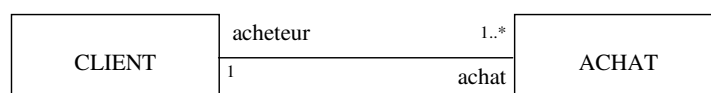


NFP107 (Partie SQL)
EXAMEN, 1^{ère} session – juin 2008
(Tout document écrit autorisé)

Considérons une société de la grande distribution désirant dresser le profil commercial de ses clients. Pour ce faire elle met en place une base de données contenant l'ensemble de ses clients possédant une carte de fidélité de l'enseigne et les achats qu'ils effectuent. Les informations importantes pour l'enseigne sont la référence des produits achetés, la quantité, le montant dépensé pour le produit et le montant total par jour. Ces informations doivent être conservées pour chaque visite d'un client. Chaque mois, le profil du « *client du mois* » est dressé. Ce « client » récapitule pour chaque type de produit le produit le plus acheté dans le mois ainsi que la quantité moyenne acquise par les clients. Afin de ne pas compliquer le schéma de la base ce client particulier est inséré dans la relation « CLIENT » avec un identifiant commençant par les lettres « CM » suivi du mois et de l'année sur deux lettres chacun. Les autres champs ne sont pas renseignés. L'identifiant d'un client « normal » commence par « N » suivi de cinq chiffres.

Le diagramme de classes de cette application est le suivant :



Un client est caractérisé par son nom, son prénom, son identifiant, sa date de naissance, son adresse et son code postal. Un achat est caractérisé par la référence du produit acheté, la quantité et le montant total de l'achat. La référence du produit est composée de sept caractères dont les trois premiers correspondent au type du produit. Par exemple, « ELM0001 » correspond à un produit électroménager et « HIF1021 » à de la HIFI. Le fait qu'un client vienne plusieurs fois dans une même journée ne présente pas d'intérêt dans ce problème et ce fait n'est pas pris en compte. On considérera donc que l'ensemble des achats d'une même journée correspond à une visite.

Un schéma relationnel correspondant au schéma conceptuel ci-dessus est le suivant :

CLIENT = {IDCLI, NOM, PRENOM, DDN¹, ARD, CPOSTAL}

ACHAT² = {REFPROD, QTE, MONTANT, DACH³, #IDCLI}

I : Compréhension du schéma

1. Quelle(s) contrainte(s) faudrait-il implanter pour mettre en œuvre au mieux le schéma UML ci-dessus ? indiquez celles qui peuvent être implantées en SQL et celles qui ne le peuvent pas ? dans ce dernier cas, vous expliquerez pourquoi ? (0,75 pt)

La clef étrangère « IDCLI » de la relation « ACHAT » doit être non « null » afin d'assurer que tout achat est fait par un client (0,5 pt). Toutefois, la contrainte qui dit qu'un client a réalisé au moins un achat ne peut pas être vérifiée (0,25 pt).

2. Proposez un exemple pertinent de cette base de données en fournissant quelques enregistrements pour chaque relation. (0,5 pt)

CLIENT	IDCLI	NOM	PRENOM	DDN	ARD	CPOSTAL
	N00001	Dupont	Louis	17/12/1980	Rue des Lilas	31100
	CN0508					
	N00002	Dupin	Pierre	11/10/1953	Rue des Lys	31120

¹ Date de naissance

² Cette relation n'a pas de clef primaire car un identifiant n'a pas d'intérêt dans le cadre de ce sujet

³ Date à laquelle l'achat est effectué. La date est une chaîne de **huit caractères** ayant le format suivant : « JJMMAAAA ». Les différents constituants de la date peuvent être extraits avec la fonction « substr ».

ACHAT	IDCLI	REFPROD	QTE	MONTANT	DACH
	N00001	ELM0001	1	100,00	12/05/2008
	N00001	ELM0002	1	90,00	12/05/2008
	N00001	HIF1021	1	600,00	12/05/2008
	N00002	HIFI1022	3	30,00	13/05/2008
	CN0508	HIFI1022	2	20,00	01/06/2008

II : Donnez les requêtes suivantes dans le langage SQL

1. Calculez pour chaque client et chaque visite le montant dépensé (IDCLI, DACH, le montant) (0,75 pt)

```
select IDCLI, DACH, sum(MONTANT) /
from ACHAT
group by IDCLI, DACH //
```

2. Pour chaque « client du mois », son identifiant et la liste des achats qui lui sont attribués. Le résultat devra être classé par ordre alphabétique (0,75 pt)

```
select IDCLI, REFPROD, QTE, MONTANT /
from ACHAT
where IDCLI like 'CN%' /
order by IDCLI ; /
```

Enlever 0,25 pt si la clause where teste la valeur NULL sur le nom

3. Les articles achetés par le client « N12345 » le « 12122007 » (0,5 pt)

```
select *
from ACHAT
where IDCLI = 'N12345' and DACH = '12122007' ;
```

4. Les nom et prénom des clients habitant dans la zone de code postal « 31200 » (0,5 pt)

```
select NOM, PRENOM
from CLIENT
where CPOSTAL = '31200' ;
```

5. La référence et le nombre **total** de produits vendus pour le mois de juin 2007 (0,75 pt)

```
select REFPROD, sum(QTE) /
from ACHAT
where DACH like "%062007" /
group by REFPROD / ;
```

Cette requête ne doit pas contenir de group by /

6. L'identifiant des clients « normaux » ayant acheté au mois de juin 2007 la totalité des produits attribués au « client du mois » de juin 2007. Attention ces produits peuvent avoir été achetés sur plusieurs journées. (1,5 pts)

```
select IDCLI from CLIENT /
where not exists / ( select REFPROD from ACHAT where IDCLI = 'CN0607' /
minus /
select REFPROD from ACHAT where ACHAT.IDCLI = CLIENT.IDCLI /
and DACH like "%062007" / )
and IDCLI like 'N%' ;
```

7. Le nom et l'adresse des clients n'ayant jamais acheté des produits HIFI dans le magasin (minus, not exists, not in) (2,5 pts)

select IDCLI, NOM, ADR from CLIENT /

where IDCLI not in / (select IDCLI from ACHAT where REFPROD like "HIF%") / ; (0,75 pt)

select IDCLI, NOM, ADR from CLIENT /

where not exists / (select * from ACHAT
 where REFPROD like "HIF%"
 and ACHAT.IDCLI = CLIENT.IDCLI /) ; (0,75 pt)

select IDCLI, NOM, ADR from CLIENT /

minus /

select IDCLI, NOM, ADR from ACHAT, CLIENT /

where REFPROD like "HIF%" /
 and ACHAT.IDCLI = CLIENT.IDCLI / ; (1,0 pt)

8. L'identifiant des produits qui n'ont pas été vendus au mois de décembre 2007 (0,75 pt)

select REFPROD from ACHAT /

minus /

select REFPROD from ACHAT

where DACH like "%122007" ; /

9. Vu le nombre de données manipulées et en tenant compte des informations données en cours, proposez quelques solutions pour avoir les meilleures performances possibles pour cette application tout en optimisant l'occupation sur disque. (0,75 pt)

Proposer pour chaque requête complexe plusieurs solutions et les tester afin de choisir la plus performante ;

Eviter les parties de requêtes inutiles ;

Eviter les donner redondantes ;

Etudier les plans de tests ;

... ;