

Bases de données

Ecole Marocaine des Sciences de l'Ingénieur

© Yousra Lembachar

Variables dans SQL

Variables dans SQL

```
1 • SELECT
2     Etudiant.idEtudiant, nom, note, nomMatiere
3 FROM
4     ETUDIANT,
5     MATIERE,
6     NOTE
7 WHERE
8     NOTE.idEtudiant = ETUDIANT.idEtudiant
9     AND MATIERE.idMatiere = NOTE.idMatiere;
```

160% 48:9

Filter: File: 


idEtudiant	nom	note	nomMatiere
1	Samantha	15	Maths
2	Francis	20	Maths
3	Charles	13	Maths
4	Penelope	14	Maths
5	Craig	10	Maths
6	Steve	8	Maths
10	Patricia	5	Maths
1	Samantha	20	Sociologie
10	Patricia	4	Sociologie
11	Patricia	7	Nutrition

Les identifiants, les noms,
les notes et les matières
respectives des étudiants
ayant une note

Variables dans SQL

```
1 • SELECT
2     E.idEtudiant, nom, note, nomMatiere
3 FROM
4     ETUDIANT E,
5     MATIERE M,
6     NOTE N
7 WHERE
8     N.idEtudiant = E.idEtudiant
9     AND M.idMatiere = N.idMatiere;
```

160% 39:9

Filter: File: 

idEtudiant	nom	note	nomMatiere
1	Samantha	15	Maths
2	Francis	20	Maths
3	Charles	13	Maths
4	Penelope	14	Maths
5	Craig	10	Maths
6	Steve	8	Maths
10	Patricia	5	Maths
1	Samantha	20	Sociologie
10	Patricia	4	Sociologie
11	Patricia	7	Nutrition

Les identifiants, les noms,
les notes et les matières
respectives des étudiants
ayant une note

Variables dans SQL

```
1 • SELECT
2     E1.idEtudiant, E1.nom, E2.idEtudiant, E2.nom
3 FROM
4     ETUDIANT E1,
5     ETUDIANT E2
6 WHERE
7     E1.nom = E2.nom
8     AND E1.idEtudiant <> E2.idEtudiant
```

Les paires d'étudiants qui ont le même nom

160% 43:8

Filter: File:

idEtudiant	nom	idEtudiant	nom
▶ 11	Patricia	10	Patricia
10	Patricia	11	Patricia

```
1 • SELECT
2     E1.idEtudiant, E1.nom, E2.idEtudiant, E2.nom
3 FROM
4     ETUDIANT E1,
5     ETUDIANT E2
6 WHERE
7     E1.nom = E2.nom
8     AND E1.idEtudiant < E2.idEtudiant
```

Les paires d'étudiants qui ont le même nom sans répétition

160% 28:8

Filter: File:

idEtudiant	nom	idEtudiant	nom
▶ 10	Patricia	11	Patricia

Les fonctions d'agrégation dans SQL

SELECT FONC_AGGR(COL),...

FROM nomTable1, nomTable2,...

WHERE condition

GROUP BY COL

HAVING condition

FONC_AGGR(COL) : **MIN** | **MAX** | **AVG** | **SUM** | **COUNT**

La fonction COUNT

```
1 • SELECT
2     COUNT(note)
3 FROM
4     NOTE ;
```

Le nombre de notes

Filter:	10:4
Filter: <input type="text"/>	File:
COUNT(note)	
▶ 8	

```
1 • SELECT
2     COUNT(idEtudiant)
3 FROM
4     ETUDIANT ;
```

Le nombre d'étudiants


Filter:	1:1
Filter: <input type="text"/>	File:
COUNT(idEtudiant)	
▶ 8	

La fonction COUNT

Le nombre d'étudiants avec
une note > 10

```
1 • SELECT
2     COUNT(idEtudiant)
3 FROM
4     NOTE
5 WHERE
6     note > 10
7 ;
```

1:1

Filter: File: 

COUNT(idEtudiant)
▶ 4

Les fonctions MIN & MAX

```
1 • SELECT
2     min(note)
3 FROM
4     NOTE
```

La note minimum

9:4

Filter: File:

min(note)
▶ 9

```
1 • SELECT
2     max(note)
3 FROM
4     NOTE
```

La note maximum

8:2

Filter: File:



max(note)
▶ 20

La fonction AVG

La moyenne des notes

```
1 • SELECT
2     AVG(note)
3 FROM
4     NOTE
```

13:2

Filter:  File: 

AVG(note)
▶ 12.0000

La fonction SUM

La somme des notes

```
1 • SELECT
2     sum(note)
3 FROM
4     NOTE
```

9:4

Filter: File:

sum(note)
▶ 96

```
1 • SELECT
2     sum(note)/count(idEtudiant), avg(note)
3 FROM
4     NOTE
```

42:2

Filter: File:

sum(note)/cou...	avg(note)
▶ 12.0000	12.0000

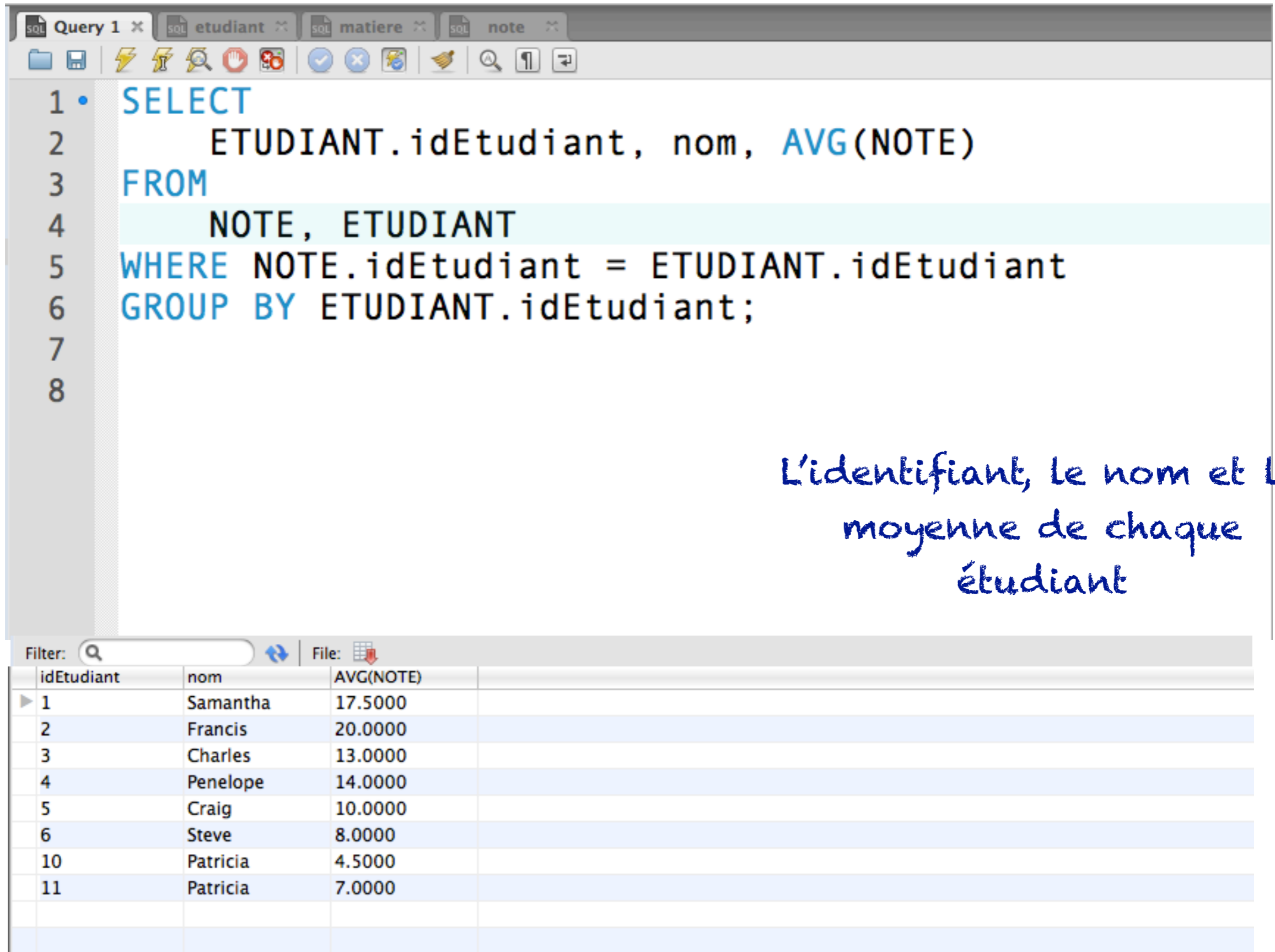
La clause GROUP BY

```
Query 1 x | etudiant x | matiere x | note x
SELECT
  idEtudiant, AVG(NOTE)
FROM
  NOTE
GROUP BY idEtudiant;
```

L'identifiant et la moyenne
de chaque étudiant

idEtudiant	AVG(NOTE)
1	17.5000
2	20.0000
3	13.0000
4	14.0000
5	10.0000
6	8.0000
10	4.5000
11	7.0000

La clause GROUP BY



The screenshot shows a SQL IDE window with a query editor and a results grid. The query editor contains the following SQL code:

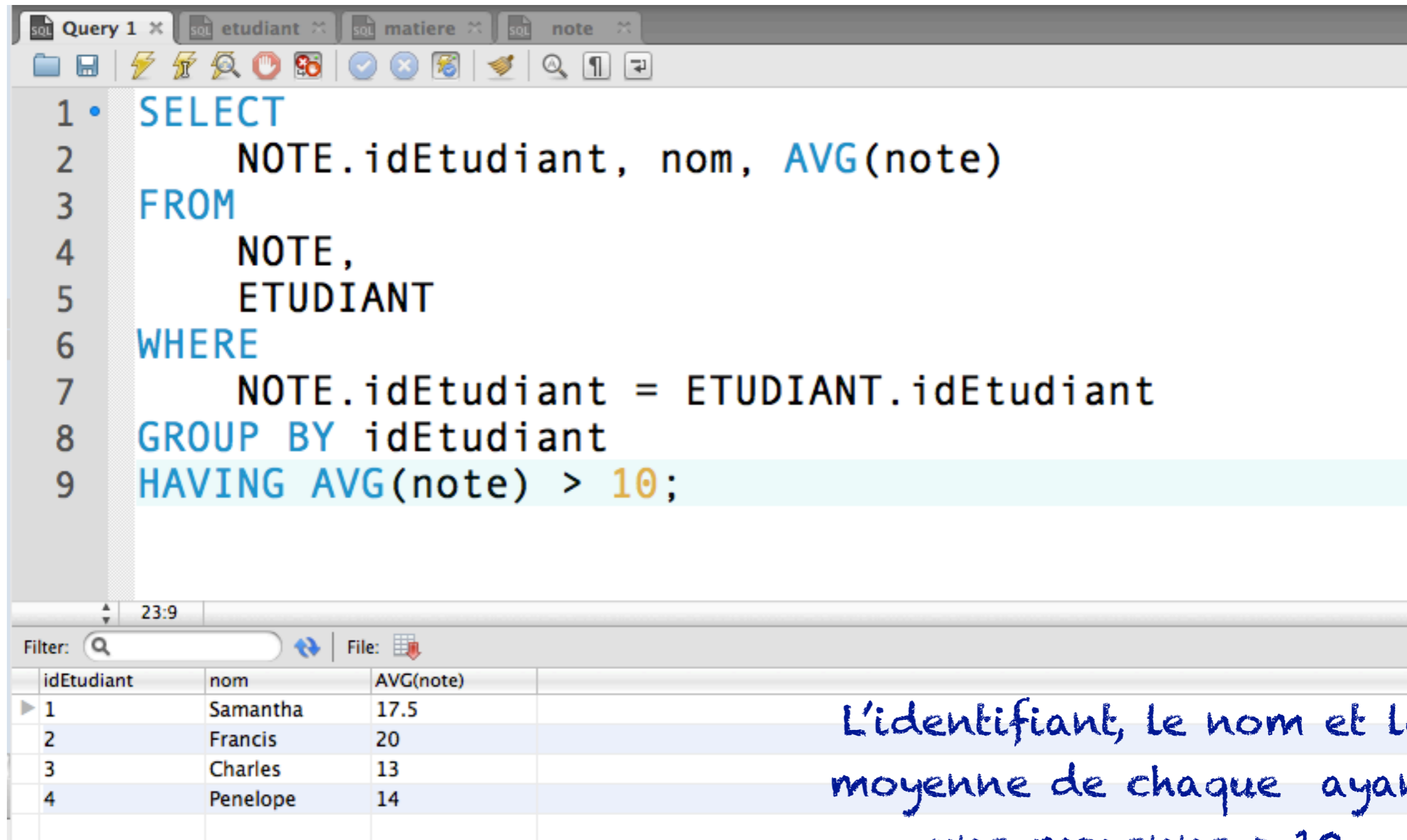
```
1 • SELECT
2     ETUDIANT.idEtudiant, nom, AVG(NOTE)
3 FROM
4     NOTE, ETUDIANT
5 WHERE NOTE.idEtudiant = ETUDIANT.idEtudiant
6 GROUP BY ETUDIANT.idEtudiant;
7
8
```

The results grid displays the following data:

idEtudiant	nom	AVG(NOTE)
1	Samantha	17.5000
2	Francis	20.0000
3	Charles	13.0000
4	Penelope	14.0000
5	Craig	10.0000
6	Steve	8.0000
10	Patricia	4.5000
11	Patricia	7.0000

L'identifiant, le nom et la
moyenne de chaque
étudiant

La clause HAVING



The screenshot shows a SQL IDE window with a query editor and a results table. The query is as follows:

```
1 • SELECT
2     NOTE.idEtudiant, nom, AVG(note)
3 FROM
4     NOTE,
5     ETUDIANT
6 WHERE
7     NOTE.idEtudiant = ETUDIANT.idEtudiant
8 GROUP BY idEtudiant
9 HAVING AVG(note) > 10;
```

The results table below the query shows the following data:

idEtudiant	nom	AVG(note)
1	Samantha	17.5
2	Francis	20
3	Charles	13
4	Penelope	14

Handwritten blue text on the right side of the results table reads: "L'identifiant, le nom et la moyenne de chaque ayant une moyenne > 10".

Opérateurs ensemblistes

UNION

```
1 • SELECT nom as chaine FROM ETUDIANT  
2 UNION  
3 SELECT nomMatiere as chaine FROM MATIERE;
```

160% 42:3

Filter:



File:



chaine

▶ Samantha

Francis

Charles

Penelope

Craig

Steve

Dave

Cameron

Patricia

Maths

Sociologie

Nutrition

Art Moderne

Art Contempo...

Art Classique

Tous les noms des étudiants
et des matières

INTERSECT, EXCEPT

- Clauses non supportées par MySQL
- INTERSECT pour l'intersection de deux ensembles
- EXCEPT pour la différence entre deux ensembles