_DATE('01/01/14', 'DD/MM/YY')
        AND TO_DATE('31/01/14', 'DD/MM/YY')
        GROUP BY A.ID_ACTIVITE)
```

**Ici une solution plus simple et plus performante techniquement mais qui est potentiellement fausse fonctionnellement.**
**En effet cette requête ne fait qu'une seule passe sur les tables mais ne renvoie qu'une seule ligne, ce qui peut être gênant en cas d'égalité.**

```
SELECT ss.NOM FROM (
SELECT A.NOM, COUNT(*)
FROM  ACTIVITE A, SEANCE S
WHERE A.ID_ACTIVITE = S.ID_ACTIVITE
AND TRUNC(S.DATE_SEANCE) BETWEEN TO_DATE('01/01/14',
'DD/MM/YY') AND TO_DATE('31/01/14', 'DD/MM/YY')
GROUP BY A.ID_ACTIVITE, A.NOM
ORDER BY 2 DESC
) ss
WHERE ROWNUM =1
```