|  |
| --- |
| EFREI Ing-1-NEW |
| Mini Projet Base de données |
| Partie 2 : Insertion & Interrogation des données |

|  |
| --- |
| Grégoire BELLEPERCHE & Thomas QUERREC  05/10/2025 |

Table des matières

[1) Contexte métier minimal et rôle utilisateur 2](#_Toc211810322)

[2) Questions métier → types de requêtes (ce qu’on cherche et pourquoi) 2](#_Toc211810323)

[3) Pourquoi ce scénario “marche” pour l’évaluation 4](#_Toc211810324)

Après création et insertion des données dans la base de donnée. On s’est demandé comment relier nos tables à un véritable besoin. Plutôt que d’enchaîner des requêtes “gratuites”, on a posé un rôle (utilisateur) et des questions métier. L’idée : si on comprend *pourquoi* on interroge, on écrit des requêtes plus simples, plus propres, et surtout alignées avec le cahier des charges (projections/sélections, agrégations, jointures variées, sous-requêtes).

# 1) Contexte métier minimal et rôle utilisateur

* Contexte narratif : le monde de Solarys (nos référentiels : royaumes, espèces, lignées, ressources, guildes, classes, etc. déjà modélisés/peuplés).
* Rôle utilisateur : l’Archiviste royal conseille le Roi. Il doit extraire des informations pour orienter trois décisions : (I) fiscalité/ressources, (II) capacité militaire/titres, (III) santé publique/expéditions.
* Principe : chaque requête répond à un *ordre du Roi* clairement énoncé en commentaire dans 4\_interrogation.sql. Nous avons gardé un style “RP” pour la vidéo mais une structure 100% SQL et vérifiable.

# 2) Questions métier → types de requêtes (ce qu’on cherche et pourquoi)

**A. Projections / Sélections (filtrer & lister vite)**

Problème : l’Archiviste doit rapidement repérer *qui taxer / qui surveiller*.  
Notre approche (5 requêtes) :

* Lister les royaumes pacifiques et trier par richesse (repérer les gros contributeurs potentiels).
* Trouver les métiers liés à la magie (repérage réglementaire via LIKE).
* Dresser la liste DISTINCT des rôles de classes (vue synthétique des profils de combat).
* Relever les ressources “magiques” (base pour taxes spécifiques).
* Repérer les royaumes “moyens” (borne basse/haute sans BETWEEN, pour montrer la maîtrise des opérateurs).

**B. Agrégations (mesurer & comparer)**

Problème : prendre des décisions sans chiffres n’a pas de sens.  
Notre approche (5 requêtes) :

* Moyenne des richesses (niveau macro).
* Compte des espèces sapientes (population “intelligente” à gouverner).
* Répartition des ressources par initiale avec GROUP BY + HAVING (simple, lisible, conforme aux consignes).
* Effectifs par guilde (filtrage >= 3 pour repérer les organisations actives).
* Nombre de classes par rôle (équilibre DPS/Tank/Soin/Support).  
  Idée clé : l’Archiviste obtient un tableau de bord minimal pour briefer le Roi, en s’appuyant sur nos PK/FK et index de jointure.

**C. Jointures (diversifier comme demandé)**

Problème : naviguer entre référentiels et relations n–n.  
Notre approche (5 requêtes) :

* INNER JOIN (Personnage↔Royaume) : registre d’origine.
* LEFT JOIN (Personnage↔Lignée) : montrer les persos sans lignée (données manquantes mais utiles).
* RIGHT JOIN (Ressource↔RoyaumeRessource↔Royaume) : rattacher disponibilité/importance par royaume.
* CROSS JOIN (Classe×Métier) : vision cartésienne contrôlée (utile pour matrices “profils × filières”).
* FULL OUTER JOIN (émulé) : UNION d’un LEFT et d’un RIGHT pour lister toutes les guildes et tous les membres, même isolés (MySQL ne fournit pas FULL OUTER natif, on documente la solution).

**D. Sous-requêtes (EXISTS / ANY / ALL, comparaisons à une moyenne)**

Problème : exprimer des règles “métiers” naturelles (existe/n’existe pas, au-dessus d’un seuil relatif).  
Notre approche (≥5 requêtes, on en a fait + pour couvrir large) :

* Royaumes > moyenne de richesse (comparaison à sous-requête + ORDER BY).
* NOT EXISTS : lignées sans descendant ; métiers non attribués (vérifications d’intégrité “métiers”).
* EXISTS : royaumes avec ressource rare ; personnages avec titre actif.
* Santé : liste des personnages malades (jointure + tri par maladie).
* NOT EXISTS : guildes sans bataille (alerte “inactivité”).
* ANY : armées au-dessus de la moyenne de leur propre royaume (comparaison corrélée).
* NOT EXISTS : expéditions sans membres (nettoyage “piquenique royal”).

# 3) Respect des consignes (Scénario)

* Il relie chaque requête à un besoin utilisateur explicite (le Roi/gouvernance), ce que le sujet demande : un scénario d’utilisation avec liste de données à extraire.
* Il couvre toutes les catégories imposées (≥5 chacune) et diversifie les jointures (INNER/LEFT/RIGHT/CROSS et FULL OUTER émulé).
* Il montre les bonnes pratiques SQL : DISTINCT à bon escient, LIKE pour masques, GROUP BY/HAVING, EXISTS/NOT EXISTS, comparaison à la moyenne, ALL/ANY pour exprimer des contraintes relatives, et tri quand l’usage décisionnel l’exige (ex. priorisation fiscale).