



Projet Fil rouge

Motoclub Millau Passion

Jalon 1

|  |  |
| --- | --- |
| Emilie Paniagua – CDA5 | 27 août 2021 |

# Remerciements

Je remercie l'association AMIO et tout particulièrement l'ESRP 2ISA de m'avoir donné ma chance. Malgré une longue attente liée au COVID et au confinement de mars 2020, qui a retardé l'entrée en formation d'une huitaine de mois, j'intègre une formation qui va me permettre de trouver une issue professionnelle et personnelle.

Je tiens à renouveler donc ma gratitude à l'équipe pédagogique et aux formateurs, sans oublier le personnel de l'hébergement, de la restauration, la structure psycho médico sociale, tout ce qui fait de l'ESRP un endroit convivial, propice à travailler et à s'épanouir. Je mesure la chance que j'ai d'étudier dans la ville de Millau, qui est au demeurant un cadre agréable, proche de la nature et des commerces.

Je remercie également mes camarades de promotion, pour leur soutien et la bonne humeur quotidienne.

Je remercie également mes parents et mes amis développeurs, qui ont cru dès le début en mes capacités à progresser dans ce métier.

Résumé

PROJET Moto Club Millau Passion

Moto Club Millau Passion (MCMP) est un club organisant des sorties en deux roues. MCMP souhaite structurer son système d'information afin de communiquer avec ses adhérents, gérer les cotisations, publier des nouvelles, et autres besoins consignés dans le **cahier des charges**.

Pour ce faire, je dois analyser lesbesoins du client. Je rédige un schéma de cas d'utilisation en **UML**, un scénario d'utilisation, un **dictionnaire des données**, des règles de gestion. Ces outils me servent à recenser les besoins évoqués afin de construire **un modèle de données (MCD)** avec **Merise** permettant la création d'une **base de données.**

Je crée ensuite des tables, contraintes, et autres objets nécessaires à l'architecture de la base. Des jeux de données insérés dans celle-ci me permettent ensuite de formuler des **requêtes SQL** de test afin d'évaluer la bonne  gestion des informations, le tout sous le **SGBD Oracle**.

Tout au long du projet, je consigne mes actions, lesquelles requièrent de fréquentes itérations, et une certaine agilité qui permet de revenir aux étapes précédentes afin de peaufiner à la fois la base et les différentes données de test.

Enfin, j'établis des règles de **gestion de la sécurité** pour l'utilisation de la base de données, avec la mise en place de droits différents pour les utilisateurs, et de programmes en PLSQL.

**Mots Clés :** cahier des charges, UML, dictionnaire des données, MCD, Merise, Base de données, requêtes, SQL, SGBD, Oracle, gestion de la sécurité, PLSQL.

Emilie Paniagua

CDA5 – 2ISA

Table des matières

[Gestion du temps : 4](#_Toc80802582)

[Suivi des modifications GIT 6](#_Toc80802583)

[Diagramme des cas d'utilisation 7](#_Toc80802584)

[Exemple de scénario significatif 8](#_Toc80802585)

[Dictionnaire de données et les règles de gestion 9](#_Toc80802586)

[Diagrammes d'occurrence, cardinalités 12](#_Toc80802587)

[Modèle Logique de Données (MLD) 13](#_Toc80802588)

[Schéma de la base de données : 15](#_Toc80802589)

[Requêtes de test SQL 16](#_Toc80802590)

[PLSQL 17](#_Toc80802593)

[Droits 18](#_Toc80802594)

[Datapump – sauvegarde de la base de données 18](#_Toc80802595)

Gestion du projet

Planning et suivi de l'activité

Etat d'avancement

Conception de la base de données

Dictionnaire des données

Règles de gestion

Modèles des données

Mise en place de la base de données

Création des tables et des autres objets de la base (vues, index, …)

Elaboration du jeu d'essai en partant des scénarios les plus représentatifs

♣ Réalisation

Requêtes et procédures les plus significatives et résultats obtenus

♣ Gestion de la sécurité

Mise en place des droits utilisateurs

Sauvegarde de la base de test.

♣ Bilan

Apports et difficultés rencontrées

Perspectives

(Le dossier de projet hors annexes doit comporter entre 15 et 20 pages)

Remerciements

Présentation personnelle

Liste des compétences du référentiel couvertes par ce jalon :

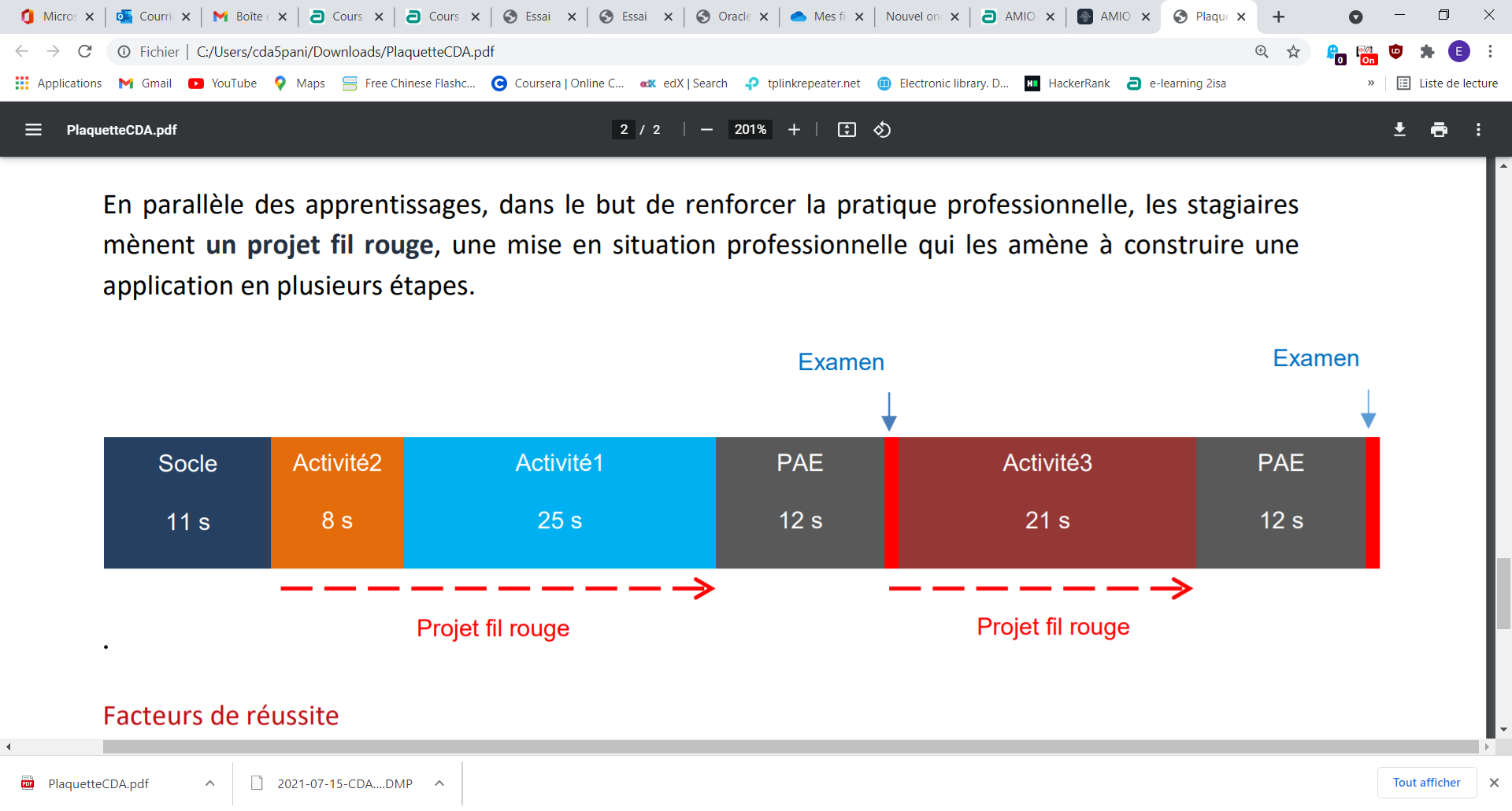
Concevoir une base de données

Mettre en place une base de données

Développer des composants dans le langage d'une base de données.

Chaque jalon se conclut par un EPCF, un examen partiel en cours de formation.

Nous en sommes au jalon 1



**Jalon 1**

Jalon 2

Jalon 3

Jalon 4

Expression des besoins du projet

Le fil rouge nous a été présenté en amphi, pour son lancement, le 21 juin 2021. C'est un projet qui permet de récapituler et d'ancrer les différentes compétences acquises en cours.

## Gestion du temps :

Concevoir la base de données nécessite de jongler de façon récursive avec trois étapes principales de la création du projet :

* Les règles de gestion et le dictionnaire des données issu du cahier des charges et de la FAQ
* Le MCD
* Les create table et insertions de valeurs dans la base de données

Au fur et à mesure de la lecture du cahier des charges, de la compréhension de la FAQ et des échanges avec les formateurs, j'en déduis que le travail avance suivant une boucle que l'on peut appeler boucle de feedback (ou roue de Deming, voir image ci-dessous),



Figure 1 : Boucle de Feedback : comment comprendre son fonctionnement

La boucle de feedback fonctionne suivant 4 micro-étapes:

* **planifier (plan),** où l'on envisage la tâche à réaliser, les paramètres nécessaires et la technique à employer
* **faire (do),** on met en œuvre la technique avec les paramètres envisagés
* **vérifier (check),**  on vérifie si l'étape a bien produit les résultats escomptés
* **agir (act),** suivant les résultats obtenus, on passe à l'étape suivante.

**Exemple pratique sur une boucle :**

* **Plan :** j'envisage d'ajouter des attributs à une entité Adhérents que je souhaite créer sous Looping. J'ajoute l'id, le nom, le prénom, la date de naissance, la photo de profil, le numéro de voie, le nom de rue, le code postal, la ville, le pays. J'envisage de créer une table Organisateur, qui hériterait de la table Adhérents.
* **Do :** je codifie ces informations dans le Dictionnaire des données et je les rentre dans le MCD sur Looping.
* **Check :** Je vérifie en échangeant avec mes formateurs si l'architecture est intéressante. Ceux-ci me disent que l'héritage est difficile à exploiter sous looping et Oracle.
* **Act :** Je décide donc de fusionner les deux tables Adhérent et Organisateur en une table Membre, et de recourir à un booléen (Nombre 1 pour vrai, 0 pour faux sous Oracle) pour départager si le membre est ou non un organisateur.

**Exemple avec une boucle sur les insertions de données :**

* **Plan :** Je me suis rendue compte qu'il manquait le login, le mot de passe, et un numéro de téléphone dans membres. Je planifie l'ajout de ces attributs dans le MCD et le dictionnaire de données.
* **Do :** Je codifie et rentre les données dans le MCD, et comme l'étendue de mes autres tables est prête, je copie le code de création des tables et je l'insère dans la base oracle. J'insère également des données dans mon fichier insert correspondant au login, mot de passe et numéro de téléphone de membres.
* **Check :** Je vérifie que les informations ont été écrites proprement en base.
* **Act :** Je sauvegarde mes fichiers de création de table et d'inserts de valeurs sur des fichiers texte.

En conséquence, par la nature même de ces itérations, il est très difficile d'estimer et de prévoir le temps que l'on va passer à chacune desdites étapes. Je vais donc me livrer à cet exercice mais en conscience qu'il a ses limites. Pour ce faire, j'ai établi un tableau de suivi sur Trello où j'ai répertorié l'avancée quotidienne des différentes tâches que j'ai entreprises. (nota : j'ai perdu l'accès au tableau Trello, il ne me reste plus que la prise d'écran ci-dessous pour évaluer le temps passé, je choisis donc de transférer le suivi sur une feuille excel classique)

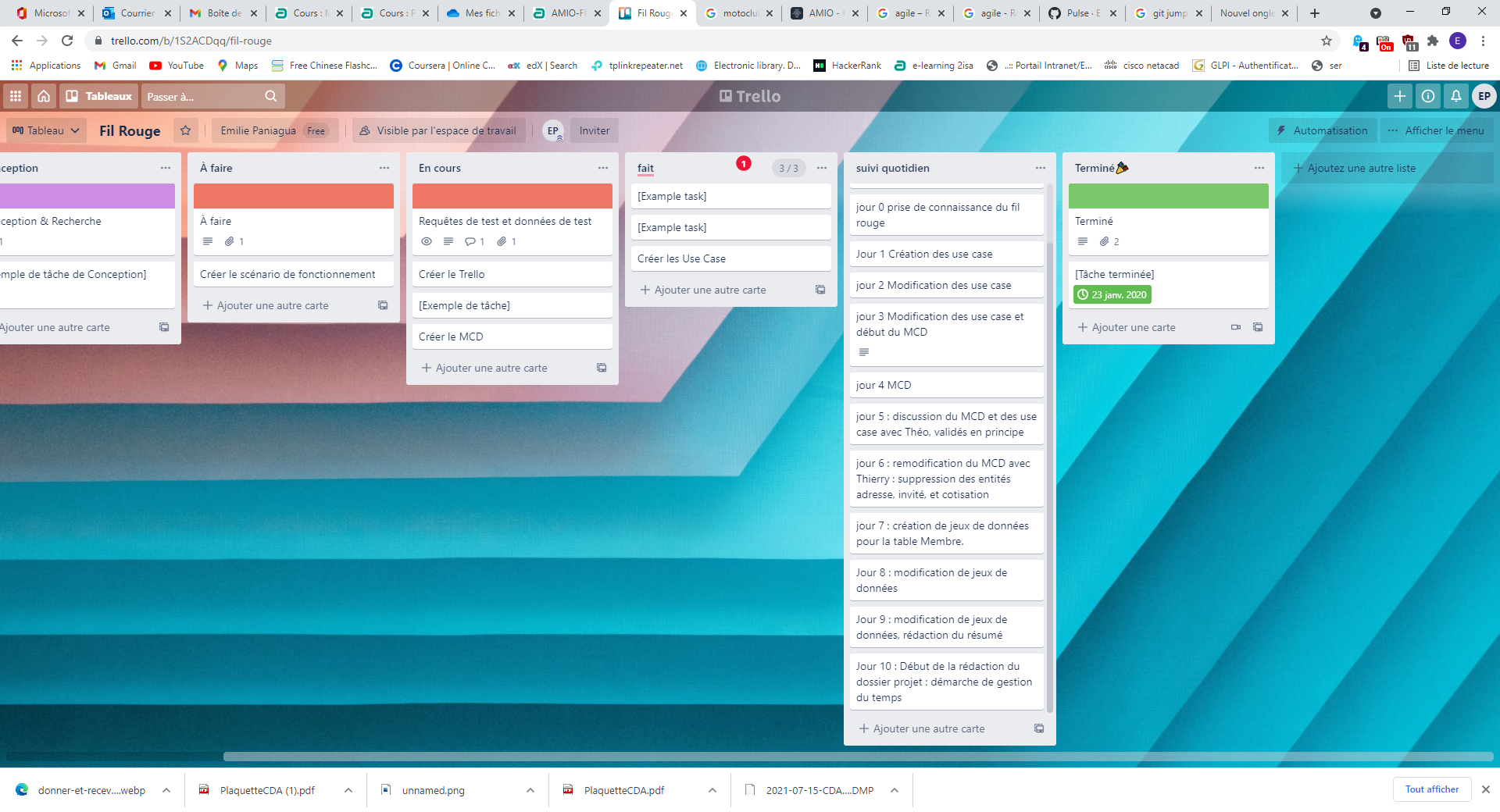


Figure 2 : impression d'écran du suivi des tâches au 15 juillet 2021

(ERRATUM : le suivi quotidien est fait en demi-journées de 4 heures et non en journées entières)

Tableau excel de gestion du temps

## Suivi des modifications GIT

Pour suivre et sauvegarder les modifications que j'apporte à mon dossier fil rouge, je me sers de Git et d'un [repository sur Github](https://github.com/E1000y/FilRougeMCMP).

(voir également [la liste des commits du projet](https://github.com/E1000y/FilRougeMCMP/commits/main))

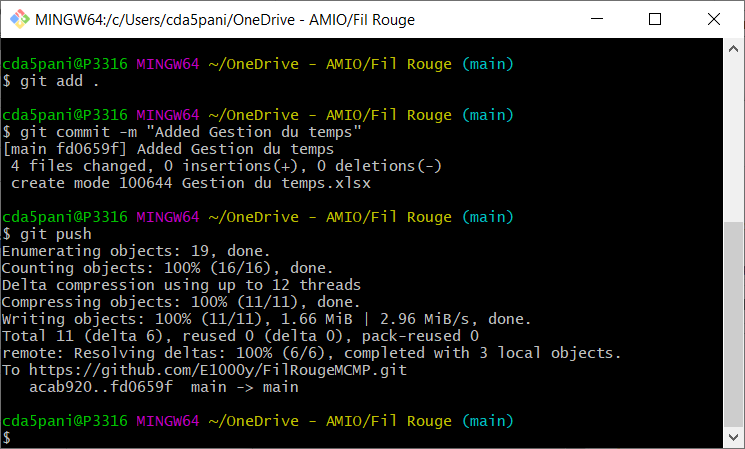
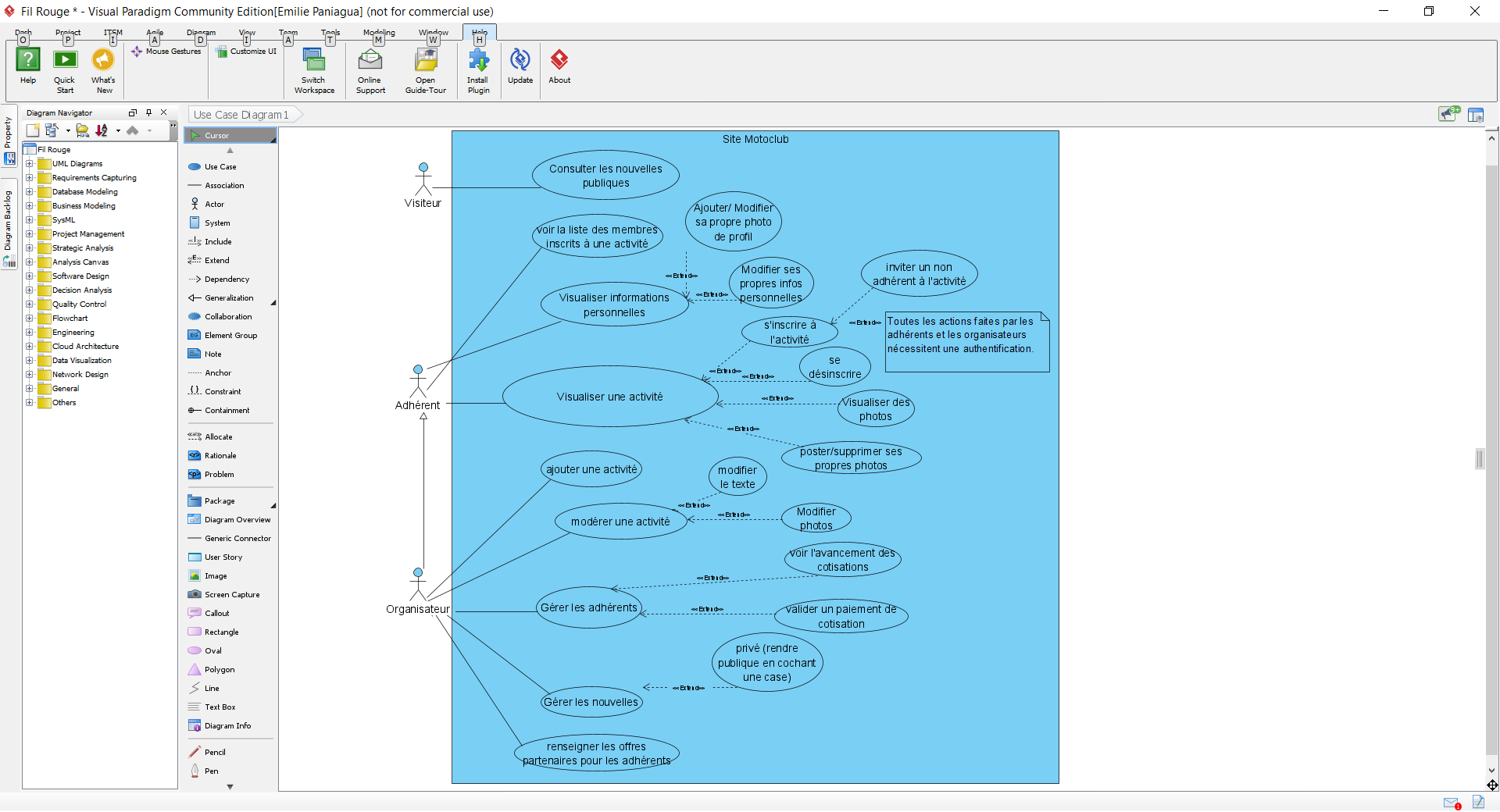


Figure 3 : Gestion des versions avec Git, exemple

## Diagramme des cas d'utilisation



Trois acteurs interviennent dans le système :

* Les visiteurs, qui n'ont pour possibilité que de consulter les informations publiques du site,
* Les adhérents, qui participent aux sorties de l'association, pour lesquelles ils payent une cotisation annuelle comme dans un club classique, et qui peuvent agir sur leur propre profil,
* Les organisateurs, qui sont les membres du bureau de l'assocation. Ce sont eux qui font vivre le site en publiant des propositions de sorties à moto, auxquelles les adhérents vont s'inscrire. Ils disposent également des droits pour gérer les adhérents.

Du diagramme des cas d'utilisations vont découler des scénarios d'analyse des fonctionnalités de l'application dont un exemple est donné ci-dessous :

## Exemple de scénario significatif

Cas : "Ajouter sa photo de profil"

**Objectif :** Permettre d'ajouter sa photo de profil.

**Acteurs :** l'adhérent

**Date :** le 17/08/2021

**Responsable :** Emilie Paniagua

**Version :** 3.0

**Préconditions :** l'adhérent visualise ses informations personnelles. Le système présente une photo de profil par défaut.

**Scénario :**

1. L'adhérent choisit une photo depuis sa galerie ou son ordinateur
2. L'adhérent valide son choix de photo
3. Le système stocke la photo associée à l'utilisateur

**Alternatives :**

1. L'adhérent ne choisit pas de photo de profil
2. Le système laisse un avatar par défaut

**Exceptions :**

Si le système n'arrive temporairement pas à joindre la base de données, il affiche un message d'erreur du type : "connexion à la base de données impossible, veuillez réessayer plus tard".

**Postconditions :**

L'adhérent a correctement inséré sa photo

Le système affiche sa page de profil, comprenant sa photo correctement affichée.

## Dictionnaire de données et les règles de gestion

La lecture du cahier des charges permet de tirer des informations importantes à coder dans la base de données. Je fais évoluer conjointement cette liste qui s'appelle dictionnaire de données, et la conception sous le logiciel Looping de ma base de données- Modèle Conceptuel de Données ou MCD, voir plus loin.

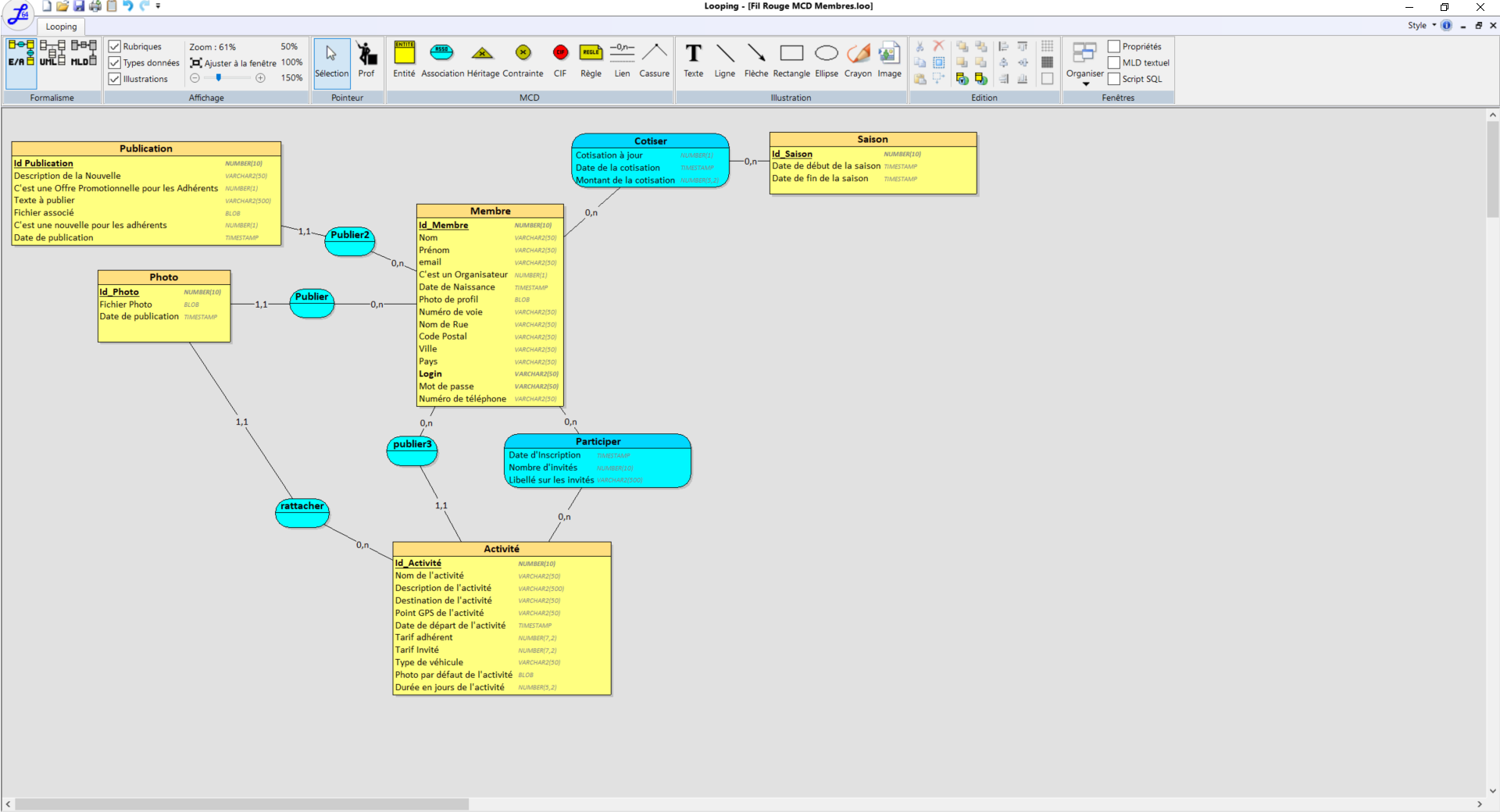
|  |  |  |
| --- | --- | --- |
| **Libellé** | **Code** | **Type** |
|  |  |  |
| Id du membre : adhérent ou organisateur | Id\_Membre | number(10) |
| nom du membre | Nom | Varchar2(50) |
| prénom du membre | Prenom | Varchar2(50) |
| email du membre | email | Varchar2(50) |
| Est un organisateur | IsOrganisateur | Number(1) |
| Date de naissance | Date\_Naissance | Timestamp |
| Numéro de voie | No\_Voie | Varchar2(50) |
| Nom de rue | Nom\_De\_Rue | Varchar2(50) |
| Code postal | Code\_Postal | Varchar2(50) |
| Ville | Ville | Varchar2(50) |
| Pays | Pays | Varchar2(50) |
| Login | Login | Varchar2(50) |
| Mot de Passe | mdp | Varchar2(50) |
| Numéro de Téléphone | NoTel | Varchar2(50) |
|  |  |  |
| id activité | Id\_Activite | number(10) |
| nom de l'activité | Nom\_Activite | Varchar2(50) |
| Description de l'activité | Description\_Activite | Varchar2(50) |
| Destination Activité | Destination\_Activite | Varchar2(50) |
| Point GPS Activité | Point\_GPS\_Activite | Varchar2(50) |
| Date de l'activité | Date\_Activite | timestamp |
| Tarif Adhérent | Tarif\_Adherent | Number(7,2) |
| Tarif invité | Tarif\_Invite | Number(7,2) |
| Type de véhicule | Type\_Vehicule | Varchar2(50) |
| photo par défaut de l'activité | Photo\_Activite | BLOB |
| Durée en jours de l'activité | Duree\_Activite | Number(5,2) |
|  |  |  |
| Date d'inscription | Date\_Inscription | Timestamp |
| Nombre d'invités | Nombre\_invites | Number(10) |
| Libellé sur les invités | Libelle\_invites | Varchar2(500) |
|  |  |  |
| id photo | Id\_Photo | Number(10) |
| fichier photo | Fichier\_Photo | BLOB |
| date de publication | Date\_Publi\_Photo | Timestamp |
|  |  |  |
| id de la publication | Id\_Publication | Number(10) |
| Description de la nouvelle | Description\_Nouvelle | Varchar2(50) |
| C'est une Offre Promotionnelle pour les Adhérents | isOffrePromotionnelleAdherents | Number(1) |
| Texte à publier | Texte | Varchar2(500) |
| C'est une nouvelle pour les adhérents | IsNouvelleAdherents | Number(1) |
| Fichier associé | Fichier\_Associe | BLOB |
| Date de la publication | Date\_Publi\_Nouvelle | Timestamp |
|  |  |  |
| Cotisation à jour | Cotisation\_A\_Jour | Number(1) |
| Date de la cotisation | Date\_Cotisation | Timestamp |
| Montant de la cotisation | Montant\_Cotisation | Number(5,2) |
|  |  |  |
| Id Saison | Id\_Saison | Number(10) |
| Date de début de la saison | Date\_Debut\_Saison | Timestamp |
| Date de fin de la saison | Date\_Fin\_Saison | Timestamp |

Valeurs calculées : nombre de participants à une activité donnée TODO : mettre dans le tableau

|  |  |
| --- | --- |
| RGFAQ01 | les activités comptent maximum 50 participants |
| RGFAQ02 | On peut s'inscrire maxi 5 jours avant la date de début de l'activité |
| RGFAQ03 | l'application doit savoir si un adhérent a payé ou pas |
| RGFAQ04 | le paiement ne se fait pas par le biais de l'application |
| RGFAQ05 | Les activités ne sont visibles que par les adhérents, mais il peut y avoir une news qui parle d'une activité |
| RGFAQ06 | une saison dure un an (du 1er septembre au 31 août suivant) |
| RGFAQ07 | le nombre maximum de participants à une activité est fixé à 50 |
| RGFAQ08 | la date maxi d'inscription à une activité est fixée à la date de l'activité moins 5 jours. |
| RGFAQ09 | on doit savoir qui écrit une nouvelle |
|  |  |
|  |  |
| RG00 | L'adhésion est obligatoire pour participer aux animations |
| RG01 | une personne inscrite à une activité peut annuler son inscription avant la date limite d'inscription |
| RG02 | Un adhérent ne peut pas publier d'information |
| RG03 | un organisateur est un adhérent, qui sont tous deux membres |
| RG04 | Tout adhérent a possibilité d'inscrire des invités non adhérents |
| RG05 | Toute activité ne comporte pas nécessairement de participation financière |
| RG06 | Si une participation est demandée, il y a un tarif adhérent et un tarif invité |

Le dictionnaire de données et le MCD étant créés en même temps. L'un prenant appui sur l'autre pour la réflexion de la construction de l'ensemble, voici ci-dessus la version finale du dictionnaire de données qui donne une bonne visibilité à la structure à adopter pour construire les tables du MCD.

Présentation du Modèle Conceptuel de Données MCD:



## Diagrammes d'occurrence, cardinalités

Le MCD est constitué de tables qui peuvent avoir des relations entre elles. Ces relations sont caractérisées par des cardinalités.

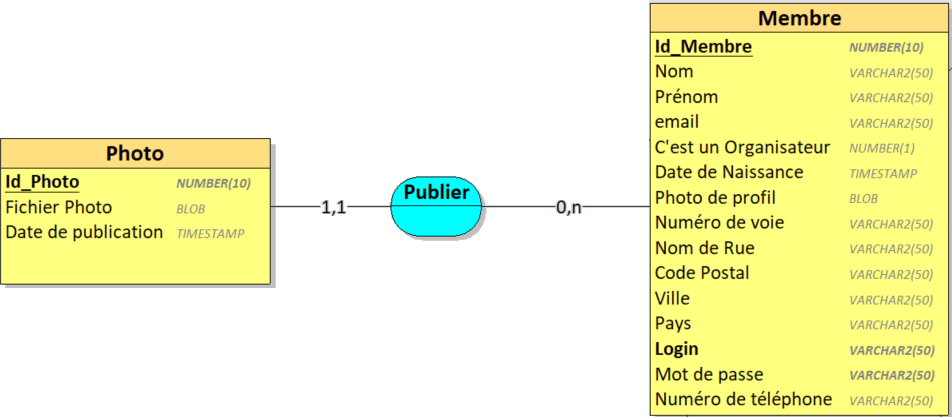
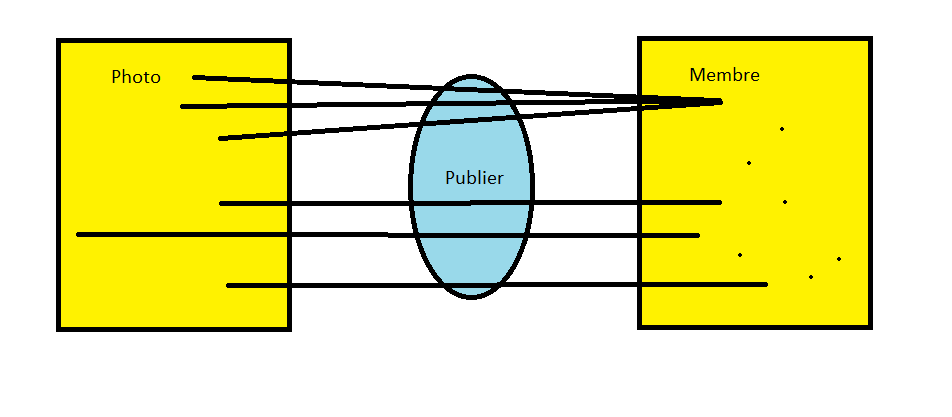


Figure 4 Deux tables (en jaune) et une association (en bleu) forment un diagramme d'occurrence

les chiffres de part et d'autre de la bulle bleue centrale se lisent de la manière suivante : une photo est publiée par (1,1) un et un seul membre, tandis qu'un membre peut publier (0,n) de zéro à un nombre non spécifié de photos.



Lu autrement, il n'existe pas de photos qui n'aient pas été publiées par un membre (pas de points libres dans photo), tandis qu'il existe des membres (points libres dans membre) qui n'ont pas publié de photo.

## Modèle Logique de Données (MLD)

On a besoin du Modèle Logique de Données pour concevoir les scripts de création des tables SQL. Afin de le produire, on reprend les éléments du MCD, et on décompose chaque attribut dans chaque table correspondante, avec chaque relation si celle-ci engendre une table. Par exemple, participer et cotiser sont des relations qui deviennent des tables car elles contiennent des attributs.

Les clés primaires sont en gras souligné (**Clé Primaire**), les clés étrangères sont en bleu italique préfixé d'un mot-dièse (*#Clé\_Etrangère*). Les valeurs dont le nom est en gras sont uniques, le type des valeurs est indiqué en italique à côté de leur nom. Quand le type des valeurs est gras, la valeur est non nulle. Exemple : dans Membre, le login est unique et non null, tandis que le mot de passe n'est que non null.

**Membre = (Id\_Membre *NUMBER(10)***,

Nom *VARCHAR2(50)*,

Prenom *VARCHAR2(50)*,

email *VARCHAR2(50)*,

IsOrganisateur *NUMBER(1)*,

Date\_Naissance *TIMESTAMP*,

Photo\_Profil *BLOB*,

No\_Voie *VARCHAR2(50)*,

Nom\_De\_Rue *VARCHAR2(50)*,

Code\_Postal *VARCHAR2(50)*,

Ville *VARCHAR2(50)*,

Pays *VARCHAR2(50)*,

**Login *VARCHAR2(50)***,

mdp ***VARCHAR2(50)***,

NoTel *VARCHAR2(50)***);**

**Activite = (Id\_Activite *NUMBER(10)***,

Nom\_Activite *VARCHAR2(50)*,

Description\_Activite *VARCHAR2(500)*,

Destination\_Activite *VARCHAR2(50)*,

Point\_GPS\_Activite *VARCHAR2(50)*,

Date\_Activite *TIMESTAMP*,

Tarif\_Adherent *NUMBER(7,2)*,

Tarif\_Invite *NUMBER(7,2)*,

Type\_Vehicule *VARCHAR2(50)*,

Photo\_Activite *BLOB*,

Duree\_Activite *NUMBER(5,2),*

*#Id\_Membre***);**

**Photo = (Id\_Photo *NUMBER(10)***,

Fichier\_Photo *BLOB*,

Date\_Publi\_Photo *TIMESTAMP,*

*#Id\_Activite,*

*#Id\_Membre***);**

**Saison = (Id\_Saison *NUMBER(10)***,

Date\_Debut\_Saison *TIMESTAMP*,

Date\_Fin\_Saison *TIMESTAMP***);**

**Publication = (Id\_Publication *NUMBER(10)***,

Description\_Nouvelle *VARCHAR2(50)*,

IsOffrePromotionnelleAdherents *NUMBER(1)*,

Texte *VARCHAR2(500)*,

Fichier\_Associe *BLOB*,

IsNouvelleAdherents *NUMBER(1)*,

Date\_Publi\_Nouvelle *TIMESTAMP,*

*#Id\_Membre***);**

**Participer = (*#Id\_Membre,***

***#Id\_Activite***,

Date\_Inscription *TIMESTAMP*,

Nombre\_Invites *NUMBER(10)*,

Libelle\_Invites *VARCHAR2(500)***);**

**Cotiser = (*#Id\_Membre,***

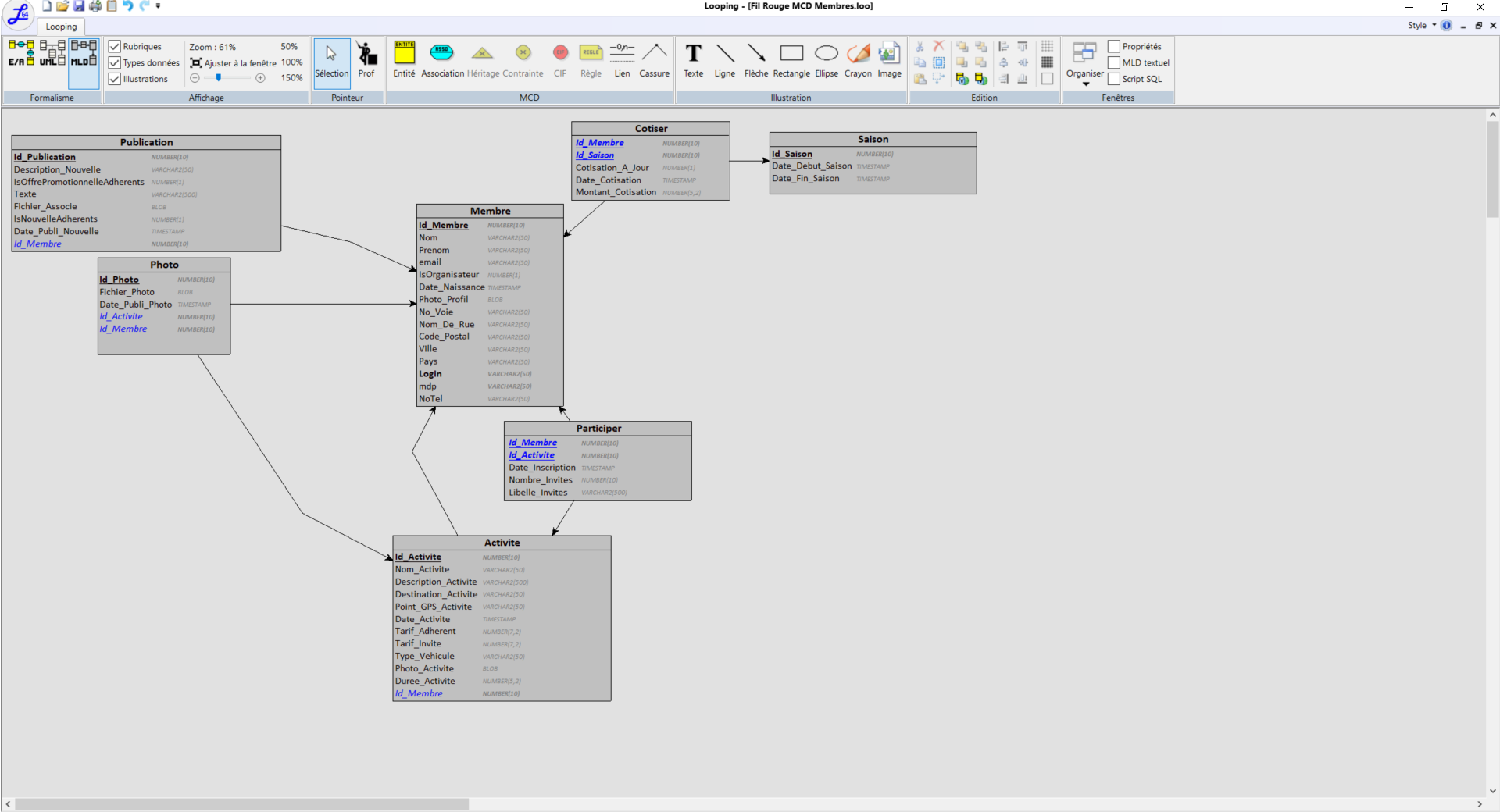
***#Id\_Saison***,

Cotisation\_A\_Jour *NUMBER(1)*,

Date\_Cotisation *TIMESTAMP*,

Montant\_Cotisation *NUMBER(5,2)***);**

## Schéma de la base de données :



## Requêtes de test SQL

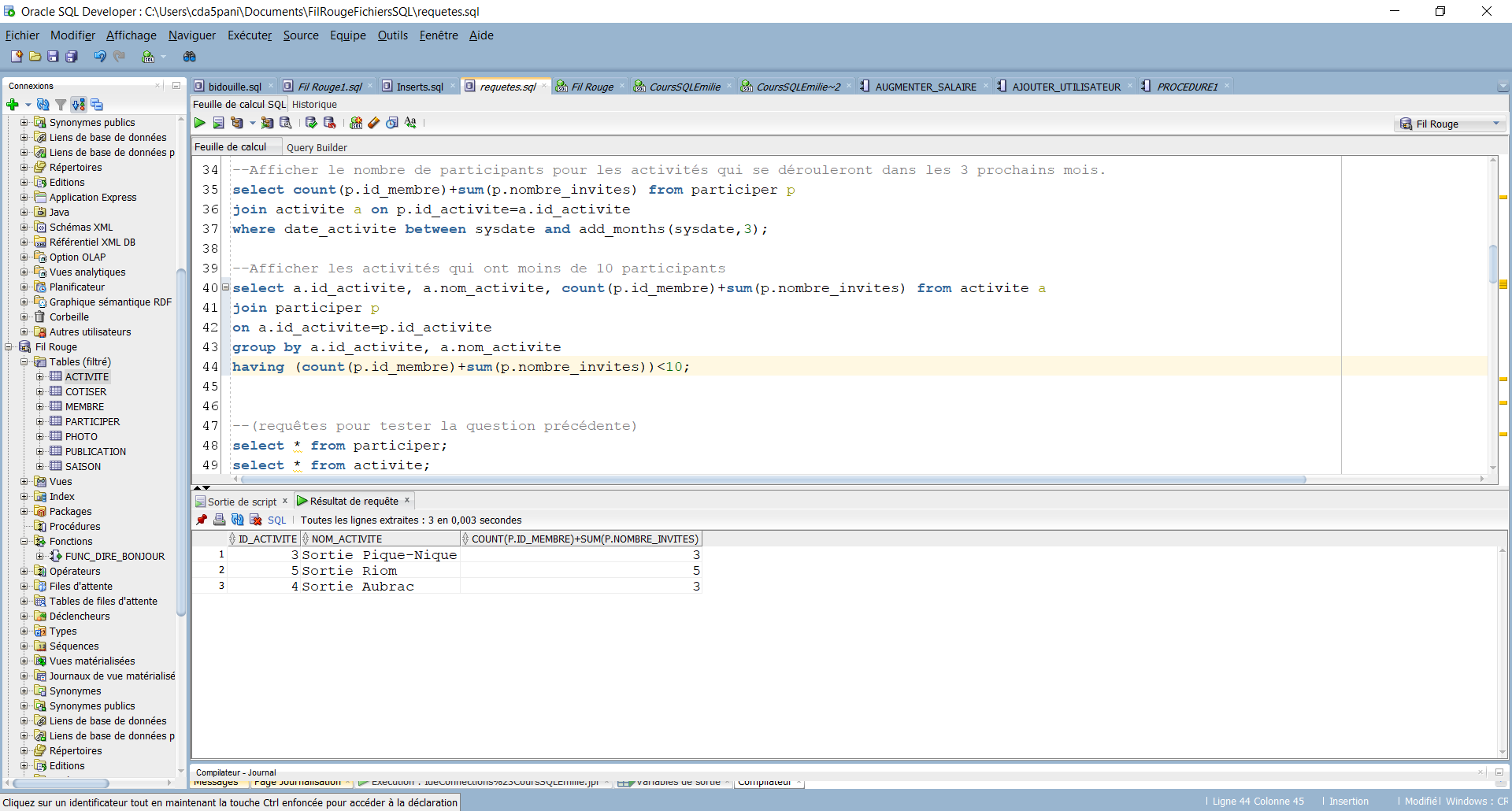
### Afficher les activités qui ont moins de 10 participants

select a.id\_activite, a.nom\_activite, count(p.id\_membre)+sum(p.nombre\_invites) from activite a

join participer p on a.id\_activite=p.id\_activite

group by a.id\_activite, a.nom\_activite

having (count(p.id\_membre)+sum(p.nombre\_invites))<10;



### En fin de saison, pour permettre de faire le bilan moral de l'association, afficher toutes les activités avec lenombre de participants adherents et le nombre de participants invites. Les trier par audience décroissante.

select a.nom\_activite, count(p.id\_membre), sum(p.nombre\_invites)

from activite a join participer p

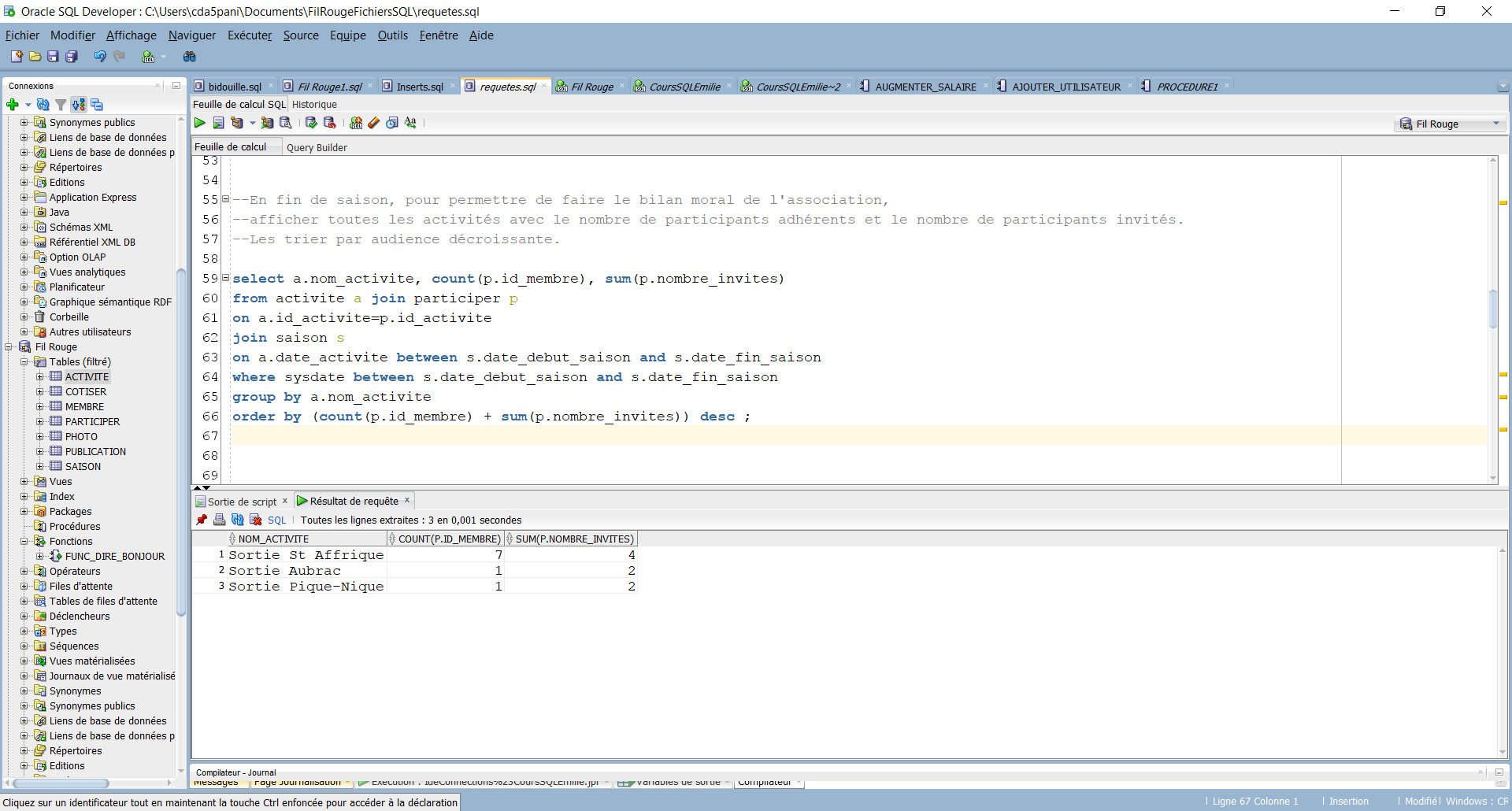
on a.id\_activite=p.id\_activite

join saison s on a.date\_activite between s.date\_debut\_saison and s.date\_fin\_saison

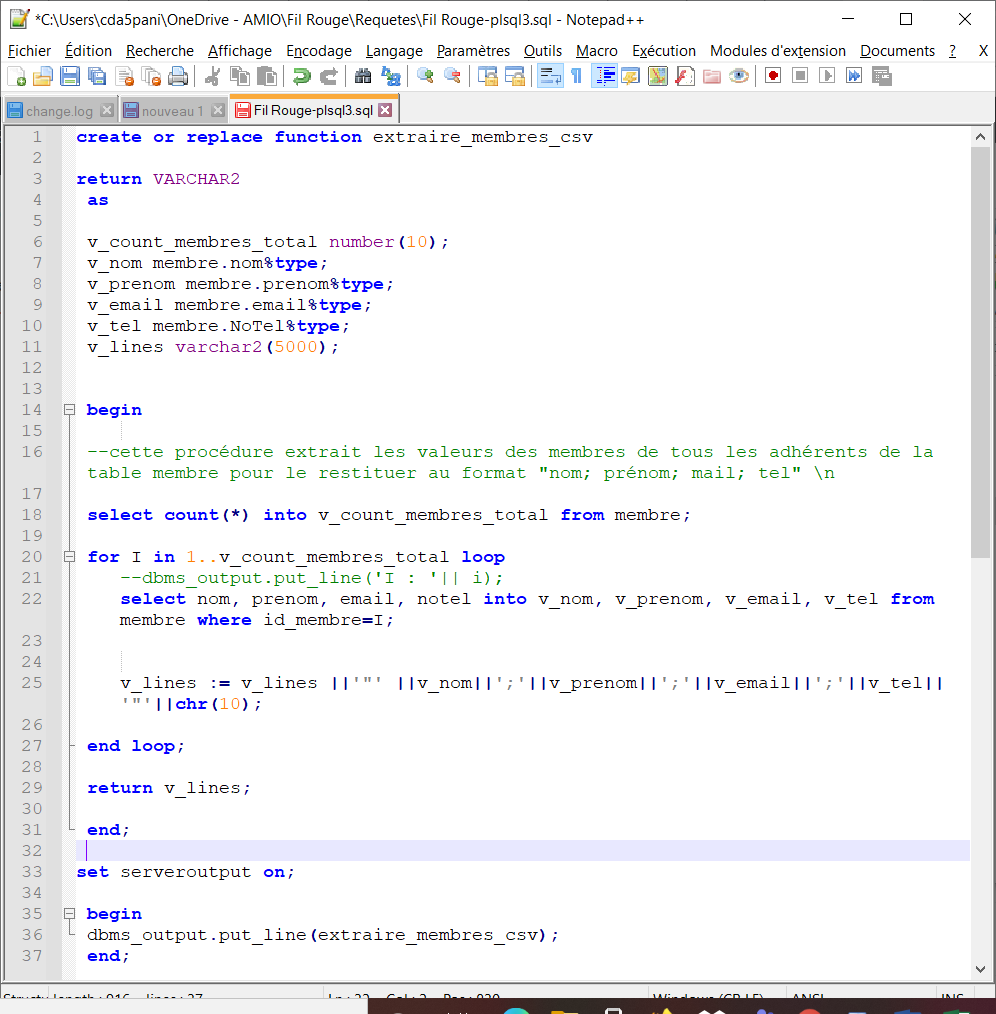
where sysdate between s.date\_debut\_saison and s.date\_fin\_saison

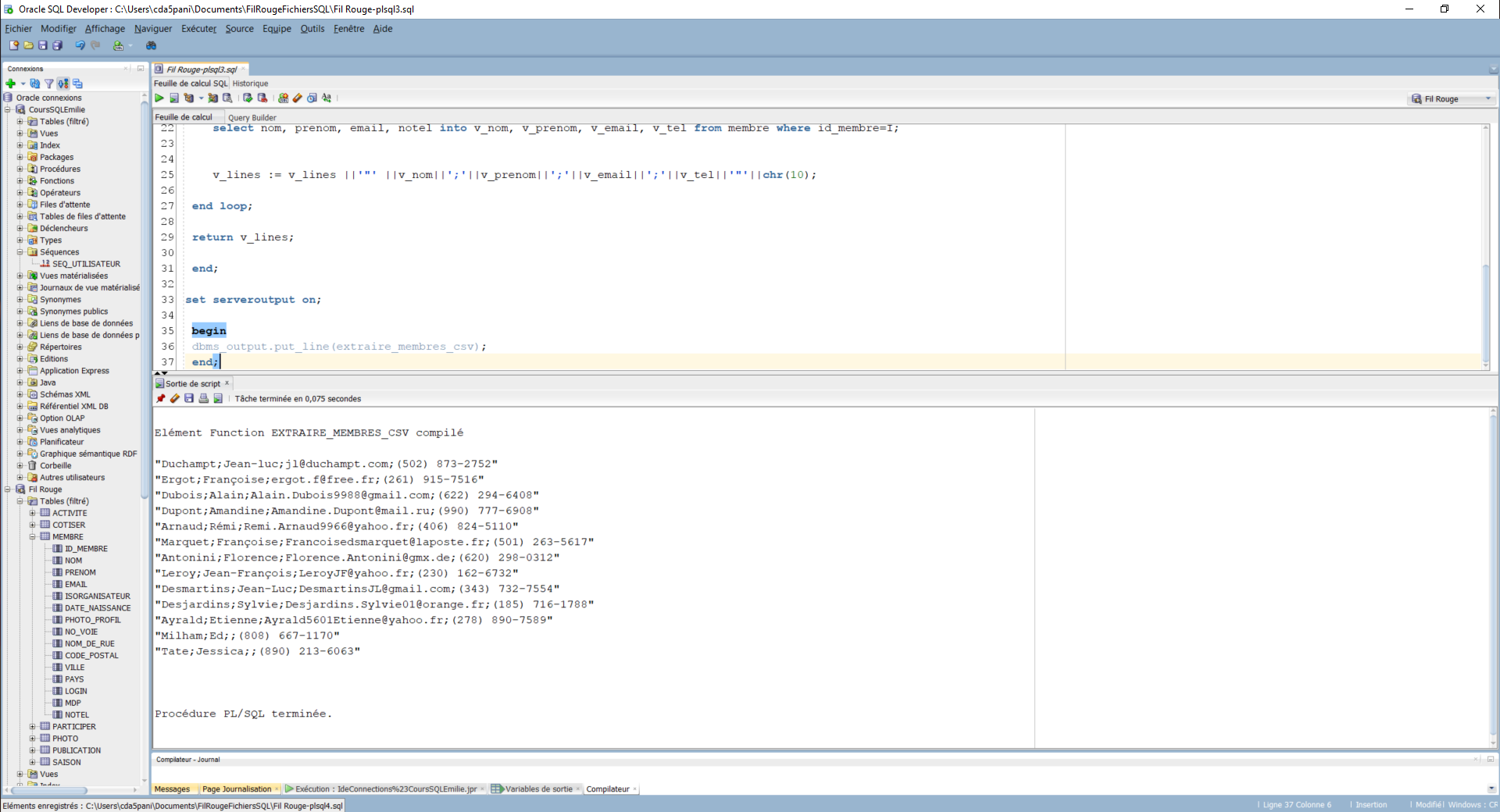
group by a.nom\_activite

order by (count(p.id\_membre) + sum(p.nombre\_invites)) desc ;



## PLSQL





## Droits

## Datapump – sauvegarde de la base de données

L'export de la base de donnée se fait en théorie en une ligne de commande sous cmd avec la commande ci-dessous :

*expdp cda5pani\_proj/cda5pani\_proj@oradev*

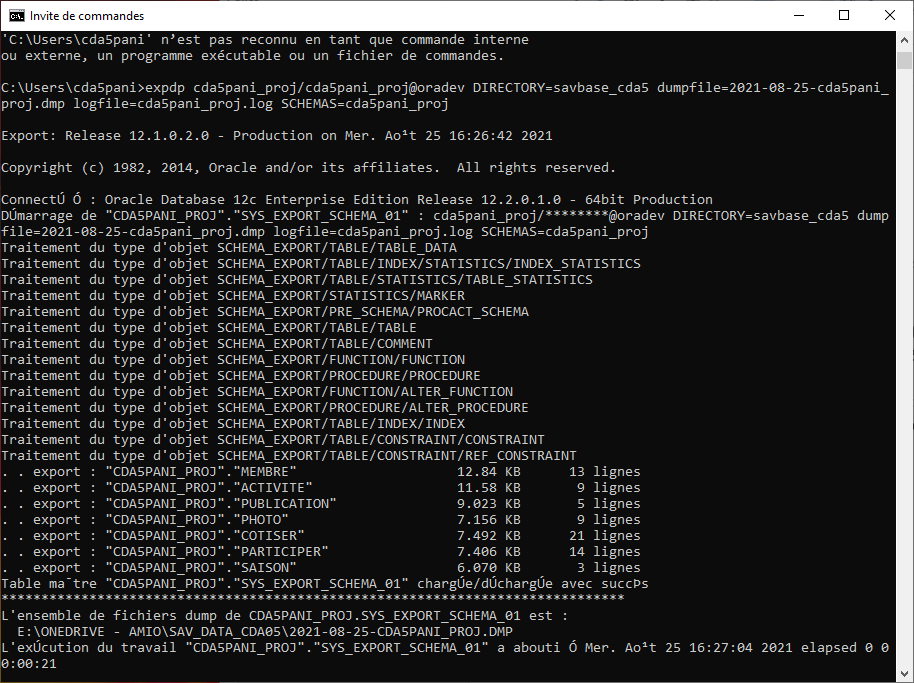
*DIRECTORY=savbase\_cda5*

*dumpfile=2021-08-25-cda5pani\_proj.dmp*

*logfile=cda5pani\_proj.log*

*SCHEMAS=cda5pani\_proj*

L'export se fait sur une machine distante et est accessible en pratique [ici](https://amio-my.sharepoint.com/personal/credo_adm_2isa_net/_layouts/15/onedrive.aspx?ct=1626333157046&or=OWA%2DNT&cid=41280d38%2D3073%2D5036%2Dbd7c%2Dd1df1e5bbfdf&originalPath=aHR0cHM6Ly9hbWlvLW15LnNoYXJlcG9pbnQuY29tLzpmOi9nL3BlcnNvbmFsL2NyZWRvX2FkbV8yaXNhX25ldC9Fckg0c1gtcjVBQk5pM2xaajZVdWVCWUI1d1dRSmtuMHR3dExZLUdHNGJUUTNnP3J0aW1lPWJuVC03RjlIMlVn&OR=Teams%2DHL&CT=1629899652609&sortField=Modified&isAscending=false&id=%2Fpersonal%2Fcredo%5Fadm%5F2isa%5Fnet%2FDocuments%2FSAV%5FDATA%5FCDA05), mais dans les faits je ne vois pas mon fichier sur la page en question, malgré le message de succès de la commande.



Bilan