CM 1 : Requêtes simples

La base de données `eleve.db` est disponible sur la page du cours.

Q1 Afficher tout le contenu de la table Eleve.

```
SELECT * FROM Eleve;
```

ID	Prenom	Nom	DdN	Portable	Sex	Taille
1	Jean	Dubois	14/01/94	06.42.12.48.13	M	174
2	Bruno	Dupont	06/04/97	06.65.84.23.00	M	168
4	Marion	Dubois	12/02/99	NULL	F	170
6	Simone	Brun	NULL	06.14.75.64.82	F	168
3	Jean	Lapont	28/10/97	NULL	M	186
12	Alexandre	Caston	21/09/97	06.42.62.50.31	NULL	162

Q2 Afficher tout le contenu de la table Eleve.

```
SELECT * FROM Eleve;
```

ID	Prenom	Nom	DdN	Portable	Sex	Taille
1	Jean	Dubois	14/01/94	06.42.12.48.13	M	174
2	Bruno	Dupont	06/04/97	06.65.84.23.00	M	168
4	Marion	Dubois	12/02/99	NULL	F	170
6	Simone	Brun	NULL	06.14.75.64.82	F	168
3	Jean	Lapont	28/10/97	NULL	M	186
12	Alexandre	Caston	21/09/97	06.42.62.50.31	NULL	162

Q3 Afficher les premières 4 lignes de la table Eleve.

```
SELECT * FROM Eleve LIMIT 4;
```

ID	Prenom	Nom	DdN	Portable	Sex	Taille
1	Jean	Dubois	14/01/94	06.42.12.48.13	M	174
2	Bruno	Dupont	06/04/97	06.65.84.23.00	M	168
4	Marion	Dubois	12/02/99	NULL	F	170
6	Simone	Brun	NULL	06.14.75.64.82	F	168

Q4 Afficher les prénoms des élèves.

```
SELECT Prenom FROM Eleve;
```

Prenom
Jean
Bruno
Marion
Simone
Jean
Alexandre

Q5 Afficher les prénoms des élèves sans repetition

```
SELECT DISTINCT Prenom FROM Eleve;
```

Prenom
Jean
Bruno
Marion
Simone
Alexandre

Q6 Trouver le numéro de portable de Jean

```
SELECT Portable FROM Eleve WHERE Prenom = 'Jean';
```

Portable
06.42.12.48.13
NULL

Q7 Trouver le numéro de portable de Jean Dubois

```
SELECT Portable FROM Eleve WHERE Prenom = 'Jean' AND Nom = 'Dubois';
```

Portable
06.42.12.48.13

Q8 Trouver le nom et prenom de la personne dont le numéro de portable est 06.14.75.64.82.

```
SELECT Nom, Prenom FROM Eleve WHERE Portable = '06.14.75.64.82';
```

Nom	Prenom
Brun	Simone

Q9 Trouver les prénoms de membres des familles Dupont et Dubois

```
SELECT Prenom FROM Eleve WHERE Nom = 'Dupont' OR Nom = 'Dubois';
```

ou encore

```
SELECT Prenom FROM Eleve WHERE Nom IN ('Dupont','Dubois');
```

Prenom
Jean
Bruno
Marion

Q10 Trouver les élèves qui font entre 170 et 180 cm de taille.

```
SELECT Prenom, Nom FROM Eleve WHERE Taille >= 170 AND Taille <= 180;
```

ou encore

```
SELECT Prenom, Nom FROM Eleve WHERE Taille BETWEEN 170 AND 180;
```

Prenom	Nom
Jean	Dubois
Marion	Dubois

Q11 Trouver les personnes qui ont “pont” dans leur nom de famille.

```
SELECT Prenom, Nom FROM Eleve WHERE Nom LIKE '%pont%';
```

Prenom	Nom
Bruno	Dupont
Jean	Lapont

Q12 Afficher la taille en mètres de tout le monde (avec leur prénoms et noms)

```
SELECT Prenom, Nom, Taille/100.0 FROM Eleve;
```

Prenom	Nom	Taille/100.0
Jean	Dubois	1.74
Bruno	Dupont	1.68
Marion	Dubois	1.7
Simone	Brun	1.68
Jean	Lapont	1.86
Alexandre	Caston	1.62

Q13 Afficher les noms complètes des filles avec leur civilité (Mlle.)

```
SELECT 'Mlle. ' || Prenom || ' ' || Nom AS "Nom complet"
  FROM Eleve
 WHERE Sex = 'F';
```

Nom complet
Mlle. Marion Dubois
Mlle. Simone Brun

Q14 Trouver la moyenne de tailles des élèves

```
SELECT AVG(Taille) FROM Eleve;
```

AVG(Taille)
171.333333333333

Q15 Trouver la somme des tailles de tous les élèves

```
SELECT SUM(Taille) FROM Eleve;
```

SUM(Taille)
1028

Q16 Trouver le minimum et maximum de tailles des garçons

```
SELECT MIN(Taille), MAX(Taille) FROM Eleve WHERE Sex = 'M';
```

MIN(Taille)	MAX(Taille)
168	186

Q17 Trouver le nombre des garçons

```
SELECT COUNT(*) FROM Eleve WHERE Sex = 'M';
```

COUNT(*)
3

Q18 Trouver le nombre des prénoms uniques masculins

```
SELECT COUNT(DISTINCT Prenom) FROM Eleve WHERE Sex = 'M';
```

COUNT(DISTINCT Prenom)
2

Q19 Ordonner les etudiants par leur taille (l'ordre croissant)

```
SELECT Prenom, Nom FROM Eleve ORDER BY Taille;
```

Prenom	Nom
Alexandre	Caston
Bruno	Dupont
Simone	Brun
Marion	Dubois
Jean	Dubois
Jean	Lapont

Q20 Ordonner les etudiants par leur taille (l'ordre decroissant)

```
SELECT Prenom, Nom FROM Eleve ORDER BY Taille DESC;
```

Prenom	Nom
Jean	Lapont
Jean	Dubois
Marion	Dubois
Bruno	Dupont
Simone	Brun
Alexandre	Caston

Q21 Ordonner les élèves par leur nom de famille et leur prenom (l'ordre croissant)

```
SELECT Prenom, Nom FROM Eleve ORDER BY Prenom, Nom;
```

Prenom	Nom
Alexandre	Caston
Bruno	Dupont
Jean	Dubois
Jean	Lapont
Marion	Dubois
Simone	Brun

Q22 Afficher les élèves qui n'ont pas (communiqué) de téléphone portable

```
SELECT Prenom, Nom FROM Eleve WHERE Portable IS NULL;
```

Prenom	Nom
Marion	Dubois
Jean	Lapont

Q23 Afficher un carnet de numéros portables (nom complet, numero portable) contenant tous les etudiants qui ont un portable.

```
SELECT Prenom || ' ' || Nom As "Nom complet", Portable
  FROM Eleve
 WHERE Portable IS NOT NULL;
```

Nom complet	Portable
Jean Dubois	06.42.12.48.13
Bruno Dupont	06.65.84.23.00
Simone Brun	06.14.75.64.82
Alexandre Caston	06.42.62.50.31

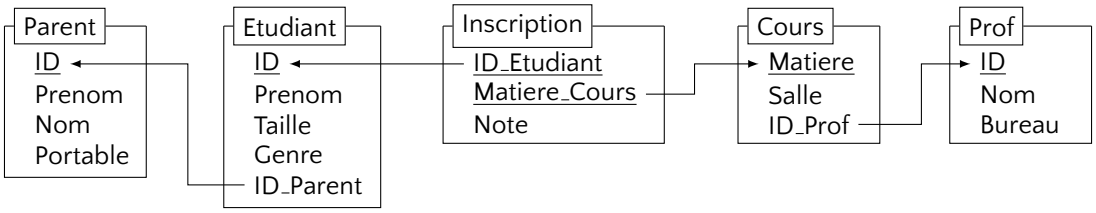
Q24 Afficher les noms complets de tous les élèves avec leur civilité

```
SELECT 'M. ' || Prenom || ' ' || Nom AS "Nom complet" FROM Eleve WHERE Sex = 'M'
 UNION
SELECT 'Mlle. ' || Prenom || ' ' || Nom FROM Eleve WHERE Sex = 'F';
```

Nom complet
M. Bruno Dupont
M. Jean Dubois
M. Jean Lapont
Mlle. Marion Dubois
Mlle. Simone Brun

CM 2 : Requêtes de jointure

La base de données `fac.db` est disponible sur la page du cours. Son schéma est présenté ci-dessous.



Les contenus des tables sont suivants.

Etudiant

ID	Prenom	Taille	Genre	ID_Parent
1	Jean	178	H	1
3	Jean	162	H	3
4	Marie	159	H	1
5	Paul	161	F	2
6	Luc	161	H	NULL
8	Marion	159	F	4

Parent

ID	Prenom	Nom	Portable
1	Bruno	Dubois	06.14.21.56.34
2	Constance	Dupont	06.41.21.32.14
3	Adèle	Martin	06.84.81.96.12

Cours

Matiere	Salle	ID_Prof
SQL	B2.461	11
HTML	A2.061	46
IA	A1.423	11

Prof

ID	Nom	Bureau
11	Sławek	D.42
46	Fabien	C.21
57	Marc	D.42

Inscription

ID_Etudiant	Note	Matiere_Cours
1	12.0	SQL
1	12.0	HTML
1	NULL	IA
3	15.0	SQL
4	16.0	SQL
4	17.0	HTML
4	12.0	IA
6	11.0	SQL
6	NULL	IA
8	16.0	HTML
8	NULL	IA

Q1 Afficher l'information complète sur les cours avec l'information complète sur les professeurs qui les assurent

```
SELECT * FROM Cours JOIN Prof ON (Cours.ID_Prof=Prof.ID);
```

Matiere	Salle	ID_Prof	ID	Nom	Bureau
SQL	B2.461	11	11	Sławek	D.42
HTML	A2.061	46	46	Fabien	C.21
IA	A1.423	11	11	Sławek	D.42

Q2 Afficher l'information complète sur les cours avec les noms des professeurs qui les assurent.

```
SELECT Cours.*, Prof.Nom FROM Cours JOIN Prof ON (Cours.ID_Prof=Prof.ID);
```

Matiere	Salle	ID_Prof	Nom
SQL	B2.461	11	Sławek
HTML	A2.061	46	Fabien
IA	A1.423	11	Sławek

Q3 Afficher, sans repetition, les noms des professeurs qui enseignent un (ou plusieurs) cours.

```
SELECT DISTINCT Prof.Nom FROM Cours JOIN Prof ON (Cours.ID_Prof=Prof.ID);
```

Nom
Sławek
Fabien

Q4 Afficher les nom des professeurs qui enseignent des cours dans la salle B2.461.

```
SELECT Prof.Nom
  FROM Cours
  JOIN Prof ON (Cours.ID_Prof=Prof.ID)
 WHERE Cours.Salle='B2.461';
```

ou en utilisant les aliases pour rendre la requêtes plus courte

```
SELECT P.Nom
  FROM Cours AS C
  JOIN Prof AS P ON (C.ID_Prof=P.ID)
  WHERE C.Salle='B2.461';
```

Nom
Sławek

Q5 Afficher le prénom et le nom des étudiants

```
SELECT Etudiant.Prenom, Parent.Nom
  FROM Etudiant JOIN Parent ON (Etudiant.ID_Parent=Parent.ID);
```

Prenom	Nom
Jean	Dubois
Jean	Martin
Marie	Dubois
Paul	Dupont

Q6 Afficher la liste des notes de Marie

```
SELECT Inscription.Matiere_Cours, Inscription.Note
  FROM Etudiant
  JOIN Inscription ON (Etudiant.ID=Inscription.ID_Etudiant)
 WHERE Etudiant.Prenom='Marie';
```

Matiere_Cours	Note
HTML	17.0
IA	12.0
SQL	16.0

Q7 Calculer la moyenne des notes de Marie

```
SELECT AVG(Inscription.Note)
  FROM Etudiant
  JOIN Inscription ON (Etudiant.ID=Inscription.ID_Etudiant)
 WHERE Etudiant.Prenom='Marie';
```

AVG(Inscription.Note)
15.0

Q8 Afficher les cours et les nom les professeurs qui n'ont pas renseigne tous les notes

```
SELECT DISTINCT Cours.Matiere, Prof.Nom
  FROM Inscription
  JOIN Cours ON (Inscription.Matiere_Cours=Cours.Matiere)
  JOIN Prof ON (Cours.ID_Prof=Prof.ID)
 WHERE Inscription.Note IS NULL;
```

Matiere	Nom
IA	Sławek

Produit cartesien

Q9 Construire le Produit cartésien des tables Cours et Prof

```
SELECT * FROM Cours, Prof;
```

Matiere	Salle	ID_Prof	ID	Nom	Bureau
SQL	B2.461	11	11	Sławek	D.42
SQL	B2.461	11	46	Fabien	C.21
SQL	B2.461	11	57	Marc	D.42
HTML	A2.061	46	11	Sławek	D.42
HTML	A2.061	46	46	Fabien	C.21
HTML	A2.061	46	57	Marc	D.42
IA	A1.423	11	11	Sławek	D.42
IA	A1.423	11	46	Fabien	C.21
IA	A1.423	11	57	Marc	D.42

Q10 Afficher l'information complète sur les cours avec l'information complète sur les professeurs qui les assurent, en utilisant la jointure croisée.

```
SELECT * FROM Cours, Prof WHERE (Cours.ID_Prof=Prof.ID);
```

Matiere	Salle	ID_Prof	ID	Nom	Bureau
SQL	B2.461	11	11	Sławek	D.42
HTML	A2.061	46	46	Fabien	C.21
IA	A1.423	11	11	Sławek	D.42

Jointure asymétrique

Q11 Afficher le prénom et si renseigné le nom de famille de tout étudiant

```
SELECT Etudiant.Prenom, Parent.Nom
  FROM Etudiant LEFT OUTER JOIN Parent ON (Etudiant.ID_Parent=Parent.ID);
```

Prenom	Nom
Jean	Dubois
Jean	Martin
Marie	Dubois
Paul	Dupont
Luc	NULL
Marion	NULL

Q12 Afficher la liste des nom complets des étudiants qui suivent le cours de HTML dans l'ordre décroisant de leur notes.

```
          SELECT Etudiant.Prenom, Parent.Nom, Inscription.Note
            FROM Etudiant
LEFT OUTER JOIN Parent ON (Etudiant.ID_Parent=Parent.ID)
            JOIN Inscription ON (Etudiant.ID=Inscription.ID_Etudiant)
           WHERE Matiere_Cours='HTML'
        ORDER BY Inscription.Note DESC;
```

Prenom	Nom	Note
Marie	Dubois	17.0
Marion	NULL	16.0
Jean	Dubois	12.0

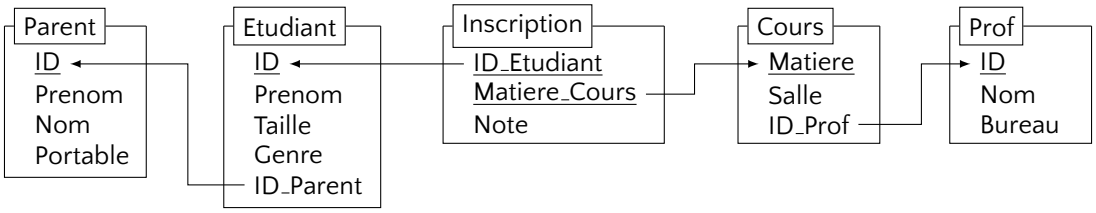
Q13 Afficher les noms des professeurs qui n'enseignent aucun cours (sans répétition).

```
SELECT DISTINCT Prof.Nom
  FROM Prof LEFT OUTER JOIN Cours ON (Prof.ID=Cours.ID_Prof)
 WHERE Cours.Matiere IS NULL;
```

Nom
Marc

CM 3 : Requêtes de jointure avancées

La base de données `fac.db` est disponible sur la page du cours. Son schéma est présenté ci-dessous.



Les contenus des tables sont suivants.

Etudiant

ID	Prenom	Taille	Genre	ID_Parent
1	Jean	178	H	1
3	Jean	162	H	3
4	Marie	159	H	1
5	Paul	161	F	2
6	Luc	161	H	NULL
8	Marion	159	F	4

Parent

ID	Prenom	Nom	Portable
1	Bruno	Dubois	06.14.21.56.34
2	Constance	Dupont	06.41.21.32.14
3	Adèle	Martin	06.84.81.96.12

Cours

Matiere	Salle	ID_Prof
SQL	B2.461	11
HTML	A2.061	46
IA	A1.423	11

Prof

ID	Nom	Bureau
11	Sławek	D.42
46	Fabien	C.21
57	Marc	D.42

Inscription

ID_Etudiant	Note	Matiere_Cours
1	12.0	SQL
1	12.0	HTML
1	NULL	IA
3	15.0	SQL
4	16.0	SQL
4	17.0	HTML
4	12.0	IA
6	11.0	SQL
6	NULL	IA
8	16.0	HTML
8	NULL	IA

Q1 Afficher les noms des professeurs qui enseignent un (ou plusieurs) cours (sans répétition).

```
SELECT DISTINCT Prof.Nom FROM Cours JOIN Prof ON (Cours.ID_Prof=Prof.ID);
```

ou

```
SELECT Prof.Nom
  FROM Prof
 WHERE ID IN (SELECT ID_Prof FROM Cours);
```

ou encore

```
SELECT Prof.Nom
  FROM Prof
 WHERE EXISTS (SELECT * FROM Cours WHERE Cours.ID_Prof=Prof.ID);
```

Nom
Sławek
Fabien

Q2 Afficher les noms des professeurs qui n'enseignent aucun cours (sans répétition).

```
SELECT Prof.Nom
  FROM Prof LEFT OUTER JOIN Cours ON (Prof.ID=Cours.ID_Prof)
 WHERE Cours.Matiere IS NULL;
```

ou

```
SELECT Prof.Nom
  FROM Prof
 WHERE ID NOT IN (SELECT ID_Prof FROM Cours);
```

ou encore

```
SELECT Prof.Nom
  FROM Prof
 WHERE NOT EXISTS (SELECT * FROM Cours WHERE Cours.ID_Prof=Prof.ID);
```

Nom
Marc

Q3 Afficher les étudiants dont le parent n'est pas renseigné dans la base de données.

```
SELECT Etudiant.*
  FROM Etudiant LEFT OUTER JOIN Parent ON (Etudiant.ID_Parent=Parent.ID)
 WHERE Parent.ID IS NULL;
```

ou

```
SELECT *
  FROM Etudiant
 WHERE ID_Parent IS NULL OR ID_Parent NOT IN (SELECT ID FROM Parent);
```

ou encore

```
SELECT *
  FROM Etudiant
 WHERE NOT EXISTS (
   SELECT *
   FROM Parent
   WHERE Parent.ID = Etudiant.ID_Parent
);
```

ID	Prenom	Taille	Genre	ID_Parent
6	Luc	161	H	NULL
8	Marion	159	F	4

Q4 Pour tout cours afficher sa matière avec le nom de prof qui l'assure.

```
SELECT Cours.Matiere, Prof.Nom
  FROM Cours JOIN Prof ON (Cours.ID_Prof = Prof.ID);
```

ou

```
SELECT C.Matiere, P.Nom
  FROM Cours AS C JOIN Prof AS P ON (C.ID_Prof = P.ID);
```

ou encore plus simplement

```
SELECT C.Matiere, P.Nom
  FROM Cours C JOIN Prof P ON (C.ID_Prof = P.ID);
```

Matiere	Nom
SQL	Sławek
HTML	Fabien
IA	Sławek

Q5 Afficher le nom complet de tout étudiant qui suit un cours assuré par un professeur dont le bureau est D.42

```
          SELECT DISTINCT E.Prenom, P.Nom
            FROM Etudiant E
LEFT OUTER JOIN Parent P ON (E.ID_Parent = P.ID)
            JOIN Inscription I ON (E.ID = I.ID_Etudiant)
            JOIN Cours C ON (I.Matiere_Cours = C.Matiere)
            JOIN Prof Pr ON (C.ID_Prof = Pr.ID)
           WHERE Pr.Bureau = 'D.42';
```

Prenom	Nom
Jean	Dubois
Jean	Martin
Marie	Dubois
Luc	NULL
Marion	NULL

Q6 Pour tout étudiant afficher son prénom et le numéro de portable de son parent. La colonne avec le numéro de portable doit s'appeler Portable de Parent.

```
SELECT Etudiant.Prenom, Parent.Portable AS "Portable de Parent"
  FROM Etudiant LEFT OUTER JOIN Parent ON (Etudiant.ID_Parent=Parent.ID);
```

Prenom	Portable de Parent
Jean	06.14.21.56.34
Jean	06.84.81.96.12
Marie	06.14.21.56.34
Paul	06.41.21.32.14
Luc	NULL
Marion	NULL

Q7 Afficher les paires de nom de professeurs qui partagent le bureau.

```
SELECT P1.Nom, P2.Nom
  FROM Prof AS P1 JOIN Prof AS P2
 WHERE P1.Bureau = P2.Bureau
   AND P1.ID <> P2.ID;
```

Nom	Nom
Sławek	Marc
Marc	Sławek

Q8 Idem mais sans répéter les même deux professeurs

```
SELECT P1.Nom, P2.Nom
  FROM Prof AS P1 JOIN Prof AS P2
 WHERE P1.Bureau = P2.Bureau
   AND P1.ID < P2.ID;
```

Nom	Nom
Sławek	Marc

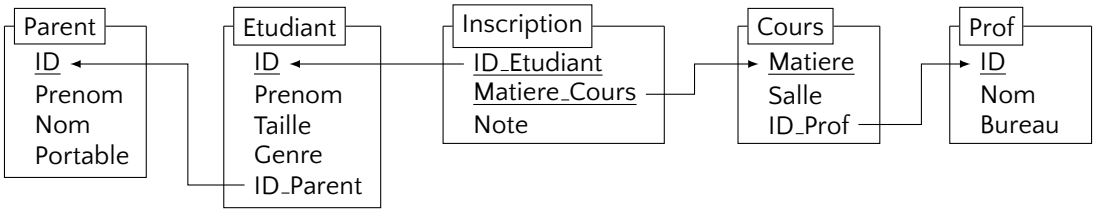
Q9 Afficher les nom complètes des étudiants qui suivent au moins deux cours différents assurés par le même professeur (sans répétition).

```
          SELECT DISTINCT E.Prenom, P.Nom
            FROM Etudiant AS E
LEFT OUTER JOIN Parent AS P ON (E.ID_Parent = P.ID)
            JOIN Inscription AS I1 ON (E.ID = I1.ID_Etudiant)
            JOIN Cours AS C1 ON (I1.Matiere_Cours = C1.Matiere)
            JOIN Inscription AS I2 ON (E.ID = I2.ID_Etudiant)
            JOIN Cours AS C2 ON (I2.Matiere_Cours = C2.Matiere)
           WHERE I1.Matiere_Cours <> I2.Matiere_Cours
             AND C1.ID_Prof = C2.ID_Prof;
```

Prenom	Nom
Jean	Dubois
Marie	Dubois
Luc	NULL

CM 4 : Requêtes de groupage et d'agrégation

La base de données `fac.db` est disponible sur la page du cours. Son schéma est présenté ci-dessous.



Les contenus des tables sont suivants.

Etudiant

ID	Prenom	Taille	Genre	ID_Parent
1	Jean	178	H	1
3	Jean	162	H	3
4	Marie	159	H	1
5	Paul	161	F	2
6	Luc	161	H	NULL
8	Marion	159	F	4

Parent

ID	Prenom	Nom	Portable
1	Bruno	Dubois	06.14.21.56.34
2	Constance	Dupont	06.41.21.32.14
3	Adèle	Martin	06.84.81.96.12

Cours

Matiere	Salle	ID_Prof
SQL	B2.461	11
HTML	A2.061	46
IA	A1.423	11

Prof

ID	Nom	Bureau
11	Sławek	D.42
46	Fabien	C.21
57	Marc	D.42

Inscription

ID_Etudiant	Note	Matiere_Cours
1	12.0	SQL
1	12.0	HTML
1	NULL	IA
3	15.0	SQL
4	16.0	SQL
4	17.0	HTML
4	12.0	IA
6	11.0	SQL
6	NULL	IA
8	16.0	HTML
8	NULL	IA

Q1 Créer une vue qui complète les informations sur les étudiants dans la table `Etudiant` avec le nom de famille (pris de la table `Parent`).

```
CREATE VIEW EtudiantNom AS
SELECT Etudiant.ID, Etudiant.Prenom, Parent.Nom,
       Etudiant.Taille, Etudiant.Genre, Etudiant.ID_Parent
  FROM Etudiant LEFT OUTER JOIN Parent ON (Etudiant.ID_Parent = Parent.ID);
```

EtudiantNom

ID	Prenom	Nom	Taille	Genre	ID_Parent
1	Jean	Dubois	178	H	1
3	Jean	Martin	162	H	3
4	Marie	Dubois	159	H	1
5	Paul	Dupont	161	F	2
6	Luc	NULL	161	H	NULL
8	Marion	NULL	159	F	4

Q2 Afficher la taille moyenne pour tout sexe d'étudiant

```
SELECT Genre, AVG(Taille) FROM Etudiant GROUP BY Genre;
```

Genre	AVG(Taille)
F	160.0
H	165.0

Q3 Afficher la taille moyenne, la taille minimale ainsi que la taille maximale pour tout sexe d'étudiant

```
SELECT Genre, AVG(Taille), MIN(Taille), MAX(Taille) FROM Etudiant GROUP BY Genre;
```

Genre	AVG(Taille)	MIN(Taille)	MAX(Taille)
F	160.0	159	161
H	165.0	159	178

Q4 Afficher la moyenne des notes de tout étudiant

```
SELECT ID_Etudiant, AVG(Note) FROM Inscription GROUP BY ID_Etudiant;
```

ID_Etudiant	AVG(Note)
1	12.0
3	15.0
4	15.0
6	11.0
8	16.0

Q5 Idem + ordonner les résultats par la moyenne (l'ordre décroissant)

```
  SELECT ID_Etudiant, AVG(Note) AS Moyenne
    FROM Inscription
GROUP BY ID_Etudiant
ORDER BY Moyenne DESC;
```

ID_Etudiant	Moyenne
8	16.0
3	15.0
4	15.0
1	12.0
6	11.0

Q6 Pour tout étudiant afficher sa moyenne ainsi que le nombre de cours auquels il est inscrits et le nombre des notes qui sont renseignés.

```
  SELECT ID_Etudiant, AVG(Note), COUNT(*) AS Inscriptions, COUNT(Note) AS Notes
    FROM Inscription
GROUP BY ID_Etudiant;
```

ID_Etudiant	AVG(Note)	Inscriptions	Notes
1	12.0	3	2
3	15.0	1	1
4	15.0	3	3
6	11.0	2	1
8	16.0	2	1

Q7 Afficher les moyennes des étudiants dont toutes les notes sont renseignés.

```
  SELECT ID_Etudiant, AVG(Note)
    FROM Inscription
GROUP BY ID_Etudiant
  HAVING COUNT(*) = COUNT(Note);
```

ID_Etudiant	AVG(Note)
3	15.0
4	15.0

Q8 Pour tout étudiant afficher son prénom, son nom de famille (s'il est renseigné) et sa moyenne

```
   SELECT Etudiant.Prenom, Parent.Nom, AVG(Note) AS Moyenne
    FROM Etudiant LEFT OUTER JOIN Parent ON (Etudiant.ID_Parent = Parent.ID)
    JOIN Inscription ON (Etudiant.ID = Inscription.ID_Etudiant)
GROUP BY Etudiant.Prenom, Parent.Nom;
```

ou plus simplement en utilisant la vue `EtudiantNom`

```
  SELECT EtudiantNom.Prenom, EtudiantNom.Nom, AVG(Note) AS Moyenne
    FROM EtudiantNom
    JOIN Inscription ON (EtudiantNom.ID = Inscription.ID_Etudiant)
GROUP BY EtudiantNom.Prenom, EtudiantNom.Nom
ORDER BY Moyenne DESC;
```

Prenom	Nom	Moyenne
Marion	NULL	16.0
Jean	Martin	15.0
Marie	Dubois	15.0
Jean	Dubois	12.0
Luc	NULL	11.0

Q9 Calculer la moyenne de tout cours

```
SELECT Matiere_Cours, AVG(Note) FROM Inscription GROUP BY Matiere_Cours;
```

Matiere_Cours	AVG(Note)
HTML	15.0
IA	12.0
SQL	13.5

Q10 Calculer la statistique qui compare les moyennes des étudiants et des étudiantes pour tous les cours.

```
  SELECT Matiere_Cours, Genre, AVG(Note)
    FROM Inscription
    JOIN Etudiant ON (Etudiant.ID = Inscription.ID_Etudiant)
GROUP BY Matiere_Cours, Genre
ORDER BY Matiere_Cours, Genre;
```

Matiere_Cours	Genre	AVG(Note)
HTML	F	16.0
HTML	H	14.5
IA	F	NULL
IA	H	12.0
SQL	H	13.5

Q11 Pour tout professeur compter le nombre des cours qu'il assure.

```
  SELECT Prof.Nom, COUNT(Cours.Matiere) AS "Cours Enseignes"
    FROM Prof LEFT OUTER JOIN Cours ON (Prof.ID = Cours.ID_Prof)
GROUP BY Nom;
```

Nom	Cours Enseignes
Fabien	1
Marc	0
Sławek	2

Q12 Pour tout professeur calculer le nombre d'étudiant qu'il encadre lors de ces cours (si un professeur encadre le même étudiant aux plusieurs cours, cet étudiant doit être compté uniquement une fois).

```
SELECT Prof.NOM, COUNT(S.ID_Etudiant) AS "Etudiants Encadres"
FROM Prof
LEFT OUTER JOIN (
   SELECT DISTINCT Cours.ID_Prof, Inscription.ID_Etudiant
   FROM Cours
   JOIN Inscription ON (Cours.Matiere = Inscription.Matiere_Cours)
) AS S ON (Prof.ID = S.ID_Prof)
GROUP BY Prof.Nom;
```

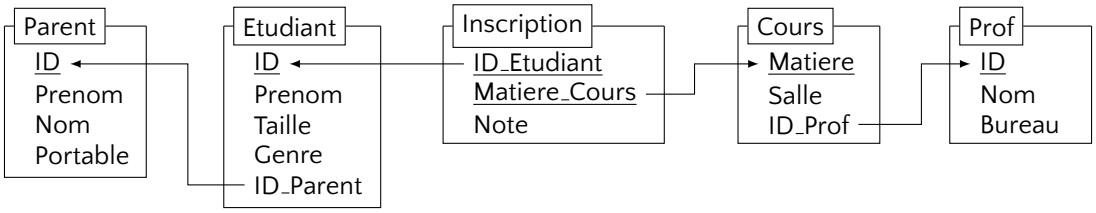
ou sans sous-expressions

```
SELECT Prof.NOM, COUNT(DISTINCT Inscription.ID_Etudiant) AS "Etudiants Encadres"
FROM Prof
LEFT OUTER JOIN Cours ON (Prof.ID = Cours.ID_Prof)
LEFT OUTER JOIN Inscription ON (Cours.Matiere = Inscription.Matiere_Cours)
GROUP BY Prof.Nom;
```

Nom	Etudiants Encadres
Fabien	3
Marc	0
Sławek	5

CM 5 : Requêtes de manipulation de données

La base de données `fac.db` est disponible sur la page du cours. Son schéma est présenté ci-dessous.



Les contenus initiales des tables sont suivants.

Etudiant

ID	Prenom	Taille	Genre	ID_Parent
1	Jean	178	H	1
3	Jean	162	H	3
4	Marie	159	H	1
5	Paul	161	F	2
6	Luc	161	H	NULL
8	Marion	159	F	4

Parent

ID	Prenom	Nom	Portable
1	Bruno	Dubois	06.14.21.56.34
2	Constance	Dupont	06.41.21.32.14
3	Adèle	Martin	06.84.81.96.12

Cours

Matiere	Salle	ID_Prof
SQL	B2.461	11
HTML	A2.061	46
IA	A1.423	11

Prof

ID	Nom	Bureau
11	Sławek	D.42
46	Fabien	C.21
57	Marc	D.42

Inscription

ID_Etudiant	Note	Matiere_Cours
1	12.0	SQL
1	12.0	HTML
1	NULL	IA
3	15.0	SQL
4	16.0	SQL
4	17.0	HTML
4	12.0	IA
6	11.0	SQL
6	NULL	IA
8	16.0	HTML
8	NULL	IA

Q1 Dans la table Parent ajouter le parent de Marion. Son nom est Fabrice Degassi et son numéro de portable est *06.43.92.36.93*. Observation : dans la table Etudiant la ligne de Marion utilise l'identifiant de parent *4*. Par conséquent, c'est aussi l'identifiant à utiliser pour Fabrice dans la table Parent.

```
INSERT INTO Parent VALUES (4,'Fabrice','Degassi','06.43.92.36.93');
```

Parent

ID	Prenom	Nom	Portable
1	Bruno	Dubois	06.14.21.56.34
2	Constance	Dupont	06.41.21.32.14
3	Adèle	Martin	06.84.81.96.12
4	Fabrice	Degassi	06.43.92.36.93

Q2 Inscrire Jean Martin (ID 3) au cours de HTML.

```
INSERT INTO Inscription(ID_Etudiant,Matiere_Cours) VALUES (3,'HTML');
```

Inscription

ID_Etudiant	Note	Matiere_Cours
1	12.0	SQL
1	12.0	HTML
1	NULL	IA
3	15.0	SQL
4	16.0	SQL
4	17.0	HTML
4	12.0	IA
6	11.0	SQL
6	NULL	IA
8	16.0	HTML
8	NULL	IA
3	NULL	HTML

Q3 Inscrire Paul Dupont (ID 5) à tous les cours

```
INSERT INTO Inscription(ID_Etudiant,Matiere_Cours)
SELECT 5, Matiere FROM Cours;
```

Inscription

ID_Etudiant	Note	Matiere_Cours
1	12.0	SQL
1	12.0	HTML
1	NULL	IA
3	15.0	SQL
4	16.0	SQL
4	17.0	HTML
4	12.0	IA
6	11.0	SQL
6	NULL	IA
8	16.0	HTML
8	NULL	IA
3	NULL	HTML
5	NULL	HTML
5	NULL	IA
5	NULL	SQL

Q4 Ajouter une nouvelle information sur Luc (ID 6) : son parent est Adèle Martin (ID 3).

```
UPDATE Etudiant
   SET ID_Parent = 3
 WHERE ID = 6;
```

Etudiant

ID	Prenom	Taille	Genre	ID_Parent
1	Jean	178	H	1
3	Jean	162	H	3
4	Marie	159	H	1
5	Paul	161	F	2
6	Luc	161	H	3
8	Marion	159	F	4

Q5 Augmenter de 2 toute note du cours HTML.

```
UPDATE Inscription
   SET Note = Note + 2
 WHERE Matiere_Cours = 'HTML';
```

Inscription

ID_Etudiant	Note	Matiere_Cours
1	12.0	SQL
1	14.0	HTML
1	NULL	IA
3	15.0	SQL
4	16.0	SQL
4	19.0	HTML
4	12.0	IA
6	11.0	SQL
6	NULL	IA
8	18.0	HTML
8	NULL	IA
3	NULL	HTML
5	NULL	HTML
5	NULL	IA
5	NULL	SQL

Q6 Augmenter de 1 toute note de tout cours enseigné par Sławek (ID 11).

```
UPDATE Inscription
   SET Note = Note + 1
 WHERE Matiere_Cours IN (SELECT Matiere FROM Cours WHERE ID_Prof = 11);
```

Inscription

ID_Etudiant	Note	Matiere_Cours
1	13.0	SQL
1	14.0	HTML
1	NULL	IA
3	16.0	SQL
4	17.0	SQL
4	19.0	HTML
4	13.0	IA
6	12.0	SQL
6	NULL	IA
8	18.0	HTML
8	NULL	IA
3	NULL	HTML
5	NULL	HTML
5	NULL	IA
5	NULL	SQL

Q7 Ajouter une colonne textuelle *Nom* à la table Etudiant.

```
ALTER TABLE Etudiant
ADD COLUMN Nom TEXT;
```

Etudiant

ID	Prenom	Taille	Genre	ID_Parent	Nom
1	Jean	178	H	1	NULL
3	Jean	162	H	3	NULL
4	Marie	159	H	1	NULL
5	Paul	161	F	2	NULL
6	Luc	161	H	3	NULL
8	Marion	159	F	4	NULL

Q8 Transmettre les noms de famille de la table Parent à la table Etudiant.

```
UPDATE Etudiant
   SET Nom = Parent.Nom
  FROM Parent
 WHERE Parent.ID = Etudiant.ID_Parent;
```

Etudiant

ID	Prenom	Taille	Genre	ID_Parent	Nom
1	Jean	178	H	1	Dubois
3	Jean	162	H	3	Martin
4	Marie	159	H	1	Dubois
5	Paul	161	F	2	Dupont
6	Luc	161	H	3	Martin
8	Marion	159	F	4	Degassi

Q9 Supprimer Marc de la table Prof.

```
DELETE FROM Prof
      WHERE Nom = 'Marc';
```

Prof

ID	Nom	Bureau
11	Sławek	D.42
46	Fabien	C.21

Q10 Supprimer toutes les inscriptions aux cours enseignés par Fabien.

```
DELETE FROM Inscription
      WHERE Matiere_Cours IN (
         SELECT Cours.Matiere
         FROM Cours
         JOIN Prof ON Cours.ID_Prof = Prof.ID
         WHERE Prof.Nom = 'Fabien'
);
```

Inscription

ID_Etudiant	Note	Matiere_Cours
1	13.0	SQL
1	NULL	IA
3	16.0	SQL
4	17.0	SQL
4	13.0	IA
6	12.0	SQL
6	NULL	IA
8	NULL	IA
5	NULL	IA
5	NULL	SQL