
8TRD157 – Base de données avancées

Travail #2 – Programmation sous Oracle

Objectifs

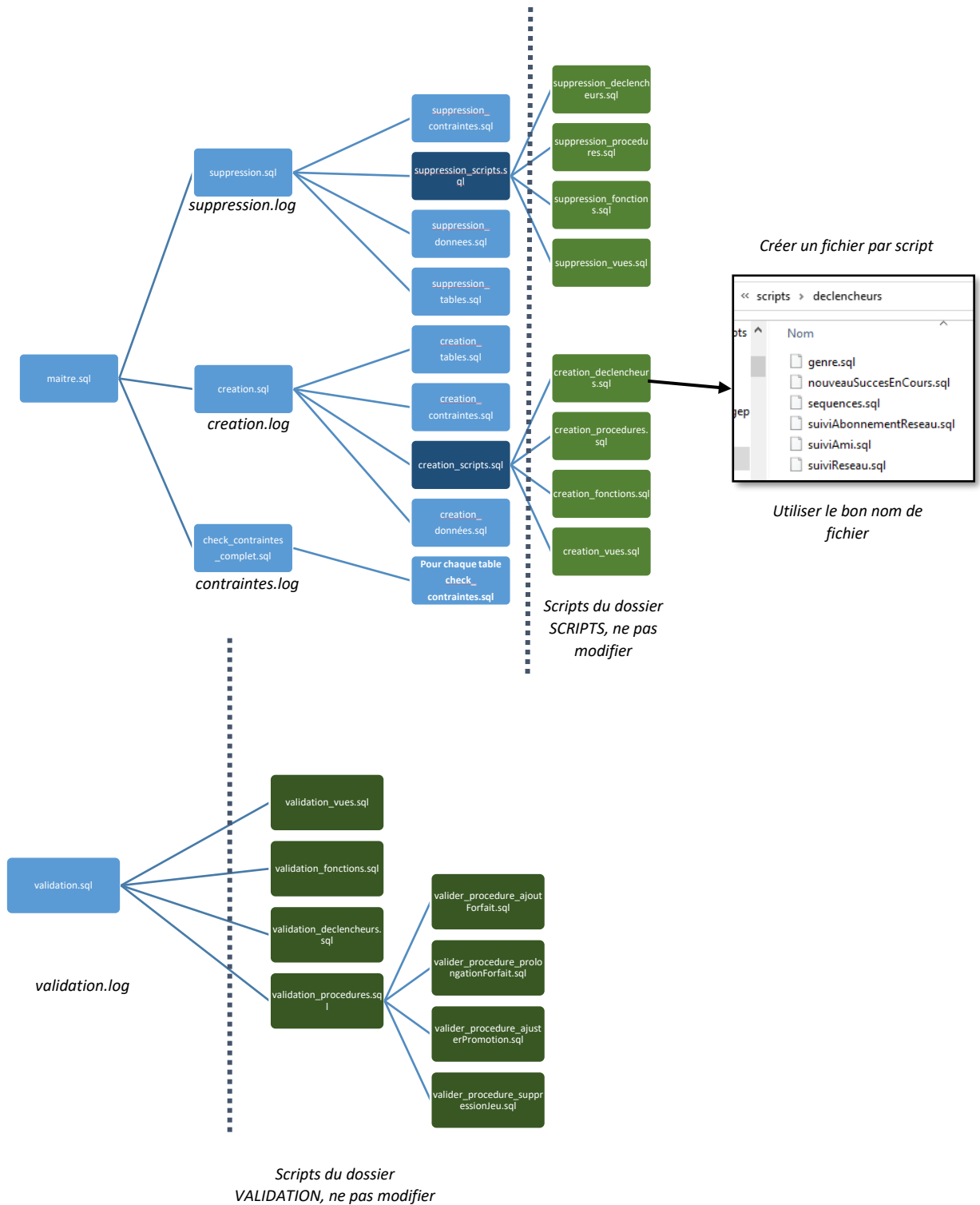
- À partir d'un modèle conception et d'un modèle logique d'entité-relation, être en mesure de comprendre une base de données
- À partir de scripts fournis, être en mesure de générer les tables, les contraintes et les données d'une base de données
- Programmer les scripts PL/SQL permettant la gestion de cette base de données

Consignes générales

- Individuellement ou en équipe de 2
- **Attention particulière au plagiat et à la fraude**
- Méthode de remise : tous les scripts compressés fonctionnels incluant les fichiers .log
- Pondération : 20%
- Date de remise : 9 novembre 2020

Mise en situation

- Utiliser les modèles conceptuels et logiques fournis pour comprendre la mise en situation
-
- Utiliser les scripts SQL fournis pour construire la base de données de départ
- Programmer les scripts demandés
 - Chaque script est indépendant et peut être réalisé dans le désordre.
 - Chaque script défini sa propre mise en situation
 - Utiliser les noms de script fournis (vous pouvez toutefois ajuster la syntaxe)
- Utiliser les scripts de validation fournis
 - Important car ils seront utilisés pour la correction (**voir page suivante**)
- Vous devrez retourner la structure de code, compressé, avec vos scripts « intégrés » à l'intérieur.
 - Le script « maitre.sql » doit fonctionner
 - Le script « validation.sql » doit fonctionner
- Attention, vous pouvez seulement « ajuster » les scripts dans les dossiers « scripts » et « validation » (par exemple, modifier avec vos propres noms de fichiers et d'objets)



Section #1 : Les procédures stockées

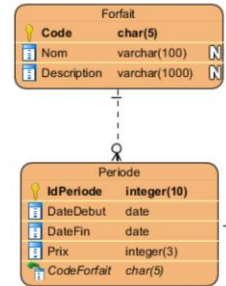
Procédure « ajoutForfait »

Paramètre(s)

- IN : Code, Nom, Description, Prix

Détails

- Insertion d'un forfait avec une date de début correspondant à la date du jour
- La durée d'un forfait est d'un an
- Prévoir un nom par défaut (ex : Forfait) en cas de valeur nulle



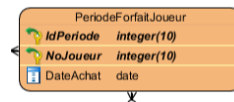
Procédure « prolongationForfait »

Paramètre(s)

- IN : CodeForfait
- OUT : Le nombre de joueurs touchés

Détails

- Tous les joueurs avec ce forfait actif ont une prolongation de deux mois (ajouter deux mois à la date d'achat)
- Tous les forfaits ont une durée d'un an à partir de la date d'achat



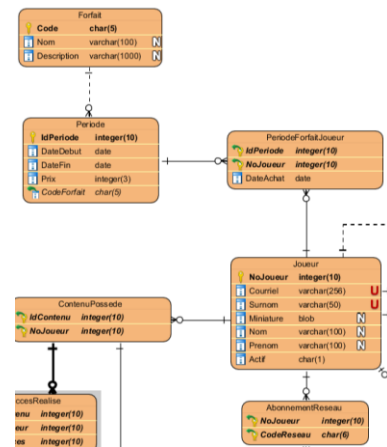
Procédure « ajusterPromotion »

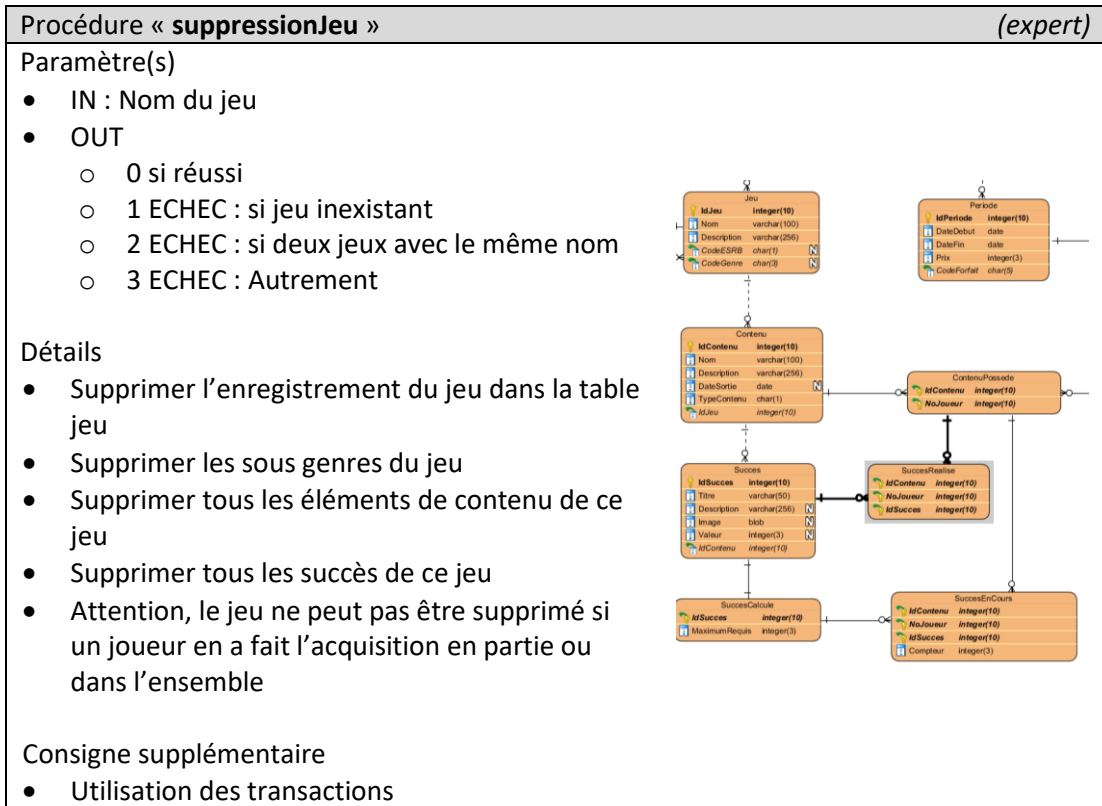
Paramètre(s)

- OUT : Code de forfait
- OUT : Nouveau prix (ou le prix courant si la valeur n'est pas mise à jour)

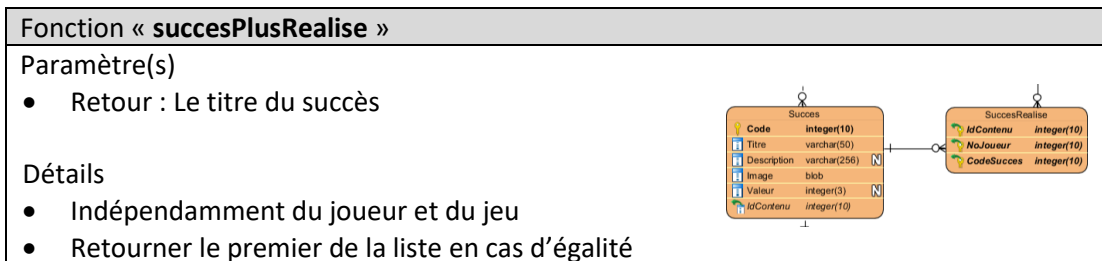
Détails

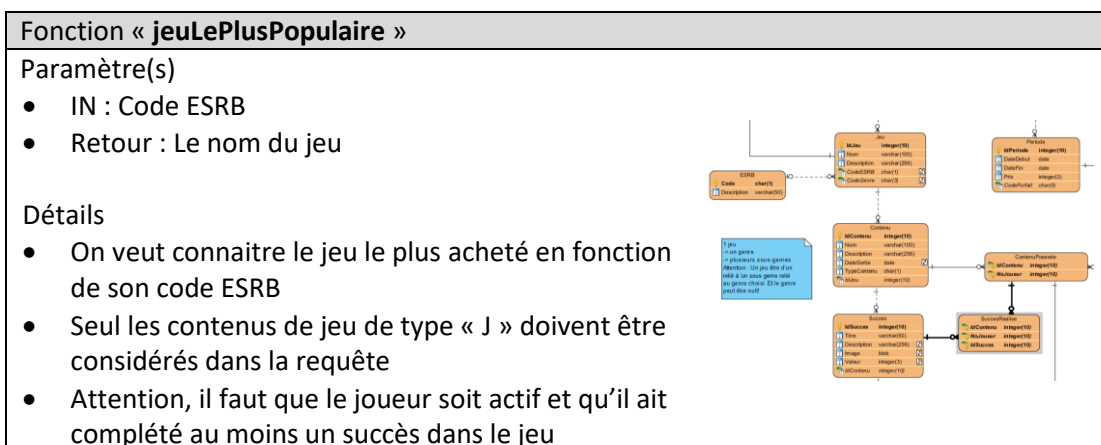
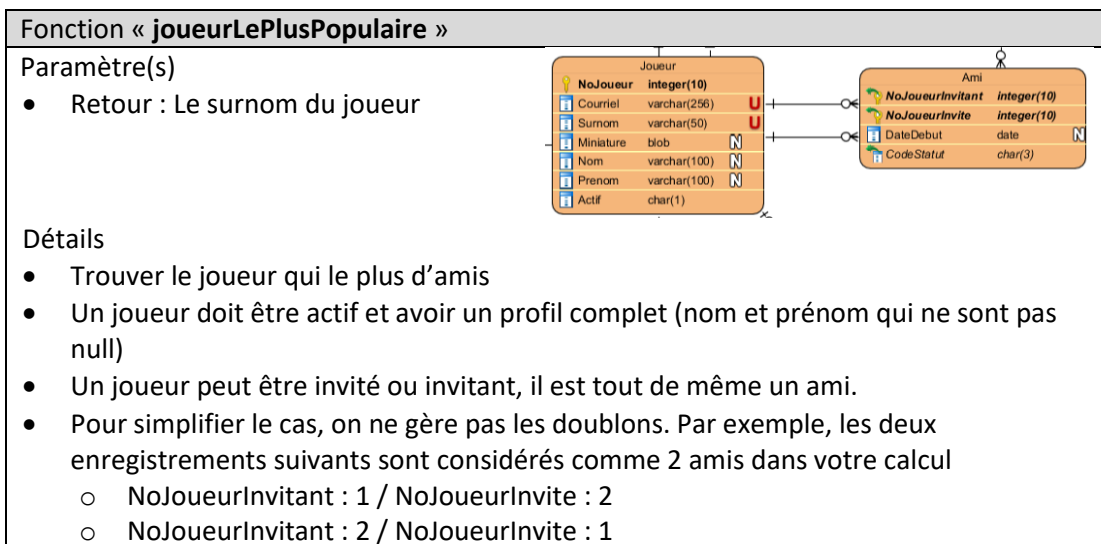
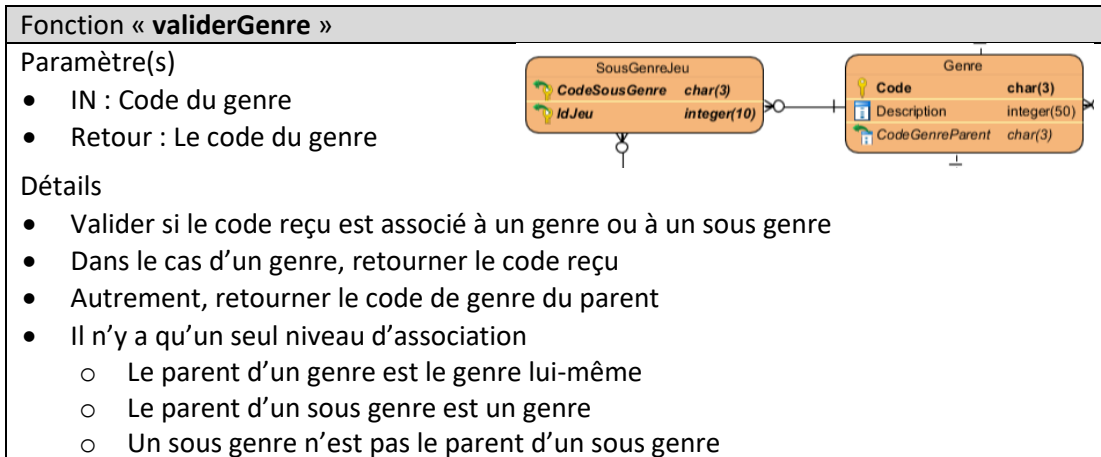
- Vérifier le forfait le plus utilisé par les joueurs abonnés au réseau actif le plus populaire (utiliser la table période)
- Pour simplifier le cas, ne pas considérer si la période ciblée est valide ou pas (date valide)
- Si la période n'est pas la plus chère, augmenter le prix de 10%.





Section #2 : Les fonctions



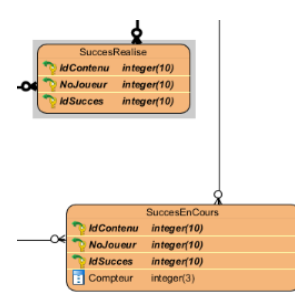


Section #3 : Les déclencheurs

Un lien utile pour cette section

http://www.java2s.com/Tutorial/Oracle/0560_Trigger/ifupdatingordeletingifinsertingorupdating.htm

Gestion des noAutomatique			
Table	Champ	Séquence	Nom de la procédure
JEU	IdJeu	JEU_SEQ	JEU_ID_TRG
CONTENU	IdContenu	CONTENU_SEQ	CONTENU_ID_TRG
SUCCES	IdSucces	SUCCES_SEQ	SUCCES_ID_TRG
PERIODE	IdPeriode	PERIODE_SEQ	PERIODE_ID_TRG
JOUEUR	NoJoueur	JOUEUR_SEQ	JOUEUR_ID_TRG
SUIVI_AMITIE	IdSuivi	SUIVI_AMITIE_SEQ	SUIVI_AMITIE_ID_TRG
SUIVI_RESEAU	IdSuivi	SUIVI_RESEAU_SEQ	SUIVI_RESEAU_ID_TRG

Déclencheur «NouveauSuccesEnCours_TRG »	
Critères de déclenchement	
<ul style="list-style-type: none"> Table : SUCCES_EN_COURS Évènement : Before insert 	
Détails	
<ul style="list-style-type: none"> Il faut empêcher de pouvoir ajouter un succès en cours qui a déjà été complété Réflexion : Il faut que l'insertion soit annulée 	
<p><i>Je voudrais seulement spécifier qu'il s'agit d'une mauvaise pratique mais que, quelque fois, on doit contourner les règles de bonnes pratiques. C'est pourquoi on réalise ce déclencheur à titre expérimental</i></p> <p>https://stackoverflow.com/questions/41092940/prevent-insert-trigger</p>	
	

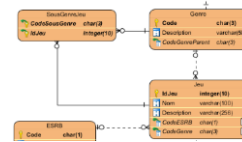
Déclencheur « **Genre_TRG** »

Critères de déclenchement

- Table : Jeu
- Évènement : update

Détails

- Il faut s'assurer que les genres/sous genres du jeu soient toujours conformes aux règles suivantes
 - Si on supprime/modifie le genre, on supprime aussi les sous genres
 - Attention : le genre doit être réellement être modifié dans le « update » (pas seulement mettre à jour avec la même valeur)

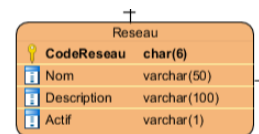
Déclencheur « **Suivi_Reseau_TRG** »

Critères de déclenchement

- Table : Reseau
- Évènement : After update/insert

Détails

- À chaque opération sur la table réseau, on copie une partie des informations de l'enregistrement touché dans la table de suivi (utiliser :NEW)
- Dans la table « SUIVI_RESEAU » on insère le code de réseau, la date de suivi (sysdate) et l'action (défini au point suivant)
- Pour l'insertion, le code est « CRE »
- Pour la mise à jour, si le réseau est actif, le code est « ACT »
- Pour la mise à jour, si le réseau est inactif, le code est « DES »



Petit détail

- Un petit « *problème* » de conception va empêcher la suppression des réseaux et des joueurs dès qu'une entrée sera dans la table de suivi (il y a des FK... Oupsss...)

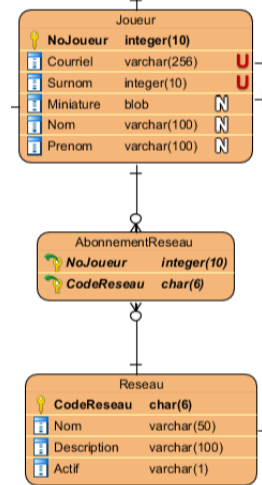
Déclencheur « **Suivi_Abonnement_Reseau_TRG** »

Critères de déclenchement

- Table : ABONNEMENT_RESEAU
- Évènement : After insert/delete

Détails

- À chaque opération sur la table AbonnementReseau, on copie une partie des informations de l'enregistrement touché dans la table de suivi
- Dans la table « SUIVI_RESEAU » on insère le code de réseau, la date de suivi (sysdate) et l'action (défini au point suivant) et le NoJoueur
- Pour l'insertion, le code est « ABO » (utiliser :NEW)
- Pour la suppression, le code est « DSA » (utiliser :OLD)

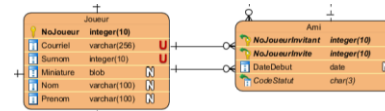
Déclencheur « **Suivi_Ami_TRG** »

Critères de déclenchement

- Table : AMI
- Évènement : After update/insert/delete

Détails

- À chaque opération sur la table AMI, on copie une partie des informations de l'enregistrement touché dans la table de suivi
- Dans la table « SUIVI_AMITIE » on insère le NoJoueurInvitant, NoJoueurInvite, la date de suivi (sysdate) et le code de statut
- Pour l'insertion et la mise à jour, utiliser :NEW
- Pour la suppression, utiliser :OLD



Section #4 : Les vues

Vue « MesSucces »

Les champs

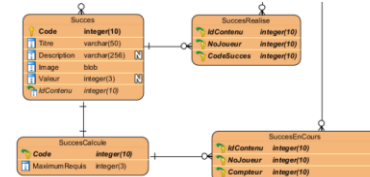
- SuccèsRealisé (Titre, Description, Valeur, NoJoueur)
- SuccèsEnCours (Titre, Description, Valeur, NoJoueur)

Détails

- Cette vue ne fait pas de distinction entre les types de succès

Suggestion

- Utiliser une UNION



Vues de « Suivi »

Les tables de suivis sont couvertes par trois vues distinctes

Vue « **suiviReseau** »

- Table source : SUIVI_RESEAU (avec jointure sur RESEAU)
- Fourni des informations de suivi sur les entrées concernant seulement les réseaux (donc dont le NoJoueur est null)
- Les champs : Moment, Action, CodeReseau, Reseau.NomReseau
- Trier les entrées en fonction du « moment »

Vue « **suiviReseauJoueur** »

- Table source : SUIVI_RESEAU (avec jointure sur JOUEUR)
- Fourni des informations de suivi sur les entrées concernant les joueurs (donc dont le NoJoueur n'est pas null)
- Les champs : Moment, Action, CodeReseau, Joueur.Surnom
- Trier les entrées en fonction du « moment »

Vue « **suiviAmitiInvitation** »

- Table source : SUIVI_AMITIE (avec jointure sur JOUEUR)
- Fourni des informations de suivi sur les joueurs « invitants »
- Les champs : Moment, Surnom du joueur invitant, NoJoueurInvitant, NoJoueurInvite, DateSuivi, CodeStatut
- Trier les entrées en fonction du « moment »



Critères d'évaluation

Critères	Pondération
Les procédures stockées	25
Les fonctions	25
Les déclencheurs	25
Les vues	25
TOTAL	100