BaDMat 1.0
Mode d'emploi

Myrtille Grulois *

Été 2020

*Université populaire et citoyenne de Roubaix

Sommaire

Introduction

Ce manuel d'utilisation concerne la première version du logiciel BaDMat de l'UPC de Roubaix. Les explications concernent à la fois la structure de la base de données et les différentes fonctions disponibles. Pour toute action d'ordre général, se reporter au préambule, qui contient des instructions valables pour la plupart des fonctions (sauf mention contraire).

1 Préambule : ce qu'il est utile de savoir (faire)

1.1 Ouverture de l'invite de commande

Bon je vais pas trop détailler, comme on n'a pas encore décidé de l'OS à utiliser...

1.2 Lancement d'un script

Tout d'abord, précisons que dans notre cas, un script correspond à une fonctionnalité. Chaque fonction décrite ci-dessous est donc lancée à partir d'un fichier différent. Pour lancer un script (à compléter, dépend 1. de l'OS 2. de comment fonctionne Python sur l'OS)

1.3 Utilisateurs et connexion à la base de données via Python

Lorsque vous souhaitez consulter la base de données, il vous suffit d'entrer *user* lorsqu'on vous demande l'utilisateur. Pour toute autre action, vous aurez besoin d'un compte avec des droits étendus. Pour cela, reportez-vous à la personne chargée de créer les comptes, ou bien à la section correspondante de ce manuel. Une fois que vous disposez d'un nom d'utilisateur et d'un mot de passe, vous pouvez effectuer le script de votre choix en entrant vos identifiants lorsqu'ils sont demandés. Pour la recherche, continuez à utiliser l'utilisateur commun *user*.

1.4 Interactions avec le programme

Pour la plupart des instructions, le programme nécessite que vous rentriez une lettre en majuscule. Tapez la lettre correspondant à votre choix parmi les propositions de la question, puis validez en tapant *Entrée*. Parfois, on vous demandera également de taper *Entrée* pour toutes les colonnes que vous souhaitez passer, et n'importe quelle lettre pour toute autre action. De manière générale, une instruction précède à chaque fois les commandes à entrer pour vous guider.

1.5 Définition des termes utilisés

On parlera plus loin de table lorsqu'on parlera de catégorie enregistrée dans la base de données : ainsi, il y aura par exemple une table pour les différents matériaux, et une autre pour les pièces qui sont composées de ces matériaux.

Tout au long de la suite, ligne désignera une entrée dans la base de données. Colonne désignera un des possibles critères de cette ligne (le nom, la taille, la quantité...)

id	t_m_date	t_m_nom	t_m_famille	t_m_prix
1	2020-07-20	sapin	organique	12
2	2020-07-13	chêne	organique	12
3	2020-07-13	cuivre	métal	32

Dans l'exemple ci-dessus, vous avez un extrait de la table des matériaux. Une ligne de cette table (par exemple celle d'ID 1 décrivant le sapin) correspond à une entrée dans la table, donc à un matériau. Elle sera nommée. . . ligne. On nommera colonne, par exemple, le critère ID. Pour aller plus loin, la colonne ID est de type numérique, tandis que la colonne t_m_famille est de type textuel. Reportez-vous à la section **Base de données** pour connaître plus précisément les types de chaque colonne, ainsi qu'aux **Annexes** pour une description de chaque type.

2 Fonctions

Cette section regroupe les différentes fonctions implémentées, ainsi que la façon de les exécuter, et les différentes options possibles.

2.1 Recherche

La fonction principale de cette base de données est la fonction de recherche. Pour l'exécuter, lancer le script *search.py*, puis se connecter à la base de données. Pour vous connecter, utilisez *user* comme identifiant, pour lequel on ne vous demandera pas de mot de passe. Vous devrez commencer par choisir dans quelle table est-ce que la recherche s'effectuera. Il existe deux types de recherche : la recherche simple et la recheche avancée.

2.1.1 Recherche simple

La recherche simple est à privilégier si vous ne souhaitez inclure qu'une seule colonne dans votre recherche. Vous aurez toujours la possibilité de choisir quelle(s) colonne(s) vous souhaitez afficher dans les résultats.

Recherche textuelle
Si la colonne que vous sélectionnez est de type textuel, alors vous pourrez entrer un mot ou une expression.

Recherche numérique
Si la colonne que vous sélectionnez est de type numérique, vous pourrez effectuer une recherche autour d'une valeur donnée, ou bien une recherche sur un intervalle. Si vous choisissez la recherche numérique simple, vous aurez toutefois la possibilité d'entrer une tolérance. La recherche s'effectuera alors sur l'intervalle de longueur de deux fois votre tolérance qui entoure votre valeur. Si vous rentrez une valeur décimale, pensez bien à saisir un point (.) et non une virgule.

Recherche par date
Vous pouvez également effectuer une recherche sur la date.

2.1.2 Recherche avancée

La recherche avancée vous prendra plus de temps. Elle permet à la fois d'effectuer une recherche sur plusieurs colonnes, mais également de rechercher plusieurs critères dans la même colonne (pour les colonnes de texte).

Recherche textuelle
Si la colonne que vous sélectionnez est de type textuel, alors vous pourrez entrer un mot ou une expression, qui sera recherché dans cette colonne. Toutefois, vous aurez ensuite la possibilité d'ajouter des mots supplémentaires, que vous pourrez moduler : c'est-à-dire imposer que le mot saisi ne se trouve pas dans les résultats de votre recherche, ou bien cherche le premier terme ou bien le deuxième, etc.

Recherche numérique
La recherche numérique avancée est identique à celle de la recherche simplifiée. Si la colonne que vous sélectionnez est de type numérique, vous pourrez effectuer une recherche autour d'une valeur donnée, ou bien une recherche sur un intervalle. Si vous choisissez la recherche numérique simple, vous aurez toutefois la possibilité d'entrer une tolérance. La recherche s'effectuera alors sur l'intervalle de longueur de deux fois votre tolérance qui entoure votre valeur.

Recherche par date
Vous pouvez également effectuer une recherche sur la date.

Une fois que vous aurez entré vos critères de recherche, vous pourrez également choisir les colonnes à afficher, et ce quel que soit le type de recherche préalablement choisi. Pour cela, vous avez trois possibilités : afficher les colonnes par catégorie (description de ces dernières dans l'annexe Structure initiale complète de la base de données, afficher toutes les colonnes (ce type d'affichage est déconseillé la plupart du temps), ou encore afficher des colonnes que vous sélectionnerez manuellement.

2.2 Ajouter une nouvelle ligne

Lorsque vous souhaitez ajouter une nouvelle ligne, il vous faut utiliser le script *addRow.py*. Connectez-vous, puis indiquez quelle table vous souhaitez compléter. On vous demandera ensuite d'entrer un par un les critères que vous souhaitez compléter. Si pour un critère donné vous ne savez pas, vous pouvez simplement taper "Entrée". Cela laissera vide la colonne concernée. Toutefois, certaines colonnes ne peuvent pas être laissées vides. Pour celles-là (plus de précisions dans l'annexe Structure initiale complète de la base de données), il faut impérativement que vous entriez une valeur.

La complétion de certaines colonnes est conditionnée par des termes préalablement rentrés. Par exemple, il ne vous est pas possible d'entrer une pièce composée d'un matériau qui n'est pas présent dans la base de données. Il faudra d'abord que vous créiez le matériau, puis vous pourrez ajouter la pièce associée. Le même principe s'applique pour la famille de matériaux. Un certain nombre de termes est autorisé (organique, métal, alliage, composite, polymère, céramique). Si vous essayez de rentrer un autre terme, vous obtiendrez une erreur. La liste complète des colonnes contraintes est consultable dans l'annexe Structure initiale complète de la base de données. Pensez à tenir cette annexe à jour si vous souhaitez continuer à ce qu'elle soit pertinente. En outre, le script *printTerms.py* vous permet d'afficher la liste actuelle des termes autorisés.

Si vous entrez une valeur décimale dans les colonnes qui l'acceptent, pensez bien à utiliser le point et non la virgule comme séparateur.

2.3 Modifier une ligne existante

Utilisez pour cela le script *modifyRow.py*. Après vous être connecté et avoir sélectionné la table sur laquelle vous souhaitez travailler, il faut que vous connaissiez l'ID exact de la ligne à modifier. Si ce n'est pas le cas, commencez plutôt par effectuer une recherche, dans laquelle vous sélectionnerez la colonne ID parmi les colonnes à afficher.

Modifier une ligne se passe à peu près comme pour en ajouter une. On vous propose tous les critères un par un, à vous de les compléter.

Attention, une modification de ligne est définitive ! Si vous remplacez la valeur actuelle de la colonne, celle-ci est totalement effacée pour être remplacée par votre nouvelle valeur. Il vous est donc conseillé d'effectuer des sauvegardes régulières de la base de données (pour cela, se reporter à la section Sauvegarde et restauration de la base de données).

2.4 Supprimer une ligne existante

Le script à utiliser est le script *delete.py*. Après connexion et choix de la table, il vous faudra, là encore, renseigner l'ID exact de la ligne à supprimer.

Attention, une suppression de ligne est définitive, soyez sûrs de ce que vous faites ! En cas d'erreur, pensez à restaurer une sauvegarde précédemment effectuée (pour plus d'informations, consultez la section Sauvegarde et restauration de la base de données).

2.5 Fusionner deux lignes existantes

Utilisez pour cela le script *merge.py*. Connectez-vous et choisissez la table. Bien entendu, vous ne pouvez fusionner deux lignes que si elles se trouvent dans la même table. Vous aurez besoin des ID des deux lignes à fusionner.

Le programme vous proposera automatiquement une possibilité de fusion. Pour celle-là, il privilégiera de remplir au maximum les colonnes, et, en cas de doute, de conserver la valeur de la ligne dont vous aurez donné l'ID en premier. Ainsi, si les deux valeurs sont identiques, il vous proposera automatiquement cette valeur. Si l'une des lignes n'est pas remplie, il suggérera automatiquement l'autre valeur. En cas de remplissage différent pour les deux lignes, il vous proposera par défaut la valeur présente sur la première ligne. Vous verrez entre parenthèses la valeur que contenait la deuxième ligne.

Si vous souhaitez modifier la valeur proposée par défaut (que ce soit pour la remplacer par la valeur entre parenthèses ou par une autre valeur), il vous suffit de taper la valeur de remplacement et de saisir "Entrée". Encore une fois, des instructions assez détaillées devraient vous guider tout au long de l'exécution du programme.

Attention, une fusion de lignes est définitive ! L'une de deux sera supprimée, et l'autre modifiée avec les valeurs que vous avez choisies. Vous ne pourrez pas revenir en arrière après avoir validé la fusion, à moins d'utiliser une sauvegarde (pour plus d'informations, référez-vous à la section 3.3.1 Sauvegarde et restauration de la base de données).

2.6 Autoriser de nouveaux termes lors de la complétion de ligne

Certaines colonnes, afin de faciliter les recherches et d'éviter que la base de données ne se transforme en un fouillis innommable, n'acceptent que les termes figurant sur une liste remplie au préalable. Toutefois, cette liste n'est pas exhaustive. Si vous souhaitez à ajouter de nouveaux mots, vous pouvez utiliser le script *addTerm.py*.

Précautions d'usage : le but de cette limitation à une liste pour certaines colonnes est de limiter le nombre de termes autorisés. Bien entendu, il se peut qu'il y ait des termes nécessaires qui aient été oubliés, et qui doivent être ajoutés. Mais prenez la peine de consulter la liste actuelle en utilisant le script *printTerms.py* avant d'ajouter le terme que vous souhaitez. Peut-être un synonyme y est-il déjà présent, ou bien vous avez oublié un accent... A des fins de simplification, il est préférable de ne pas entrer le dictionnaire dans son entièreté dans cette liste, et de rester concis.

2.7 Création d'un nouvel utilisateur

3 Base de données

3.1 Créer la base de données

Cette action n'est à effectuer qu'une seule fois. Ouvrir le terminal et se connecter à mysql.

Taper `CREATE DATABASE TestDB;` et valider.

Si la réponse est du style `Query OK, 1 row affected (0.00 sec)`, c'est que la requête a réussi, et que la base de données est prête à être créée.

Vous pouvez alors exécuter le script *createDB.py*. Ce script va créer la base de données avec toutes ses tables et ses colonnes. Toutefois elle sera vide, ce sera à vous d'y ajouter des lignes.

Si vous avez récupéré une sauvegarde et que vous souhaitez donc compléter automatiquement les tables lors de la création de la base de données, vous préférerez utiliser le script *backup.py* (après avoir effectué, bien entendu, la première étape de ce paragraphe). Pour plus d'informations sur ce dernier script, consultez la section Sauvegarde et restauration de la base de données.

3.2 Structure de la base de données

La base de données comporte plusieurs tables. Les deux plus importantes pour nous sont celle des **Materiaux** et celle des **Pieces**. **Materiaux** a pour vocation de recenser tous les matériaux possibles et leurs différentes propriétés, physiques, optiques, mécaniques, acoustiques... **Pieces** quant à elle recense toutes les "pièces" de construction et de réparation qui peuvent être trouvées sur place. Celles-ci allant de la simple planche de bois au tambour d'une marque précise de machine à laver en passant par des vis et des poignées de portes.

Vous trouverez en annexe un tableau reprenant les unités utilisées pour chacune des colonnes de ces deux tables. Les unités demandées sont absolument à respecter, sans quoi la base de données ne présenterait plus aucune cohérence.

A titre informatif, d'autres tables existent aussi. La table **Words** vous sera d'une certaine utilité si vous souhaitez compléter la base de données. C'est elle qui contient les termes autorisés pour certaines colonnes des autres tables qui sont restrictives. Pour la compléter, le script *addTerm.py* est à votre disposition, tandis que pour la consulter, vous pouvez utiliser le script *printTerms.py*

Vous n'interagirez jamais directement avec les tables **NameMateriaux** et **NamePieces**. Elles ont une vocation descriptive, et sont réutilisées par les scripts pour fournir des informations d'affichage, d'unités, mais également des informations sur les données qui peuvent être entrées dans les colonnes correspondantes des tables **Materiaux** et **Pieces**. Les informations qu'elles contiennent peuvent se lire facilement dans l'annexe Structure initiale complète de la base de données.

3.3 Manipulation de la base de données avec Python

3.3.1 Sauvegarde et restauration de la base de données

Il est conseillé d'effectuer régulièrement des sauvegardes de la base de données. Pour cela, utilisez le script *backup.py*. Vous ne pouvez enregistrer qu'une seule sauvegarde chaque jour, puisque le programme utilise pour cela la date. Faites donc attention : il est conseillé d'effectuer une sauvegarde préalable avant toute altération de la base de données, a fortiori s'il s'agit de suppression, de fusion ou de modification, puisque ces opérations écrasent définitivement les données précédentes, et ne sont pas récupérables autrement que grâce à cette backup.

Vous pouvez utiliser le même script pour restaurer une sauvegarde qui a été effectuée récemment. Soit la sauvegarde du jour, soit vous pouvez choisir manuellement la date que vous souhaitez. Les fichiers de sauvegarde sont des fichiers que vous pouvez consulter avec n'importe quel éditeur de texte. Leur structure n'est pas très lisible, mais vous permet tout de même de retrouver certaines informations, si vous avez un doute sur la date de la sauvegarde que vous souhaitez restaurer.

Notez bien que restaurer une sauvegarde permet d'annuler toutes les modifications qui ont été effectuées depuis, donc écrase toutes les nouvelles données entrées depuis.

3.3.2 Modifications de la structure de la base de données

Le script *modifyDb.py* va vous permettre d'effectuer des modifications structurelles de la base de données. Parmi ces modifications, nous pouvons nommer l'ajout et la suppression de colonnes, mais également la modification de colonnes déjà existantes.

Prenez garde lors de ces manipulations. D'une part, vous risquez de perdre des données (pensez donc à effectuer des sauvegardes). D'autre part, certaines modifications sont impossibles (notamment à cause des lignes déjà complétées, qui imposent certaines restrictions sur les modifications possibles). Vous risquez donc de vous heurter fréquemment à des erreurs si vous abusez de ce script. Il est conseillé d'en limiter l'utilisation au maximum, et de bien réfléchir à ses interventions avant de les entamer.

La modification d'une colonne existante peut se faire de deux façons différentes : soit en modifiant les données de remplissage de la colonne (donc le type, la valeur par défaut et si elle a le droit d'être laissée vide ou non), soit ce que nous appellerons les "métadonnées" de la colonne (à savoir son titre, si elle dépend ou non des termes autorisés, sa catégorie d'affichage, et son unité si elle en a une).

3.4 Connexion manuelle à MySQL

Pour certaines actions rares, telles que prendre connaissance de la taille des données stockées, il faut passer par MySQL directement.

Pour cela, le processus est le suivant :

Ouvrez l'invite de commande.

Connectez vous à MySQL : `mysql -u votreidentifiant -p` puis entrez votre mot de passe.

.1 Structure initiale complète de la base de données

.2 Description des types possibles pour chaque colonne

Différents types existent en SQL.

Il y a d'une part les types numériques :

- INT (et ses variantes TINYINT, SMALLINT, MEDIUMINT, BIGINT qui représentent des entiers de différentes tailles. TINYINT va de -64 à 64, tandis que BIGINT peut aller jusqu'à 10^{18}.
- FLOAT, qui représente un nombre décimal
- DOUBLE, qui représente un nombre décimal plus précis, mais prenant également plus de place en mémoire.

On trouve ensuite les types textuels :

- VARCHAR(x), avec x le nombre de caractères maximum autorisés pour ce champ.
- TEXT, qui permet de stocker jusqu'à près de 200.000 caractères, mais n'accepte pas les valeurs par défaut (nous y reviendront ci-après).

Puis les autres types. Dans notre cas, nous ne faisons appel qu'au type DATE, qui permet d'afficher une date.

Nous en arrivons maintenant aux attributs possibles. Comme évoqué plus haut, il est possible d'attribuer une valeur par défaut à une colonne. Cette valeur par défaut doit bien entendu être compatible avec le type renseigné. Si jamais la colonne est laissée vide, alors elle sera automatiquement complétée avec la valeur prévue.

Un autre attribut possible est d'imposer à MySQL de refuser ou d'accepter une valeur vide. Pour certaines colonnes, il peut être pertinent de refuser qu'elle soit laissée vide, et d'imposer à celui qui la complète d'entrer une valeur (à moins qu'une valeur par défaut ne soit définie).