Quelle est la différence entre TABLE et MATRICE ?
A quoi peut ressembler le bulletin de notes du futur ?

Par : Abdel YEZZA, Ph.D.
Date : oct. 2014/mise à jour déc. 2014 puis déc. 2023

Contenu

Liste des graphes, formules et illustrations

1.Introduction

Cette question je l'ai posé à plusieurs personnes ayant des métiers relativement distincts bien qu'elles soient toutes en informatique. A ma surprise, très peu parmi elles connaissent vraiment la différence entre une matrice et une table. Evidemment, ce sont les collaborateurs qui travaillent sur les bases de données qui sont sensés pouvoir différencier entre ces deux éléments. Justement une base de données relationnelle est généralement formée de ce qu'on appelle des tables. Chaque table est constituée de champs qui représentent en quelque sorte l'entête de la table et des enregistrements formés des valeurs de chaque champ s'affichant successivement après l'entête comme illustré ci-après :

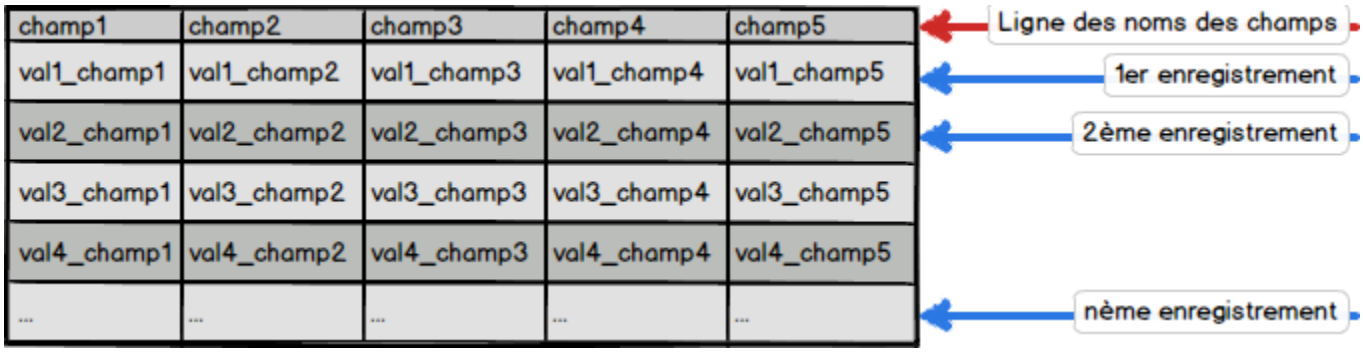


Fig 1. Schéma d'une table

La notion d'une **table** semble être simple, qu'en est-il à propos d'une **matrice** ? Avant de répondre à cette question, voici quelques constations que l'on peut confirmer à propos d'une table :

1. Elle possède une ligne fixe (l'entête) qui représente les champs de la table
2. Le reste des lignes représentent les enregistrements formés des valeurs de chaque champ
3. Une table peut être vue comme étant un ensemble de vecteurs (enregistrements, ou tableaux en termes techniques) de même dimension finie (= le nombre de champs)

Qu'est-ce que c'est une matrice alors ? Une matrice par rapport à une table possède en plus des champs (1ère ligne des colonnes) des noms de champs attribués aux lignes de sorte à ce que chaque entrée à l'intersection d'un **champ (colonne) j** et d'une **ligne (étiquette) i** représente la valeur de l'entrée à l'**intersection (i, j)** que l'on peut noter **Val(i,j)** comme illustré ci-dessous :

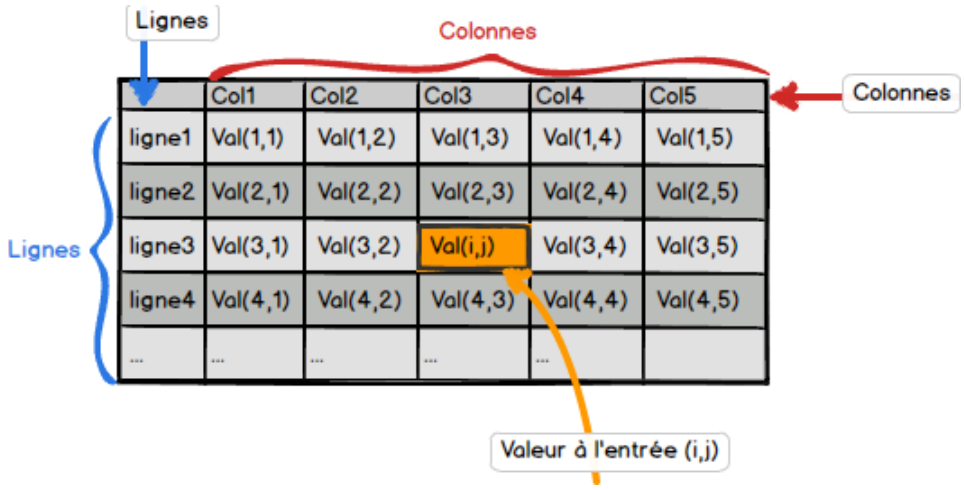


Fig 2. Schéma d'une matrice

Chaque entrée peut être une valeur définie ou le résultat d'une fonction ***f*** appliquée à la paire **(i, j)** que l'on peut noter ***f*(i, j)**.

On peut confirmer les constatations suivantes concernant une matrice :

- Elle est formée de colonnes et de lignes fixes
- On ne parle pas d'enregistrements, mais plutôt d'entrées aux intersections des lignes et colonnes pouvant être des valeurs définies ou des fonctions des paires (i, j) : ***f*(i,j)**.
- Une matrice peut être vue comme étant un ensemble résultant d'un produit des lignes et des colonnes **LxC** dont la dimension (nombre d'entrées) est le produit des dimensions des lignes et des colonnes : **nombre de lignes X Nombre de colonnes.**

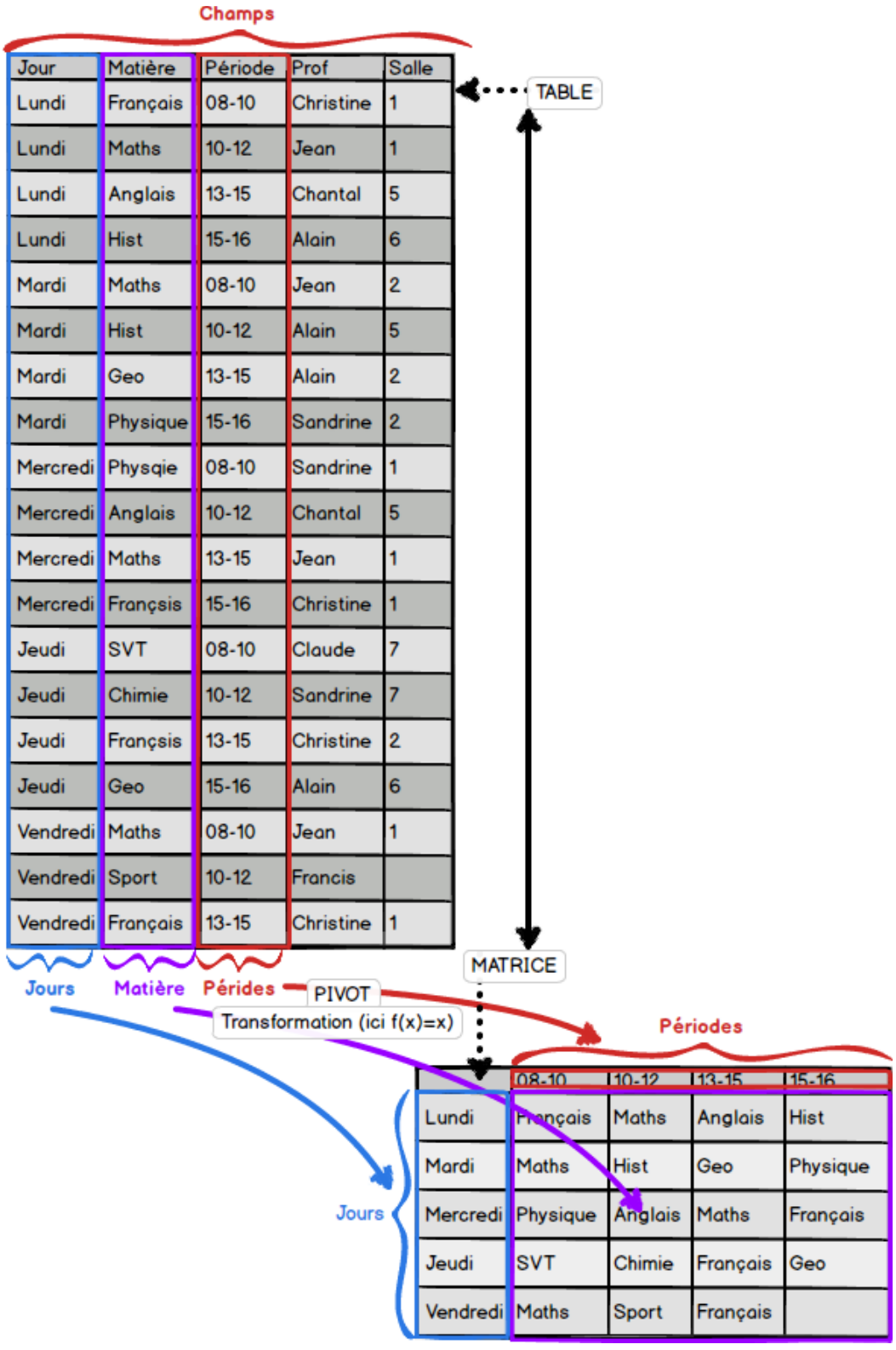
D'un point de vue technique une table est clairement plus appropriée car elle peut être facilement conçue dans une base de données ou tout simplement un tableur. En revanche, d'un point de vue utilisateur final une table est moins parlante qu'une matrice, car cette dernière réalise une association entre des éléments qui possèdent généralement un sens. Voici un exemple simple :

Exemple 1 : Si vous connaissez déjà un tableur comme **MS EXCEL**, chaque **onglet** de point de vue du produit lui-même est en fait une matrice (grille à deux dimensions). Les lignes sont fixes et représentées par 1,2,3,....et les colonnes sont fixes et représentées par des lettres A,B,C,.... Chaque cellule de l'onglet est repérée d'une manière unique par l'intersection de sa ligne par sa colonne. Pour les versions récentes de **MS EXCEL**, le nombre de lignes peut aller jusqu'à 2^20=1 048 576 et le nombre de colonnes peut aller jusqu'à 2^14 = 16 384.

En tant que développeur travaillant sur une base de données, il peut arriver pour une question d'usage que vos utilisateurs ont besoin de présenter les données d'une manière matricielle plutôt que sous forme de tableau. Cela pose alors la question naturelle : Comment passer d'une table à une matrice et vice-versa ?

2.La relation entre une table et une matrice

Dans ce chapitre, nous allons voir comment on peut passer d'une table à une matrice et vice-versa. Autrement dit, comment transformer une vue sous forme de table en une vue matricielle ? Le schéma suivant illustre cette transformation pour un exemple concret sur lequel nous allons travailler dans la suite :



Jour	Matière	Période	Prof	Salle
Lundi	Français	08-10	Christine	1
Lundi	Maths	10-12	Jean	1
Lundi	Anglais	13-15	Chantal	5
Lundi	Hist	15-16	Alain	6
Mardi	Maths	08-10	Jean	2
Mardi	Hist	10-12	Alain	5
Mardi	Geo	13-15	Alain	2
Mardi	Physique	15-16	Sandrine	2
Mercredi	Physqie	08-10	Sandrine	1
Mercredi	Anglais	10-12	Chantal	5
Mercredi	Maths	13-15	Jean	1
Mercredi	Françsis	15-16	Christine	1
Jeudi	SVT	08-10	Claude	7
Jeudi	Chimie	10-12	Sandrine	7
Jeudi	Françsis	13-15	Christine	2
Jeudi	Geo	15-16	Alain	6
Vendredi	Maths	08-10	Jean	1
Vendredi	Sport	10-12	Francis	
Vendredi	Français	13-15	Christine	1

	08-10	10-12	13-15	15-16
Lundi	Français	Maths	Anglais	Hist
Mardi	Maths	Hist	Geo	Physique
Mercredi	Physique	Anglais	Maths	Français
Jeudi	SVT	Chimie	Français	Geo
Vendredi	Maths	Sport	Français	

Fig 3. Passage d'une table à une matrice et vice-versa

1. La matrice peut être le résultat d'une transformation issue de plusieurs tables à la fois et non nécessairement d'une table unique
2. Les valeurs des entrées de la matrice peuvent combiner plusieurs colonnes sources
3. Chaque entrée de la matrice peut être fonction non seulement des valeurs de la ligne et de la colonne correspondantes, mais aussi des valeurs issues d'autres colonnes

Remarque :

ATTENTION ! Passer d'une table à une matrice sans appliquer de transformation sur la **colonne** qui devient les entrées de la matrice permet l'inverse, autrement dit, de passer de la matrice à la table. En revanche, si la **colonne** qui devient les entrées de la matrice subit une transformation, le passage inverse n'est pas garanti sauf si la transformation inverse est possible, i.e., retrouver les valeurs de la colonne des entrées de la matrice.

Passer de la TABLE à la MATRICE est une opération appelée PIVOT (pivoter une colonne en ligne des champs en l'occurrence ici les périodes 08-09, 10-12, ...). Autrement dit toutes les valeurs distinctes de la colonne Periode de la TABLE deviennent des noms des colonnes dans la MATRICE, et les valeurs des intersections Jour x Periode représentent tout simplement dans notre cas la Matière correspondante dans la TABLE. L'inverse passer de la MATRICE à la TABLE est appelée UNPIVOT. Le PIVOT est plutôt simple et constitue une commande SQL standard en langage SQL dans la plupart des produits SGBD comme ORACLE, SQL SERVER, MS ACCESS et même MS EXCEL sur lequel un exemple sera fourni. Le UNPIVOT, en revanche n'est pas standard et simple à mettre en place. Un lien vers un complément dans MS EXCEL vous sera fourni dans le chapitre qui suit. A noter que dans le SGBD open source **MySql** la commande **PIVOT** n'existe pas, mais il y a un moyen indirect pour le réaliser. Une recherche sur Internet vous apportera rapidement la réponse. D'ailleurs une section lui est réservée dans la suite de cet article dans laquelle je vous montre une astuce basée sur des fonctions mathématiques et inspirée d'un exemple plus simple que le présent disponible à cette adresse Internet : MySQL/Pivot table.

D'une manière générale, la construction de la table à partir de la matrice correspondante n'est possible que sous certaines conditions que voici :

- La fonction qui évalue les entrées de la matrice doit être réversible, autrement dit il existe une fonction inverse permettant de trouver toutes les valeurs des autres colonnes (à l'exception de la colonne correspondant aux lignes et la colonne qui a subi le pivot dans la matrice).
- La fonction doit avoir toutes les colonnes de la table comme variables à l'exception des 2 colonnes correspondant aux lignes de la matrice (en bleu) et celle qui & été pivotée (en rouge).

Si vous êtes familier avec l'utilisation du **tableau croisé dynamique dans MS EXCEL**, ce dernier ce n'est qu'une **requête PIVOT** gérée en arrière-plan d'une manière transparente par **EXCEL** qui présente plutôt des moyens graphiques plus simple à comprendre et utiliser. En revanche, les tableaux croisés dynamiques générés sont uniquement en lecture, autrement dit, vous ne pouvez pas modifier le contenu des entrées des matrices résultantes. En conséquence OUI c'est dynamique; mais dans un sens uniquement pour lire la table et générer la matrice ! Si vous utilisez la même fonctionnalité dans **MS ACCESS**, par magie, il vous est possible de voir la requête PIVOT derrière le tableau croisé dynamique, quel bonheur !

3.Comment construire un calendrier scolaire dans EXCEL ?

Nous prenons comme exemple celui de la figure du chapitre précédent pour montrer que le passage de la TABLE vers la MATRICE peut se faire en fait par le biais d'une requête SQL simple comme suit :

```
TRANSFORM First([Calendrier$].Matiere)
SELECT Jour
FROM [Calendrier$]
GROUP BY Jour
PIVOT 'Période : ' & [Calendrier$].Periode
```

Ici **Calendrier** est l'onglet qui contient la table. Si vous voulez en plus d'afficher la matière, le nom du Prof et le numéro de la salle de chaque matière, la requête SQL peut être écrite comme suit :

```
TRANSFORM First([Calendrier$].Matiere & chr(10) & chr(13) &  'Prof : ' & [Calendrier$].Prof
& chr(10) & chr(13) & 'Salle : ' & [Calendrier$].Salle)
SELECT Jour
FROM [Calendrier$]
GROUP BY Jour
PIVOT 'Période : ' & [Calendrier$].Periode
```

A noter que la chaine **chr(10) & chr(13)** représente une fin et un saut de ligne. Le résultat de cette requête donne la matrice suivante représentant tout simplement un emploie de temps scolaire :

Périodes → Jour ↓	**Période : 08-10**	**Période : 10-12**	**Période : 13-15**	**Période : 15-16**
Lundi	Français Prof : Christine Salle : 1	Maths Prof : Jean Salle : 1	Anglais Prof : Chantal Salle : 5	Hist Prof : Alain Salle : 6
Mardi	Maths Prof : Jean Salle : 2	Hist Prof : Alain Salle : 5	Geo Prof : Alain Salle : 2	Physique Prof : Sandrine Salle : 2
Mercredi	Physique Prof : Sandrine Salle : 1	Anglais Prof : Chantal Salle : 5	Maths Prof : Jean Salle : 1	Français Prof : Christine Salle : 1
Jeudi	SVT Prof : Claude Salle : 7	Chimie Prof : Sandrine Salle : 7	Français Prof : Christine Salle : 2	Geo Prof : Alain Salle : 6
Vendredi	Maths Prof : Jean Salle : 1	Sport Prof : Francis Salle :	Français Prof : Christine Salle : 1	

L'exemple ci-dessus est téléchargeable ici :

calendrier.xlsm

Afin de réaliser l'inverse, passer de la **MATRICE** à la **TABLE**, vous pouvez télécharger le complément suivant à l'adresse suivante :http://tduhameau.files.wordpress.com/2012/09/table2db.xla. Je l'ai testé et il fonctionne parfaitement sur notre exemple.

4.Comment construire un bulletin de notes dans EXCEL ?

Afin de rester dans le même thème que celui de l'exemple précédent, autrement dit l'éducation, nous allons présenter un exemple dont le but est de construire une matrice représentant les moyennes trimestrielles et annuelle de toutes les matières. Graphiquement l'objectif est présenté par le schéma suivant :

SOURCE : Table stockant les notes saisies des élèves

Eleve	Matiere	Note	Date	Trim	coefficient	Note_Coeff	Note cum
Sandrine DUPONT	Physique	5,50	01/09/2023	1	0,2	1,10	1,10
Simon DUPONT	Physique	14,30	01/09/2023	1	1,0	14,30	15,40
Luc DUPONT	Chimie	12,10	02/09/2023	1	0,4	4,84	20,24
Luc DUPONT	Hist	15,40	02/09/2023	1	0,4	6,16	26,40
Sandrine DUPONT	Chimie	5,50	02/09/2023	1	0,7	3,85	30,25
Sandrine DUPONT	Geo	10,00	02/09/2023	1	1,0	10,00	40,25
Yasmine DUPONT	Chimie	18,00	02/09/2023	1	0,4	7,20	47,45
Yasmine DUPONT	Hist	20,00	02/09/2023	1	0,4	8,00	55,45
Luc DUPONT	Français	7,70	03/09/2023	1	0,5	3,85	59,30
Luc DUPONT	Hist	17,60	03/09/2023	1	1,0	17,60	76,90
Luc DUPONT	Hist	12,10	03/09/2023	1	0,4	4,84	81,74
Nora DUPONT	Geo	5,50	03/09/2023	1	1,0	5,50	87,24
Nora DUPONT	Maths	11,00	03/09/2023	1	0,5	5,50	92,74

Fig 4. Table des Notes

La table contient les colonnes :

- **Eleve** : Nom de l'élève
- **Matiere** : Nom de la matière
- **Note** : La note attribuée pour la matière correspondante (sur la base de 20 par défaut, on aurait pu rendre la note max comme colonne supplémentaire fixée par le Professeur)
- **Date** : Date de la note
- **Trim** : Cette colonne représente le trimestre de l'année scolaire (3 en tout) est calculée en fonction de la date (en principe, cette colonne ne doit pas être stockée dans une base de données, car elle est calculée sur la base d'une colonne stockée !)
- **Coefficient** : Pondération attribuée à la note par le Professeur
- **Note_Coeff** : Note pondérée selon le Coefficient = Note x Coefficient (normalement ne doit pas être stockée !)

CIBLE : Matrice des moyennes par élève

Bulletin de notes annuel

Sélectionner un élève : Yasmine DUPONT

Matiere	Trim 1	Trim 2	Trim 3	Moyenne par matière	Moyenne classe par matière	Indicateur
Anglais	14,26	14,89	14,33	14,49	13,60	
Chimie	14,67	17,50	16,10	16,09	13,97	
Français	14,84	14,96	13,78	14,53	13,47	
Geo	12,60	16,73	14,59	14,64	14,49	
Hist	13,63	15,23	13,64	14,17	13,92	
Maths	15,29	14,42	15,54	15,08	12,98	
Physique	14,83	15,29	15,46	15,19	13,76	
Sport	17,23	16,63	14,29	16,05	14,57	
SVT	17,48	14,44	13,64	15,19	13,52	
Moyenne Trimestrielle :	14,98	15,57	14,60	15,05	13,81	
Moyenne Générale classe :	15,05					

Fig 5. Matrice des notes et moyennes trimestrielles

Cette matrice affiche pour chaque élève sélectionné :

- La moyenne par matière et par trimestre
- La moyenne générale par trimestre
- La moyenne générale de l'année

Cette matrice peut générer des graphes type RADAR par exemple comme vous pouvez l'examiner dans le fichier EXCEL en téléchargement ou le voir ci-après :

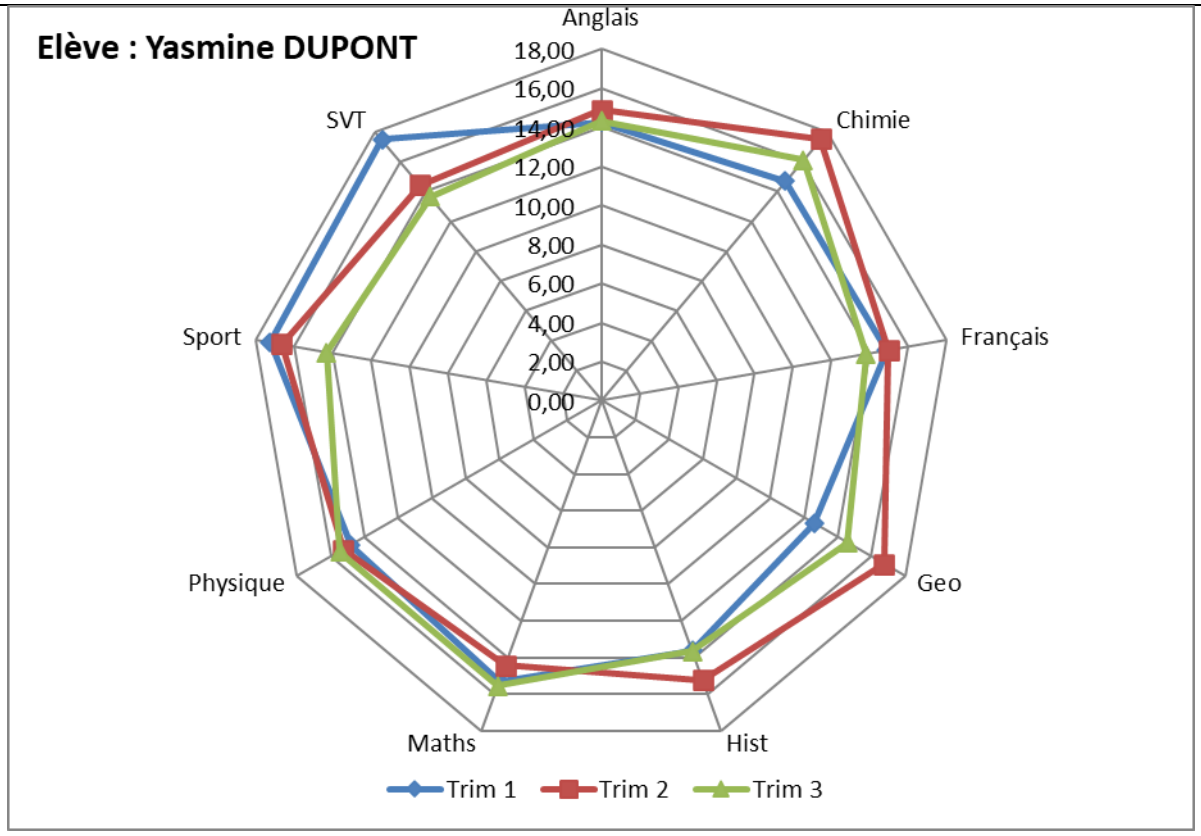


Fig 6. **RADAR des moyennes trimestrielles de toutes les matières**

La question est la suivante : Comment obtenir la cible (matrice) à partir de la source (table) ?

Comme nous l'avons déjà expliqué dans l'exemple de la section précédente, il suffit d'appliquer un PIVOT comme suit :

- La colonne **Matiere** représente les lignes de la matrice
- La colonne **Trim** est celle qui est pivotée comme étant les colonnes de la matrice
- Les **entrées** de la matrice est une fonction qui calcule la moyenne de chaque matière pour chaque trimestre, donc elle est fonction de la note et du coefficient de la note

Plus précisément, la requête PIVOT est la suivante :

```
TRANSFORM (SUM([Notes$].Note_Coeff))/SUM([Notes$].coefficient)
            SELECT Matiere
            FROM [Notes$]
            WHERE Eleve='" & Eleve & "'"
            GROUP BY Matiere
            PIVOT 'Trim ' & [Notes$].Trim
```

Ici **Notes** est l'onglet qui contient la table source. Autrement dit pour chaque matière la fonction f qui calcule les entrées est la suivante :

```
f(Note_Coeff, coefficient) = (SUM([Notes$].Note_Coeff))/SUM([Notes$].coefficient)
```

Ceci n'est juste que la somme des notes pondérées divisée par la somme des coefficients.

La requête SQL précédente ne doit pas normalement utiliser la colonne Note_Coeff et elle peut être exprimée comme suit :

```
TRANSFORM (SUM([Notes$].coefficient*[Notes$].Note))/SUM([Notes$].coefficient)
            SELECT Matiere
            FROM [Notes$]
            WHERE Eleve='" & Eleve & "'"
            GROUP BY Matiere
            PIVOT 'Trim ' & [Notes$].Trim
```

L'exemple complet ci-dessus est téléchargeable ici :

notes- v0.1.xlsm

5.Comment construire un bulletin de notes dans MySql ?

Afin de pouvoir réaliser l'astuce fournie dans cette section, vous devez maîtriser l'environnement du SGBD MySql disponible en téléchargement classique en accès WEB comme par exemple le package : XAMPP ou en accès locale sous Workbench : Workbench MySql.

Sachant que le PIVOT n'existe pas en tant que commande native dans le langage SQL de MySql, l'astuce est de combiner deux éléments :

- Utiliser la commande **SELECT** standard
- Utiliser deux fonctions mathématiques **SIGN** (fonction signe) et **ABS** (fonction valeur absolue) définies ci-dessous.

Rappelons d'abord les deux fonctions :

$$SIGN(x) = \begin{cases} -1\ Si\ x < 0 \\ 0\ Si\ x = 0 \\ +1\ Si\ x > 0 \end{cases}$$

$$ABS(x) = \begin{cases} -x\ Si\ x < 0 \\ x\ Si\ x \geq 0 \end{cases}$$

A partir de ces deux fonctions standards on peut déduire que :

$$1 - ABS(SIGN(x-1)) = \begin{cases} 1\ Si\ x = 1 \\ 0\ Sinon \end{cases}$$

et d'une manière générale :

$$a - ABS(SIGN(x-a)) = \begin{cases} 1\ Si\ x = a \\ 0\ Sinon \end{cases}$$

Pour commencer reprenons notre exemple MS EXCEL de la section précédente et commençons par une requête SQL simple. Pour cela, télécharger le **fichier CSV** représentant l**a table notes dans le fichier EXCEL** : notes.csv. Vous commencez par l'importer dans le SGBD MySql dans une **base de données appelée test** par exemple et dans **une nouvelle table appelée notes** comme indiqué ci-dessous :

Console de gestion MySql phPMyAdmin :

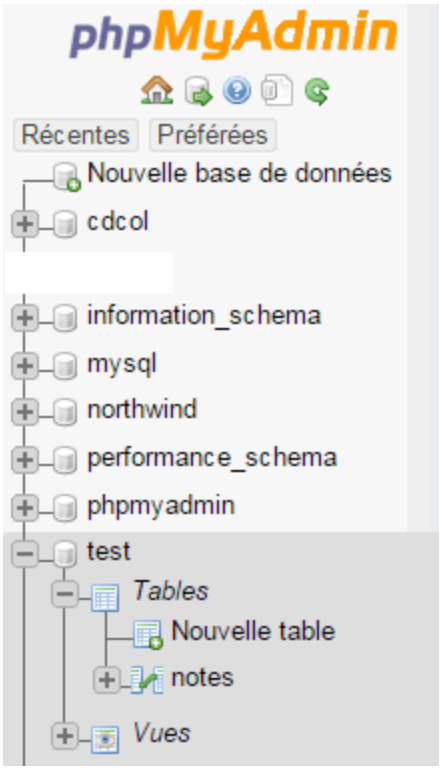


Fig 7. Console WEB phpMyAdmin de MySql

Table notes de la base de données test :

Eleve	Matiere	Note	Date	Trim	coefficient
Luc DUPONT	Anglais	8.80	2014-09-12	1	1.00
Luc DUPONT	Anglais	20.00	2014-12-29	1	0.80
Luc DUPONT	Anglais	20.00	2014-09-21	1	0.30
Luc DUPONT	Anglais	12.10	2014-09-22	1	0.70
Luc DUPONT	Anglais	5.50	2014-11-05	1	0.30
Luc DUPONT	Anglais	8.80	2014-10-11	1	0.40
Luc DUPONT	Anglais	13.20	2014-09-23	1	0.90
Luc DUPONT	Anglais	14.30	2014-09-08	1	0.60
Luc DUPONT	Anglais	13.20	2014-10-13	1	0.60
Luc DUPONT	Anglais	19.80	2014-11-04	1	1.00
Luc DUPONT	Anglais	16.50	2015-02-20	2	0.60
Luc DUPONT	Anglais	6.60	2015-03-20	2	0.80
Luc DUPONT	Anglais	20.00	2015-02-22	2	0.90
Luc DUPONT	Anglais	14.30	2015-03-24	2	0.80
Luc DUPONT	Anglais	7.70	2015-01-23	2	0.20
Luc DUPONT	Anglais	15.40	2015-01-25	2	0.70
Luc DUPONT	Anglais	9.90	2015-03-13	2	0.30
Luc DUPONT	Anglais	19.80	2015-03-30	2	0.60
Luc DUPONT	Anglais	9.90	2015-03-06	2	0.60
Luc DUPONT	Anglais	19.80	2015-02-17	2	0.10
Luc DUPONT	Anglais	7.70	2015-05-10	3	0.60

Fig 8. Table MySql **notes**

Dans l'onglet SQL de phpMyAdmin exécuter la requête suivante :

```
SELECT Matiere,
       (sum((1-ABS(SIGN(Trim - 1)))*coefficient*Note)/sum((1-ABS(SIGN(Trim - 1)))*coefficient)) AS Trim1
       FROM notes
       WHERE Eleve='Pierre DUPONT'
       GROUP BY Matiere
```

Vous devez avoir le résultat suivant (partiellement affiché) :

Matiere	Trim1
Anglais	16.571154
Chimie	16.378788
Français	13.804878
Geo	16.104878
Hist	18.018750
Maths	12.171739
Physique	11.476087
Sport	17.211538
SVT	15.978571

A partir de cet exemple simple affichant les moyennes du 1er trimestre uniquement, on peut le généraliser pour les autres trimestres et exécuter la requête suivante :

```
SELECT Matiere,
       (sum((1-ABS(SIGN(Trim - 1)))*coefficient*Note)/sum((1-ABS(SIGN(Trim - 1)))*coefficient)) AS Trim1,
       (sum((1-ABS(SIGN(Trim - 2)))*coefficient*Note)/sum((1-ABS(SIGN(Trim - 2)))*coefficient)) AS Trim2,
       (sum((1-ABS(SIGN(Trim - 3)))*coefficient*Note)/sum((1-ABS(SIGN(Trim - 3)))*coefficient)) AS Trim3
       FROM notes
       WHERE Eleve='Pierre DUPONT'
       GROUP BY Matiere
```

pour obtenir la matrice des moyennes des trois trimestres :

Matiere	Trim1	Trim2	Trim3
Anglais	16.571154	11.798387	14.258490
Chimie	16.378788	15.803333	16.193940
Français	13.804878	11.315517	16.156667
Geo	16.104878	15.026230	15.422727
Hist	18.018750	12.650000	12.404878
Maths	12.171739	12.780000	13.029545
Physique	11.476087	13.558209	14.478379
Sport	17.211538	18.800000	11.900000
SVT	15.978571	14.343750	11.000000

Afin de comparer avec les résultats obtenus dans l'exemple sous MS EXCEL de la section précédente, remplacer la condition ***Eleve="Pierre DUPONT"*** par ***Eleve='Simon DUPONT'*** et utiliser la fonction d'agrégation Round pour arrondir les moyennes à 2 décimales comme suit :

```
SELECT Matiere,
Round((sum((1-ABS(SIGN(Trim - 1)))*coefficient*Note)/sum((1-ABS(SIGN(Trim - 1)))*coefficient)), 2) AS Trim1,
Round((sum((1-ABS(SIGN(Trim - 2)))*coefficient*Note)/sum((1-ABS(SIGN(Trim - 2)))*coefficient)), 2) AS Trim2,
Round((sum((1-ABS(SIGN(Trim - 3)))*coefficient*Note)/sum((1-ABS(SIGN(Trim - 3)))*coefficient)), 2) AS Trim3
FROM notes
WHERE Eleve='Simon DUPONT'
GROUP BY Matiere
```

Vérifier que vous obtenez bien la matrice suivante identique à celle obtenue dans EXCEL :

Matiere	Trim1	Trim2	Trim3
Anglais	12.59	13.99	13.06
Chimie	9.28	12.84	12.33
Français	11.82	10.43	16.49
Geo	14.55	15.34	15.40
Hist	12.05	15.30	13.44
Maths	11.32	12.84	11.63
Physique	17.63	16.37	11.84
Sport	14.07	13.69	16.78
SVT	15.55	10.92	10.44

Maintenant vous savez tout ou presque à propos des requêtes PIVOT et quelles sont leurs utilisations, à vous d'appliquer ce que cet article vous a appris et en guise d'exercice, au lieu d'utiliser la requête PIVOT dans EXCEL utilisez plutôt la technique de cette section et comparer les résultats obtenus à la section précédente.

6.Comment automatiser la construction du bulletin de notes dans MySql ?

Afin d'automatiser la construction via une requête PIVOT du bulletin de notes pour n'importe quel élève comme le cas de notre exemple dans MS EXCEL, il est souhaitable de ne pas inclure en dur le nom de l'élève dans la requête SQL. Autrement dit, trouver un moyen pour le passer en paramètre à la requête et obtenir la matrice du bulletin de notes automatiquement.

Une des manières pour générer automatiquement le relevé de notes est de disposer d'une procédure qui accepte le nom de l'élève comme paramètre. Heureusement cette fonctionnalité existe dans MySql et elle est appelée **'PROCEDURE STOCKEE**'. Il s'agit de définir une procédure acceptant le nom de l'élève comme paramètre qui retourne le bulletin et pouvant être appelée à tout moment. Pour ce faire, il suffit de suivre les étapes suivantes dans **phpMyAdmin** :

1. Sélectionner la base de données **test** dans phpMyAdmin
2. Se positionner sur l'**onglet Procédures stockées**
3. Cliquer sur le lien **Ajouter une procédure**
4. La fenêtre suivante s'affiche :

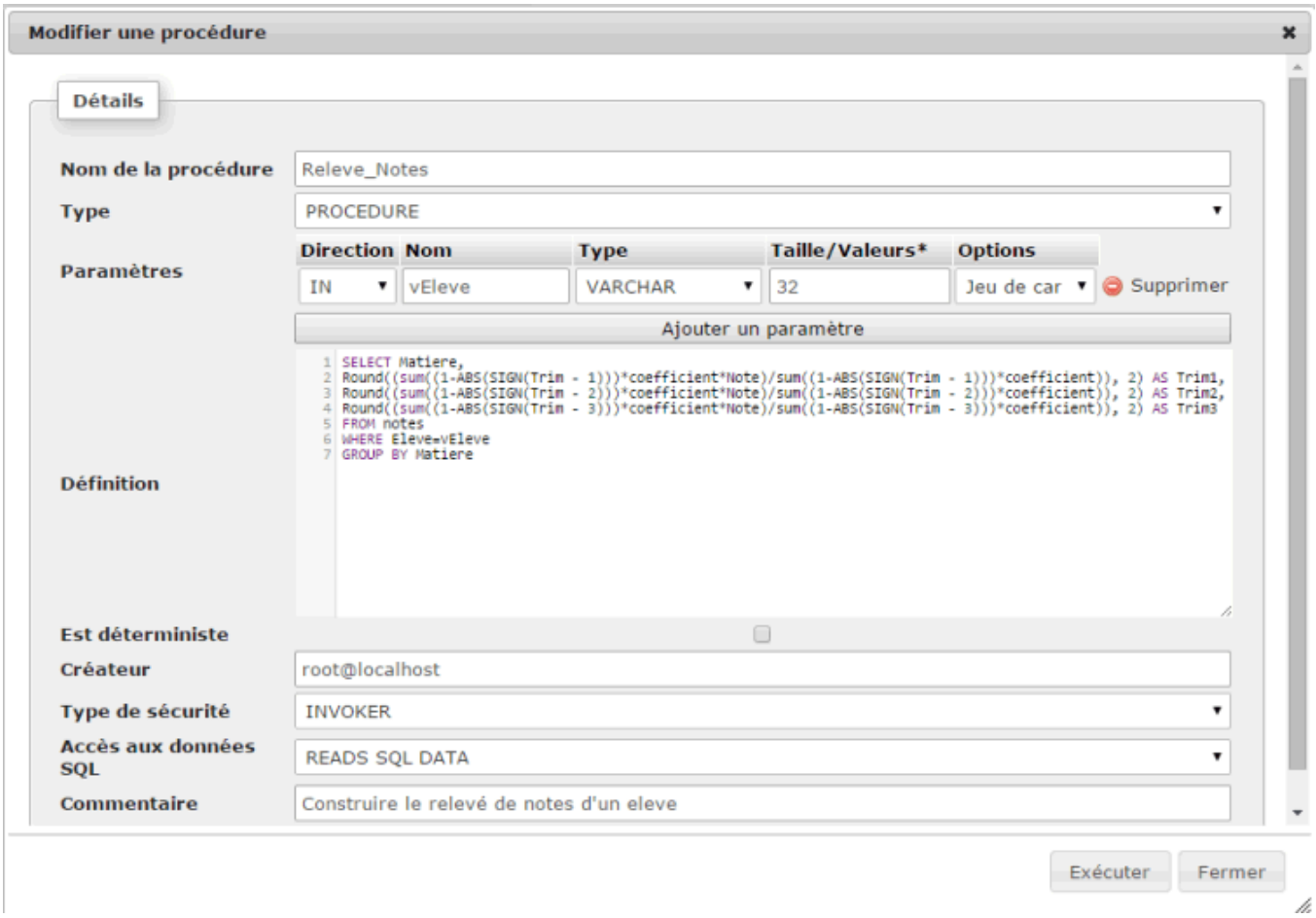
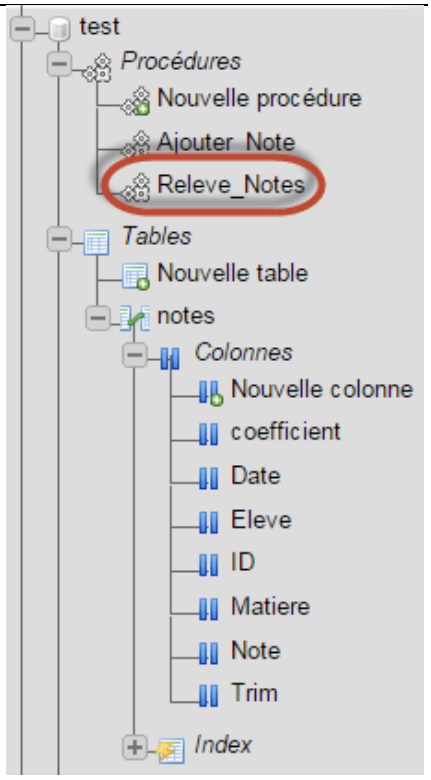


Fig 9. Ajouter une procédure stockée

5. Taper le nom de la procédure : **Releve_Note**
6. Ajouter le paramètre : **vEleve** tel qu'indiqué ci-dessous
7. Taper dans **Définition** la requête SQL :

```
SELECT Matiere,
Round((sum((1-ABS(SIGN(Trim - 1)))*coefficient*Note)/sum((1-ABS(SIGN(Trim - 1)))*coefficient)), 2) AS Trim1,
Round((sum((1-ABS(SIGN(Trim - 2)))*coefficient*Note)/sum((1-ABS(SIGN(Trim - 2)))*coefficient)), 2) AS Trim2,
Round((sum((1-ABS(SIGN(Trim - 3)))*coefficient*Note)/sum((1-ABS(SIGN(Trim - 3)))*coefficient)), 2) AS Trim3
FROM notes
WHERE Eleve=vEleve
GROUP BY Matiere
```

8. Puis cliquer sur **Exécuter** pour enregistrer la procédure stockée qui apparaît comme ci-contre :



9. Pour appeler cette procédure stockée afin de générer le bulletin d'un élève, vous cliquez sur la procédure et tapez le nom de l'élève puis cliquez sur **Exécuter** :

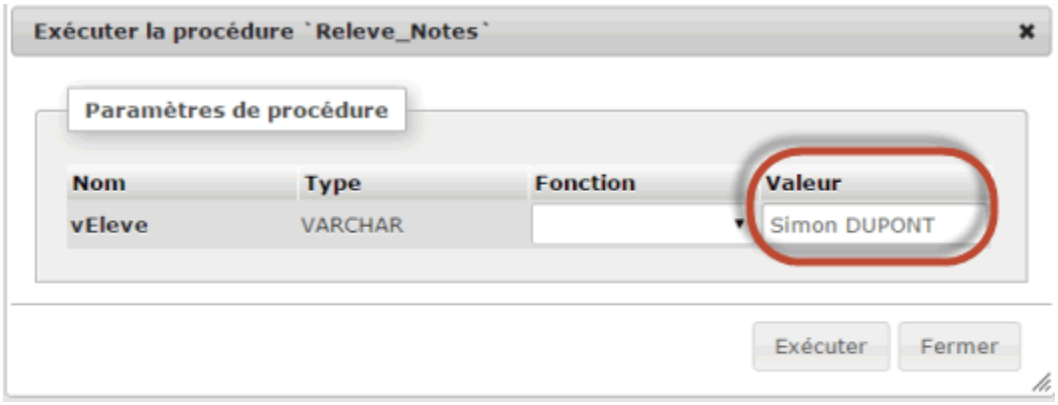


Fig 10. Exécuter une procédure stockée

ou vous tapez dans l'onglet SQL la requête suivante :

```
CALL Releve_Notes('Simon DUPONT');
```

Vous pouvez maintenant appeler cette procédure stockée dans n'importe quel programme capable d'accéder et exécuter les procédures stockées dans MySql comme PHP, ASP etc.

7.Utiliser POWER BI pour produire un bulletin de notes du futur !

La dernière mouture de Microsoft de l'un de leurs produits phares POWER BI faisant partie d'une suite appelée « Microsoft Fabric », peut produire en un minimum d'effort les mêmes résultats que MS EXCEL et beaucoup plus.

Le premier intérêt de passer à **POWER BI**, réside dans sa capacité à produire un tableau de bord attractif et interactif à la fois peu importe les sources de données utilisées (tables issues d'Excel ou de bases de données diverses). Les étapes suivantes sont réalisées dans l'ordre suivant :

1. Connecter POWER BI aux données sources même dont les formats peuvent être hétérogènes (la liste est longue)
2. Réaliser les différents ajustements, configurations et transformations (via **POWER QUERY** produit pouvant être connecté à EXCEL aussi)
3. Définir et construire le schéma du modèle (ajouter éventuellement des tables descriptives appelées communément « Tables de dimension », définir toutes les relations entre les tables). Cette étape est cruciale, car elle permet la contextualisation, la cohérence des données et la dynamique dans les visualisations (étape suivante).
4. Construire et produire des tableaux de bord visuels et dynamiques

Le tout se fait qu'une seule fois et toute mise à jour des données sources est répercutées sur toute la chaine précédente.

POWER BI se charge de rafraichir tout le cycle sans aucun effort supplémentaire ! C'est ce que cette section présente comme résultat sans aller dans les détails laissés aux spécialistes de ce produit, notamment quand il s'agit d'utiliser le nouveau langage **DAX (Data Analysis Expressions)** qui pourrait être nécessaire dans certaines situations moins banales.

Dans cette partie, nous reprenons une seule source de données, celle utilisée dans la section Un autre exemple plus complexe dans EXCEL (Comment construire un bulletin de notes ?). Le tableau de bord produit est présenté dans la page suivante, se rapprochant de ce qui pourrait être le bulletin de notes du futur :

1. Complètement digitalisé
2. Interactif
3. Intuitif

Je vous laisse explorer les multiples usages de ce produit, impressionnant au premier abord, mais dont l'utilisation deviendra plus naturelle, une fois on a bien compris les principes ainsi que les approches, et approfondi la compréhension. Les domaines d'utilisation de ce produit sont multiples dès le moment où il s'agit des données qui doivent être exploitées, collectées, analysées, nettoyées, transformées, améliorées, corrélées puis synthétisées dans des tableaux de bord facilement lisibles et compréhensibles qui seront in fine partagée dans des rapports. Le partage est réalisé au travers le service de publication WEB dans l'organisation de l'entreprise rattachée au compte utilisant POWER BI, comme par exemple dans un salon TEAMS d'un groupe de travail ou projet.

Le fichier POWER BI qui a servi à produire le bulletin de notes ci-dessous est téléchargeable ici :

Calendrier-Notes.pbix

QUELLE EST LA DIFFERENCE ENTRE TABLE ET MATRICE ? Par Abdel YEZZA, Ph.D

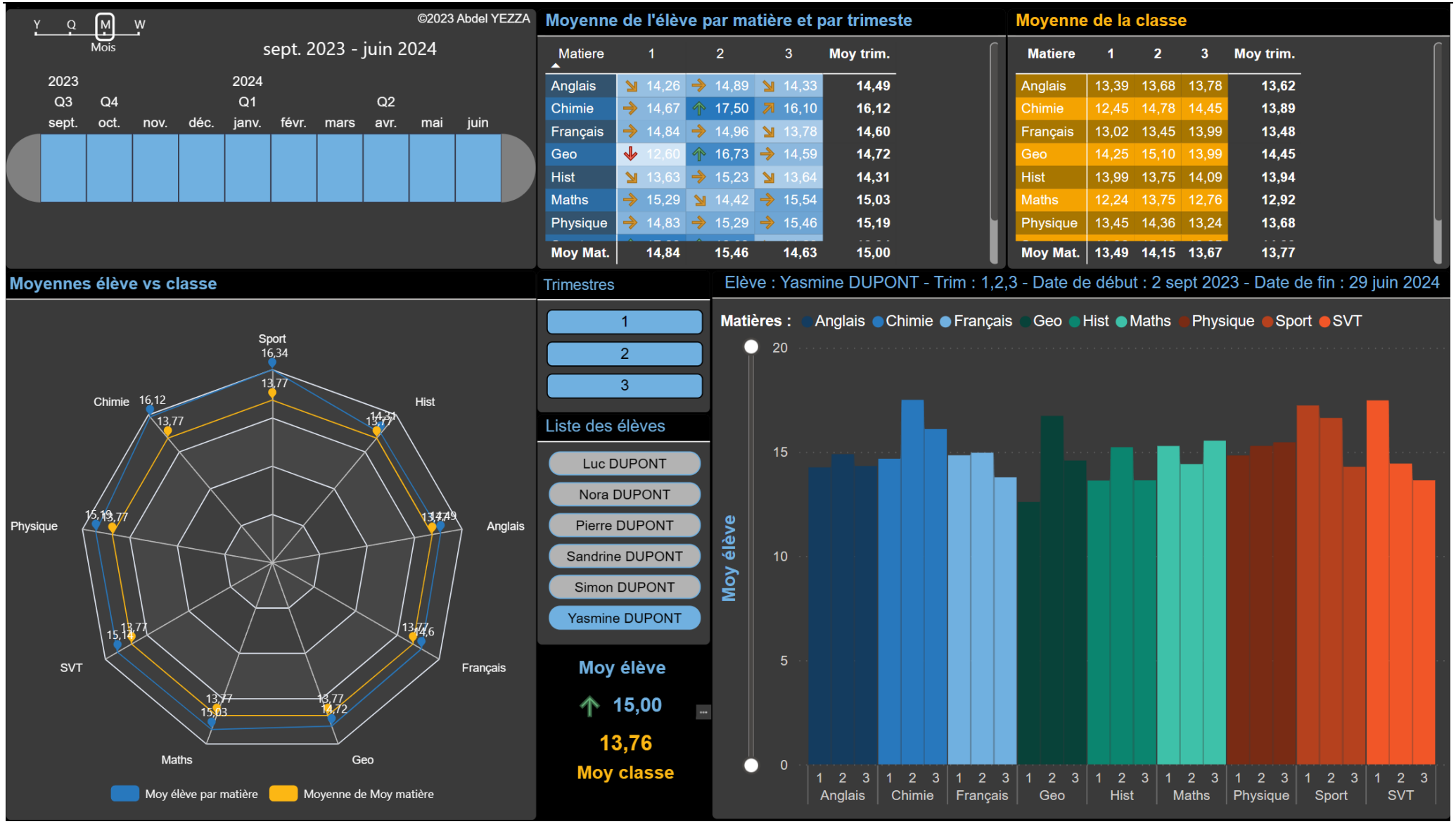


Fig 11. Bulletin de notes du futur !

Calendrier de la semaine

Jour	08-10	10-12	13-15	15-16
Lundi	Français Prof : Christine Salle : 1	Maths Prof : Jean Salle : 1	Anglais Prof : Chantal Salle : 5	Hist Prof : Alain Salle : 6
Mardi	Maths Prof : Jean Salle : 2	Hist Prof : Alain Salle : 5	Geo Prof : Alain Salle : 2	Physique Prof : Sandrine Salle : 2
Mercredi	Physique Prof : Sandrine Salle : 1	Anglais Prof : Chantal Salle : 5	Maths Prof : Jean Salle : 1	Françsis Prof : Christine Salle : 1
Jeudi	SVT Prof : Claude Salle : 7	Chimie Prof : Sandrine Salle : 7	Françsis Prof : Christine Salle : 2	Geo Prof : Alain Salle : 6
Vendredi	Maths Prof : Jean Salle : 1	Sport Prof : Francis Salle :	Français Prof : Christine Salle : 1	

Fig 12. Calendrier de la classe

8.Conclusion

Si vous êtes développeur, vous devez être capable de passer de la table à la matrice et éventuellement l'inverse selon le contexte et les exigences de votre application. Cependant, un utilisateur final va certainement préférer une vue matricielle plus parlante et plus simple à comprendre qu'une table avec des enregistrements dont certains champs sont répétitifs.

L'avantage du PIVOT est non seulement le passage d'une vue de table vers une vue matricielle, mais aussi appliquer des fonctions d'agrégation standards ou personnalisées comme nous l'avons fait dans deux exemples sous EXCEL et le SGBD MySql. Cet article vous a montré les principales relations entre une **TABLE** et une **MATRICE** et peut vous donner des idées pour trouver un algorithme simple et performant afin de réaliser le 'UNPIVOT' de la même manière que le PIVOT via des requêtes SQL dans le cas où les entrées de la matrice sont issues directement des colonnes.

Pour conclure, l'utilisation de POWER BI s'avère faciliter significativement la production non seulement des matrices à partir de plusieurs tables, mais surtout des tableaux de bord de qualité, synthétiques, soignés et expressifs bien que l'exemple qui a été donné est de la base école au sens propre du mot !