8TRD157 – Base de données avancées Travail #2 – Programmation sous Oracle

Objectifs

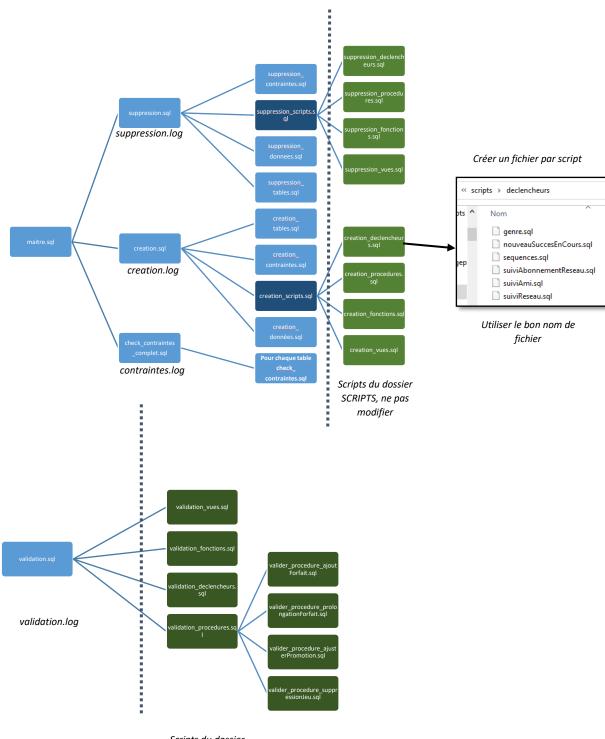
- À partir d'un modèle conception et d'un modèle logique d'entité-relation, être en mesure de comprendre une base de données
- À partir de scripts fournis, être en mesure de générer les tables, les contraintes et les données d'une base de données
- Programmer les scripts PL/SQL permettant la gestion 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
 - O Chaque script est indépendant et peut être réalisé dans le désordre.
 - o Chaque script défini sa propre mise en situation
 - o Utiliser les noms de script fournis (vous pouvez toutefois ajuster la syntaxe)
- Utiliser les scripts de validation fournis
 - o 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)



Scripts du dossier VALIDATION, ne pas modifier

Section #1 : Les procédures stockées

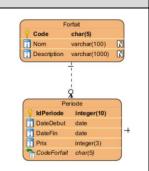
Procédure « ajoutForfait »

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

PeriodeForfaltJoueur IdPeriode Integer(10) NoJoueur Integer(10) DateAchat date

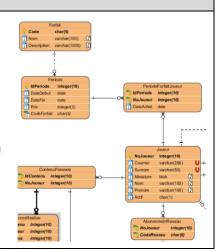
Procédure « ajusterPromotion »

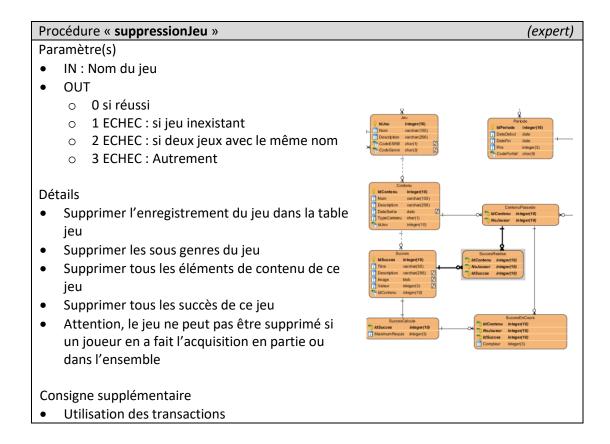
Paramètre(s)

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

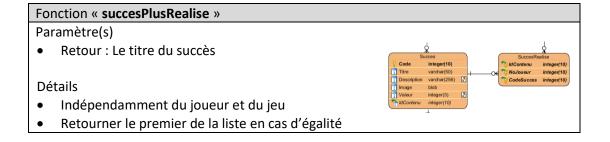
Détails

- 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



SousGenreJeu
CodeSousGenre char(3)

integer(10)

integer(50)

Description

CodeGenreParent char(3)

Fonction « validerGenre »

Paramètre(s)

- IN: Code du genre
- Retour : Le code du genre

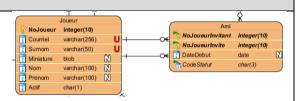
Détails

- Valider si le code reçu est associé à un genre ou à un sous genre
- Dans le cas d'un genre, retourner le code reçu
- Autrement, retourner le code de genre du parent
- Il n'y a qu'un seul niveau d'association
 - o Le parent d'un genre est le genre lui-même
 - o Le parent d'un sous genre est un genre
 - o Un sous genre n'est pas le parent d'un sous genre

Fonction « joueurLePlusPopulaire »

Paramètre(s)

• Retour : Le surnom du joueur



Détails

- Trouver le joueur qui le plus d'amis
- Un joueur doit être actif et avoir un profil complet (nom et prénom qui ne sont pas null)
- Un joueur peut être invité ou invitant, il est tout de même un ami.
- Pour simplifier le cas, on ne gère pas les doublons. Par exemple, les deux enregistrements suivants sont considérés comme 2 amis dans votre calcul
 - NoJoueurInvitant: 1 / NoJoueurInvite: 2
 - NoJoueurInvitant : 2 / NoJoueurInvite : 1

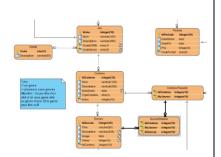
Fonction « jeuLePlusPopulaire »

Paramètre(s)

- IN : Code ESRB
- Retour : Le nom du jeu

Détails

- On veut connaître le jeu le plus acheté en fonction de son code ESRB
- Seul les contenus de jeu de type « J » doivent être considérés dans la requête
- Attention, il faut que le joueur soit actif et qu'il ait complété au moins un succès dans le jeu



Section #3: Les déclencheurs

Un lien utile pour cette section

http://www.java2s.com/Tutorial/Oracle/0560 Trigger/ifupdatingordeletingifinsertingorup dating.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

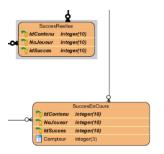
Table : SUCCES_EN_COURSÉvènement : Before insert

Détails

- Il faut empêcher de pouvoir ajouter un succès en cours qui a déjà été complété
- Réflexion : Il faut que l'insertion soit annulée

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

 $\underline{\text{https://stackoverflow.com/questions/41092940/prevent}}\\ -\text{insert-trigger}$



Déclencheur « Genre_TRG »

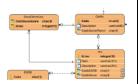
Critères de déclenchement

• 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 »

Reseau CodeReseau char(6) Nom varchar(50) Description varchar(100) Actif varchar(1)

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 integer(10) Table: ABONNEMENT RESEAU Courriel Surnom integer(10) Évènement : After insert/delete Miniature varchar(100) varchar(100) Détails • À chaque opération sur la table AbonnementReseau, on copie une partie des informations de l'enregistrement touché dans NoJoueur integer(10 la table de suivi CodeReseau char(6) 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) Nom varchar(50) Description varchar(100) Pour la suppression, le code est « DSA » (utiliser :OLD) Actif varchar(1)

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 »

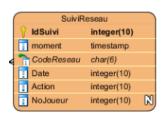
- Table source : SUIVI_RESEAU (avec jointure sur RESEAU)
- Fourni des informations de suivi sur les entrées concernant seulement les réseaux (donc dont le NoJoueur est null)
- Les champs : Moment, Action, CodeReseau, Reseau.NomReseau
- Trier les entrées en fonction du « moment »

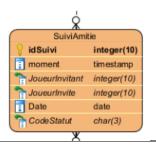
Vue « suiviReseauJoueur »

- 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 « suiviAmitieInvitation »

- 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