TEST SHEET

|  |  |
| --- | --- |
| Request ID | 001 |
| Request Title | Mis en place de la base de données |
| Applicant | Sarah Hautot |
| Request Date | 17/04/2024 |

Result:

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **TEST CONDUCTED BY** | **DATE / SPRINT** | **VERSION** | **RESULT (OK / NOK)** | | | |
| Internal | Rev. 1 | Rev. 2 | Rev. 3 |
| Tom Dunet | 17/04/2024 | Préparation | OK |  |  |  |
|  |  |  |  |  |  |  |
| Test performed in a second language | |  |  |  |  |  |

Summary:

|  |
| --- |
| Script de création de la base de données, des triggers.  Insertion de données test aussi |

Initial test conditions:

|  |
| --- |
| Base de données MYSQL vide |

Test steps:

|  | **ACTION** | **RESULT** | **COMMENT** | **TEST (OK / NOK)** | | | |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Internal | Rev. 1 | Rev. 2 | Rev. 3 |
| 1 | Connecter vous à votre serveur MYSQL depuis le dossier src/sql. | Vous êtes connecter à MYSQL. |  | OK |  |  |  |
| 2 | Créer la base de données ‘Maitai’ avec la commande « **CREATE DATABASE Maitai** » puis connecter vous avec « **USE Maitai** » | Vous êtes dans la base de données Maitai. |  | OK |  |  |  |
| 3 | Exécuter le fichier create.sql pour créer les tables avec la commande « **source create.sql** » | 14 lignes « Query OK» doivent apparaitre. Aucune erreur doit venir.  Les warnings sont normal. |  | OK |  |  |  |
| 4 | Exécuter le fichier fonction.sql pour créer les triggers sur les tables avec la commande « **source fonction.sql** » | 8 lignes « Query OK » doivent apparaitre. Aucune erreur doit venir.  Les warnings sont normal. |  | OK |  |  |  |
| 5 | Exécuter le fichier insert.sql pour insérer des données dans les tables avec la commande « **source insert.sql** » | Les données doivent s’insérer sans problème.  Les warnings sont normal. | Pour libProd était à 30 caractère, mais ce n’était pas assez donc passé à 255 | OK |  |  |  |
| 6 | La table UtilisateurDroit est rempli  (utiliser la commande **SELECT**) | La table UtilisateurDroit à un total de 10 lignes, dont deux utilisateur admin avec le droit 1. |  | OK |  |  |  |
| 7 | Désactiver le compte de admin1 avec la commande « **UPDATE Utilisateur SET actif = 0 WHERE login = 'admin1';**» | Le compte doit maintenant être désactiver (vérifiable avec la commande **SELECT**) |  | OK |  |  |  |
| 8 | Désactiver le compte de admin2 avec la commande « **UPDATE Utilisateur SET actif = 0 WHERE login = 'admin2';**» | Un message d’erreur doit apparaitre : « *ERROR 1644 (45000): Impossible de désactiver le dernier compte actif »* |  | OK |  |  |  |
| 9 | Regarder que la table Ticket comporte les données. | La table Ticket à un total de 12 lignes, concernant 4 clients. |  | OK |  |  |  |
| 10 | Vérifier que le client 4 ne soit pas présent. Puis, ajouter un ticket sans prix totaux fournis. « **INSERT INTO Ticket (idProd, idCli, qa) VALUES (1,4,3);** » | Le client 4 est maintenant présent et le prix du ticket est de 119.97€ pour le client 4 et le produit 1 (le prix total a été calculé automatiquement selon la quantité et son prix unitaire). | N'a pas l’attribue présent dans la table client. | OK |  |  |  |
| 11 | Ajouter un ticket sans prix totaux fournis. « **INSERT INTO Ticket VALUES (2,4,3, 10.00);** » | Le prix du ticket est de 10€, comme insérer par la base. |  | OK |  |  |  |
| 12 | Insérer un tuple dans le fichier historique pour simuler une exportation de fichier secu avec la commande « **INSERT INTO Historique (chemin, type, idCli) VALUES ('TestSecu','SECU', 4);** » | Les commandes du client 4 sont toujours présents. |  | OK |  |  |  |
| 13 | Insérer un tuple dans le fichier historique pour simuler une exportation de ticket avec la commande « **INSERT INTO Historique (chemin, type, idCli) VALUES ('TestSecu','TICKET', 4);** » | Les commandes du client 4 ont était supprimés. |  | OK |  |  |  |

Remarks:

|  |
| --- |
| Fonctionnelle.  Augmentation des caractères pour libProd. |