**Médiathèque**

**[30 minutes - 6pts]**

Une médiathèque veut s'informatiser pour gérer plus efficacement son catalogue, contenant des livres, des films, et des albums de musique.

Les informations à enregistrer pour les livres sont : les titres, le (ou les) auteur(s), la date de parution, l'éditeur, la date d'impression et le nombre de pages. Les informations à enregistrer pour les films sont : le nom, le (ou les) réalisateur(s), au maximum six acteurs ayant joué dans le film avec leur rôle (il peut n'y avoir aucun acteur), le studio de production et la date de sortie. Les informations à gérer pour les albums sont : le nom de l'album, l'auteur, la maison de production, l'année de sortie, et les supports disponibles à la médiathèque (CD et/ou vinyle).

Pour chaque auteur, réalisateur, acteur ou musicien d'une œuvre, on veut gérer son nom, son prénom, son ou ses noms d'artiste (s'il en a), et sa date de naissance. Un album de musique peut avoir pour auteur :

* un artiste (ex : Jimmy Hendrix, Mozart),
* ou un groupe (ex : les Rolling Stones ou Franz Ferdinand) dont on gérera le nom, l'année de formation, et les informations sur les différents membres.

Livres, films et albums sont identifiés de manière unique (un film ne peut pas avoir le même identifiant qu'un album) par un code interne à la médiathèque. On veut aussi pouvoir gérer les adaptations cinématographiques (quels films sont les adaptations de quels livres). On veut enfin pouvoir retrouver facilement la BO correspondant à un film donné (la bande originale est l'album de musique du film).

**Question**

Réalisez un *package* permettant de modéliser en UML le catalogue de la médiathèque.

[**Solution**](http://www4.utc.fr/~nf17/DOCS/examens/2010median/co/medianP2010exo1.html)

Exemple

[Package Catalogue](http://www4.utc.fr/~nf17/DOCS/examens/2010median/res/exo1-Q1.pdf" \t "_blank)

Remarque

* Pour plus de lisibilité, il est conseillé de nommer toutes les associations, même lorsque cela semble évident à comprendre (sauf compositions, agrégations et héritages).
* Le nom d'artiste aurait pu être posé en attribut multivalué
* L'agrégation est utilisée ici (losange blanc) : ce type d'association est différent d'une composition (il n'y a pas de relation de dépendance entre le composé et la composant) et se traite comme une association classique, avec une terminologie un peu plus précise (notion de tout-partie).
* On pose comme hypothèse :
  + qu'un film peut être l'adaptation de plusieurs livres à la fois (par exemple deux volumes d'une série).
  + qu'un film peut avoir plusieurs BO (soit en plusieurs volumes, soit des versions remixées ou harmoniques, par exemple).

La médiathèque s'étend sur trois étages, chacun comprenant plusieurs rayons à thème (Romans Contemporains, Musique Classique, etc.). Certains de ces rayons peuvent comprendre à la fois des albums, des livres et des films (comme le rayon Science Fiction, ou le rayon Thriller). Dans un rayon, les œuvres centrales ou récentes sont disposées sur des présentoirs (chacun pouvant recueillir un certain nombre d'œuvres : films, livres et/ou albums), tandis que les autres sont rangées sur des étagères identifiées par un code particulier. Chaque étagère peut comprendre un certain nombre d'œuvres, mais d'une seule catégorie (on a des étagères de livres et des étagères d'albums, mais pas d'étagères comprenant à la fois des livres et des albums).

**Question**

Réalisez un second *package* permettant de modéliser l'organisation de la bibliothèque et de ses moyens de rangement.

[**Solution**](http://www4.utc.fr/~nf17/DOCS/examens/2010median/co/medianP2010exo1.html)

Exemple

[Package Organisation](http://www4.utc.fr/~nf17/DOCS/examens/2010median/res/exo1-Q2.pdf" \t "_blank)

Remarque

* On a pris ici comme hypothèse que les rangements étaient mobiles, et donc indépendants des rayons (d'où l'agrégation et non la composition).

# Carnets de voyages

## [30 minutes - 6pts]

Soit la relation suivante, donnée avec une couverture minimale de ses DFE :

carnetDeVoyage (numAuteur, nomAuteur, prenomAuteur, numVille, nomVille, nomPays, description)

* numAuteur→nomAuteur
* numAuteur→prenomAuteur
* numVille→nomVille
* numVille→nomPays
* numAuteur, numVille→description

## Question

Rappeler la définition formelle d'une clé.

Énoncez la ou les clés existantes.

Pour chaque clé, énoncer les DF qui prouvent que c'est une clé et explicitez les axiomes d'Armstrong utilisés pour établir ces DF.

Dites en quelle forme normale est la relation (montrez pourquoi).

[**Solution**](http://www4.utc.fr/~nf17/DOCS/examens/2010median/co/medianP2010exo2.html)

Clé

Une clé est un groupe d'attributs minimal qui détermine tous les attributs de la relation.

Il y a une unique clé (numAuteur, numVille).

* numAuteur, numVille→nomAuteur
* numAuteur, numVille→prenomAuteur
* numAuteur, numVille→nomVille
* numAuteur, numVille→nomPays
* numAuteur, numVille→description

La réflexivité et la transitivité sont utilisées à chaque fois de la même façon, par exemple, pour le premier :

numAuteur, numVille→numAuteur ET numAuteur→nomAuteur DONC numAuteur, numVille→nomAuteur

NF

La relation est en 1NF, on a identifié une clé et les attributs sont atomiques. Elle n'est pas en 2NF car des attributs faisant partie de la clé déterminent d'autres attributs, par exemple : numAuteur→nomAuteur.

## Question

Proposez un schéma normalisé en 3NF, sans perte, en faisant apparaître les clés.

[**Solution**](http://www4.utc.fr/~nf17/DOCS/examens/2010median/co/medianP2010exo2.html)

carnetDeVoyage(#numAuteur=>auteur, #numVille=>ville, description)

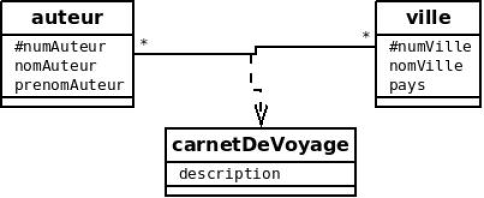
auteur(#numAuteur, nomAuteur, prenomAuteur)

ville(#numVille, nomVille, nomPays)

## Question

Rétro-concevez le modèle UML qui aurait permis d'arriver directement à ce résultat normalisé.

[**Solution**](http://www4.utc.fr/~nf17/DOCS/examens/2010median/co/medianP2010exo2.html)



*Modèle des carnets de voyage*

## Question

Écrivez le code SQL permettant de créer ce schéma normalisé dans une base de données standard.

[**Solution**](http://www4.utc.fr/~nf17/DOCS/examens/2010median/co/medianP2010exo2.html)

CREATE TABLE auteur (

numAuteur INTEGER PRIMARY KEY,

nomAuteur VARCHAR,

prenomAuteur VARCHAR);

CREATE TABLE ville (

numVille INTEGER PRIMARY KEY,

nomVille VARCHAR,

nomPays VARCHAR);

CREATE TABLE carnetDeVoyage (

numAuteur INTEGER REFERENCES auteur(numAuteur),

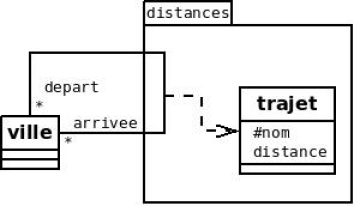
numVille INTEGER REFERENCES ville(numVille),

description VARCHAR,

PRIMARY KEY (numAuteur, numVille));

## Question

De nouvelles informations doivent être ajoutées à la base, afin de connaître les distances entre les villes. Ces nouvelles spécifications sont présentées sous la forme d'un *package* complémentaire. Effectuez le passage au relationnel de ce schéma (n'ajoutez que ce qui est nouveau). Expliquez ce que permet la clé #nom de trajet.



*Modèle des distances*

[**Solution**](http://www4.utc.fr/~nf17/DOCS/examens/2010median/co/medianP2010exo2.html)

trajet (#nom, #depart=>ville, #arrivee=>ville, distance)

#nom permet de définir plusieurs trajets différents entre deux mêmes villes.

## Question

Écrivez le code SQL permettant d'implémenter le *package* distances.

Puis, donnez tous les droits sur l'ensemble du schéma à un utilisateur Admin, et les droits en lecture sur l'ensemble du schéma à tous les utilisateurs de la base.

[**Solution**](http://www4.utc.fr/~nf17/DOCS/examens/2010median/co/medianP2010exo2.html)

CREATE TABLE trajet (

nom VARCHAR,

depart INTEGER REFERENCES ville(numVille),

arrivee INTEGER REFERENCES ville(numVille),

distance INTEGER NOT NULL,

PRIMARY KEY (nom, depart, arrivee));

GRANT ALL PRIVILEGES ON auteur TO Admin ;

GRANT ALL PRIVILEGES ON ville TO Admin ;

GRANT ALL PRIVILEGES ON carnetDeVoyage TO Admin ;

GRANT ALL PRIVILEGES ON trajet TO Admin ;

GRANT SELECT ON auteur TO PUBLIC ;

GRANT SELECT ON ville TO PUBLIC ;

GRANT SELECT ON carnetDeVoyage TO PUBLIC ;

GRANT SELECT ON trajet TO PUBLIC ;

# Gestion documentaire

## [45 min - 6 pts]

Soit le schéma relationnel suivant permettant une gestion électronique de documents (GED) :

document (#id, title, creator=>user)

user (#login, pwd, name)

version (#num, #doc=>document, contrib=>user)

NB : L'on utilisera ni les opérateurs ensemblistes, ni les sous-requêtes.

## Question

En **algèbre relationnel** et en **SQL**, afficher toutes les informations présentes dans la base de données.

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| *Résultat* | | | | | | | | |
| id | title | creator | login | pwd | name | num | doc | contrib |
| 001 | Le Capital | karl | karl | marx | Karl Marx | 1 | 001 | karl |
| ... |  |  |  |  |  |  |  |  |

[**Solution**](http://www4.utc.fr/~nf17/DOCS/examens/2010median/co/medianP2010exo3.html)

Algèbre relationnel

R1=produit(document, user)

R2=produit(R1, version)

SQL

SELECT \*

FROM document, user, version ;

## Question

En **algèbre relationnel** et en **SQL**, afficher le nom de l'auteur du document dont le titre est "Le Capital".

|  |
| --- |
| *Résultat* |
| name |
| Karl Marx |

[**Solution**](http://www4.utc.fr/~nf17/DOCS/examens/2010median/co/medianP2010exo3.html)

Algèbre relationnel

R1=restriction(document, title='Le Capital')

R2=jointure(R1, user, R1.creator=user.login)

R3=projection(R2, name)

SQL

SELECT name

FROM document, user

WHERE document.creator=user.login AND document.title='Le Capital';

## Question

En **algèbre relationnel** et en **SQL**, afficher les logins des users qui sont à la fois creator et contrib et qui commencent par "k".

|  |
| --- |
| *Résultat* |
| login |
| karl |
| ... |

[**Solution**](http://www4.utc.fr/~nf17/DOCS/examens/2010median/co/medianP2010exo3.html)

Algèbre relationnel

R1=restriction(document, creator COMME 'k%');

R2=jointure(R1, version, R1.creator=version.contrib)

R3=projection(R2, creator)

SQL

SELECT document.creator AS login

FROM document, version

WHERE document.creator=version.contrib AND document.creator LIKE 'k%';

## Question

En **algèbre relationnel** et en **SQL**, Afficher les logins des users qui n'ont ni créé, ni contribué.

|  |
| --- |
| *Résultat* |
| login |
| paul |
| ... |

[**Solution**](http://www4.utc.fr/~nf17/DOCS/examens/2010median/co/medianP2010exo3.html)

Algèbre relationnel

R1=jointureExterneGauche(user, document, user.login=document.creator)

R2=jointureExterneGauche(R1, version, user.login=version.contrib)

R3=restriction(R2, document.creator=NULL ET version.contrib=NULL)

R4=projection(R3, user.login)

SQL

SELECT login

FROM (user LEFT JOIN document ON user.login=document.creator) LEFT JOIN version ON user.login=version.contrib

WHERE document.creator IS NULL AND version.contrib IS NULL

## Question

En **SQL**, afficher la dernière version (celle qui a le numéro le plus élevé) de chaque document, la liste sera triée par id des documents croissants.

|  |  |
| --- | --- |
| *Résultat* | |
| id | last\_version |
| 001 | 2 |
| ... | ... |

[**Solution**](http://www4.utc.fr/~nf17/DOCS/examens/2010median/co/medianP2010exo3.html)

SQL

SELECT doc AS id, MAX(num) AS last\_version

FROM version

GROUP BY doc

ORDER BY doc;

## Question

En **SQL**, afficher le nombre de contributions pour chaque utilisateur, en face de leur nom, à condition que l'utilisateur ait contribué au moins 2 fois, la liste sera triée par nombre de contributions décroissant.

|  |  |
| --- | --- |
| *Résultat* | |
| name | nb\_contributions |
| Karl Marx | 14 |
| ... | ... |

[**Solution**](http://www4.utc.fr/~nf17/DOCS/examens/2010median/co/medianP2010exo3.html)

SQL

SELECT user.name, COUNT(\*) AS nb\_contributions

FROM user, version

WHERE user.login=version.contrib

GROUP BY user.name

HAVING nb\_contributions>=2

ORDER BY nb\_contributions DESC;