Contrôle de connaissances de Bases de données 2 **07 mars 2013.**

☐ **0**	☐ **0**	☐ **0**
☐ **1**	☐ **1**	☐ **1**
☐ **2**	☐ **2**	☐ **2**
☐ **3**	☐ **3**	☐ **3**
☐ **4**	☐ **4**	☐ **4**
☐ **5**	☐ **5**	☐ **5**
☐ **6**	☐ **6**	☐ **6**
☐ **7**	☐ **7**	☐ **7**
☐ **8**	☐ **8**	☐ **8**
☐ **9**	☐ **9**	☐ **9**

⟵ codez votre numéro de code ci-contre, et inscrivez votre nom et prénom ci-dessous.

Nom et prénom :

.....

.....

Durée : 80 minutes.
Aucun document n'est autorisé.
Les questions faisant apparaître le symbole ♣ peuvent présenter zéro, une ou plusieurs bonnes réponses. Les autres questions ont une unique bonne réponse.
En générale les questions comptent pour 1 point si bonne réponse.
0,5 points négatifs seront affectés par *mauvaise* case coché avec comme plancher -1 point par question.

Requêtes SQL

Une société de taxis tient le bilan des courses qu'elle a effectuées depuis l'aéroport de Toulouse-Blagnac. Les relations sont les suivantes :

Chauffeurs(idCh, nom, prenom, tel). On mémorise le nom, le prénom, le numéro de portable (tel) de chaque chauffeur ainsi que son identifiant unique (idCh).

Courses(idCo, datec, destc, nbpers, kms, idVe, idCh). Les courses ont un identifiant (idCo), elles sont assurées par un chauffeur (idCh), à une certaine date (datec), vers une certaine destination (destc) située à une distance (kms) de l'aéroport, avec un véhicule donné (idVe). Les distances sont exprimées en kilomètres.

Vehicules(idVe, modele, cat). Les véhicules ont un identifiant (idVe), un modèle (modele) et une catégorie (cat). La catégorie est un entier entre 1 et 5 indiquant le confort du véhicule. Elle sert à calculer le prix de base d'une course réalisée avec ce véhicule.

Question 1 ♣ Dans du papier libre donner une requête affichant le nom, le prénom et le téléphone des chauffeurs qui, le 23 février, ont totalisé plus de 500 kms de déplacements.
Ne rien cocher pour cette question, ca sera le correcteur qui cochera un seul choix.

■ OK ■ 3/4 OK ■ 2/4 OK ■ 1/4 OK ☐ 0/4 OK

Question 2 ♣ Dans du papier libre donner une requête affichant l'identifiant, le modèle et la catégorie de *tous* les véhicules du parc, avec l'identifiant des personnes qui les ont conduites le 01/03/2013 *lorsqu'il y en a.*
Ne rien cocher pour cette question, ca sera le correcteur qui cochera un seul choix.

■ OK ■ 3/4 OK ■ 2/4 OK ■ 1/4 OK ☐ 0/4 OK

DF, FN et algorithmes de décomposition 1

Une association de la région nantaise organise des voyages culturels pour ses adhérents. Chaque voyage a une référence (refV). Il coûte un certain prix (prixV) et concerne une ville de départ (villeD) ainsi qu'une ville d'arrivée (villeA), pour laquelle il y a un accompagnateur (idAccomp). Une personne est responsable de l'organisation du voyage (resp). De cette personne, on mémorise son nom, son prénom et son téléphone (tel). Plusieurs trajets peuvent composer le voyage. Un trajet (trajet) concerne un point de départ (ptD), un point d'arrivée (ptA) et s'effectue avec un moyen de transport (idTrans). Par exemple, depuis Blain pour aller à Barcelone, on prendra d'abord le bus jusqu'à l'aéroport Nantes-Atlantique puis l'avion jusqu'à Barcelone. Un moyen de transport a un type (type) et est assuré par une société (idSoc).

Les dépendances fonctionnelles identifiées sont les suivantes :

(1) refV → villeD, villeA, prixV, resp

(2) villeA → idAccomp

(3) trajet → ptD, ptA, idTrans

(4) resp → nom, prenom, tel

(5) resp, villeA → refV

(6) idTrans → type, idSoc

(7) idAccomp → villeA

On considère la relation R munie de l'ensemble DF1.

R(refV, villeD, villeA, prixV, resp, idAccomp, trajet, ptD, ptA, idTrans,nom, prenom, tel, type, idSoc)

DF1 = {(1), (2), (3), (4), (5), (6), (7)}

Question 3 ♣ Etant donné l'ensemble de dependances fonctionnelles DF1 et la relation R cochez lorsque les affirmations sont vraies :

- ☐ Il peut y avoir deux accompagnateurs pour une même ville d'arrivée.
- ■ Une même société peut assurer plusieurs moyens de transports.
- ☐ Une même personne peut être responsable de plusieurs voyages vers une même ville d'arrivée.

Question 4 ♣
Cocher tous les ensembles d'attributs qui forment une clé de R, selon DF1. On utilisera la définition de clé donnée en cours.

- ☐ {villeA, trajet}
- ☐ {refv, resp, villeA, trajet}
- ■ {idAccomp, resp, trajet}
- ☐ {resp, villeA, idTrans}

Question 5 ♣ Cochez les dépendances fonctionnelles déductibles de l'ensemble DF1 :

- ■ idAccomp, resp → villeD
- ☐ villeA → refV, idAccomp
- ☐ trajet, resp → villeD

Question 6 ♣
Dans du papier libre, indiquer tous les schémas de relations obtenus en utilisant l'algorithme de Bernstein pour la relation R et l'ensemble DF1.
Ne cochez aucune case pour cette question, ca sera le correcteur qui cochera un seul choix.

■ OK ■ 3/4 OK ■ 2/4 OK ■ 1/4 OK ☐ 0/4 OK

DF, FN et algorithmes de décomposition 2

Soit le schéma de relation Voitures muni de l'ensemble de dépendances fonctionnelles DF2

Voitures(noI, modèle, marque, puissance, couleur)
DF2 = {noI → modèle ; modèle → marque ; noI → couleur}

Soit l'instance suivante de la relation, notée v.

noI	modèle	marque	puissance	couleur
AD444RT	clio	renault	70	rose
RF555ST	twingo	renault	70	rose
CV777ET	megane	renault	70	rose

Question 7 L'instance v proposée pour la relation Voitures, présente-t-elle un phénomène de redondance d'information lié au fait que le schéma de relation ne vérifie pas plus que la 1ère forme normale (1FN) ?

☐ Oui ■ Non

Question 8 ♣ Soit la décomposition en R1(noI, modèle, couleur) et R2(modèle, marque, puissance) du schèma Voitures. Les dépendances fonctionnelles associées sont les restrictions de DF2 à chaque ensemble. On note r1 et r2 les instances correspondant à la décomposition de l'instance v de Voiture. Parmi les affirmations suivantes, lesquelles sont vraies ?

- ☐ La décomposition est spi car la jointure de r1 et r2 contient exactement les tuples de v
- ☐ La décomposition n'est pas spi car la jointure de r1 et r2 contient plus de tuples que n'en contient v.
- ☐ La décomposition n'est pas spi car l'intersection de R1 et R2 ne détermine pas l'attribut puissance.
- ☐ La décomposition est spi car l'intersection de R1 et R2 détermine l'attribut marque.
- ■ La décomposition n'est pas spi car l'intersection de R1 et R2 ne détermine pas les attributs puissance, noI, couleur.

Question 9 ♣ On se propose d'utiliser l'algorithme de normalisation par décomposition pour la relation Voitures. Parmi les affirmations suivantes cocher celles qui sont vraies :

- ☐ Quel que soit l'ordre dans lequel on considère les dépendances fonctionnelles, on obtient 5 relations.
- ■ Quel que soit l'ordre dans lequel on considère les dépendances fonctionnelles, on obtient 4 relations.
- ☐ Quel que soit l'ordre dans lequel on considère les dépendances fonctionnelles, on obtient 3 relations.
- ☐ Quel que soit l'ordre dans lequel on considère les dépendances fonctionnelles, la décomposition est spdf.
- ■ La décomposition est spdf pour au moins deux ordres de considération des dfs.
- ☐ Il existe un ordre de considération des dfs dans lequel on obtient une relation R(noI, modele, couleur) au niveau de feuilles
- ■ Il existe un ordre de considération des dfs permettant d'obtenir une décomposition spi et spdf où toutes les relations sont en 3FN.

Question 10 On considère maintenant le schéma de relation *Voitures2(noI, modèle, conso, puissance, couleur)* muni de l'ensemble de dépendances DF3 = {noI → modèle ; puissance → conso; noI → couleur}.
On considère la décomposition en :

- R1(noI, modèle, couleur)
- R2(puissance, conso)
- R3(noI, puissance, couleur)

Les dépendances fonctionnelles associées sont les restrictions de DF3 à chacun des ensembles d'attributs (R1, R2, R3).
Cette décomposition est-elle spi ?

■ Oui ☐ Non

PLSQL, Triggers et Vues

Question 11 ♣ Parmi les contraintes suivantes, à votre avis lesquelles ne peuvent pas être implementées avec des triggers ?

- ☐ NOT NULL
- ☐ UNIQUE
- ☐ PRIMARY KEY
- ☐ FOREIGN KEY
- ☐ CHECK
- ■ Toutes ces contraintes peuvent être implémentées avec des triggers

Question 12 ♣ Une vue est une requête stockée qui est interrogée comme une table. Pour l'exécution de requêtes sur les vues, le SGBD traduit les requêtes à l'aide de techniques de ?

- ☐ Optimisation de requêtes
- ☐ Evaluation de requêtes
- ■ Ré-écriture de requêtes
- ■ Concaténation d'arbres relationnels

Question 13 Un trigger met en place une règle Evénément-Condition-Action (ECA). Dans le trigger nombres_table_after_insert_row utilisé dans l'une de questions de cette section, dans quelle ligne se trouve la condition ?

☐ 1 ☐ 2 ☐ 3 ■ 4

Question 14 Considerez la relation vide *nombres(a number, b number)* ainsi que le trigger suivant.[a]

```
CREATE OR REPLACE TRIGGER nombres_table_after_insert_row
AFTER INSERT ON nombres
FOR EACH ROW
WHEN (new.a * new.b > 10)
BEGIN
  INSERT INTO nombres VALUES(:new.a-1, :new.b+1);
END;
/
```

Quel tuple ne vas pas générer l'insertion de deux tuples suplémentaires dans la relation nombres ?

■ (3,4) ☐ (3,5) ☐ (4,3) ☐ (3,8)

[a] Dans ce trigger on fait abstraction du code pour générer de transactions autonomes afin d'éviter le problème de table mutante.

Question 15 Considerez la relation *nombres(a number, b number)* avec comme tuples pour les attributs $(a, b) = (1, 2), (2, 3), (3, 4), (4, 5), (5, 6)$.
Considerez le code PLSQL suivant.

```
CREATE OR REPLACE FUNCTION calculer(nombre IN NUMBER)
RETURN NUMBER
IS
  cal NUMBER DEFAULT 1;
BEGIN
  FOR i IN 1..nombre
  LOOP
      cal:=cal*i;
  END LOOP;
  RETURN cal;
END;
/
CREATE OR REPLACE PROCEDURE upd_relation IS
     CURSOR C1 IS SELECT * FROM nombres;
BEGIN
    FOR nb IN C1
    LOOP
        UPDATE nombres SET b=calculer(nb.a) WHERE a=nb.a;
    END LOOP;
END;
/
```

Quelles seront les tuples de la relation après avoir appelé la procedure upd_relation ?

- ☐ (1,1),(2,4),(3,9),(4,16),(5,25)
- ■ (1,1),(2,2),(3,6),(4,24),(5,120)
- ☐ (1,2),(2,6),(3,12),(4,20),(5,30)

Question 16 Considérez la relation *contacts(nom, prenom, adresse, téléphone, relation)* où l'attribut relation peut avoir comme valeurs (famille|ami|connaissance|VIP). La vue contacts_famille suivante, peut-elle être mise à jour correctement et directement ?
CREATE VIEW contacts_famille AS
SELECT nom, prenom, telephone, adresse
FROM contacts
WHERE relation ='famille';

☐ Oui ■ Non

Objet Relationnel

Question 17 ♣ Dans l'approche objet-relationnel, l'utilisateur a la possibilité de créer de types. Les types définis peuvent être pour créer :

- ☐ Une fonction
- ☐ Une procèdure
- ■ Une table
- ☐ Une vue
- ■ Un attribut d'une table
- ■ Un attribut dans un autre type

Question 18 ♣ Un type peut être constitué :

- ☐ Des types
- ■ D'un constructeur
- ■ Des méthodes
- ☐ Des contraintes
- ■ Des attributs