UNIVERSITÉ DE MONTPELLIER
-
FACULTÉ DES SCIENCES

Session : 1	Durée de l'épreuve : 2 heures
Date : janvier 2019	Documents autorisés : tous
Mention Informatique	Matériel utilisé : aucun
Master $2^{ème}$ année : Administration BD (HMIN328)	

1 BD exemple : les courses nautiques

1.1 Enoncé

La base de données considérée, se propose de gérer de manière très partielle les courses de bateaux au grand large. Seuls les courses, les épreuves au sein des courses et les skippers (navigateurs) qui participent à ces épreuves, sont pris en charge par la modélisation. Ainsi les navigateurs sont identifiés par un numéro, et caractérisés par un nom, un prénom, un genre, un pays (nationalité) et une date de naissance. Une course est identifiée à la fois par un libellé de course (route du rhum, vendée globe, transat Jacques Vabre, ...) et une date de départ, et est caractérisée par un lieu de départ et un lieu de destination. Une épreuve dépend d'une course, et est identifiée par un numéro, et caractérisée par une catégorie de bateau. Il y a plusieurs épreuves dans une course, et donc plusieurs classements dans une course, en fonction de la catégorie du bateau. Ainsi, pour la toute dernière route du rhum de novembre 2018, la modélisation distingue 4 épreuves, une par catégorie de bateau (Ultime, R-Multi, Imoca et Class40). Un navigateur ne participe qu'à une épreuve à la fois, dans une catégorie bien spécifique mais peut participer dans le temps, à différentes épreuves. Une épreuve met en concurrence différents navigateurs par catégorie de bateau. Un navigateur concourt sur un bateau qui possède un nom et obtient un classement à l'issue de sa participation à l'épreuve. Le schéma relationnel de la base de données vous est donné.

1.2 Schéma Relationnel

Les attributs portant les contraintes de clés primaires sont en gras. Les attributs portant les contraintes de clés primaires sont en gras. Les contraintes de clés étrangères sont précisées à l'aide de ♯. Les types des attributs vous sont également indiqués.

- Course(**libelle VARCHAR(25), date_depart DATE**, lieu_depart VARCHAR(20), lieu_arrivee VARCHAR(20))
- Epreuve(**numEpreuve NUMBER(3)**, categorie_bateau VARCHAR(10), ♯libelle VARCHAR(25), ♯date_depart DATE,)
- Navigateur(**numNavigateur NUMBER(5)**, nom VARCHAR(15), prenom VARCHAR(15), genre VARCHAR(1), dateNaissance DATE, pays VARCHAR(15))
- Participe(**♯numEpreuve NUMBER(3), ♯numNavigateur NUMBER(5)**, classement integer, nomBateau VARCHAR(25))

2 Optimisation de requêtes

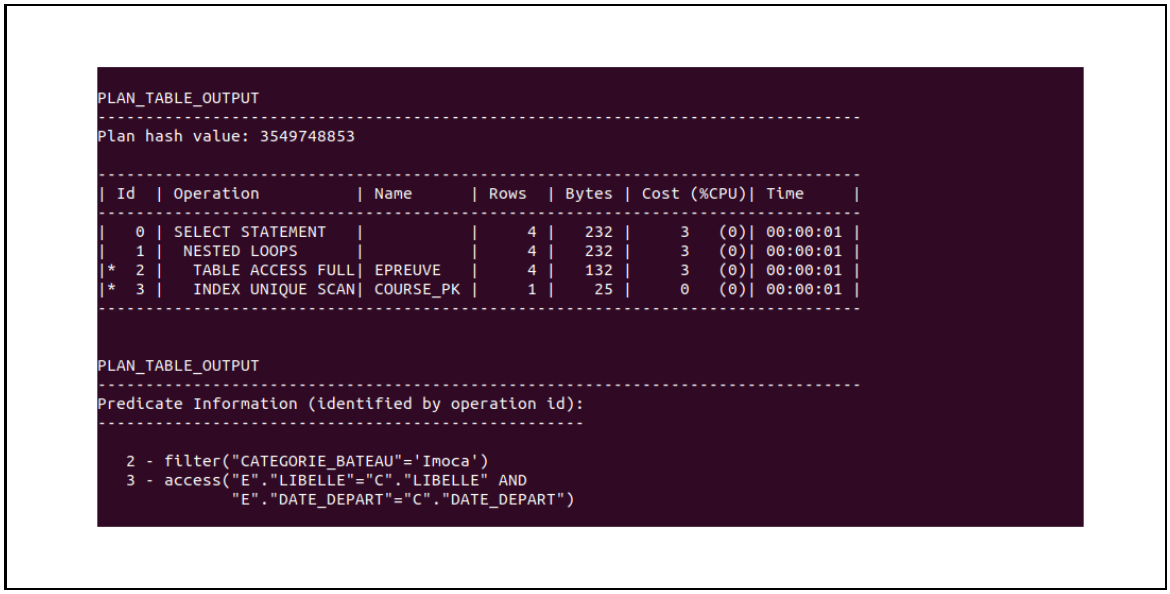
2.1 Plan d'exécution d'une requête

2.1.1 Question 1 (3 points)

Expliquer la sémantique associée à la requête suivante et décrivez le plan d'exécution obtenu (figure 2.1.1). En particulier, vous indiquerez si un index est utilisé pour améliorer les performances du calcul. Vous donnerez le nom de cet index et désignerez les attributs

sur lesquels s'applique cet index. Quel est l'opérateur physique exploité pour le calcul de la jointure ? Vous construirez l' arbre algébrique associé au plan d'exécution.

```
explain plan for select e.date_depart from epreuve e, course c
where e.libelle = c.libelle and e.date_depart = c.date_depart
and categorie_bateau = 'Imoca';

select plan_table_output from table (dbms_xplan.display()) ;
```

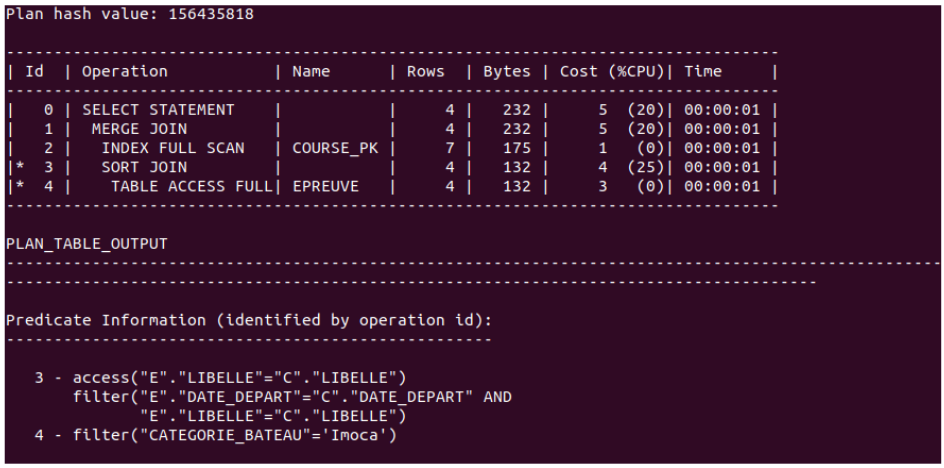


```
PLAN_TABLE_OUTPUT
--------------------------------------------------------------------------------
Plan hash value: 3549748853

--------------------------------------------------------------------------------
| Id  | Operation          | Name      | Rows  | Bytes | Cost (%CPU)| Time     |
--------------------------------------------------------------------------------
|   0 | SELECT STATEMENT   |           |     4 |   232 |     3   (0)| 00:00:01 |
|   1 |  NESTED LOOPS      |           |     4 |   232 |     3   (0)| 00:00:01 |
|*  2 |   TABLE ACCESS FULL| EPREUVE   |     4 |   132 |     3   (0)| 00:00:01 |
|*  3 |   INDEX UNIQUE SCAN| COURSE_PK |     1 |    25 |     0   (0)| 00:00:01 |
--------------------------------------------------------------------------------


PLAN_TABLE_OUTPUT
--------------------------------------------------------------------------------
Predicate Information (identified by operation id):
---------------------------------------------------

   2 - filter("CATEGORIE_BATEAU"='Imoca')
   3 - access("E"."LIBELLE"="C"."LIBELLE" AND
              "E"."DATE_DEPART"="C"."DATE_DEPART")
```

FIGURE 1 – Premier plan d'exécution de la requête

2.1.2 Question 2 (2 points)

Donner les différents opérateurs physiques disponibles sous Oracle pour le calcul d'une jointure. Comment modifieriez vous la requête précédente pour que l'opérateur physique exploité soit un tri-fusion (sort merge) ? Le plan d'exécution correspondant à l'utilisation de l'algorithme de tri-fusion est donné en figure 2.1.2. Vous expliquerez les raisons qui ont poussé l'optimiseur à ne pas faire le choix de cet algorithme.



```
Plan hash value: 156435818

-------------------------------------------------------------------------------
| Id  | Operation           | Name      | Rows  | Bytes | Cost (%CPU)| Time     |
-------------------------------------------------------------------------------
|   0 | SELECT STATEMENT    |           |     4 |   232 |     5  (20)| 00:00:01 |
|   1 |  MERGE JOIN         |           |     4 |   232 |     5  (20)| 00:00:01 |
|   2 |   INDEX FULL SCAN   | COURSE_PK |     7 |   175 |     1   (0)| 00:00:01 |
|*  3 |   SORT JOIN         |           |     4 |   132 |     4  (25)| 00:00:01 |
|*  4 |    TABLE ACCESS FULL| EPREUVE   |     4 |   132 |     3   (0)| 00:00:01 |
-------------------------------------------------------------------------------

PLAN_TABLE_OUTPUT
-------------------------------------------------------------------------------------------------------------------
-------------------------------------------------------------------------------

Predicate Information (identified by operation id):
---------------------------------------------------

   3 - access("E"."LIBELLE"="C"."LIBELLE")
       filter("E"."DATE_DEPART"="C"."DATE_DEPART" AND
              "E"."LIBELLE"="C"."LIBELLE")
   4 - filter("CATEGORIE_BATEAU"='Imoca')
```

Figure 2 – Second plan d'exécution de la requête

2.2 Question 3 : différentes écritures d'une même requête (3 points)

Vous écrirez en SQL et de deux manières différentes, la requête : `donnez le nom et le prénom des navigateurs qui ont terminé soit premier, soit second (classement = 1 ou classement = 2) d'une épreuve d'une course au départ de Saint Malo (lieu_depart = 'Saint Malo')`. Vous en donnerez les expressions algébriques et vous formulerez votre avis sur la requête qui vous semble la plus efficace.

3 Index

3.1 Index implicite (3 points)

L'index COURSE_PK est analysé. Des instructions SQL, que vous aurez à commenter, sont données. Le résultat de la requête de consultation de la vue index_stats est également fourni.

FIGURE 3 – Consultation de la vue index_stats après analyse de l'index

Vous répondrez aux questions suivantes avec concision :

1. Quelle est la structure de données sur laquelle s'appuie cet index ? Est ce le type d'index le plus fréquemment utilisé sous Oracle ? Précisez en les intérêts.
2. Cet index est t'il dit unique ? quelle est l'information dans le résultat de la requête sur la vue index_stats qui peut vous renseigner à ce sujet ?
3. Expliquer la signification des attributs BTREE_SPACE, MOST_REPEATED_KEY, LF_ROWS, BR_ROWS, HEIGHT et commentez les valeurs obtenues.

3.2 Index explicite (2 points)

1. Donner l'ordre SQL de création d'index secondaire sur les attributs libelle et date_depart de la table Epreuve
2. Cet index est t'il unique ? Pourquoi choisir précisemment ces attributs pour la définition de cet index ?
3. Donner un exemple de requête de consultation qui pourrait être efficacement servie par ce nouvel index

4 Architecture Oracle (5 points)

4.1 Eléments mis en jeu

Une figure incomplète de l'organisation physique d'un serveur de données Oracle vous est donnée (vous la reproduirez sur votre copie). Vous compléterez les informations manquantes en ce qui concerne les structures et les processus mis en jeu. Les principales vues du méta-schéma qui servent à renseigner les différentes structures mémoire et processus d'arrière plan (ou autres processus) sont listées, vous positionnerez également ces vues sur la figure.

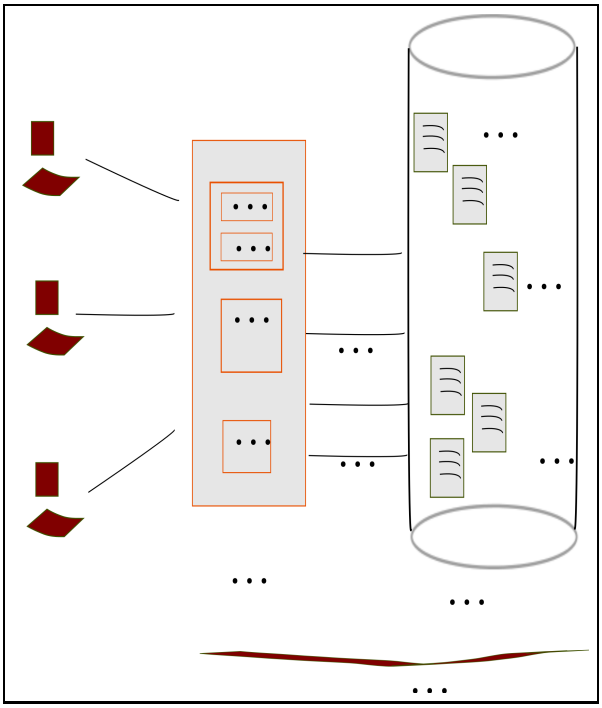


Figure 4 – Architecture physique Oracle (structures et processus)

4.1.1 Vues à positionner sur la figure

1. v$instance, v$database, v$bgprocess, v$sga, v$sgastat, v$sql, vsql_plan, vdatafile, v$logfile, v$bh, v$librarycache

4.2 Structures mémoire à détailler

Vous expliquerez de manière très brève les rôles principaux dévolus aux structures mémoires listées ci-dessous

1. Database buffer cache
2. Redo log buffer
3. Data dictionary cache
4. Library cache

5 Transaction (2 points)

Une séquence d'ordres SQL vous est fournie. Vous définirez le rôle général des ordres commit et rollback. Vous en expliquerez les effets sur la séquence proposée. Quels sont les effets des ordres qui seront rendus permanents au sein de la base de données

```
alter table Navigateur add constraint domiGenre check (genre in ('f','h'));
rollback;
insert into Navigateur values
        (21, 'Peyron','Loic','h','1-dec-1959','France');
insert into Navigateur values
        (22, 'MacArthur','Ellen','f','8-jul-1976','Royaume-Uni');
rollback;
update Navigateur set nom = 'Loick' where numNavigateur = 21;
exit;
```

6 Annexe

```
INSERT INTO COURSE VALUES
       ('Route du Rhum','4-nov-2018','Saint Malo','Pointe a Pitre');
INSERT INTO COURSE VALUES
       ('Route du Rhum','2-nov-2014','Saint Malo','Pointe a Pitre');
INSERT INTO COURSE VALUES
       ('Route du Rhum','9-nov-2002','Saint Malo','Pointe a Pitre');

INSERT INTO EPREUVE VALUES
       (1, 'Ultime', 'Route du Rhum','4-nov-2018');
INSERT INTO EPREUVE VALUES
       (2, 'Class40', 'Route du Rhum','4-nov-2018');
INSERT INTO EPREUVE VALUES
       (3, 'R-Multi', 'Route du Rhum','4-nov-2018');

INSERT INTO NAVIGATEUR VALUES
       (1, 'Peyron','Loick','h','1-dec-1959','France');
INSERT INTO NAVIGATEUR VALUES
       (2, 'MacArthur','Ellen','f','8-jul-1976','Royaume-Uni');
INSERT INTO NAVIGATEUR VALUES
       (3, 'Merron','Miranda','f','2-jul-1969','Royaume-Uni');

INSERT INTO PARTICIPE VALUES
       (3,1,4,'Olmix');
INSERT INTO PARTICIPE VALUES
       (3,6,1,'Happy');
INSERT INTO PARTICIPE VALUES
       (4,1,1,'Banque populaire VII');
```