Jour 1 :

Su – (pour se connecter au root, mdp : NetUser27)

Systemectl status mysql

Mysql -u root -p

CREATE USER ‘…’@’localhost’ IDENTIFIED BY ‘…’; (pour se connecter au server local) (NetUser27) (etudiant)

GRANT ALL PRIVILEGES ON \*.\* TO ‘etudiant’@’localhost’ WITH GRANT OPTION;

Flush privileges;

Mysql -u etudiant -p

CREATE DATABASE …; (gestion\_etudiants)

Show databases;

Use gestion\_etudiants; (permet d’entrer dans la base de données gestion\_etudiants);

CREATE TABLE … ( (etudiants)

* … … AUTO\_INCREMENT PRIMARY KEY (pour que à chaque lignes, le numéro augmente de 1) (ID\_etudiant INT)
* Nom VARCHAR(30) NOT NULL,
* Prenom VARCHAR(30) NOT NULL,
* Date\_naissance DATE,
* Email VARCHAR(50) unique (pas de virgule si c’est terminer)
* ) ;

SHOW TABLES;

DESCRIBE etudiants; (affiche en détails les données de etudiants)

Select \* from etudiants;

Jour 2 :

INSERT INTO etudiants (nom, prenom, date\_naissance, email)

-VALUES

-(‘FLAN’, ‘Mathis’, ‘2006-11-10’, ‘toto@toto.fr’) ; (ceci sont un exemple de données à entrer et qui vont être mises dans le tableau)

SELECT \* from étudiants ; (sert afficher le tableau) (si on remplace l’étoile par le nom d’une variable, cela va afficher uniquement la variables choisis)

Une image contenant texte, reçu, Police, ligne

Description générée automatiquement

DELETE FROM etudiants ;

-WHERE id\_etudiant = 2 ; (efface la données 2 dans la colonne id\_etudiant)

UPDATE etudiants

-SET email = ‘toto@ipi.fr’

-WHERE id\_etudiant = 1; (renome le mail ou l’id\_etudiant est égale à 1)

ALTER TABLE etuduants

-RENAME COLUMN email to mail; (renome le titre d’une colonne)

ALTER TABLES etudiants

-ADD ville varchar(30); (créer une nouvelle colonne dans etudiants)

Describe etudiants; (affiche en détails les données de etudiants)

Une image contenant texte, capture d’écran, Police, nombre

Description générée automatiquement

ALTER TABLE etudiants ADD ville\_naissance varchar(30) AFTER prenom; (ajoute la colonne ville\_naissance après la colonne prénom)

ALTER TABLES etudiants

-DROP COLUMN ville; (supprime la colonne ville)

Exercices :

DATABASE : cours.

Une image contenant texte, reçu, Police, capture d’écran

Description générée automatiquement3a)

Une image contenant texte, reçu, capture d’écran, Police

Description générée automatiquement3b)

3c) Une image contenant texte, reçu, capture d’écran, Police

Description générée automatiquement

Une image contenant texte, Police, reçu, blanc

Description générée automatiquement3d)

Une image contenant texte, Police, reçu, capture d’écran

Description générée automatiquement3e)

Une image contenant texte, Police, reçu, blanc

Description générée automatiquement3f)

Une image contenant texte, reçu, Police, capture d’écran

Description générée automatiquement3g)

Une image contenant texte, reçu, Police, capture d’écran

Description générée automatiquement3h)

Jour 3 :

(en root) apt update (vérifie combien de paquets peuvent être mis à jour)

Apt upgrade (met à jour les paquet)

Apt install phpmyadmin (NetUser27)

Sur firefox (<http://127.0.0.1/phpmyadmin/> (utilisateur : etudiant ; mdp : NetUser27)

Si nécessaire (

MYSQL -u root -p

ALTER USER ‘root’@’localhost’ IDENTIFIED BY ‘NetUser27’;

) cela sert à changer de mot de passe dans la base de donnée ‘root’

Exercices :

CREATE TABLE inscriptions (

id\_inscription INT AUTO\_INCREMENT PRIMARY KEY,

id\_etudiant INT,

id\_cours INT,

date\_inscription DATE,

FOREIGN KEY (id\_etudiant) REFERENCES etudiants(id\_etudiant),

FOREIGN KEY (id\_cours) REFERENCES cours(id\_cours)

);

Dans “insérer” ont choisi les valeurs

a) Quels cours suit un étudiant

SELECT e.nom, c.nom\_cours

FROM etudiants e (où va-t-on les chercher)

JOIN inscription i ON e.id\_etudiant = i.id\_etudiant

JOIN cours c ON i.id\_cours = c.id\_cours ;

WHERE e.nom = ‘FLAN’; (on définit un exemple)

Une image contenant texte, Police, capture d’écran, blanc

Description générée automatiquement

b) Quels étudiants sont inscris à un cours ?

Dans la base de donnée gestion\_etudiants

SELECT c.nom\_cours, e.nom, e. prenom

FROM etudiants e

JOIN inscription i ON e.id\_etudiant = i.id\_inscription

JOIN cours c ON i.id\_cours = c.id\_cours

WHERE c.nom\_cours = ‘informatique‘;

Une image contenant texte, Police, capture d’écran, blanc

Description générée automatiquement

c)

d) Quand un étudiant s’est t’il inscrit en cours ?

SELECT e.nom, e.prenom, i.date\_inscription, c.nom\_cours

FROM etudiants e

JOIN inscription i ON e.id\_etudiant = i.id\_inscription

JOIN cours c ON i.id\_cours = c.id\_cours

WHERE c.nom\_cours = ‘informatique‘;

Une image contenant texte, Police, capture d’écran, blanc

Description générée automatiquement

e)Quels étudiants ne sont inscris à aucun cours ?

SELECT e.nom, e. prenom

FROM etudiants e

LEFT JOIN inscription i ON e.id\_etudiant = i.id\_etudiant

WHERE i.id\_cours IS NULL;

f)

SELECT c.nom\_cours, COUNT(i.id\_idetudiant) AS nombre\_etudiants

FROM cours c

LEFT JOIN inscription i ON c.id\_cours = i.id\_cours

GROUP BY c.nom\_cours;

Jour 4 :

Création de la table étudiant :

CREATE TABLE professeur (

Id\_professeur INT AUTO\_INCREMENT PRIMARY KEY,

Nom VARCHAR(30) NOT NULL,

Prenom VARCHAR(30) NOT NULL,

);

Importation de la clef étrangère id\_professeur dans cours :

ALTER TABLE cours

ADD COLUMN id\_professeur INT;

ALTER TABLE cours

ADD FOREIGN KEY (id\_professeur) REFERENCES professeur(id\_professeur);

Faire en sorte qu’un professeur soit attribuer à un cours

UPDATE cours

SET id\_professeur = 1  
WHERE nom\_cours = ‘mathématiques’ ;

-Lister les étudiants avec leurs cours et les professeurs associés :

(les étudiants sint inscrits à des cours dans la table inscription (etudiant > inscription) / sur des enregistrement, on va cherché le cours en question (inscription > cours) / les professeurs sont assoicier à des cours (cours > professeur))

SELECT e.nom AS nom\_etudiant (pour éviter de le confondre avec le nom du prof), e.prenom AS prenom\_etudiant, c.nom\_cours, p.nom AS nom\_prof, p.prenom AS prenom\_prof

FROM etudiants e

JOIN inscription i ON e.id\_etudiant = i.id\_inscription

JOIN cours c ON i.id\_cours = c.id\_cours

JOIN professeur p ON c.id\_professeur = p.id\_professeur

-Lister les professeurs qui n’ont pas encore de cours attribué :

SELECT p.nom AS nom\_prof, p.prenom AS prenom\_prof, c.nom\_cours

FROM professeur p

LEFT JOIN cours c ON p.id\_professeur = c.id\_professeur

WHERE c.id\_cours IS NULL;

-Compter le nombre d’étudiants inscrits par professeur :

prenomJOIN cours c ON p.id\_professeur = c.id\_professeur

JOIN inscription i ON c.id\_cours = i.id\_cours

GROUP BY p.nom

Jour 5 :

CREATE TABLE salles (

id\_salle INT AUTO\_INCREMENT PRIMARY KEY,

nom VARCHAR(30) NOT NULL,

capacité INT NOT NULL

);

ALTER TABLE cours

ADD COLUMN id\_salle INT;

ALTER TABLE cours

ADD FOREIGN KEY (id\_salle) REFERENCES salles(id\_salle);

UPDATE cours

SET id\_salle = 1  
WHERE nom\_cours = ‘Iinformatique’ ;

UPDATE cours

SET id\_salle = 2  
WHERE nom\_cours = ‘Anglais’ ;

UPDATE cours

SET id\_salle = 3  
WHERE nom\_cours = ‘Physique’ ;

UPDATE cours

SET id\_salle = 4   
WHERE nom\_cours = ‘Mathématiques’ ;

Tous les cours avec leurs salles atribuer :

SELECT c.nom\_cours, s.id\_salle, s.capacité

FROM salles s

JOIN cours c ON s.id\_salle = c.id\_salle

Trouver les salles non utilisées :

SELECT s.id\_salle, s.nom

FROM salles s

LEFT JOIN cours c ON s.id\_salle = c.id\_salle

WHERE c.id\_salle IS NULL;

Lister les prof et la salle où ils enseignent :

SELECT p.nom, s.nom

FROM salles s

JOIN cours c ON s.id\_salle = c.id\_salle

JOIN professeur p ON c.id\_professeur = p.id\_professeur

BONUS : Les salles avec le nombre de cours qui s’y déroulent :

SELECT s.nom, COUNT(c.id\_cours) AS nombre\_cours

FROM salles s

JOIN cours c ON s.id\_salle = c.id\_salle

GROUP BY s.nom ;