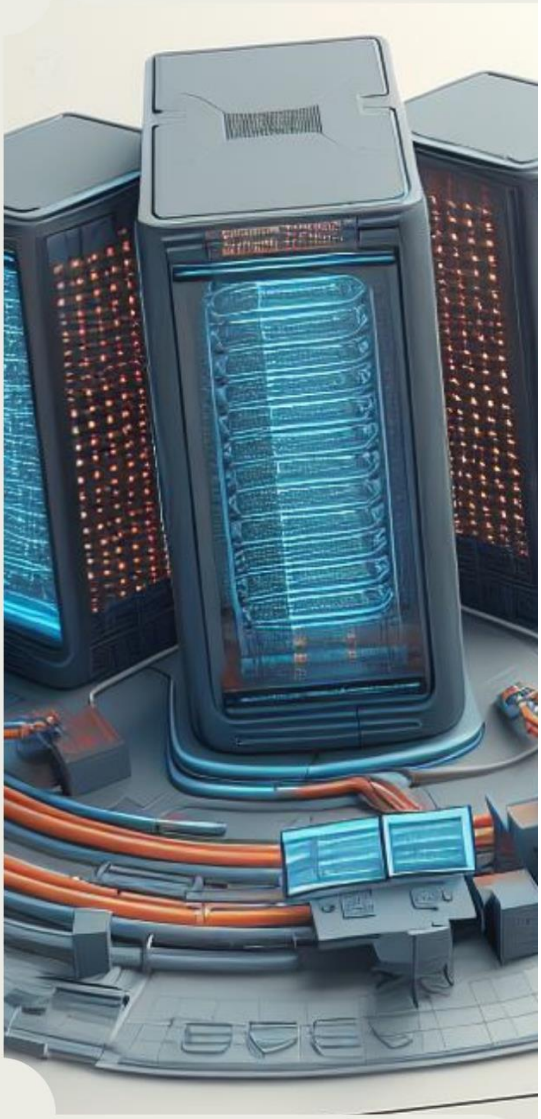
CONCEPTION ET CRÉATION D'UNE BASE DE DONNÉES



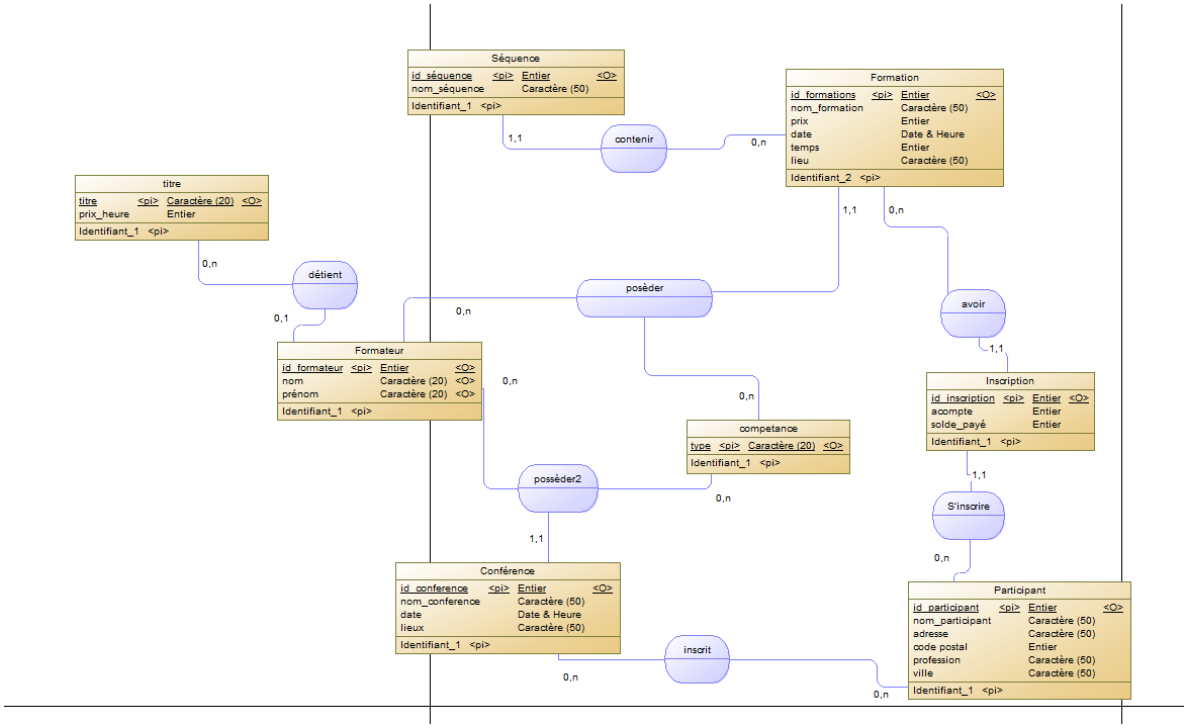
Vellozzi Mathieu
El Habachi Ilyas

2023 / 2024

Sommaire :

I) Modèle Conceptuelle des Données (MCD)

Tout d'abord, pour réaliser ce projet, nous avons dû créer un Modèle Conceptuel de Données. Cela nous a permis d'avoir une idée générale de la structure de la base de données.



Nous avons opté pour la création de deux relations ternaires : l'une impliquant les tables "Formateur", "Conférence" et "Compétence", et l'autre impliquant les tables "Formateur", "Formation" et "Compétence". Cette structure a été choisie en raison de la dépendance entre le sujet d'une formation et la compétence requise pour enseigner cette formation. Ainsi, un formateur doit posséder la compétence associée à une formation donnée pour pouvoir dispenser cette formation.

Le même concept s'applique à la relation entre les tables "Formateur", "Conférence" et "Compétence".

Les seules différences entre une formation et une conférence sont le fait qu'une formation est payante contrairement à la conférence, et qu'une formation peut être divisée en séquences.

Dans ces conférences et formations, on peut retrouver des participants. Ces derniers peuvent participer à plusieurs conférences et formations en même temps. Pour tenir compte de leurs inscriptions, nous avons choisi de créer une table "Inscription".

Chaque formateur possède un titre. Cette information nous sert à calculer son indemnité à la fin de chaque mois.

II) SQL

a. Création des tables

Ensuite, une fois que nous avons terminé la création du modèle conceptuel de données, nous sommes passés à la réalisation du modèle physique de données en utilisant Oracle SQL. Grâce à une fonctionnalité de PowerMC, l'outil que nous avons utilisé pour créer le MCD, nous avons pu générer les requêtes nécessaires pour la création, la suppression et la modification des tables.

b. Insertion des données

Avec l'objectif final de tester notre base de données, nous avons décidé de remplir nos tables avec des données aléatoires. Nous avons essayé de recréer tous les cas possibles pour nous rapprocher le plus possible des différentes situations et des cas de la vie réelle.

Pour remplir les tables, nous avons dû suivre un ordre bien précis en raison des différents cas de dépendance entre les tables.

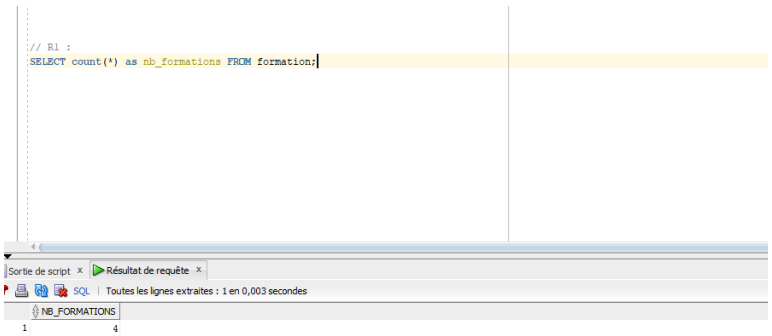
Exemple :

Pour remplir la table “POSSEDER”, la table qui définit pour chaque formation son formateur, nous avons dû avant créer les différents formateurs et les différentes formations. Et pour créer les différents formateurs nous avons dû créer avant les différents titres.

c. Requêtes

Afin de vérifier notre base de données, nous avons répondu à huit requêtes.

- Requête 1 : Afficher toutes les formations de notre base de données



- Requête 2 : Afficher toutes les formations de 2023

```
// R2 :
SELECT COUNT(*) AS nb_formation_en_2023 FROM formation WHERE EXTRACT(YEAR FROM date_formation) = 2023;
```

Sortie de script | Résultat de requête

SQL | Toutes les lignes extraites : 1 en 14,32 secondes

	NB_FORMATION_EN_2023
1	2

- Requête 3 : Afficher les participant de la conférence du 15/10/2023

```
// R3 :
SELECT * FROM participant
    INNER JOIN inscrit ON participant.id_participant = inscrit.id_participant
    INNER JOIN conference ON inscrit.id_conference = conference.id_conference
    WHERE date_conference = '15/10/2023';
```

Sortie de script | Résultat de requête

SQL | Toutes les lignes extraites : 0 en 0,017 secondes

ID_PARTI...	NOM_PAR...	ADRESSE	CODE_PO...	PROFESSI...	VILLE	ID_CONF...	ID_PARTI...	ID_CONF...	NOM_CO...	DATE_CO...	LIEUX

- Requête 3 : A la date précédente il n'y a eu aucune conférence donc nous avons changé la data pour faire si d'afficher une conférence

```
// R3.2 :
SELECT * FROM participant
    INNER JOIN inscrit ON participant.id_participant = inscrit.id_participant
    INNER JOIN conference ON inscrit.id_conference = conference.id_conference
    WHERE date_conference = '12/01/2023';
```

Sortie de script | Résultat de requête

SQL | Toutes les lignes extraites : 3 en 0,005 secondes

	ID_PARTICIPANT	NOM_PARTICIPANT	ADRESSE	CODE_POSTAL	PROFESSION	VILLE
1	205	Mathieu Lambert	1212 Avenue de la Mer	33000	Avocat	Bordeaux
2	206	Emma Rousseau	1313 Boulevard de la Plage	67000	Journaliste	Strasbourg
3	207	Hugo Girard	1414 Rue du Mont	69000	Médecin	Lyon

- Requête 4 : Les participant de la formation N° qui n'ont pas encore paye l'intégralité de la formation

```
// R4 :
SELECT participant.id_participant, participant.nom_participant, inscription.id_formations, formation.nom_formation, formation.prix-(inscription.solde_paye + inscription.acompte) AS reste_a_payer
    FROM participant
    INNER JOIN inscription ON participant.id_participant = inscription.id_participant
    INNER JOIN formation ON inscription.id_formations = formation.id_formations
    WHERE formation.id_formations = 101 AND (inscription.solde_paye + inscription.acompte) < formation.prix;
```

Sortie de script | Résultat de requête

SQL | Toutes les lignes extraites : 3 en 0,008 secondes

	ID_PARTICIPANT	NOM_PARTICIPANT	ID_FORMATIONS	NOM_FORMATION	RESTE_A_PAYER
1	201	Thomas Martin	101	Rééducation neurologique	12
2	202	Sophie Lefevre	101	Rééducation neurologique	46
3	203	Alexandre Dubois	101	Rééducation neurologique	22

Même dans ce cas nous avons changé un peu la consigne pour faire si qu'on est au moins un participent

- Requête 5 : Afficher les 3 premiers codes postaux où il y a le plus de participants toute formations confondues.

```
// R5 :
SELECT code_postal, COUNT(*) AS nb_personnes FROM participant
GROUP BY code_postal
HAVING COUNT(*) = (SELECT MAX(COUNT(*)) FROM participant GROUP BY code_postal)
    OR COUNT(*) = ( SELECT MAX(COUNT(*)) FROM participant GROUP BY code_postal HAVING COUNT(*) != (SELECT MAX(COUNT(*)) FROM participant GROUP BY code_postal))
    OR COUNT(*) = (SELECT MAX(COUNT(*)) FROM participant GROUP BY code_postal HAVING COUNT(*) != (SELECT MAX(COUNT(*)) FROM participant GROUP BY code_postal HAVING COUNT(*) !=
        (SELECT MAX(COUNT(*)) FROM participant GROUP BY code_postal)) AND COUNT(*) != (SELECT MAX(COUNT(*)) FROM participant GROUP BY code_postal));
```

Sortie de script | Résultat de requête

SQL | Toutes les lignes extraites : 3 en 0,007 secondes

	CODE_POSTAL	NB_PERSONNES
1	75000	5
2	69000	3
3	13000	4

Nous avons découvert juste au moment de la rédaction de ce document qu'elle existe une solution beaucoup plus simple pour écrire cette requête qui utilise une fonctionnalité de la commande GROUP BY.

- Requête 6 : La conférence la plus populaire, celle qui a été attiré le plus de participants.

```
// R6 :
SELECT conference.id_conference, conference.nom_conference, conference.date_conference, COUNT(*) as nb_participants from conference
    INNER JOIN inscrit ON inscrit.id_conference = conference.id_conference
    INNER JOIN participant ON participant.id_participant = inscrit.id_participant
    GROUP BY conference.id_conference, conference.nom_conference, conference.date_conference
    HAVING COUNT(*) = (SELECT MAX(COUNT(*)) from conference
                       INNER JOIN inscrit ON inscrit.id_conference = conference.id_conference
                       INNER JOIN participant ON participant.id_participant = inscrit.id_participant
                       GROUP BY conference.id_conference);
```

Sortie de script | Résultat de requête

SQL | Toutes les lignes extraites : 2 en 5,548 secondes

	ID_CONFERENCE	NOM_CONFERENCE	DATE_CONFERENCE	NB_PARTICIPANTS
1	400	Intelligence émotionnelle	12/01/23	3
2	402	Médecine énergétique	05/01/24	3

- Requête 7 : Afficher le professeur le plus populaire, celui qui a eu le plus de participants à son cours

```
// R7 :
SELECT formateur.id_formateur, formateur.nom, formateur.prenom, COUNT(*) AS nb_fan FROM formateur
    INNER JOIN poseder ON formateur.id_formateur = poseder.id_formateur
    INNER JOIN inscription ON poseder.id_formations = inscription.id_formations
    GROUP BY formateur.id_formateur, formateur.nom, formateur.prenom
    HAVING COUNT(*) = (SELECT MAX(COUNT(*)) FROM formateur
                       INNER JOIN poseder ON formateur.id_formateur = poseder.id_formateur
                       INNER JOIN inscription ON poseder.id_formations = inscription.id_formations
                       GROUP BY formateur.id_formateur);
```

Sortie de script | Résultat de requête

SQL | Toutes les lignes extraites : 1 en 18,234 secondes

	ID_FORMATEUR	NOM	PRENOM	NB_FAN
1	2	Vellozzi	Mathieu	3

- Requête 8 : Afficher la liste des participants qui doivent de l'argent au centre de formation.

```
// R8 :
SELECT participant.id_participant, participant.nom_participant, formation.prix-(inscription.solde_paye + inscription.acompte) AS reste_a_payer FROM participant
    INNER JOIN inscription ON participant.id_participant = inscription.id_participant
    INNER JOIN formation ON formation.id_formations = inscription.id_formations
    WHERE (inscription.solde_paye + inscription.acompte) < formation.prix;
```

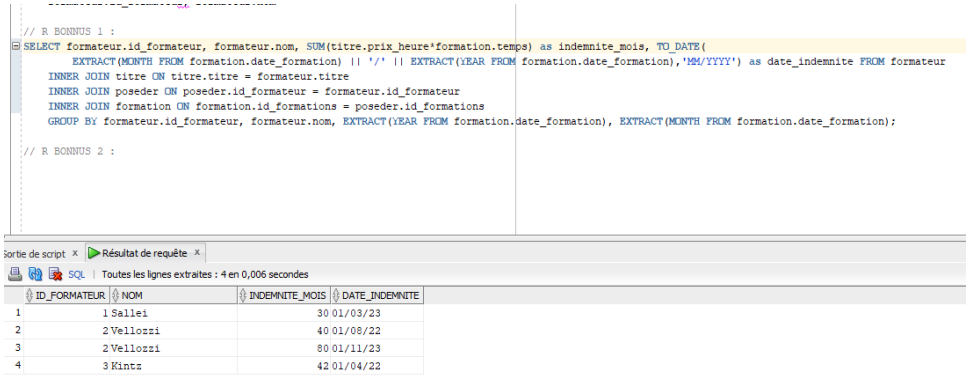
Sortie de script | Résultat de requête

SQL | Toutes les lignes extraites : 4 en 3,961 secondes

	ID_PARTICIPANT	NOM_PARTICIPANT	RESTE_A_PAYER
1	201	Thomas Martin	12
2	202	Sophie Lefevre	46
3	203	Alexandre Dubois	22
4	205	Mathieu Lambert	30

- Requête Bonus :

Nous avons choisi de faire une requête bonus qui affiche les indemnités de chaque mois pour chaque formateur.



```
// R BONNUS 1 :
SELECT formateur.id_formateur, formateur.nom, SUM(titre.prix_heure*formation.temps) as indemnite_mois, TO_DATE(
        EXTRACT(MONTH FROM formation.date_formation) || '/' || EXTRACT(YEAR FROM formation.date_formation),'MM/YYYY') as date_indemnite FROM formateur
    INNER JOIN titre ON titre.titre = formateur.titre
    INNER JOIN poseder ON poseder.id_formateur = formateur.id_formateur
    INNER JOIN formation ON formation.id_formations = poseder.id_formations
    GROUP BY formateur.id_formateur, formateur.nom, EXTRACT(YEAR FROM formation.date_formation), EXTRACT(MONTH FROM formation.date_formation);

// R BONNUS 2 :
```

Sortie de script | Résultat de requête

SQL | Toutes les lignes extraites : 4 en 0,006 secondes

	ID_FORMATEUR	NOM	INDEMNITE_MOIS	DATE_INDEMNITE
1	1	Sallei	30	01/03/23
2	2	Vellozzi	40	01/08/22
3	2	Vellozzi	80	01/11/23
4	3	Kintz	42	01/04/22

Pour faire cela, nous avons récupéré les heures faites par chaque formateur à chaque mois et nous les avons multipliés par son tarif horaire.

III) Annexes

Nous mettons à votre disposition l'intégralité de notre code SQL en annexe, au cas où vous souhaiteriez approfondir les différentes parties de notre projet.

```
alter table FORMATEUR
   drop constraint FK_FORMATEU_DETIENT_TITRE;

alter table INSCRIPTION
   drop constraint FK_INSCRIPT_AVOIR_FORMATIO;

alter table INSCRIPTION
   drop constraint FK_INSCRIPT_S_INSCRIR_PARTICIP;

alter table INSCRIT
   drop constraint FK_INSCRIT_INSCRIT_CONFEREN;

alter table INSCRIT
   drop constraint FK_INSCRIT_INSCRIT2_PARTICIP;

alter table POSEDER
   drop constraint FK_POSEDER_POSEDER_COMPETAN;

alter table POSEDER
   drop constraint FK_POSEDER_POSEDER2_FORMATEU;

alter table POSEDER
   drop constraint FK_POSEDER_POSEDER3_FORMATIO;

alter table POSSEDER2
```

```
   drop constraint FK_POSSEDER_POSSEDER2_COMPETAN;

alter table POSSEDER2
   drop constraint FK_POSSEDER_POSSEDER3_FORMATEU;

alter table POSSEDER2
   drop constraint FK_POSSEDER_POSSEDER4_CONFEREN;

alter table SEQUENCE
   drop constraint FK_SEQUENCE_CONTENIR_FORMATIO;

drop table COMPETANCE cascade constraints;

drop table CONFERENCE cascade constraints;

drop index DETIENT_FK;

drop table FORMATEUR cascade constraints;

drop table FORMATION cascade constraints;

drop index S_INSCRIRE_FK;

drop index AVOIR_FK;

drop table INSCRIPTION cascade constraints;

drop index INSCRIT2_FK;

drop index INSCRIT_FK;
```

```
drop table INSCRIT cascade constraints;

drop table PARTICIPANT cascade constraints;

drop index POSEDER3_FK;

drop index POSEDER2_FK;

drop index POSEDER_FK;

drop table POSEDER cascade constraints;

drop index POSSEDER4_FK;

drop index POSSEDER3_FK;

drop index POSSEDER2_FK;

drop table POSSEDER2 cascade constraints;

drop index CONTENIR_FK;

drop table SEQUENCE cascade constraints;

drop table TITRE cascade constraints;

/* ////////////////////////////////////////////////////////////////////////// */
/*==============================================================*/
/* Table : COMPETANCE                                           */
```

```
/*==============================================================*/
create table COMPETANCE
(
  TYPE              CHAR(20)          not null,
  constraint PK_COMPETANCE primary key (TYPE)
);
/*==============================================================*/
/* ////////////////////////////////////////////////////////////// */

/* ////////////////////////////////////////////////////////////// */
/*==============================================================*/
/* Table : CONFERENCE                                           */
/*==============================================================*/
create table CONFERENCE
(
  ID_CONFERENCE       INTEGER           not null,
  NOM_CONFERENCE      CHAR(50),
  DATE_CONFERENCE           DATE,
  LIEUX             CHAR(50),
  constraint PK_CONFERENCE primary key (ID_CONFERENCE)
);
/*==============================================================*/
/* ////////////////////////////////////////////////////////////// */

/* ////////////////////////////////////////////////////////////// */
/*==============================================================*/
/* Table : FORMATEUR                                            */
/*==============================================================*/
create table FORMATEUR
```

```
(
  ID_FORMATEUR       INTEGER            not null,
  TITRE              CHAR(20),
  NOM                CHAR(20)           not null,
  PRENOM             CHAR(20)           not null,
  constraint PK_FORMATEUR primary key (ID_FORMATEUR)
);
/*==============================================================*/

/*--------------------------------------------------------------*/
/* Index : DETIENT_FK                                           */
/*--------------------------------------------------------------*/
create index DETIENT_FK on FORMATEUR (
  TITRE ASC
);
/*--------------------------------------------------------------*/
/* ////////////////////////////////////////////////////////////// */

/* ////////////////////////////////////////////////////////////// */
/*==============================================================*/
/* Table : FORMATION                                            */
/*==============================================================*/
create table FORMATION
(
  ID_FORMATIONS      INTEGER            not null,
  NOM_FORMATION      CHAR(50),
  PRIX               INTEGER,
  DATE_FORMATION     DATE,
  TEMPS              INTEGER,
```

```
  LIEU              CHAR(50),
  constraint PK_FORMATION primary key (ID_FORMATIONS)
);
/*==============================================================*/
/* ////////////////////////////////////////////////////////////// */

/* ////////////////////////////////////////////////////////////// */
/*==============================================================*/
/* Table : INSCRIPTION                                          */
/*==============================================================*/
create table INSCRIPTION
(
  ID_INSCRIPTION      INTEGER           not null,
  ID_FORMATIONS       INTEGER           not null,
  ID_PARTICIPANT      INTEGER           not null,
  ACOMPTE             INTEGER,
  SOLDE_PAYE          INTEGER,
  constraint PK_INSCRIPTION primary key (ID_INSCRIPTION)
);
/*==============================================================*/

/*--------------------------------------------------------------*/
/* Index : AVOIR_FK                                             */
/*--------------------------------------------------------------*/
create index AVOIR_FK on INSCRIPTION (
  ID_FORMATIONS ASC
);
/*--------------------------------------------------------------*/
```

```
/*------------------------------------------------------------*/
/* Index : S_INSCRIRE_FK                                      */
/*------------------------------------------------------------*/
create index S_INSCRIRE_FK on INSCRIPTION (
  ID_PARTICIPANT ASC
);
/*------------------------------------------------------------*/
/* ////////////////////////////////////////////////////////////////////// */

/* ////////////////////////////////////////////////////////////////////// */
/*============================================================*/
/* Table : INSCRIT                                            */
/*============================================================*/
create table INSCRIT
(
  ID_CONFERENCE       INTEGER            not null,
  ID_PARTICIPANT      INTEGER            not null,
  constraint PK_INSCRIT primary key (ID_CONFERENCE, ID_PARTICIPANT)
);
/*============================================================*/

/*------------------------------------------------------------*/
/* Index : INSCRIT_FK                                         */
/*------------------------------------------------------------*/
create index INSCRIT_FK on INSCRIT (
  ID_CONFERENCE ASC
);
/*------------------------------------------------------------*/
```

```
/*------------------------------------------------------------*/
/* Index : INSCRIT2_FK                                        */
/*------------------------------------------------------------*/
create index INSCRIT2_FK on INSCRIT (
  ID_PARTICIPANT ASC
);
/*------------------------------------------------------------*/
/* ////////////////////////////////////////////////////////////// */

/* ////////////////////////////////////////////////////////////// */
/*============================================================*/
/* Table : PARTICIPANT                                        */
/*============================================================*/
create table PARTICIPANT
(
  ID_PARTICIPANT      INTEGER            not null,
  NOM_PARTICIPANT     CHAR(50),
  ADRESSE             CHAR(50),
  CODE_POSTAL         INTEGER,
  PROFESSION          CHAR(50),
  VILLE               CHAR(50),
  constraint PK_PARTICIPANT primary key (ID_PARTICIPANT)
);
/*============================================================*/
/* ////////////////////////////////////////////////////////////// */

/* ////////////////////////////////////////////////////////////// */
/*============================================================*/
/* Table : POSEDER                                            */
```

```
/*==============================================================*/
create table POSEDER
(
  TYPE              CHAR(20)          not null,
  ID_FORMATEUR      INTEGER           not null,
  ID_FORMATIONS     INTEGER           not null,
  constraint PK_POSEDER primary key (TYPE, ID_FORMATEUR, ID_FORMATIONS)
);
/*==============================================================*/

/*--------------------------------------------------------*/
/* Index : POSEDER_FK                                     */
/*--------------------------------------------------------*/
create index POSEDER_FK on POSEDER (
  TYPE ASC
);
/*--------------------------------------------------------*/

/*--------------------------------------------------------*/
/* Index : POSEDER2_FK                                    */
/*--------------------------------------------------------*/
create index POSEDER2_FK on POSEDER (
  ID_FORMATEUR ASC
);
/*--------------------------------------------------------*/

/*--------------------------------------------------------*/
/* Index : POSEDER3_FK                                    */
/*--------------------------------------------------------*/
```

```
create index POSEDER3_FK on POSEDER (
  ID_FORMATIONS ASC
);
/*------------------------------------------------------------*/
/* ////////////////////////////////////////////////////////////////////// */

/* ////////////////////////////////////////////////////////////////////// */
/*============================================================*/
/* Table : POSSEDER2                                          */
/*============================================================*/
create table POSSEDER2
(
  TYPE              CHAR(20)          not null,
  ID_FORMATEUR      INTEGER           not null,
  ID_CONFERENCE     INTEGER           not null,
  constraint PK_POSSEDER2 primary key (TYPE, ID_FORMATEUR, ID_CONFERENCE)
);
/*============================================================*/

/*------------------------------------------------------------*/
/* Index : POSSEDER2_FK                                       */
/*------------------------------------------------------------*/
create index POSSEDER2_FK on POSSEDER2 (
  TYPE ASC
);
/*------------------------------------------------------------*/

/*------------------------------------------------------------*/
/* Index : POSSEDER3_FK                                       */
```

```
/*------------------------------------------------------------*/
create index POSSEDER3_FK on POSSEDER2 (
  ID_FORMATEUR ASC
);
/*------------------------------------------------------------*/

/*------------------------------------------------------------*/
/* Index : POSSEDER4_FK                                       */
/*------------------------------------------------------------*/
create index POSSEDER4_FK on POSSEDER2 (
  ID_CONFERENCE ASC
);
/*------------------------------------------------------------*/
/* ////////////////////////////////////////////////////////////////////// */

/* ////////////////////////////////////////////////////////////////////// */
/*============================================================*/
/* Table : SEQUENCE                                           */
/*============================================================*/
create table SEQUENCE
(
  ID_SEQUENCE         INTEGER             not null,
  ID_FORMATIONS       INTEGER             not null,
  NOM_SEQUENCE        CHAR(50),
  constraint PK_SEQUENCE primary key (ID_SEQUENCE)
);
/*============================================================*/

/*------------------------------------------------------------*/
```

```
/* Index : CONTENIR_FK                                */
/*----------------------------------------------------------*/
create index CONTENIR_FK on SEQUENCE (
  ID_FORMATIONS ASC
);
/*----------------------------------------------------------*/
/* ////////////////////////////////////////////////////////////////////////// */

/* ////////////////////////////////////////////////////////////////////////// */
/*==========================================================*/
/* Table : TITRE                                  */
/*==========================================================*/
create table TITRE
(
  TITRE             CHAR(20)          not null,
  PRIX_HEURE        INTEGER,
  constraint PK_TITRE primary key (TITRE)
);
/*==========================================================*/
/* ////////////////////////////////////////////////////////////////////////// */

alter table FORMATEUR
  add constraint FK_FORMATEU_DETIENT_TITRE foreign key (TITRE)
    references TITRE (TITRE);

alter table INSCRIPTION
  add constraint FK_INSCRIPT_AVOIR_FORMATIO foreign key (ID_FORMATIONS)
    references FORMATION (ID_FORMATIONS);
```

```
alter table INSCRIPTION
  add constraint FK_INSCRIPT_S_INSCRIR_PARTICIP foreign key (ID_PARTICIPANT)
    references PARTICIPANT (ID_PARTICIPANT);

alter table INSCRIT
  add constraint FK_INSCRIT_INSCRIT_CONFEREN foreign key (ID_CONFERENCE)
    references CONFERENCE (ID_CONFERENCE);

alter table INSCRIT
  add constraint FK_INSCRIT_INSCRIT2_PARTICIP foreign key (ID_PARTICIPANT)
    references PARTICIPANT (ID_PARTICIPANT);

alter table POSEDER
  add constraint FK_POSEDER_POSEDER_COMPETAN foreign key (TYPE)
    references COMPETANCE (TYPE);

alter table POSEDER
  add constraint FK_POSEDER_POSEDER2_FORMATEU foreign key (ID_FORMATEUR)
    references FORMATEUR (ID_FORMATEUR);

alter table POSEDER
  add constraint FK_POSEDER_POSEDER3_FORMATIO foreign key (ID_FORMATIONS)
    references FORMATION (ID_FORMATIONS);

alter table POSSEDER2
  add constraint FK_POSSEDER_POSSEDER2_COMPETAN foreign key (TYPE)
    references COMPETANCE (TYPE);

alter table POSSEDER2
  add constraint FK_POSSEDER_POSSEDER3_FORMATEU foreign key (ID_FORMATEUR)
```

```
    references FORMATEUR (ID_FORMATEUR);

alter table POSSEDER2
  add constraint FK_POSSEDER_POSSEDER4_CONFEREN foreign key (ID_CONFERENCE)
    references CONFERENCE (ID_CONFERENCE);

alter table SEQUENCE
  add constraint FK_SEQUENCE_CONTENIR_FORMATIO foreign key (ID_FORMATIONS)
    references FORMATION (ID_FORMATIONS);
commit;


------------------------------ INSERTIONS DONNEES ------------------------------

/*----------------table Titre----------------*/
INSERT INTO titre VALUES ('master',20);
INSERT INTO titre VALUES ('docteur',30);
INSERT INTO titre VALUES ('etudiant',7);
INSERT INTO titre VALUES ('BUT',13);
/*-----------------------------------------*/

/*----------------Competance----------------*/
INSERT INTO competance VALUES ('psychologue');
INSERT INTO competance VALUES ('physique');
INSERT INTO competance VALUES ('familial');
INSERT INTO competance VALUES ('holistique');
/*-----------------------------------------*/

/*----------------Formateur----------------*/
INSERT INTO formateur VALUES (1,'docteur','Sallei','Jean-Pierre');
```

```
INSERT INTO formateur VALUES (2,'master','Vellozzi','Mathieu');

INSERT INTO formateur VALUES (3,'etudiant','Kintz','Dylan');

INSERT INTO formateur VALUES (4,'etudiant','Silveira Ramos','olivier');

/*----------------------------------------*/


/*----------------Formation----------------*/

INSERT INTO formation VALUES (100, 'Gestion de la douleur', 145, '12/03/2023', 1,
'amphithéâtre1 Lyon');

INSERT INTO formation VALUES (101, 'Rééducation neurologique', 112, '05/08/2022', 2,
'amphithéâtre2 Lyon');

INSERT INTO formation VALUES (102, 'Communication efficace', 175, '18/11/2023', 4,
'amphithéâtre1 Lyon');

INSERT INTO formation VALUES (103, 'Nutrition holistique', 130, '22/04/2022', 6,
'amphithéâtre2 Lyon');

/*----------------------------------------*/


/*----------------Participant----------------*/

INSERT INTO participant VALUES (200, 'Alice Dupont', '123 Rue de la République', '75000',
'Ingénieur', 'Paris');

INSERT INTO participant VALUES (201, 'Thomas Martin', '456 Avenue des Fleurs', '69000',
'Professeur', 'Lyon');

INSERT INTO participant VALUES (202, 'Sophie Lefevre', '789 Rue de la Libération', '13000',
'Infirmière', 'Marseille');

INSERT INTO participant VALUES (203, 'Alexandre Dubois', '1010 Boulevard du Soleil',
'31000', 'Artisan', 'Toulouse');

INSERT INTO participant VALUES (204, 'Camille Leroux', '1111 Rue des Étoiles', '44000',
'Étudiant', 'Nantes');

INSERT INTO participant VALUES (205, 'Mathieu Lambert', '1212 Avenue de la Mer', '33000',
'Avocat', 'Bordeaux');

INSERT INTO participant VALUES (206, 'Emma Rousseau', '1313 Boulevard de la Plage',
'67000', 'Journaliste', 'Strasbourg');

INSERT INTO participant VALUES (207, 'Hugo Girard', '1414 Rue du Mont', '69000',
'Médecin', 'Lyon');
```

```
INSERT INTO participant VALUES (208, 'Chloé Moreau', '1515 Avenue des Champs', '75000', 'Architecte', 'Paris');

INSERT INTO participant VALUES (209, 'Lucas Perrin', '1616 Rue de la Forêt', '13000', 'Pompier', 'Marseille');

INSERT INTO participant VALUES (210, 'Léa Lemoine', '1717 Boulevard des Collines', '33000', 'Enseignant', 'Bordeaux');

INSERT INTO participant VALUES (211, 'Louis Martin', '1818 Rue des Vignes', '59000', 'Pharmacien', 'Lille');

INSERT INTO participant VALUES (212, 'Elise Leroy', '1919 Avenue de la Cascade', '69000', 'Psychologue', 'Lyon');

INSERT INTO participant VALUES (213, 'Arthur Dupuis', '2020 Boulevard des Montagnes', '75000', 'Chef cuisinier', 'Paris');

INSERT INTO participant VALUES (214, 'Manon Leclerc', '2121 Rue des Roses', '13000', 'Agriculteur', 'Marseille');

INSERT INTO participant VALUES (215, 'Arthur Dupon', '2021 Boulevard des Montagnes', '75000', 'Chef cuisinier', 'Paris');

INSERT INTO participant VALUES (218, 'Manoa Dupuis', '2022 Boulevard des Montagnes', '13000', 'Chef cuisinier', 'Paris');

INSERT INTO participant VALUES (217, 'Mathieu Dubois', '2023 Boulevard des Montagnes', '75000', 'Chef cuisinier', 'Paris');

/*----------------------------------------*/


/*----------Insciption----------*/

INSERT INTO inscription VALUES (300, 100, 200, 30, 115);

INSERT INTO inscription VALUES (301, 101, 201, 50, 50);

INSERT INTO inscription VALUES (302, 101, 202, 36, 30);

INSERT INTO inscription VALUES (303, 101, 203, 60, 30);

INSERT INTO inscription VALUES (304, 103, 204, 70, 100);

INSERT INTO inscription VALUES (305, 103, 205, 90, 10);

/*----------------------------------------*/


/*----------Conference----------*/
```

```
INSERT INTO conference VALUES (400, 'Intelligence émotionnelle', '12/01/2023', 'amphithéâtre1 Lyon');

INSERT INTO conference VALUES (401, 'Psychologie positive','12/07/2023','amphithéâtre1 Lyon');

INSERT INTO conference VALUES (402, 'Médecine énergétique', '05/01/2024','amphithéâtre2 Lyon');

INSERT INTO conference VALUES (403, 'Communication Familiale','10/01/2023','amphithéâtre1 Lyon');

INSERT INTO conference VALUES (404, 'Éducation Parentale','20/04/2021','amphithéâtre2 Lyon');

/*----------------------------------------*/


/*----------------Poseder-----------------*/

INSERT INTO poseder VALUES ('physique', 1, 100);

INSERT INTO poseder VALUES ('physique', 2, 101);

INSERT INTO poseder VALUES ('psychologue', 2, 102);

INSERT INTO poseder VALUES ('holistique', 3, 103);

/*----------------------------------------*/


/*----------------posseder2----------------*/

INSERT INTO posseder2 VALUES ('psychologue', 2, 400);

INSERT INTO posseder2 VALUES ('psychologue', 2, 401);

INSERT INTO posseder2 VALUES ('holistique', 2, 402);

INSERT INTO posseder2 VALUES ('familial', 4, 403);

INSERT INTO posseder2 VALUES ('familial', 4, 404);

/*----------------------------------------*/


/*-----------------Instant----------------*/

INSERT INTO inscrit VALUES (400, 205);

INSERT INTO inscrit VALUES (400, 206);

INSERT INTO inscrit VALUES (400, 207);
```

```
INSERT INTO inscrit VALUES (401, 208);
INSERT INTO inscrit VALUES (401, 209);
INSERT INTO inscrit VALUES (402, 210);
INSERT INTO inscrit VALUES (402, 211);
INSERT INTO inscrit VALUES (402, 212);
INSERT INTO inscrit VALUES (403, 213);
INSERT INTO inscrit VALUES (403, 214);
INSERT INTO inscrit VALUES (404, 214);
/*----------------------------------------*/

commit;
select * from formation;
select * from formateur;
select * from participant;
select * from conference;
select * from inscrit;
select * from inscription;
select * from poseder;

// R1 :
SELECT count(*) as nb_formations FROM formation;

// R2 :
SELECT COUNT(*) AS nb_formation_en_2023 FROM formation WHERE EXTRACT(YEAR
FROM date_formation) = 2023;

// R3 :
SELECT * FROM participant
   INNER JOIN inscrit ON participant.id_participant = inscrit.id_participant
   INNER JOIN conference ON inscrit.id_conference = conference.id_conference
```

```
   WHERE date_conference = '15/10/2023';


// R3.2 :

SELECT * FROM participant

   INNER JOIN inscrit ON participant.id_participant = inscrit.id_participant

   INNER JOIN conference ON inscrit.id_conference = conference.id_conference

   WHERE date_conference = '12/01/2023';


// R4 :

SELECT participant.id_participant, participant.nom_participant, inscription.id_formations,
formation.nom_formation, formation.prix-(inscription.solde_paye + inscription.acompte) AS
reste_a_payer

   FROM participant

   INNER JOIN inscription ON participant.id_participant = inscription.id_participant

   INNER JOIN formation ON inscription.id_formations = formation.id_formations

   WHERE formation.id_formations = 101 AND (inscription.solde_paye +
inscription.acompte) < formation.prix;


// R5 :

SELECT code_postal, COUNT(*) AS nb_personnes FROM participant

GROUP BY code_postal

HAVING COUNT(*) = (SELECT MAX(COUNT(*)) FROM participant GROUP BY code_postal)

   OR COUNT(*) = ( SELECT MAX(COUNT(*)) FROM participant GROUP BY code_postal
HAVING COUNT(*) != (SELECT MAX(COUNT(*)) FROM participant GROUP BY code_postal))

   OR COUNT(*) = (SELECT MAX(COUNT(*)) FROM participant GROUP BY code_postal
HAVING COUNT(*) != (SELECT MAX(COUNT(*)) FROM participant GROUP BY code_postal
HAVING COUNT(*) !=

      (SELECT MAX(COUNT(*)) FROM participant GROUP BY code_postal)) AND COUNT(*) !=
(SELECT MAX(COUNT(*)) FROM participant GROUP BY code_postal));


// R6 :
```

```
SELECT conference.id_conference, conference.nom_conference,
conference.date_conference, COUNT(*) as nb_participants from conference
   INNER JOIN inscrit ON inscrit.id_conference = conference.id_conference
   INNER JOIN participant ON participant.id_participant = inscrit.id_participant
   GROUP BY conference.id_conference, conference.nom_conference,
conference.date_conference
   HAVING COUNT(*) = (SELECT MAX(COUNT(*)) from conference
                  INNER JOIN inscrit ON inscrit.id_conference = conference.id_conference
                  INNER JOIN participant ON participant.id_participant = inscrit.id_participant
                  GROUP BY conference.id_conference);
```

// R7 :

```
SELECT formateur.id_formateur, formateur.nom, formateur.prenom, COUNT(*) AS nb_fan
FROM formateur
   INNER JOIN poseder ON formateur.id_formateur = poseder.id_formateur
   INNER JOIN inscription ON poseder.id_formations = inscription.id_formations
   GROUP BY formateur.id_formateur, formateur.nom, formateur.prenom
   HAVING COUNT(*) = (SELECT MAX(COUNT(*)) FROM formateur
                  INNER JOIN poseder ON formateur.id_formateur = poseder.id_formateur
                  INNER JOIN inscription ON poseder.id_formations = inscription.id_formations
                  GROUP BY formateur.id_formateur);
```

// R8 :

```
SELECT participant.id_participant, participant.nom_participant, formation.prix-
(inscription.solde_paye + inscription.acompte) AS reste_a_payer FROM participant
   INNER JOIN inscription ON participant.id_participant = inscription.id_participant
   INNER JOIN formation ON formation.id_formations = inscription.id_formations
   WHERE (inscription.solde_paye + inscription.acompte) < formation.prix;
   formateur.id_formateur, formateur.nom
```

// R BONNUS 1 :

```
SELECT formateur.id_formateur, formateur.nom, SUM(titre.prix_heure*formation.temps)
as indemnite_mois, TO_DATE(
    EXTRACT(MONTH FROM formation.date_formation) || '/' || EXTRACT(YEAR FROM
formation.date_formation),'MM/YYYY') as date_indemnite FROM formateur
  INNER JOIN titre ON titre.titre = formateur.titre
  INNER JOIN poseder ON poseder.id_formateur = formateur.id_formateur
  INNER JOIN formation ON formation.id_formations = poseder.id_formations
  GROUP BY formateur.id_formateur, formateur.nom, EXTRACT(YEAR FROM
formation.date_formation), EXTRACT(MONTH FROM formation.date_formation);
```