

Bases de données TP2

AR ~ SQL / Python ~ SQLITE

Exercice 1 : Concours blanc 2017-2018

On considère la base de données « **base1** » constituée des trois relations suivantes :

Acteur (numa, nom, spec ,taille, poids)

Colonne	Description	Type
numa	Numéro de l'acteur	Entier
nom	Nom de l'acteur	Texte
spec	Spécialité de l'acteur	Texte
taille	Taille	Réel
poids	Poids	Réel

Film (numf, titre, genre, annee, duree, budget, realisateur)

Colonne	Description	Type
numf	Numéro du film	Entier
titre	Titre du film	Texte
genre	Genre du film (comique,drame ...)	Texte
annee	Année de sortie	Entier
duree	Durée	Entier
budget	budget	Réel
realisateur	Réalisateur	Texte

Jouer (#numf, #numa, desc ,sal)

Colonne	Description	Type
numf	N° du film	Entier
numa	N° de l'acteur	Entier
desc	Description du rôle	Texte
sal	Salaire pour le rôle	Réel

- Les clés primaires sont soulignées et les clés étrangères sont précédées par #.
- Tous les attributs doivent être non vides

- 1) Donner les instructions **SQL** qui permettent de créer les trois relations de la base de données « **base1** » en respectant toutes les contraintes de domaines et d'intégrités référentielles.
- 2) Ecrire un script python qui :
 - Charge le module **sqlite3** ;
 - Etablit une connexion avec la base « **base1** » ;
 - Insère dans la table **Acteur** autant de lignes que l'utilisateur le souhaite ;
 - Et affiche le contenu de la table **Acteur**.
- 3) Répondre en algèbre relationnelle puis en **SQL** aux requêtes suivantes :
 - a) Afficher le titre des films qui ont une durée supérieure à 80 min et un budget ne dépassant pas les 100000 dinars ;
 - b) Afficher les rôles joués par l'actrice « Aymen Sahli ».
- 4) Répondre en **SQL** uniquement aux questions suivantes :
 - a) Afficher l'intitulé du genre ainsi que le nombre de films correspondants ;
 - b) Afficher les genres qui ont plus que 20 films ;
 - c) Donner le titre et l'année du/de(s) film(s) le(s) plus long(s) ;
 - d) Calculer le total des salaires des acteurs qui ont joué dans « TAXI 2 » ;
 - e) Afficher les noms des acteurs qui n'ont joué dans aucun film jusqu'à maintenant ;

Exercice 2 : Concours blanc 2016-2017

Un institut veut gérer son parc informatique à l'aide d'une base de données qui décrit les installations de logiciels sur des postes de travail.

Un poste de travail est une machine sur laquelle sont installés certains logiciels, gratuits ou payants.

Pour chaque installation on enregistre la date d'installation et la durée de la licence (en mois). Lorsque le logiciel est gratuit, on saisit la valeur "infini" pour la durée de licence.

On a établi le modèle relationnel ci-dessous, dans lequel les données sont organisées sous forme de relations :

- Logiciel (**id**, nom, version, prix)
- Installer (**#id_poste, #id_logiciel**, date_installation, duree_licence)
- Poste(**id**, nom, #id_salle)
- Salle(**id**, nom, nb_prises_reseau)

Les attributs de la relation **Logiciel** sont :

- **id** : identifiant, entier naturel non nul, s'incrémente automatiquement.
- **nom** : attribut non nul de type chaîne de caractères.
- **version** : attribut non nul de type chaîne de caractères, dans la mesure où la version d'un logiciel s'écrit sous la forme de nombres séparés par un ou plusieurs points, comme 5.4.1 par exemple.
- **prix** : un nombre décimal avec deux chiffres après la virgule. Un logiciel gratuit a un prix égal à 0.

Les attributs de la relation **Installer** sont :

- **id_poste** : a le même domaine que l'attribut id de la relation poste . En effet, c'est une clé étrangère qui fait référence à la clé primaire id de la relation poste.
- **id_logiciel** : a le même domaine que l'attribut **id** de la relation logiciel . En effet, c'est une clé étrangère qui fait référence à la clé primaire id de la relation logiciel.
- **date_installation** : est un attribut non nul de type DATE.
- **duree_licence** : est un attribut non nul, de type chaîne de caractères dans la mesure où on saisit la valeur "infini" lorsque le logiciel est gratuit et un nombre entier naturel de mois sinon.

Les attributs de la relation **Poste** sont :

- **id** : identifiant, entier naturel non nul, s'incrémente automatiquement.
- **nom** : attribut non nul de type chaîne de caractères.
- **id_salle** : a le même domaine que l'attribut id de la relation salle . En effet, c'est une clé étrangère qui fait référence à la clé primaire id de la relation salle.

Les attributs de la relation **Salle** sont :

- **id** : identifiant, entier naturel non nul, s'incrémente automatiquement.
- **nom** : attribut non nul, unique, de type chaîne de caractères.
- **nb_prises_reseau** : entier naturel.

Afin de mieux manipuler la base de données de l'institut, on proposera les classes Python suivantes :

➤ une **classe Logiciel** définie par :

(a) une méthode constructeur définissant les attributs

- **nom** pour le nom du logiciel
- **version** pour la version du logiciel
- **prix** pour son prix d'achat.

(b) une méthode **enregistrer(self)** permettant d'enregistrer un logiciel dans la table Logiciel.

(c) une méthode **chercher(self)** permettant de chercher un logiciel dans la table Logiciel par son nom et sa version et retourne son identifiant.

(d) une méthode **__str__(self)** pour visualiser un logiciel (nom, version, prix).

- une **classe Poste** définie par :
 - (a) une méthode constructeur définissant les attributs
 - **nom** pour le nom du poste.
 - **liste_logiciels** pour la liste des logiciels installés sur un poste. Cette liste sera initialement vide.
 - (b) une méthode **chercher(self, idSalle)** permettant de chercher un poste dans la table Poste par son nom et la salle où il est installé et retourne son identifiant.
 - (c) une méthode **installer(self, logiciel, date, licence)** qui permet d'installer un logiciel sur un poste à une date avec une licence, données en paramètres.
 - (d) une méthode **charger_liste_logiciels(self)** permettant de mettre à jour la liste_logiciels par les logiciels installés sur un poste à partir de la base de données.
 - (e) une méthode **__str__(self)** pour visualiser la liste des logiciels installés sur un poste.

- une **classe Salle** définie par :
 - (a) une méthode constructeur définissant les attributs
 - **nom** pour le nom de la salle.
 - **liste_postes** pour la liste des postes localisés dans une salle. Cette liste sera initialement vide.
 - (b) une méthode **chercher(self)** permettant de chercher une salle dans la table Salle par son nom et retourne son identifiant.
 - (c) une méthode **installer(self, p)** qui permet d'ajouter un poste **p** à une salle.
 - (d) une méthode **charger_liste_postes(self)** permettant de mettre à jour la liste_postes par les postes installés dans une salle à partir de la base de données.
 - (e) une méthode **__str__(self)** pour visualiser les postes (nom, catégorie) d'une salle.

On supposera pour la suite des questions que l'institut a équipé 3 salles (101, 102 et 323) par 9 postes, 3 postes (P1, P2 et P3) pour chacune des salles.

La base de données de l'institut se nomme '**institut.db**', et on utilisera le module SQLite pour sa manipulation.

Questions :

1. Quelle est la requête SQL permettant de Créer la table **Installer**.
2. Définir une fonction Python **executer** permettant de retourner le résultat d'exécution d'une requête SQL de type Select.
3. Définir la classe **Logiciel**.
4. Créer une fonction **lister_salles** permettant de retourner à partir de la table Salle, une liste contenant toutes les salles de l'institut.
5. Définir un objet **scilab** représentant le logiciel *Scilab 2.3* en sa version gratuite et l'enregistrer dans la base de données.
6. Enregistrer dans la base de données l'opération d'installation du Scilab sur tous les postes de l'institut le 04/04/2017.
7. Changer la portée de l'attribut prix de la classe Logiciel du publique à privée. Réécrire seulement la/les instructions de la/les méthodes modifiées de la classe Logiciel.
8. Ajouter, à la classe Logiciel, une méthode publique nommée **getPrix(self)** permettant de retourner le prix d'achat d'un logiciel. Quelles sont les modifications à faire au niveau de la classe Logiciel pour garder la cohérence du code (Réécrire seulement la/les instructions modifiées de la classe Logiciel).
9. Ajouter, à la classe Logiciel, une méthode publique **setPrix(self, nouveau_prix)** permettant de modifier le prix d'un logiciel, dans laquelle on fait tous les traitements/contrôles nécessaires pour préserver l'intégrité des données. Réécrire seulement la/les instructions modifiées de la classe Logiciel.
10. Ajouter à la classe Poste une méthode **nb_logiciels(self)** permettant de calculer et retourner le nombre de logiciels installés sur un poste .
11. Quelle est la requête SQL permettant d'afficher les postes sur lesquels le logiciel 'Python' n'est pas installé.