1/12

E4D : ETUDE DE CAS

Durée : 5 heures **Coefficient : 5**

CAS SECOLOG

Éléments de correction

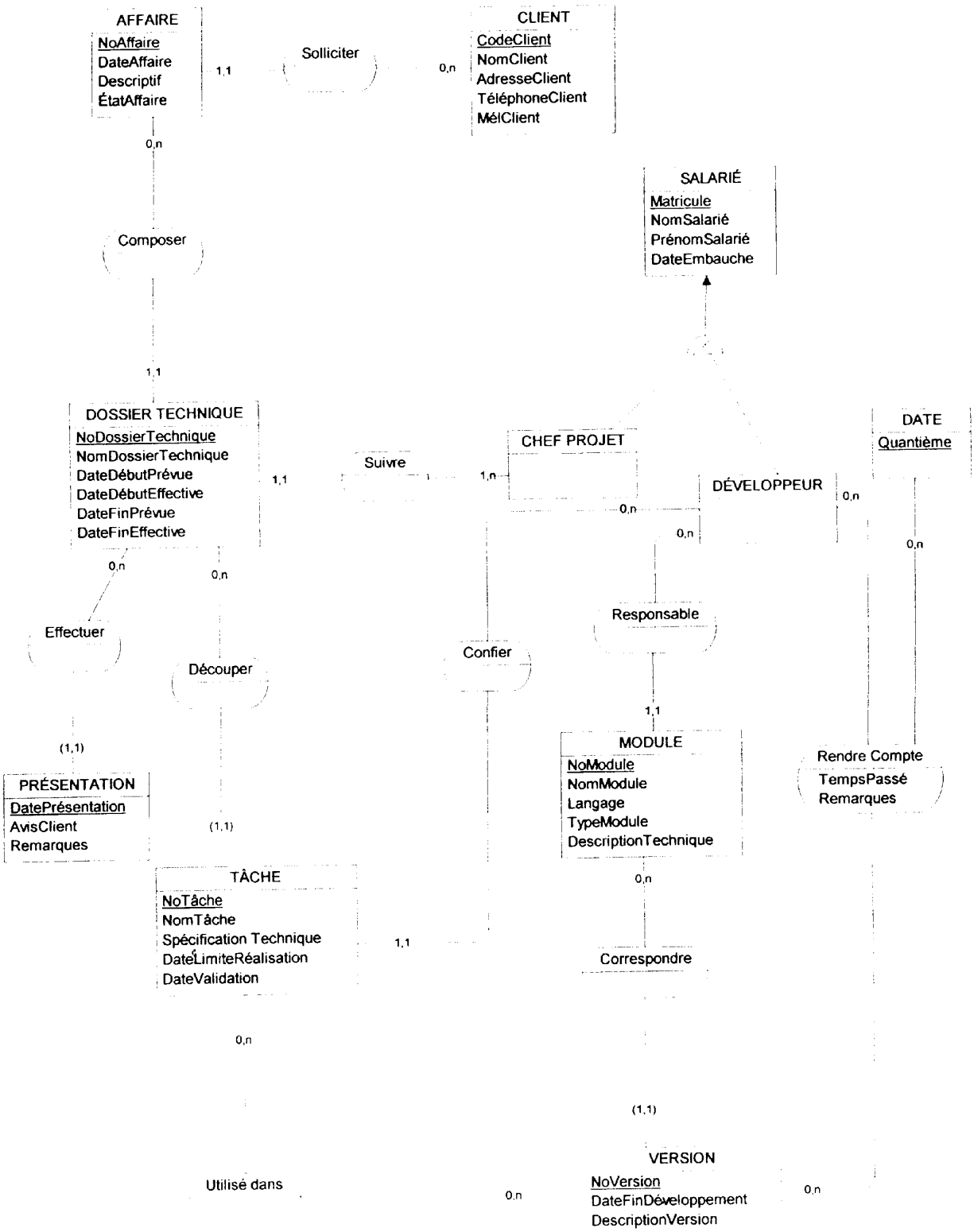
Barème

Dossier 1	Gestion des développements	4,5 points
Dossier 2	Gestion des propositions commerciales	2,5 points
Dossier 3	Gestion des formations	4 points
Dossier 4	Gestion des sessions de formation	6 points
Dossier 5	Équipement des salles de formation	3 points
	Total	20 points

Dossier 1 : Gestion des développements

4,5 points soit 22,5/100

2/12



3/12

Remarques :

- La notion de présentation peut être considérée comme une association *présenter* liant les entités *DOSSIER TECHNIQUE* et *DATE*. Dans ce cas, l'association porte les propriétés *AvisClient* et *Remarques*.
- L'état d'une affaire peut être considéré comme une entité *ÉTAT* identifiée par un code et associée à l'entité *AFFAIRE* pour mémoriser l'état actuel d'une affaire.
- La spécialisation des salariés n'est pas indispensable mais son absence conduirait à écrire des contraintes inter-relations un peu lourdes.
- On admettra l'association *valider* entre les entités *CHEF PROJET* (0,n) et *TÂCHE* (0,1), porteuse de la propriété *DateValidation*.
- On admettra la présence d'une entité *TYPE MODULE*.
- On admettra la présence d'une entité *COMPTE RENDU*, identifiée artificiellement, en lieu et place de l'association *Rendre compte*.

Bloc 1 « Affaire » ..... 7,5 points
Entités : *Affaire, Client, Dossier technique, Tâche*
Associations : *composer, découper, solliciter.*
On évalue particulièrement :
- la propriété *ÉtatAffaire* dans l'entité *Affaire* (ou une variante admise),
- la propriété *DateValidation* dans l'entité *Tâche* (ou une variante admise),
- l'identification relative de l'entité *Tâche*.

Bloc 2 « Salarié » ..... 5 points
Entités : *Salarié, Chef de projet, Développeur*
Associations : *suivre et confier.*

Bloc 3 « Module » ..... 5 points
Entités : *Module, Version*
Associations : *correspondre, Utilisé dans*
On évalue particulièrement :
- l'identification relative des versions.

Bloc 4 « Compte rendu » : ..... 3 points
Entité : *Date*
Association : *rendre compte*
(ou une variante admise)

Bloc 5 « Présentation » : » : ..... 2 points
Entité : *Présentation*
Association : *effectuer*
(ou une variante admise)*Proposition de barème :*

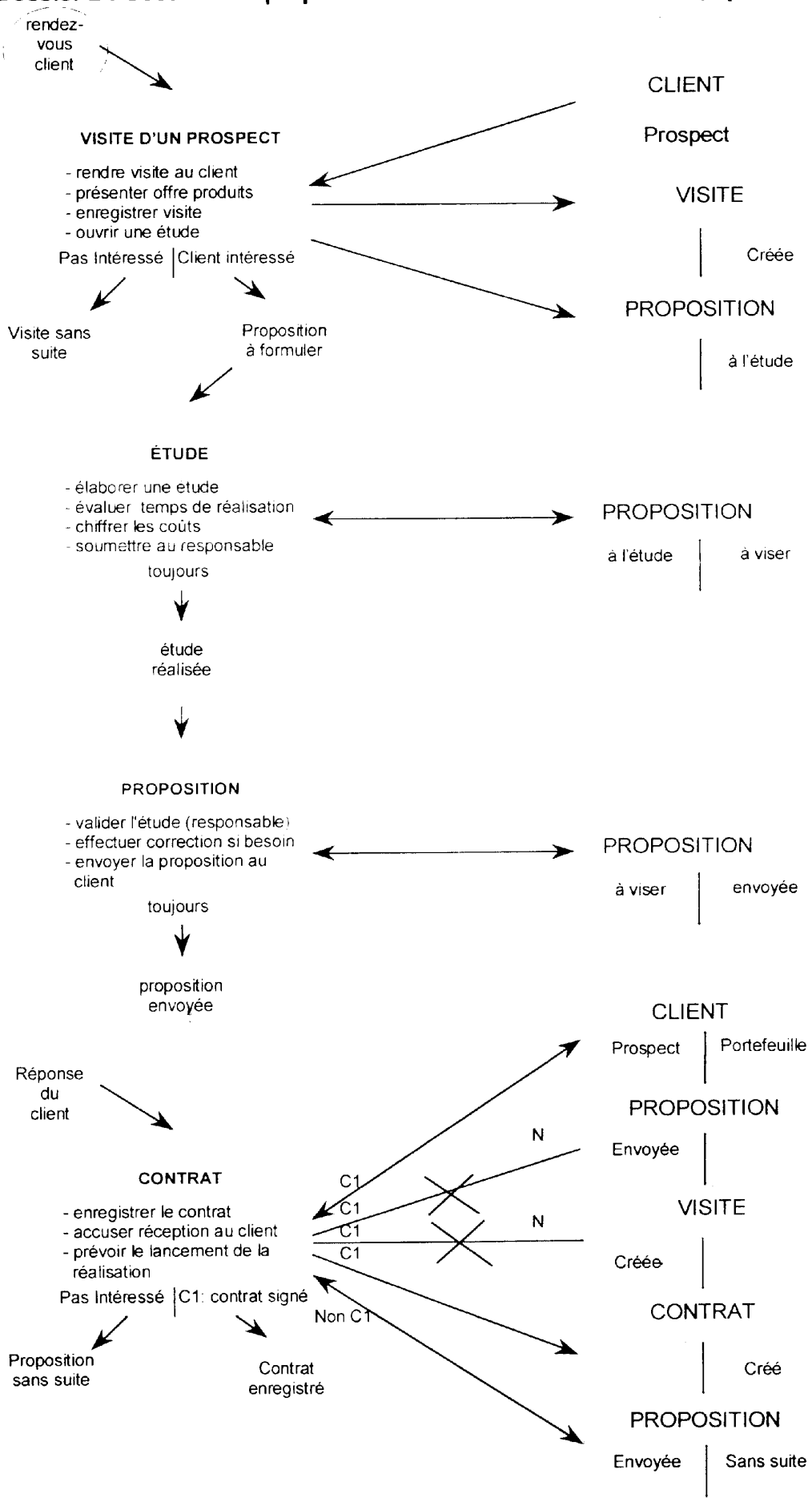
Bloc 1 « Affaire » :	**7.5 points**
Bloc 2 « Personnel » :	**5 points**
Bloc 3 « Module » :	**5 points**
Bloc 4 « Compte Rendu» :	**3 points**
Bloc 5 « Présentation » :	**2 points**
Total :	**22,5 points sur 100**

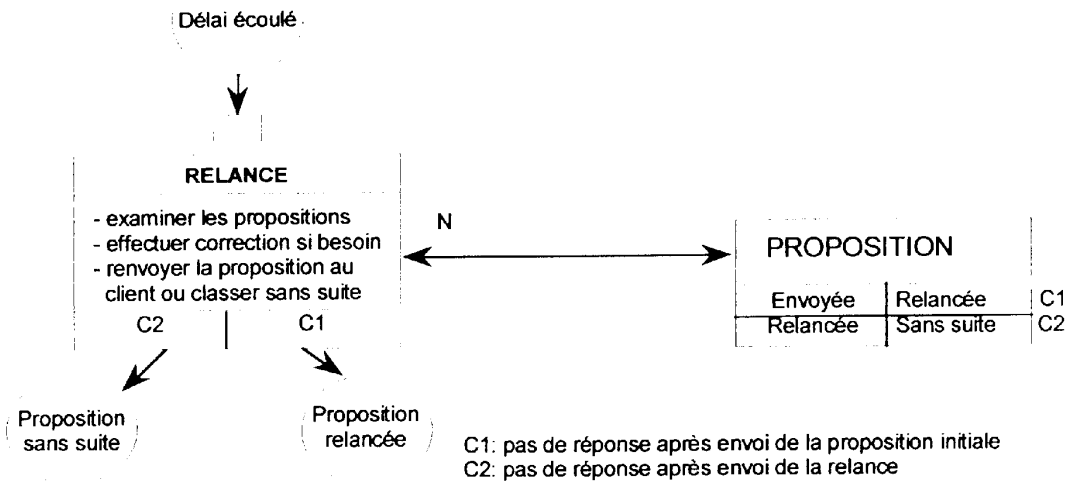
4,5/20

Dossier 2 : Gestion des propositions commerciales

2,5 points soit 12,5/100

4/12





5/12

Remarques :

- Le formalisme MCTA de Merise/2 n'est pas requis.
  - La décomposition en opérations fondée sur les états cohérents du système, faisant apparaître des événements internes entre opérations, est propre à Merise/2 : elle ne constitue pas une exigence du référentiel du BTS et ne s'impose donc pas. Il faudra veiller à ne pas pénaliser le candidat qui aurait fondé sa décomposition sur la notion d'opération ininterruptible utilisée dans Merise et l'aurait appliquée de façon cohérente : on admettra qu'un seul bloc modélise les trois opérations *VISITE*, *ÉTUDE* et *PROPOSITION* ; une solution à deux opérations VISITE/ÉTUDE et PROPOSITION, cette dernière déclenchée par l'accord du responsable, sera aussi acceptée.
  - Il conviendra cependant d'évaluer la complétude de la solution proposée par le candidat, notamment en ce qui concerne la façon dont les opérations agissent sur les données, une exigence du sujet.
- Le N, indiquant que l'opération peut traiter plusieurs occurrences, n'est pas exigé.
- La décomposition des opérations en suite d'actions n'est pas demandée.
- L'opération Relance peut être décomposée en deux opérations.

Proposition de barème :

Identification des opérations :	**5 points**
Identification des déclencheurs :	**2 points**
Identification des objets du MCD et des opérations effectuées :	**3,5 points**
Identification des conditions d'exécution :	**2 points**
Total :	**12,5 points sur 100** — **2,5/20**

6/12

Dossier 3 : Gestion des formations

a) De créer la tables SESSION en précisant les contraintes de clé primaire et de clé étrangère.

```
create table SESSION
(   CodeForm     CHAR(6)      not null    REFERENCES FORMATION,
    NoSession    NUMBER(2)    not null,
    DateSession  DATE         not null,
    Matricule    CHAR(8)      not null    REFERENCES SALARIÉ,
    PRIMARY KEY (CodeForm, NoSession)
) ;
```

b) Afficher les dates de session, les libellés de formation et le nom des produits étudiés au cours des sessions de formation suivies par le stagiaire numéro 911 pendant la période comprise entre le 1er mars 2000 et le 15 mai 2000.

```
SELECT  DateSession, LibelléForm, NomProduit
   FROM  PARTICIPER p, SESSION s, FORMATION f, PRODUIT pr
   WHERE  p.CodeForm = s.CodeForm
      AND  p.NoSession = s.NoSession
      AND  s.CodeForm = f.CodeForm
      AND  f.CodeProduit = pr.CodeProduit
      AND  NoStagiaire = 911
      AND  DateSession BETWEEN '01/03/2000' AND '15/05/2000' ;
```

La présence de la relation « STAGIAIRE » est inutile.

c) Afficher le numéro des formateurs et le nombre de sessions de formation qu'ils ont assurées. Seuls les formateurs ayant assuré au moins 10 sessions de formation feront partie du résultat.

```
SELECT Matricule, COUNT(*) AS Nombre_de_formations
FROM SESSION
GROUP BY Matricule
HAVING COUNT(*) >= 10 ;
```

L'alias de colonne n'est pas exigé.

On trouvera probablement des requêtes avec une jointure :

```
SELECT SALARIÉ.Matricule, COUNT(*) AS Nombre_de_formations
FROM SALARIÉ, SESSION
WHERE SALARIÉ.Matricule = SESSION.Matricule
GROUP BY SALARIÉ.Matricule
HAVING COUNT(*) >= 10 ;
```

d) Supprimer la deuxième session d'une formation dont le code est égal à 'F405'.

```
DELETE FROM SESSION
WHERE CodeForm = 'F405'
AND NoSession = 2 ;
```

e) Autoriser l'utilisateur DUPARC à insérer, mettre à jour et supprimer des lignes dans la table SALARIÉ.

```
GRANT INSERT, UPDATE, DELETE ON SALARIÉ TO DUPARC ;
```

ISE4D

Proposition de barème :

Requête a	**5 points**	7/12
Requête b	**4 points**	
Requête c	**4 points**	
Requête d	**4 points**	
Requête e	**3 points**	
Total :	**20 points sur 100**	**4/20**

8/12

Dossier 4 : Gestion des sessions de formation

4.1 Écrire l'algorithme d'une fonction qui compte le nombre de participants à une session de formation passée en argument.

Fonction CompteParticipants (données f : chaîne ; s : entier) : entier

```
{
    rôle des arguments :
    f : code de la formation
    s : numéro de session
}

Variables
    SQLStr : chaîne              {instruction SQL}
    jEnr : JeuEnregistrements    {objet de la classe JeuEnregistrements}
    Compteur : entier            {variable de travail}
```

3 solutions sont envisageables :

- *1e solution : Utilisation de la classe et décompte par parcours séquentiel*

```
Début
    {construction de l'instruction SQL à l'aide des arguments passés}
    SQLstr ← "SELECT * FROM PARTICIPER WHERE CodeForm =" + f + " AND NoSession = " + s + " ;"
    {appel du constructeur : instanciation de l'objet}
    jEnr.Initialiser (SQLStr)
    Compteur ← 0
    {test de l'objet : est-il vide ?}
    Si non jEnr.Fin
        alors  {parcourir le jeu d'enregistrements et compter les lignes}
            Répéter
                Compteur ← Compteur + 1
                jEnr.Avancer
            jusqu'à jEnr.Fin
    Fin si
    jEnr.Fermer
    Retourner Compteur
Fin
```

- *2e solution : Utilisation de la classe et décompte par la requête SQL*

```
Début
    CompteurParticipants ← 0
    {construction de l'instruction SQL à l'aide des arguments passés}
    SQLstr ← "SELECT COUNT(*) AS Nbre
                    FROM PARTICIPER
                    WHERE CodeForm = " + f + " AND NoSession = " + s + " ;"
    {appel du constructeur : instanciation de l'objet}
    jEnr.Initialiser (SQLStr)
    Si jEnr.ExtraitValeur("Nbre") <> Null
        alors  CompteParticipants ← jEnr.ExtraitValeur("Nbre")
    Fin si
    jEnr.Fermer
Fin
```

- *3e solution : Sans utilisation de la classe*

```
Début
    SELECT COUNT(*) into :nbre
    FROM PARTICIPER
    WHERE CodeForm = :f AND NoSession = :s ;
    Si sqlcode=0
        alors Retourner nbre
        sinon Retourner 0
    Fin si
Fin
```

9/12

Dans cet exercice, on évalue :

- La requête SQL. On sera tolérant sur la façon dont sont concaténés les chaînes et les arguments de la fonction.
- L'instanciation de l'objet par l'emploi correct du constructeur.
- Le parcours du curseur par l'emploi des méthodes de l'objet (ou la récupération de la valeur en cas d'emploi d'un COUNT).
- Le retour de la valeur obtenue dans la fonction : tous les formalismes algorithmiques sont acceptés.

4.2 Écrire l'algorithme de la procédure événementielle associée au choix d'une session (la procédure sera identifiée par l'entête cbxSessions_SurChangement()

```
Procédure cbxSessions_SurChangement()
Variables
   vCodeForm : chaîne          {variable de travail pour récupérer le code formation sélectionné}
   vNoSession : entier         {variable de travail pour récupérer le numéro de session sélectionné}
   jEnr : JeuEnregistrements   {objet de la classe JeuEnregistrements}
   SQLstr : chaîne             {instruction SQL}
   Ligne : chaîne              {ligne à ajouter dans la zone de liste}
Début
{vider la liste}
      zdlParticipants.ViderListe
{récupération des valeurs sélectionnées }
      vCodeForm ← cbxFormations.Valeur
      vNoSession ← cbxSession.Valeur
{mise à jour de l'étiquette}
      etqNbParticipants.Légende ← CompteParticipants(vCodeForm, vNoSession)
      Si CompteParticipants(vCodeForm, vNoSession) <> 0
         alors
            { instanciation d'un objet JeuEnregistrements}
            SQLStr ← "SELECT NoStagiaire, NomStagiaire, PrénomStagiaire, NomClient
               FROM STAGIAIRE s, PARTICIPER p, CLIENT c
               WHERE s.NoStagiaire = p.NoStagiaire
               AND s.NoClient = c.NoClient
               AND p.CodeForm = "  + vCodeForm + "
               AND p.NoSession = " + vNoSession + " ;"
            jEnr.Initialiser(SQLStr)
            {remplissage de la liste}
            répéter
               Ligne ← jEnr.ExtraitValeur("NoStagiaire") +" ;" +
                        jEnr.ExtraitValeur("NomStagiaire") + " ;" +
                        jEnr.ExtraitValeur("PrénomStagiaire") + " ;" +
                        jEnr.ExtraitValeur("NomClient")
               zdlParticipants.AjouteLigne(Ligne)
               jEnr.Avancer
            jusqu'à jEnr.Fin
            jEnr.Fermer
      Fin si
Fin
```

Dans cet exercice, on évalue :

- La requête SQL.
- L'instanciation du curseur.
- La « construction » d'une ligne : on évalue essentiellement l'emploi de la méthode *ExtraitValeur* et la façon dont est construite la chaîne (présence du point-virgule pour séparer les colonnes).
- Le vidage de la liste.
- L'insertion itérative des lignes dans la zone de liste.
- La mise à jour de l'étiquette par l'emploi de la fonction développée dans l'exercice précédent.

10/12

4.3 La fonction CompteParticipants étant utilisée dans plusieurs applications, il est envisagé de la stocker dans la base de données.

- **– Préciser le type de client-serveur auquel correspond cette nouvelle architecture applicative.**

Il s'agit de mettre en œuvre une architecture client-serveur de traitement.

- **– Expliquer la démarche à suivre pour créer et utiliser un traitement stocké (procédure ou fonction) dans une base de données.**

La démarche est la suivante :

1. **Création d'une fonction ou d'une procédure stockée dans la base de données, en utilisant le langage fourni avec le SGBDR.**

   Exemple avec Oracle (langage PL/SQL) :

```
CREATE FUNCTION CompteParticipants RETURN number AS
BEGIN
        /* code procédural de la fonction */
END ;
```

   Avec les SGBDR ne supportant pas l'ordre CREATE FUNCTION, on utilisera l'ordre CREATE PROCEDURE.

```
CREATE PROCEDURE CompteParticipants (NbPart number OUT) AS
BEGIN
        /* code procédural de la procédure */
END ;
```

2. **Appel de la fonction ou de la procédure stockée à partir de l'application cliente,**

selon l'une des modalités suivantes :

- /* pour une fonction */
  Resultat = CompteParticipants
- /* pour une procédure */
  EXEC (ou CALL) CompteParticipants(vCodeForm ; vNoSession ; Résultat)

Proposition de barème :

4.1 : Fonction CompteParticipants	**10 points**
4.2 : Procédure cbxSession_SurChangement()	**15 points**
4.3 : Fonction ou procédure stockée	**5 points**
Total :	**30 points sur 100** **6/20**

Dossier 5 : Équipement des salles de formation

11/12

5.1 Sans entrer dans le détail de la configuration technique des postes et du serveur, rédiger une courte note sur le choix d'une architecture qui permettra de connecter les postes des ateliers AF-1 et AF-2 au serveur (topologie, câblage, connecteurs réseau, électronique active, systèmes d'exploitation, ...).

On peut imaginer plusieurs solutions :

- Une topologie en étoile (10BaseT ou 100BaseT). La carte réseau du serveur est connectée à un *hub* (24 ports, 10 ou 100 Mbit/s) par l'intermédiaire d'un câble en paires torsadées équipées de connecteurs RJ45. La carte réseau de chaque poste est connectée au *hub* de la même façon.
  On peut aussi imaginer une solution dans laquelle on utilise deux *hubs* 12 ports (un pour chaque atelier) cascadés.
- Une topologie en bus (10Base2) – on a aucune information sur l'existant – pour l'atelier AF-2 relié à une carte réseau avec connecteur BNC sur le serveur à l'aide d'un câble coaxial fin, de connecteurs en T et de bouchons de terminaison (cette solution peut être généralisée à condition de changer la carte réseau des postes de l'atelier AF-1).
- La technologie Apple en réseau AppleTalk ...

Système d'exploitation :

- Serveur en MS Windows NT (4 ou 2000) ou NOVELL NETWARE 4 ou 5, avec postes en MS Windows NT 4 WorkStation, ou MS Windows 95, MS Windows 98, MS Windows 2000, BeOS, OS/2 WARP, ...
- Solution tout Linux avec un serveur Linux et des postes Linux ou mixte (serveur Linux, clients Windows).
- Solution tout Apple et un réseau AppleTalk.

5.2 Présenter le plan d'amortissement linéaire du serveur

La somme à amortir est de 30 000 FRF HT sur 3 ans. La date de mise en service est le 1er avril 2000.

Année	Valeur initiale	Annuité	Valeur résiduelle	Commentaire
2000	30000	7500	22500	1e annuité : 30000 * 1/3 * 270/360 = 7500
2001	22500	10000	12500	2e et 3e annuité : 30000 * 1/3 = 10000
2002	12500	10000	2500	
2003	2500	2500	0	4e annuité : 30000 * 1/3 * (360-270)/360 = 2500

12/12

5.3 Tableau des immobilisations de l'exercice 2001 : Matériel informatique

Éléments	Valeur brute à l'ouverture de l'exercice	Augmentations	Diminutions	Valeur brute à la clôture de l'exercice
Serveurs	150 000	30 000		180 000
Postes de travail	198 600			198 600
Imprimantes	60 000			60 000
Scanners	5 400			5 400
TOTAL	**414 000**	**30 000**	**0**	**444 000**

5.5 Tableau des amortissements de l'exercice 2001 : Matériel informatique

Éléments	Amortissements cumulés au début de l'exercice	Augmentations	Diminutions	Amortissements cumulés à la fin de l'exercice
Serveurs	90 000	57 500		147 500
Postes de travail	123 200	66 200		189 400
Imprimantes	25 000	20 000		45 000
Scanners	1 800	1 800		3 600
TOTAL	**240 000**	**145 500**	**0**	**385 500**

5.6 Extrait du bilan au 31/12/2001

	Brut	Amortissements	Net
Matériel informatique	444 000	385 500	58 500

Proposition de barème :

Architecture du réseau	**3 points**
Plan d'amortissement linéaire	**4 points**
Tableau des immobilisations 2001	**3 points**
Tableau des amortissements 2001	**3 points**
Extrait du bilan au 31/12/2001	**2 points**
Total :	**15 points sur 100**

3/20