Soleia Marietti, Lucie Faure-Beaulande, Benjamin Parciany, Loïc Giuntini

**Projet n°5 : Flux d’informations d’un réseau social**

**Reformulation et description du projet**

Le projet concerne un réseau social décentralisé axé uniquement sur les publications (texte, image, vidéo), sans messagerie directe. Les utilisateurs peuvent publier avec différentes options de visibilité (amis, abonnés, public) et interagir via des groupes thématiques.

L’objectif principal est de structurer une base de données permettant :

* La gestion des utilisateurs, de leurs profils et de leurs connexions.
* L’organisation des publications, en prenant en compte leur visibilité et leurs interactions.
* L’analyse des interactions pour mesurer l’engagement et recommander du contenu pertinent.
* L’optimisation de l’intégration des nouveaux utilisateurs pour maximiser leur engagement.
* L’évaluation de l’influence des membres et du rayonnement des publications dans le réseau.

**Fonctionnalités de la base de données :**

*Gestion des utilisateurs et des connexions*

* Chaque utilisateur possède un profil avec des informations basiques (nom, pseudonyme, date d’inscription, centres d’intérêt, etc.).
* Un système de connexions différenciées est mis en place :
  + Amis : Deux utilisateurs sont amis s’ils se suivent mutuellement.
  + Follower : Un utilisateur peut suivre un autre sans réciprocité.
* Les connexions indirectes sont identifiables (amis d’amis).

*Gestion des publications*

* Un utilisateur peut publier du texte, des images ou des vidéos.
* Chaque publication a une portée de visibilité configurable : amis, followers ou public
* Les interactions disponibles sur une publication sont : likes, commentaires (possibilité de réponse en thread) et partages

*Gestion des groupes thématiques*

* Les groupes sont des espaces de discussion organisés par centres d’intérêt.
* Les groupes permettent : la publication de contenus, les interactions, le suivi de l’activité des membres

*Recommandations et analyses des interactions*

La base de données supporte :

* L’identification des utilisateurs influents en analysant :
  + Le nombre de publications et interactions générées.
* La recommandation de connexions et de groupes en fonction de :
  + L’activité d’un utilisateur (contenus likés, commentés, partages).
  + Les centres d’intérêt déclarés et les groupes fréquentés.
* Le suivi de la propagation des publications, notamment via les partages indirects (ex : contenus relayés dans des groupes puis vus et likés par des utilisateurs sans connexion directe avec l’auteur initial).
* L’identification des utilisateurs interagissant en dehors de leur réseau pour repérer les comportements atypiques et élargir les recommandations.

**Questions et *axes d’analyse* :**

1. **Quels utilisateurs ont des connexions indirectes (amis d’amis) avec un utilisateur donné, et quels sont leurs intérêts communs ?**

*Connexions indirectes et intérêts communs*

* + Déterminer les "amis d’amis" en analysant les relations d’abonnement réciproques.
  + Identifier les centres d’intérêts communs via les groupes partagés.

1. **Quels groupes thématiques ont le plus d’interactions et quels types de publications génèrent le plus d’engagement ?**

*Engagement dans les groupes et types de publications populaires*

* + Mesurer l’activité dans les groupes thématiques en comptant les interactions (likes, commentaires, watchtime).
  + Identifier les formats de publication (texte, image, vidéo) qui génèrent le plus d’engagement.

1. **Comment intégrer un nouvel utilisateur et ses connexions dans le réseau existant ?**

*Intégration d’un nouvel utilisateur et recommandations*

* + Insérer un utilisateur et analyser ses centres d’intérêt (profils remplis, contacts importés, interactions).
  + Recommander des groupes et connexions pertinentes.
  + Faciliter son insertion en lui proposant des contenus et tendances adaptés.

1. **Quels utilisateurs influencent le plus les interactions dans un groupe donné ?**

*Identification des utilisateurs influents dans un groupe*

* + Analyser les interactions générées par chaque membre pour détecter les influenceurs.

1. **Comment recommander des groupes ou des connexions à un utilisateur basé sur son activité récente ?**

*Recommandations basées sur l’activité récente*

* + Suivre les actions des utilisateurs (likes, commentaires, groupes fréquentés).
  + Recommander des groupes et connexions en fonction des tendances récentes.
  + Comparer les activités des utilisateurs pour identifier des profils similaires et suggérer des communautés adaptées.

1. **Quelles publications ont atteint le plus grand nombre de vues, likes, et partages indirects via des réseaux de connexions ?**

*Propagation des publications via les partages indirects*

* + Analyser les vues, likes et partages qui proviennent de recommandations indirectes (ex. publications relayées dans les groupes).
  + Définir un mécanisme pour suivre l’origine des interactions et mesurer leur portée dans le réseau.

1. **Quels utilisateurs ont tendance à interagir en dehors de leurs cercles directs ?**

*Utilisateurs interagissant hors de leurs cercles directs*

* + Identifier les utilisateurs qui interagissent principalement avec du contenu ou des personnes en dehors de leur réseau direct (amis/abonnés).
  + Déterminer un seuil d’engagement hors cercle (ex. % d’interactions hors réseau, top 10% des plus actifs).

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

Description générée automatiquement

**Commentaire du schéma EA de la base de données :**

Un utilisateur :

* Peut ADHERER à aucun ou plusieurs groupes, ainsi la relation est de cardinalité N,N
* Peut BLOQUER aucun ou plusieurs autres utilisateurs, ainsi la relation est de cardinalité N,N récursive sur UTILISATEUR
* Peut se CONNECTER, en ami ou seulement en follower, à aucun ou plusieurs autres utilisateurs, ainsi la relation est de cardinalité N,N récursive sur UTILISATEUR
* Peut CREER aucune ou plusieurs publications, ainsi la relation est de cardinalité 0,N
* Peut INTERAGIR (commenter ou liker) sur aucune ou plusieurs publications, ainsi la relation est de cardinalité 0,N
* Peut visionner (WATCHTIME) aucune ou plusieurs publications, la relation est de cardinalité 0,N
* Peut ENVOYE une publication à un groupe, il s’agit alors d’un PARTAGE, il peut envoyer aucun ou plusieurs partages, la relation est donc 0,N

Une publication :

* Est CREEE par un unique utilisateur, la relation est de cardinalité 1,1
* Peut-être TAGUEE par un seul thème, la relation est de cardinalité 1,1
* Peut APPARTENIR à un groupe ou être indépendante, la relation est de cardinalité 0,1
* Peut recevoir aucune ou plusieurs INTERACTIONS (commentaires ou likes), la relation est de cardinalité 0,N
* Est regardé (WATCHTIME) par aucun ou plusieurs utilisateurs, la relation est de cardinalité 0,N
* Est partagé à un groupe par un utilisateur, il s’agit alors d’un PARTAGE. La publication peut être partagé aucune ou plusieurs fois, la relation est donc 0,N

Un groupe :

* Est DECRIT par un seul thème, la relation est de cardinalité 1,1
* Peut être rejoint (ADHERER) par plusieurs utilisateurs avec des rôles différent, la relation est de cardinalité 0,N
* Contient (APPARTIENT) aucune ou plusieurs publications, la relation est de cardinalité 0,N
* Peut recevoir aucun ou plusieurs PARTAGE, par un utilisateur, la relation est donc 0,N

Un thème :

* TAGUE une ou plusieurs publications, la relation est de cardinalité 1,N
* DECRIT aucun ou plusieurs groupes, la relation est de cardinalité 0,N
* Précise aucun ou plusieurs thèmes, la relation est de cardinalité 0,N
* Peut être précisé par un seul thème au maximum, la relation est de cardinalité 0,1

Un partage :

* Est reçu par un seul groupe à la fois, la relation est donc 1,1
* Est envoyé un unique utilisateur, la relation est donc 1,1
* Correspond à une unique publication, la relation est donc 1,1

**Requêtes SQL**

*Notions abordées* : fonctions, récursivité, requêtes imbriquées, sous-requêtes

*Question Bonus* : tentative d’implémentation de permissions pour le rôle d’admin dans la base

**Question 1 : Recherche d'amis indirects ayant des centres d'intérêt communs**

*Objectif :* Trouver les amis indirects des utilisateurs (les amis de leurs amis qui ne sont pas leurs amis directs) et afficher les centres d'intérêt qu'ils ont en commun.

*Etudiant* : Benjamin Parciany

* 1. Une image contenant texte, capture d’écran, Police, nombre

     Description générée automatiquement**Recherche des amis directs**

*Explication :*

* Utilisation de la CTE WITH RECURSIVE, avec une base de l’induction une jointure pour trouver les amis directs
* Récursivité pour obtenir les amis directs des amis directs précédemment trouvés
* Arrêt de profondeur de 1 avec la clause WHERE, pour ne pas avoir de doublons
  1. **Vérification du même groupe thématique**

*Explication :*

* Jointure avec ADHERER et GROUPE
* On retire les auto-relation, l’utilisateur doit être différent de son ami
* On exclut les amis directs, pour ne garder que les indirects
* GROUP BY des utilisateurs et des amis pour agréger les noms des groupes thématiques

**Question 2 : Analyse du score d'engagement des publications**

*Objectif :*Calculer l'engagement des publications et les classer par thèmes et types de publication

Etudiants : Lucie Faure-Beaulande, Soleia Marietti

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

Description générée automatiquement*

1. **Une image contenant texte, capture d’écran, Police, nombre

   Description générée automatiquementFonction « calculer\_score\_watchtime »**

*But :* Calculer un score basé sur le temps de visionnage d'une publication.

*Explication :*

1. Récupère le watchtime total et le nombre de vues de la publication depuis la table WATCHTIME.
2. Calcule la durée moyenne de visionnage par utilisateur.
3. Récupère la durée totale du contenu depuis la table publication.
4. Ratio de visionnage (%) = (Durée moyenne regardée / Durée totale du contenu) × 100.
5. Score final = ratio de visionnage × log(nombre de vues + 1) → Accentue les publications ayant plus de vues.
6. **Fonction « calculer\_score\_commentaires »**

*But :* Évaluer la popularité d’un commentaire en fonction de son engagement.

*Explication :*

1. Score des likes :
   * Somme des followers des utilisateurs qui ont liké, pondéré par 0.2.
   * Utilisation de calculer\_nombre\_followers(i.id\_utilisateur) pour prendre en compte la popularité des likers.
2. Score des réponses :
   * Nombre de réponses au commentaire, grâce à un COUNT(\*)
3. Score des likes sur les réponses :
   * Pareil que les likes normaux mais pondéré par 0.1.
   * Utilisation d’une requête imbriquée pour prendre l’interaction de type ‘like’ sur un commentaire
4. Score final = 1 + (score\_likes \* 0.5) + (score\_reponses \* 0.5) + (score\_likes\_reponses \* 0.25)
   * Un commentaire obtient au moins 1 point par défaut.
   * Les likes et réponses contribuent au score total.
5. **Fonction « calculer\_score\_likes »**

*But :* Calculer le score basé sur les likes d'une publication.

*Explication :*

1. Récupère le nombre total de likes sur la publication.
2. Utilise LOG(10, calculer\_nombre\_followers(i.id\_utilisateur) + 1) pour pondérer l’impact des utilisateurs avec beaucoup de followers.
3. **Fonction « calculer\_score\_partages »**

*But :* Mesurer l’impact des partages d’une publication.

*Explication :*

1. Récupère tous les partages d’une publication.
2. Chaque partage donne 0.5 point de base.
3. Ajoute 0.05 point par follower du partageur (calculer\_nombre\_followers(i.id\_utilisateur) \* 0.05).
4. **Fonction « calculer\_score\_engagement »**

*But* **:** Calculer un score global d’engagement basé sur le watchtime, les likes, les commentaires et les partages.

*Explication* **:**

1. Récupère les scores individuels :
   * score\_watchtime (pondéré à 35%)
   * score\_likes (25%)
   * score\_partages (15**%**)
2. Calcule du score total des commentaires (pondéré à 25%), grâce à la fonction calculer\_score\_commentaires(i.id\_interaction) codée précédemment.
3. Calcule un score final pondéré.
4. **Fonction « calculer\_nombre\_followers »**

*But* **:** Récupérer le nombre de followers d'un utilisateur.

*Explication* **:** Compte le nombre d’utilisateurs connectés à l’utilisateur donné dans la table connecter.

1. **Requêtes pour l'analyse des publications**
   1. **Classement des thèmes par engagement**

SELECT t.nom AS Groupe,

SUM(calculer\_score\_engagement(p.id\_publication)) AS score\_engagement\_total

FROM publication p

JOIN theme t ON p.id\_theme = t.id\_theme

GROUP BY t.nom

ORDER BY score\_engagement\_total DESC;

*But* **:** Trouver quels thèmes génèrent le plus d’engagement.  
*Explication***:** Regroupe les publications par thème et additionne leurs scores.

* 1. **Classement des types de publication**

SELECT p.type\_publication,

COUNT(p.id\_publication) AS nombre\_publications,

SUM(calculer\_score\_engagement(p.id\_publication)) AS engagement\_total,

AVG(calculer\_score\_engagement(p.id\_publication)) AS engagement\_moyen,

(SUM(calculer\_score\_engagement(p.id\_publication)) / COUNT(p.id\_publication)) \*

(1 + LOG(10, COUNT(p.id\_publication))) AS score\_final

FROM publication p

GROUP BY p.type\_publication

ORDER BY score\_final DESC;

*But* **:** Déterminer letype de publication le plus engageant.  
*Explication* : Calcule un score final en combinant :

* Le total d’engagement.
* L’engagement moyen.
* Un facteur de correction basé sur le nombre de publications pour éviter que les types avec peu de posts soient sous-représentés.

**Question 3 : Ajout et intégration d'un nouvel utilisateur**

*Objectif* **:** Créer un utilisateur et lui suggérer des amis, des groupes et des publications pertinentes.

*Etudiant* : Loïc Giuntini

* + 1. **Fonction « ajouter\_utilisateur »**

*But* : rajoute un utilisateur dans la table utilisateur avec les données mise en paramètre

*Explication* :

1. Crée un nouvel\_id INT qui sera l’id\_utilisateur
2. Crée un nouvel utilisateur dans la base de données avec l’id, son nom, son email et la date d’inscription
3. Renvoie le nouvel\_id crée
   * 1. **Fonction « suggerer\_ami »**

*But* : suggère des amis un nouvel utilisateur en fonction de son nom et son email

*Explication* :

1. Récupère le nouvel\_id et l’utilise pour récupérer le nom et l’email du nouvel utilisateur
2. Récupère le nom et l’email des autres utilisateurs
3. Compare le nom du nouvel utilisateur et des autres OU leur email en parsant les mails
4. Trie les résultats par ordre décroissant en fonction du nombre de connexion des autres utilisateurs et n’en donne que 5
   * 1. **Fonctions « suggerer\_groupes\_populaires » et « suggerer\_publication »**

*But* : renvoyer une table de groupe/publication populaire pour le nouvel utilisateur

*Explication* :

1. Récupère le nouvel\_id de l’utilisateur puis les différents groupes existant possédant un thème ou les différentes publications existantes
2. Trie les groupes/publication en fonction de leur score d’engagement par ordre décroissant et renvoie les 5 premiers
3. **Fonction « intégrer\_utilisateur »**

*But* : rajoute un utilisateur à la base de données et lui suggère des amis, des publications et des groupes

*Explication* :

1. Récupère le nouvel\_id grâce à la fonction 1 en attribuant un nom et un email à l’utilisateur
2. Utilise la fonction 2, 3 et 4 pour lui suggérer des amis, des groupes et des publications
3. Renvoie toutes les informations sous forme de table

**Question 4 : Calcul de l'influence d'un utilisateur**

*Objectif* : Mesurer l'impact d'un utilisateur sur la plateforme en fonction de son activité.

*Etudiant* : Loïc Giuntini

* + 1. **Fonction « calculer\_influence\_utilisateur »**

*But* : calcule l’influence d’un utilisateur et la renvoie sous forme de décimal

*Explication* :

1. Récupère l’id de l’utilisateur voulu dans les paramètres
2. Calcule le score des publications de l’utilisateur avec un poids de 0.5 en récupérant le nombre total de publication et en multipliant par 0.5 (COUNT(\*) \* 0.5)
3. Calcule ensuite le score des commentaires avec un poids de 0.3 en récupérant le nombre total de commentaire et en le multipliant par 0.3 (COUNT(\*) \* 0.3)
4. Calcule le score des partages de l’utilisateur avec un poids de 0.2 en récupérant le nombre total de commentaire et en le multipliant par 0.2 (COUNT(\*) \* 0.2)
5. Calcule enfin le score global en faisant la somme de tous les scores en définissant a 0 les score NULL grâce à COALESCE (publication \* 0.5 + commentaire \* 0.3 + publication \* 0.2)

*Requête* :

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

Description générée automatiquementUne image contenant texte, capture d’écran, Police, nombre

Description générée automatiquement

**Explication** :

Cette requête utilise la fonction décrite précédemment afin d’obtenir les 5 utilisateur les plus influents de notre jeu de données parmi les différents groupes à thème grâce aux LEFT JOIN pour récupérer les id\_utilisateur des publications ainsi que l’id des groupes. Cela implique qu’une personne peut apparaitre deux fois si elle est la plus influente dans deux groupes différents.

**Question 5 : Recommandations basées sur l'activité récente**

*Objectif* **:** Recommander des groupes pertinents à un utilisateur en fonction de son activité récente.

*Etudiant* : Soleia Marietti

* + 1. **Fonction « themes\_activite\_recente »**

*But* : Récupérer les thèmes associés aux publications avec lesquelles un utilisateur a interagi au cours des sept derniers jours

*Explication* :

* Enregistrement des actions effectuées par les utilisateurs sur des publications
* Identification des sujets ayant récemment suscité l’intérêt de l’utilisateur, grâce au JOIN avec la table publication
* Sélectionne donc les thèmes des publications avec lesquelles l’utilisateur a interagi en s’assurant que ces interactions ont eu lieu dans les sept derniers jours grâce à une condition sur la date d’interaction
* Utilisation de DISTINCT pour éliminer les doublons et ne retourner que les thèmes uniques
  + 1. **Fonction « recommander\_groupes\_recent »**

*But* : Suggérer à un utilisateur des groupes récents susceptibles de l’intéresser en se basant sur ses groupes actuels et les tendances de la plateforme

*Explication* :

* Identification des thèmes des groupes auxquels appartient l’utilisateur, donc ses centres d’intérêts
* Sélectionne une liste de groupes récemment créés, obtenue via la fonction recent\_group() permettant d’extraire les groupes nouvellement rejoint par l’utilisateur

La première approche de recommandation consiste à suggérer des groupes récents ayant un thème parent commun avec ceux des groupes déjà rejoints par l’utilisateur. L’idée est que si un utilisateur s’intéresse à un certain type de contenu, il pourrait être intéressé par des groupes abordant des sous-thèmes liés. Les groupes ainsi identifiés sont pondérés en fonction du nombre de membres qu’ils comptent, ce qui sert d’indicateur de popularité.

La deuxième approche propose des groupes récents ayant un thème apparenté mais différent de ceux déjà suivis par l’utilisateur. Contrairement à la première approche, ici l’accent est mis sur des thèmes connexes plutôt que strictement parentaux. Cette logique permet d’élargir la portée des suggestions tout en restant pertinente. Comme pour la première approche, un score est attribué en fonction du nombre de membres dans ces groupes.

Enfin, une dernière méthode de sélection repose sur les groupes récents ayant un fort engagement. Pour cela, la fonction analyse les publications des groupes récents et utilise calculer\_score\_engagement().de la question 2.

Les résultats de ces trois méthodes sont ensuite combinés via des jointures en UNION ALL et sont triés par score décroissant. Les cinq groupes ayant obtenu les scores les plus élevés sont finalement retournés, garantissant ainsi une sélection équilibrée entre pertinence thématique et activité sociale.

* + 1. **Fonction « recommander\_connexions\_globale »**

*But* : Suggérer des utilisateurs à ajouter en connexion en combinant plusieurs sources d’informations

*Explication* :

* Récupérer une première liste d’utilisateurs basée sur les interactions avec des publications communes, attribution d’un score en fonction du nombre de fois où l’utilisateur cible et les autres ont interagi avec le même contenu, pondéré par un facteur de 0.5 pour refléter l’importance de cette similarité
* Une seconde liste est générée en analysant les groupes communs, en appliquant cette fois un poids de 0.3, cela permet de privilégier les utilisateurs qui évoluent dans des cercles similaires, bien que l’appartenance à un même groupe n’implique pas forcément des interactions directes
* Une troisième source de recommandations est exploitée en prenant en compte les contacts récents, à qui un score fixe de 0.2 est attribué

Une fois ces trois ensembles de connexions obtenus, ils sont fusionnés pour éviter les doublons, et les scores sont additionnés par utilisateur. Le classement final est effectué par ordre décroissant de score ‘DESC’, garantissant que les utilisateurs ayant le plus de connexions potentielles avec l’utilisateur cible sont prioritaires. La fonction retourne les cinq meilleures suggestions, ce qui permet d’offrir des recommandations adaptées sans surcharger l’utilisateur avec trop de choix. L’approche adoptée ici assure une mise en relation pertinente en équilibrant interactions directes, contexte social et relations récentes.

**Question 6 : Publication avec le plus grand nombre d’interactions**

*Objectif***:** Calculer le nombre total de vues, de likes et de partages pour chaque publication.

*Etudiant* : Lucie Faure-Beaulande

* + 1. **Calcul du nombre de vues**

WITH VueCounts AS (

SELECT p.id\_publication, COUNT(w.temps\_debut) AS total\_vues

FROM PUBLICATION p

LEFT JOIN WATCHTIME w ON p.id\_publication = w.id\_publication

GROUP BY p.id\_publication )

*But* : Compter combien de fois une publication a été visionnée.

*Explication* : Compte les entrées de la table WATCHTIME et utilise LEFT JOIN pour s’assurer que les publications avec aucunes vues sont incluses.

* + 1. **Calcul du nombre de likes**

LikeCounts AS (

SELECT p.id\_publication, COUNT(i.id\_interaction) AS total\_likes

FROM PUBLICATION p

LEFT JOIN INTERAGIR i ON p.id\_publication = i.id\_publication

WHERE i.type\_interaction = 'like'

GROUP BY p.id\_publication ),

*But* : Compter le nombre total de likes par publication.

*Explication* : Filtre le type d’interaction et inclut les publications qui n’ont pas reçues de like grâce à LEFT JOIN.

* + 1. **Calcul du nombre de partages**

PartageCounts AS (

SELECT p.id\_publication, COUNT(pa.id\_partage) AS total\_partages

FROM PUBLICATION p

LEFT JOIN PARTAGE pa ON p.id\_publication = pa.id\_publication

GROUP BY p.id\_publication )

*But* : Compter le nombre de partages de chaque publication.

*Explication* : Inclut les publications non partagées grâce à LEFT JOIN, et compte les partages avec COUNT(pa.id\_partage).

* + 1. **Fusion des résultats et tri final**

SELECT p.id\_publication, p.contenu,

COALESCE(vc.total\_vues, 0) AS total\_vues,

COALESCE(lc.total\_likes, 0) AS total\_likes,

COALESCE(pc.total\_partages, 0) AS total\_partages,

(COALESCE(vc.total\_vues, 0) + COALESCE(lc.total\_likes, 0) +

COALESCE(pc.total\_partages, 0)) AS total\_interactions

FROM PUBLICATION p

LEFT JOIN VueCounts vc ON p.id\_publication = vc.id\_publication

LEFT JOIN LikeCounts lc ON p.id\_publication = lc.id\_publication

LEFT JOIN PartageCounts pc ON p.id\_publication = pc.id\_publication

ORDER BY total\_interactions DESC;

*But* : Fusionner les résultats des trois calculs

*Explication* :

* COALESCE est utilisé pour remplacer les valeurs NULL par 0.
* LEFT JOIN associe les statistiques de vues, likes et partages à chaque publication.

**Question 7 : Utilisateurs avec le plus d’interactions en dehors de leur cercle direct**

*Objectif***:** Mettre en place une analyse des utilisateurs hors cercle, c'est-à-dire des utilisateurs actifs sur la plateforme mais qui ne sont connectés à personne via des relations directes (connecter).

*Une image contenant texte, capture d’écran, ligne, Police

Description générée automatiquementUne image contenant texte, capture d’écran, nombre, Police

Description générée automatiquementEtudiant* : Benjamin Parciany

Une image contenant texte, capture d’écran, Police

Description générée automatiquement Une image contenant texte, capture d’écran, Police, ligne

Description générée automatiquement

* + 1. **Identifier l’engagement des utilisateurs en dehors de leur cercle d’amis**

*Explication :*

* Jointure entre UTILISATEUR, INTERAGIR ET CONNECTER
  + 1. **Fonctions « SUM(calculer\_score\_engagement(i.id\_publication)) » et « analyse\_utilisateurs\_hors\_cercle »**

*Explication* :

* Calcule le score total d’engagement pour tous les utilisateurs
* Compare les 20 utilisateurs les plus engagés hors cercle avec la moyenne de la population générale selon plusieurs indicateurs: l'ancienneté moyenne, le nombre moyen d'amis, la fréquence des interactions, l'appartenance à des groupes et le nombre de publications
* Permet aussi de déterminer si les utilisateurs préfèrent interagir avec ces vidéos, du texte ou des images

**Description des jeux de données utilisés**

Lucie et Soleïa ont tenté de récupérer des jeux de données open source en ligne pour obtenir des informations plus concrètes à apporter au projet. Cependant, aucun jeu de donnée ne correspondait entièrement à nos besoins.

Des tentatives d’assemblages de plusieurs datasets ont été effectués, mais sans résultat cohérent. De plus, l’aspect concret était déjà perdu, du fait que les datasets associés n’étaient, en réalité, pas liés. Ils ne reflétaient donc pas la réalité. Nous avons donc jugé acceptable de générer certaines données par intelligence artificielle. La base de donnée a ensuite été reprise, corrigée et complétée par Loïc et Lucie pour s’assurer de la cohérence des données.

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

Description générée automatiquementUne image contenant texte, capture d’écran, nombre, Police

Description générée automatiquementUne image contenant texte, capture d’écran, nombre, Police

Description générée automatiquementUne image contenant texte, capture d’écran, nombre, Police

Description générée automatiquementLes jeux de données que nous avons utilisés pour chaque table ont été pensé afin que chaque élément soit relié à au moins un autre élément de la base de données. Prenons l’exemple d’une partie de nos tables ADHERER, GROUPE, THEME et PUBLICATION ci-dessous.

Nous avons implémenté des thèmes dans nos jeux de données que nous avons choisi arbitrairement correspondant aux sujets de sciences informatique les plus connus. Il est possible aussi que chaque thème puisse avoir un thème parent comme par exemple, les thèmes Data Science et Programmation qui héritent tout deux du thème Technologie.

Nous pouvons ici voir que les groupes contiennent ici plusieurs membres mais aussi que chaque groupe possède bien un thème unique. En revanche, les thèmes peuvent quant à eux être bel et bien affectés à plusieurs groupes différents.

**Analyse critique du déroulement du projet**

Le projet s’est déroulé en plusieurs étapes clés, avec une progression qui a nécessité des ajustements au fil de l’avancement.

**1. Compréhension du sujet et reformulation**

Dès le départ, la reformulation du sujet et les questions qui ont émergé de l’analyse initiale nous ont permis d’approfondir notre compréhension des enjeux du projet et de la base de données. Cette étape a été essentielle, car elle nous a aidés à identifier plusieurs points flous et à soulever des questions supplémentaires à implémenter.

**2. Conception du schéma Entité-Association (EA)**

Soleïa et Benjamin ont travaillé sur la conception du schéma entité-association. Sa réalisation s’est avérée relativement simple dans un premier temps. Cependant, lors de l’écriture des requêtes, il est apparu que certaines associations devaient être revues pour mieux prendre en compte certains aspects nécessitant plus de précisions.

**3. Mise en place de l’environnement de travail**

Loïc et Lucie ont ensuite créé la base de données et mis en place un dépôt GitHub afin de centraliser et partager efficacement les informations entre tous les membres du groupe. Cette organisation a grandement facilité la collaboration.

**4. Écriture des requêtes SQL**

L’écriture des requêtes s’est globalement bien déroulée, avec une bonne coordination entre les membres. Cependant, certaines questions ont nécessité un travail plus approfondi, notamment la question 6 et la question 2. Ces dernières ont exigé un ajustement de notre modèle de données et une réflexion plus poussée, notamment sur la gestion du watchtime et des partages de publications.

**Points forts et axes d’amélioration**

* Bonne organisation avec un partage des tâches et un dépôt centralisé sur GitHub.
* Compréhension approfondie du sujet grâce à une reformulation et une analyse initiale efficace.
* Flexibilité dans l’adaptation du schéma EA en fonction des besoins des requêtes.
* Amélioration possible dans l’anticipation des besoins du schéma EA, afin d'éviter des modifications en cours de route.
* Meilleure gestion du temps sur certaines requêtes complexes, notamment en identifiant plus tôt les éléments nécessitant un approfondissement.
* Choix du logiciel de gestion de base de données à faire en amont, pour éviter les problèmes d’installations

Dans l’ensemble, le projet a bien avancé grâce à une bonne collaboration et une analyse approfondie des besoins. Quelques ajustements en amont auraient pu fluidifier le processus, mais les difficultés rencontrées ont permis d’enrichir notre réflexion et d’améliorer notre approche de conception de bases de données.