Correction de la partie BDD de Centrale 2015 :

Il s'agit de la fin du sujet, qui comporte en tout une trentaine de questions, dont certaines sont des questions de maths liées à la compréhension de la méthode d'Euler. Voici l'extrait du sujet :

IV Exploitation d'une base de données

À partir de mesures régulièrement effectuées par différents observatoires, une base de données des caractéristiques et des états des corps célestes de notre Système solaire est maintenue à jour. L'objectif de cette partie est d'extraire de cette base de données les informations nécessaires à la mise en œuvre des fonctions développées dans les parties précédentes, puis de les utiliser pour prévoir les positions futures des différentes planètes. Les données à extraire sont les masses des corps étudiés et leurs états (position et vitesse) à l'instant $t_{\min}$ du début de la simulation.

Une version simplifiée, réduite à deux tables, de la base de données du Système solaire est donnée figure 3. Les masses sont exprimées en kilogrammes, les distances en unités astronomiques ($1\,\text{au} = 1{,}5 \times 10^{11}\,\text{m}$) et les vitesses en kilomètres par seconde. Le référentiel utilisé pour exprimer les composantes des positions et des vitesses est galiléen, orthonormé et son centre est situé à proximité du Soleil.

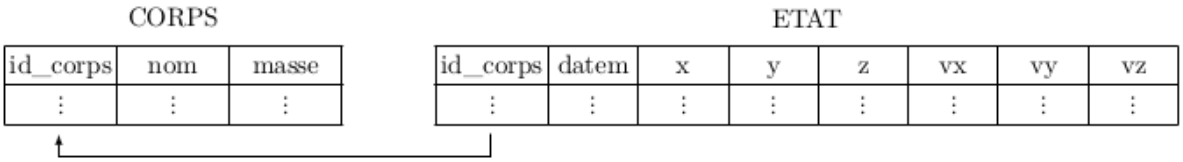

Figure 3 Schéma de la base de données

La table `CORPS` répertorie les corps étudiés, elle contient les colonnes
- `id_corps` (clé primaire) entier identifiant chaque corps ;
- `nom`, chaine de caractères, désigne le nom usuel du corps ;
- `masse` de type flottant, contient la masse du corps.

La table `ETAT` rassemble l'historique des états successifs (positions et vitesses) des corps étudiés. Elle est constituée de huit colonnes :
- `id_corps` de type entier, identifie le corps concerné ;
- `datem` est la date de la mesure, sous forme d'un entier donnant le nombre de secondes écoulées depuis un instant d'origine ;
- trois colonnes de type flottant pour les composantes de la position `x`, `y`, `z` ;
- trois colonnes de type flottant pour les composantes de la vitesse `vx`, `vy`, `vz`.

IV.A – Écrire une requête SQL qui renvoie la liste des masses de tous les corps étudiés.

IV.B – Les états des différents corps ne sont pas forcément tous déterminés exactement au même instant. Nous allons assimiler l'état initial (à la date $t_{\min}$) de chaque corps à son dernier état connu antérieur à $t_{\min}$.

Dans toute la suite, on supposera que la valeur de $t_{\min}$, sous le format utilisé dans la table `ETAT`, est accessible à toute requête SQL via l'expression `tmin()`.

IV.B.1) On souhaite d'abord vérifier que tous les corps étudiés disposent d'un état connu antérieur à `tmin()`. Le nombre de corps présents dans la base est obtenu grâce à la requête `SELECT count(*) FROM corps`. Écrire une requête SQL qui renvoie le nombre de corps qui ont au moins un état connu antérieur à `tmin()`.

IV.B.2) Écrire une requête SQL qui renvoie, pour chaque corps, son identifiant et la date de son dernier état antérieur à `tmin()`.

IV.B.3) Le résultat de la requête précédente est stocké dans une nouvelle table `date_mesure` à deux colonnes :
- `id_corps` de type entier, contient l'identifiant du corps considéré ;
- `date_der` de type entier, correspond à la date du dernier état connu du corps considéré, antérieur à `tmin()`.

Pour simplifier la simulation, on décide de négliger l'influence des corps ayant une masse strictement inférieure à une valeur fixée `masse_min()` et de ne s'intéresser qu'aux corps situés dans un cube, centré sur l'origine du référentiel de référence et d'arête `arete()` donnée. Les faces de ce cube sont parallèles aux plans formés par les axes du référentiel de référence.

Écrire une requête SQL qui renvoie la masse et l'état initial (sous la forme `masse, x, y, z, vx, vy, vz`) de chaque corps retenu pour participer à la simulation. Classez les corps dans l'ordre croissant par rapport à leur distance à l'origine du référentiel.

Une proposition de correction :

IV.A)

```
SELECT masse
FROM CORPS
```

IV. B.1)

```
SELECT COUNT(DISTINCT id_corps)
FROM ETAT
WHERE datem < tmin()
```

IV.B.2)

```
SELECT id_corps,MAX(datem)
FROM ETAT
WHERE datem < tmin()
GROUP BY id_corps
```

IV.B.3)

```
SELECT masse,x,y,z,vx,vy,vz
FROM date_mesure
JOIN CORPS ON date_mesure.id_corps =CORPS.id_corps
JOIN ETAT ON date_mesure.id_corps=ETAT.id_corps
WHERE masse > masse_min()
AND ABS(x) < arete()/2
AND ABS(y) < arete()/2
AND ABS(z) < arete()/2
ORDER BY x*x+y*y+z*z ASC
```

Remarques :

- On a calculé la distance au carré (ça évite l'utilisation d'une racine carrée – on a le même classement qu'avec la distance).

- La spécification ASC précise que le classement doit se faire par ordre croissant (c'est facultatif car c'est le comportement par défaut ; on aurait utilisé DESC pour un classement par ordre décroissant).

- Pour rebondir sur la question de l'ordre des tables pour les jointures posée par Raphaël ce matin, si on avait commencé la jointure par la table CORPS puis date_mesure, puis ETAT, il aurait fallu sélectionner les enregistrements vérifiant date_mesure.date_der=ETAT.datem.