M1 IDSM-Kharkiv – Année 2017-2018
Bases de données avancées
Examen
J. Darmont (http://eric.univ-lyon2.fr/~jdarmont/), 28/10/2017

Durée : 2 heures – Documents autorisés – Barème fourni à titre indicatif

Exercice 1 : Modélisation conceptuelle (5 points)

Une galerie d'art souhaite gérer la vente d'œuvres. Proposer un diagramme de classes UML qui modélise les spécifications ci-dessous.

Les œuvres sont caractérisées par un identifiant unique, un titre et un prix. Elles peuvent être de deux types : les tableaux ou les illustrations. Les tableaux sont caractérisés seulement par un identifiant unique. Les illustrations sont caractérisées par un identifiant unique, un nombre d'exemplaires disponibles et une date de réédition.

Les tableaux appartiennent à un type de peinture (huile, aquarelle…) caractérisé par un identifiant et un libellé. Les illustrations appartiennent à un genre (retouche photo, perspective, cartoon…) également caractérisé par un identifiant et un libellé.

Chaque œuvre a été créée par un artiste caractérisé par un identifiant unique, son nom et son prénom.

Les clients qui achètent les œuvres sont caractérisés par un identifiant unique, leur nom et leur prénom. Ils passent des commandes caractérisées par un identifiant unique et une date. Chaque commande peut concerner plusieurs œuvres dans une quantité donnée.

Exercice 2 : Requêtes SQL (5 points)

Le schéma relationnel de la base de données « événementiel » est donné ci-dessous.

MANIFESTATION (id, nom, date, tarif, id_organisme_organisateur#, id_lieu#)
ORGANISME (id, nom, type adresse)
LIEU (id, nom, type, adresse)
INTERVENANT (id, nom, prénom, coordonnées, id_organisme#)
FEEDBACK_VISITEUR (id, texte, id_manifestation#)
INTERVENIR (id_manifestation#, id_intervenant#, nb_heures)

Clés primaires
Clés étrangères#

Formuler en SQL les requêtes suivantes.

1. Liste (tous les attributs) des manifestations ayant lieu entre le 1er juillet 2017 et le 31 août 2017 inclus.
2. Tarif minimum, tarif moyen et tarif maximum de toutes les manifestations.
3. Liste des manifestations triée par date décroissante. Indiquer le nom de l'organisme organisateur et le nom du lieu (pas leurs identifiants)
4. Nombre de retours visiteurs (*feedback*) par manifestation. Indiquer le nom de la manifestation.
5. Identifiant, nom, prénom et nom d'organisme des « intervenants » qui n'interviennent dans aucune manifestation.

Exercice 3 : Programmation PL/SQL (10 points)

Le schéma relationnel de la base de données « employés » est donné ci-dessous.

DEPT (DEPTNO, DNAME, LOC)
EMP (EMPNO, ENAME, JOB, MGR#[1], HIREDATE, SAL, COMM, DEPTNO#)
SALGRADE (GRADE, LOSAL, HISAL)

1. Écrire un bloc PL/SQL anonyme qui vérifie que le plus petit salaire minimum (LOSAL) et le plus grand salaire maximum (HISAL) des échelons (GRADE) de la table SALGRADE sont inférieur et supérieur au plus petit et au plus grand salaire (SAL) de la table EMP[2], respectivement. Afficher le résultat (« OK » ou « KO ») à l'écran.

2. Écrire un bloc PL/SQL anonyme qui affiche le contenu de la table SALGRADE au format : GRADE [LOSAL, HISAL].

3. Écrire un bloc PL/SQL anonyme qui affiche le nom des employés (ENAME) de chaque département (DNAME) au format suivant.

ACCOUNTING: CLARK KING MILLER
OPERATIONS:
RESEARCH: FORD JONES SMITH
SALES: ALLEN BLAKE JAMES MARTIN TURNER WARD

4. Ajouter au bloc PL/SQL de la question 3 un traitement d'exception qui interrompt l'exécution si un département n'a aucun employé. Ne pas réécrire tout le bloc PL/SQL. Indiquer par des numéros dans la réponse à la question 3 les endroits où il faut insérer du code. Écrire le code correspondant à côté et indiquer les numéros.

5. On suppose qu'un attribut SALGRADE de type NUMBER(1) est ajouté à la table EMP. EMP.SALGRADE est clé étrangère vers SALGRADE.GRADE. Écrire un déclencheur qui, à chaque ajout ou modification dans la table EMP, affecte automatiquement une valeur à l'attribut SALGRADE. La valeur indiquée dans la requête est ignorée. La nouvelle valeur doit être lue dans la table SALGRADE (c'est GRADE). On suppose que la table SALGRADE couvre tous les salaires (SAL) possibles de la table EMP et que les intervalles [LOSAL, HISAL] sont disjoints.

Exemple :

```
    SALGRADE
GRADE LOSAL HISAL
----- ----- -----
    1   700  1200
    2  1201  1400
    3  1401  2000
    4  2001  3000
    5  3001  9999
```

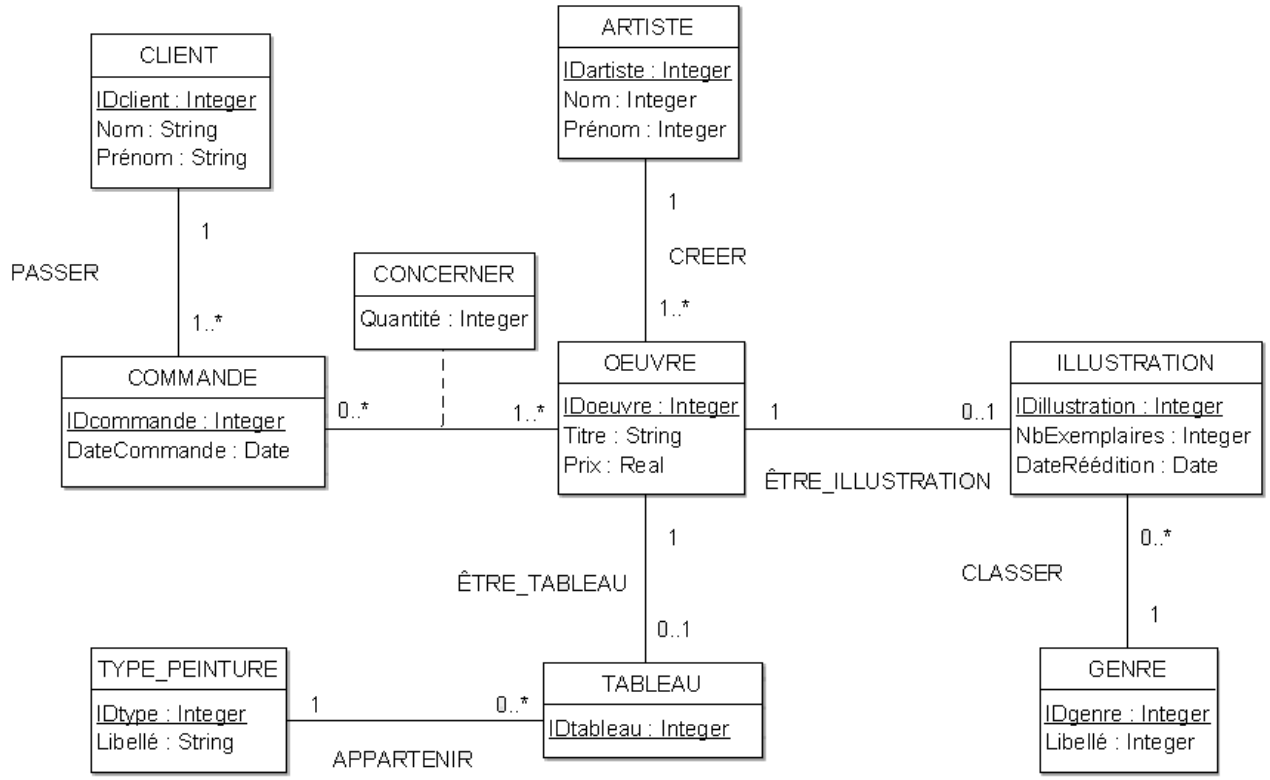
```
INSERT INTO EMP (EMPNO, ENAME, JOB, MGR, HIREDATE, SAL, COMM, DEPTNO, SALGRADE)
VALUES     (8888, 'SHEVSHENKO', 'ANALYST', 7566, '28-nov-2017', 2800, NULL, 20, NULL)
```

devient

```
<8888, 'SHEVSHENKO', 'ANALYST', 7566, '28-nov-2017', 2800, NULL, 20, 4)
```

[1] MGR# est le numéro d'employé (EMPNO) du manager de l'employé courant.
[2] MIN(LOSAL) ≤ MIN(SAL) et MAX(HISAL) ≥ MAX(SAL)

Correction exercice 1



Correction exercice 2

```
--1
SELECT *
FROM MANIFESTATION
WHERE date BETWEEN '01/07/2017' AND '31/08/2017';

--2
SELECT MIN(tarif), AVG(tarif), MAX(tarif)
FROM MANIFESTATION;

--3
SELECT m.id, m.nom, date, tarif, o.nom, l.nom
FROM MANIFESTATION m, ORGANISME o, LIEU l
WHERE id_organisme_organisateur = o.id
AND id_lieu = l.id
ORDER BY date DESC;

--4
SELECT nom, count(*)
FROM MANIFESTATION, FEEDBACK_VISITEUR
WHERE id_manifestation = id
GROUP BY nom;

--5
SELECT i.id, i.nom, prénom, o.nom
FROM INTERVENANT i, ORGANISME o
WHERE id_organisme = o.id
AND i.id NOT IN (
     SELECT id_intervenant
     FROM INTERVENIR);
```

Correction exercice 3

```
-- 1
DECLARE
  minsal EMP.SAL%TYPE;
  maxsal EMP.SAL%TYPE;
  minlo SALGRADE.LOSAL%TYPE;
  maxhi SALGRADE.HISAL%TYPE;
BEGIN
  SELECT MIN(SAL), MAX(SAL) INTO minsal, maxsal FROM EMP;
  SELECT MIN(LOSAL), MAX(HISAL) INTO minlo, maxhi FROM SALGRADE;
  IF minlo < minsal AND maxhi > maxsal THEN
    DBMS_OUTPUT.PUT_LINE('OK');
  ELSE
    DBMS_OUTPUT.PUT_LINE('KO');
  END IF;
END;

-- 2
DECLARE
  CURSOR grades IS
    SELECT * FROM SALGRADE;
  g grades%ROWTYPE;
BEGIN
  FOR g IN grades LOOP
    DBMS_OUTPUT.PUT_LINE(g.GRADE || ' [' || g.LOSAL || ', ' || g.HISAL || ']');
  END LOOP;
END;

-- 3 + 4
DECLARE
  CURSOR liste_dept IS
    SELECT DEPTNO, DNAME FROM DEPT
    ORDER BY DNAME;
  d liste_dept%ROWTYPE;
  CURSOR liste_emp(dno NUMBER) IS
    SELECT ENAME FROM EMP
    WHERE DEPTNO = dno
    ORDER BY ENAME;
  e liste_emp%ROWTYPE;
  nemp INTEGER;
  dept_vide EXCEPTION;
BEGIN
  FOR d IN liste_dept LOOP
    -- Test d'exception
    SELECT COUNT(*) INTO nemp FROM EMP WHERE DEPTNO = d.DEPTNO;
    IF nemp = 0 THEN
      RAISE dept_vide;
    END IF;
    -- Traitement courant
    DBMS_OUTPUT.PUT(d.DNAME || ':');
    OPEN liste_emp(d.DEPTNO);
    FETCH liste_emp INTO e;
    WHILE liste_emp%FOUND LOOP
      DBMS_OUTPUT.PUT(' ' || e.ENAME);
      FETCH liste_emp INTO e;
    END LOOP;
    CLOSE liste_emp;
    DBMS_OUTPUT.PUT_LINE('');
  END LOOP;
EXCEPTION
  WHEN dept_vide THEN
    RAISE_APPLICATION_ERROR(-20001, 'Un département est vide !');
END;
```

-- 5

```
CREATE OR REPLACE TRIGGER auto_salgrade
BEFORE INSERT OR UPDATE
ON EMP
FOR EACH ROW

BEGIN
  SELECT GRADE INTO :NEW.SALGRADE FROM SALGRADE
    WHERE :NEW.SAL BETWEEN LOSAL AND HISAL;
END;
```