**Université Ibn Zohr**
**École Nationale des Sciences Appliquées - Agadir**

# RAPPORT DE PROJET

## Système de Gestion Portuaire Sécurisé

---

### Équipe de Projet

**Ce travail est présenté par :** AIT BOUHMAD Omar
ABOUELAZHAR Ibrahim
BOUTEFSOUT Abd elouahab
ALIATE Anass
**Encadré par :** Pr. IDRAIS Jaafar

SDBDIA -2
ENSA AGADIR 2025-26

# Table des matières

# Chapitre 1

# Introduction et Contexte

## 1.1 Contexte du Projet

La gestion d'un port moderne nécessite la coordination efficace de multiples processus métier interdépendants : planification des escales, opérations de manutention, facturation des services et gestion des ressources humaines. Face à l'augmentation du trafic maritime et à la complexité croissante des opérations portuaires, il devient indispensable de centraliser et d'automatiser ces processus au sein d'un système d'information intégré. Le Système de Gestion Portuaire (SGP) proposé dans ce projet est une réponse complète à ces enjeux, conçu comme un système d'information fédérateur capable de supporter l'ensemble des activités d'un port commercial moderne.

Les ports contemporains doivent gérer des flux de navires extrêmement hétérogènes (commerce, passagers, pêche), chacun présentant des contraintes techniques et opérationnelles distinctes. Au-delà de la simple planification, le système doit garantir la sécurité physique des opérations (vérification des profondeurs, des longueurs de quai), l'intégrité financière (traçabilité complète des services facturés) et la conformité des règles métier qui régissent les activités portuaires.

## 1.2 Objectifs du Projet

Le Système de Gestion Portuaire vise à :

— **Centraliser l'information** : Disposer d'une base de données unique et cohérente pour l'ensemble des activités portuaires, éliminant ainsi les silos d'information et les redondances.

— **Optimiser la planification** : Améliorer l'allocation des ressources (quais, personnel, équipements) en temps réel par la visualisation de l'occupation et des disponibilités.

— **Sécuriser les opérations** : Garantir le respect des contraintes physiques et réglementaires lors de l'accostage des navires, prévenant ainsi les risques d'échouage et les accidents.

— **Fiabiliser la facturation** : Assurer une traçabilité complète des prestations pour une facturation précise et justifiée, réduisant les litiges commerciaux.

— **Professionnaliser la gestion RH** : Centraliser les données du personnel et automatiser le calcul des rémunérations, améliorant la transparence et l'équité salariale.
— **Implémenter une gouvernance des données sécurisée** : Établir des rôles et des permissions granulaires pour protéger les informations sensibles (salaires, contrats, données financières).

## 1.3 Périmètre fonctionnel

Le système couvre quatre domaines fonctionnels majeurs, correspondant aux départements clés de l'organisation portuaire :

1. **Gestion des infrastructures et planification** (Manager Quai & Capitainerie)
2. **Gestion des opérations logistiques** (Responsable Opérations)
3. **Gestion financière et facturation** (Responsable Finance)
4. **Gestion des ressources humaines** (Responsable RH)

Ce document détaille la conception, l'implémentation et la sécurisation du système, en mettant l'accent sur la robustesse architecturale et le respect des principes de gouvernance des données.

# Chapitre 2

# Analyse et Conception (Modélisation)

Ce chapitre détaille la phase de conception du système de gestion portuaire. Il définit les contraintes métier qui régissent les processus et présente la structure de données permettant de soutenir ces opérations. La conception s'appuie sur une méthodologie rigoureuse combinant l'analyse fonctionnelle et la modélisation conceptuelle.

## 2.1 Règles de Gestion

Les règles de gestion (RG) assurent la cohérence et la sécurité du système au niveau de la base de données. Elles traduisent les contraintes métier imposées par la physique des opérations portuaires, la réglementation maritime et les politiques internes de l'organisation. Chaque règle est implémentée par une combinaison appropriée de contraintes de schéma, triggers et procédures stockées.

### 2.1.1 Infrastructure et Sécurité Maritime

Les règles relatives à l'infrastructure garantissent que les navires ne seront jamais assignés à des quais incompatibles avec leurs caractéristiques techniques, prévenant ainsi les risques d'échouage ou de collision.

- **RG-INFRA-01 :** Le tirant d'eau d'un navire doit être strictement inférieur à la profondeur du bassin du quai assigné pour éviter tout risque d'échouage. En pratique, une marge de sécurité de 0,5 mètre est systématiquement appliquée (tirant_eau < profondeur_bassin - 0,5m).
- **RG-INFRA-02 :** La longueur totale du navire ne peut excéder la longueur maximale autorisée pour le quai. Cette vérification prévient les problèmes d'amarrage et de manœuvre.
- **RG-INFRA-03 :** Un quai ne peut accueillir qu'une seule escale à la fois ; tout chevauchement temporel lors de la planification est interdit. Cette règle garantit une utilisation optimale des ressources sans surcharge opérationnelle.
- **RG-INFRA-04 :** La capacité tonnage du quai doit être suffisante pour supporter le poids maximal du navire en charge. Cette contrainte est vérifiée lors de la planification et respectée lors de chaque opération de chargement/déchargement.

### 2.1.2 Opérations Logistiques

Les règles opérationnelles régissent le flux des opérations de manutention et garantissent la cohérence temporelle et la conformité des effectifs.

- **RG-OPS-01 :** Une opération de manutention ne peut être enregistrée avant l'heure d'arrivée réelle du navire à quai (ATA : Actual Time of Arrival). Cette règle est critique pour la cohérence des données opérationnelles.
- **RG-OPS-02 :** Le poids cumulé des cargaisons ajoutées ne doit pas dépasser le tonnage maximum autorisé du navire de commerce. Chaque ajout de cargaison est soumis à une vérification cumulée.
- **RG-OPS-03 :** Seul un employé dont le statut est "Actif" peut être affecté à une opération portuaire. Les employés archivés ou en congé ne peuvent pas être sollicités.
- **RG-OPS-04 :** Une opération ne peut pas se dérouler après la date de départ réelle (ATD : Actual Time of Departure) du navire. Les navires partis ne peuvent plus faire l'objet d'opérations.
- **RG-OPS-05 :** Les cargaisons dangereuses doivent être signalées dans la description de l'opération. Un tag [DANGER] est automatiquement ajouté au libellé de l'opération en cas de manutention de marchandises dangereuses.

### 2.1.3 Finance et Ressources Humaines

Les règles financières et RH protègent l'intégrité comptable et la conformité des données salariales.

- **RG-FIN-01 :** Il est interdit de supprimer une facture ayant déjà fait l'objet d'un paiement définitif. Le système force l'archivage (soft delete) ou l'annulation plutôt que la suppression physique.
- **RG-FIN-02 :** Une escale dont le statut est "Annulée" ne peut en aucun cas faire l'objet d'une facturation. Cette règle prévient la facturation d'opérations non réalisées.
- **RG-FIN-03 :** Une facture payée ne peut pas être modifiée. Le statut "Payée" est terminal et offre une protection inviolable.
- **RG-FIN-04 :** Une facture dont l'échéance est dépassée bascule automatiquement en statut "En retard" pour faciliter le suivi des impayés.
- **RG-RH-01 :** Le taux horaire d'un employé recruté doit être une valeur strictement positive. Aucun salaire négatif ou nul n'est autorisé dans la base de données.
- **RG-RH-02 :** La structure hiérarchique interdit formellement qu'un employé soit défini comme son propre supérieur. Cette vérification prévient les incohérences hiérarchiques.
- **RG-RH-03 :** Un employé ayant un historique d'affectations ne peut pas être supprimé physiquement. L'archivage (soft delete) est obligatoire pour conserver la traçabilité.

## 2.2 Dictionnaire de Données

Les entités principales identifiées pour le système sont les suivantes. Ce dictionnaire servira de référence pour toutes les opérations ultérieures sur la base de données.

**NAVIRE** : Représente l'unité physique maritime avec ses caractéristiques techniques. Elle contient les données structurelles communes à tous les bâtiments (nom, numéro IMO, pavillon, longueur, tirant d'eau). Le système gère trois spécialisations de navires selon leur utilisation métier.

**QUAI** : Infrastructure d'accueil caractérisée par ses capacités techniques (profondeur, longueur, capacité tonnage). C'est une ressource finie et limitée du port.

**ESCALE** : Point central du modèle, elle lie un navire à une infrastructure (Quai) pour une durée déterminée. Elle suit le cycle de vie complet d'une visite portuaire, de la planification à la clôture.

**EMPLOYE** : Représente le personnel portuaire affecté aux tâches opérationnelles. Elle contient les données de gestion RH (statut, taux horaire, hiérarchie).

**OPERATION** : Activité concrète de manutention (chargement, déchargement, transfert). Elle est liée à une escale et peut être associée à une cargaison.

**CARGAISON** : Lot de marchandises transporté par un navire. Elle inclut un signalement de dangerosité pour les matières réglementées.

**FACTURE** : Document financier consolidant les montants dus pour les services rendus durant une escale. Elle agrège le coût matériel et le coût humain.

**AFFECTATION** : Relation d'association entre un employé et une opération, incluant le rôle joué et les heures travaillées. Elle constitue la base du calcul de la paie.

## 2.3 Modèle Conceptuel de Données

Le diagramme de classes suivant présente l'architecture logique globale du système portuaire. Ce modèle utilise la notation UML pour représenter les entités, leurs attributs et leurs relations.

### 2.3.1 Analyse des relations clés

— **Liaison Infrastructure :** L'entité `ESCALE` assure la jointure entre un `NAVIRE` et un `QUAI` spécifique pour une durée déterminée. C'est la relation centrale autour de laquelle s'organise tout le système.
— **Flux de Réservation :** Un navire initie une `RESERVATION` qui, après validation, devient une `ESCALE` planifiée. Cette transition crée un historique traçable des demandes de port.
— **Gestion Opérationnelle :** Chaque escale donne lieu à plusieurs `OPERATIONS`. L'entité `AFFECTATION` lie ces opérations aux `EMPLOYES`, enregistrant ainsi le travail fourni.
— **Agrégation Financière :** La `FACTURE` est le résumé financier d'une `ESCALE`, intégrant le coût des opérations et la rémunération du personnel affecté.

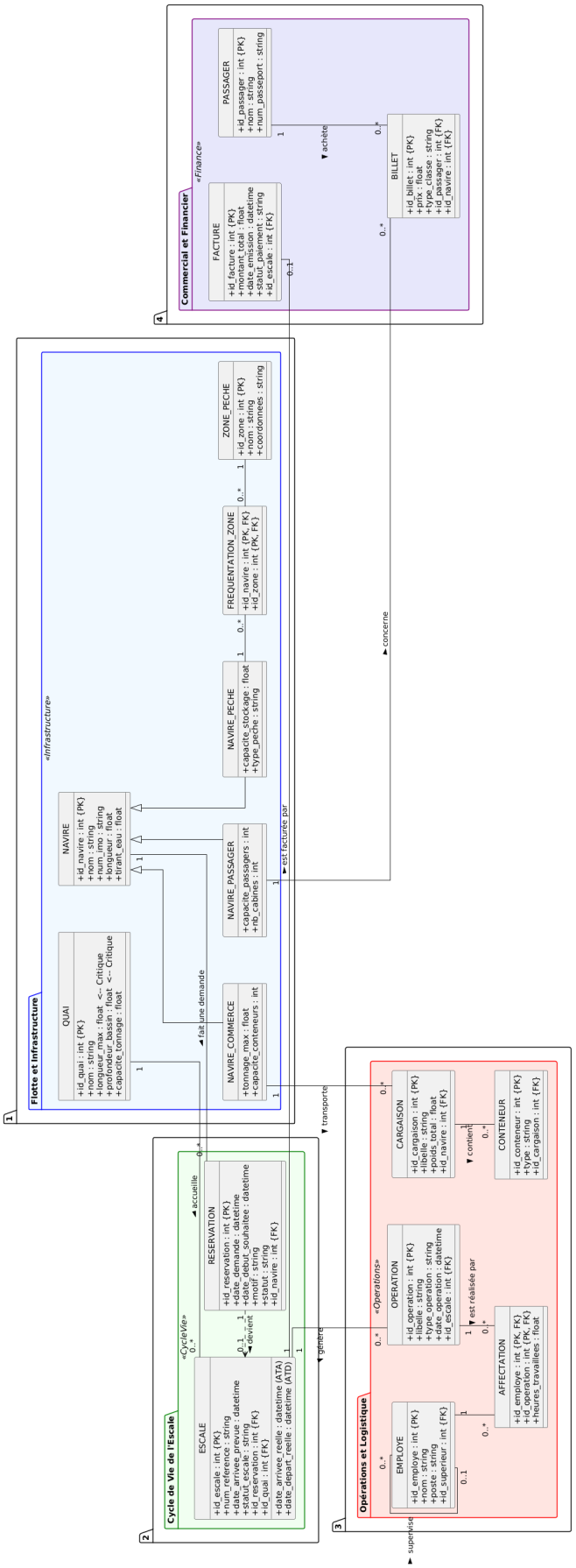


FIGURE 2.1 – Diagramme de classes UML du Système de Gestion Portuaire - Vue Détaillée des entités, attributs et relations

— **Spécialisation Métier :** Les trois types de navires (Commerce, Passager, Pêche) héritent des propriétés communes tout en ajoutant des attributs spécifiques à leur domaine.

## 2.4 Modèle Logique de Données

Le passage du modèle conceptuel (classes) vers le modèle relationnel s'appuie sur la gestion des clés primaires (PK) et étrangères (FK). Le tableau suivant résume la structure fondamentale du schéma relationnel.

| **Table** | **Clé Primaire (PK)** | **Clés Étrangères (FK)** |
|---|---|---|
| NAVIRE | `id_navire` | - |
| QUAI | `id_quai` | - |
| RESERVATION | `id_reservation` | `id_navire` |
| ESCALE | `id_escale` | `id_navire, id_quai, id_reservation` |
| OPERATION | `id_operation` | `id_escale, id_cargaison` |
| CARGAISON | `id_cargaison` | `id_navire` |
| CONTENEUR | `id_conteneur` | `id_cargaison` |
| AFFECTATION | (`id_employe, id_operation`) | `id_employe, id_operation` |
| FACTURE | `id_facture` | `id_escale` |
| BILLET | `id_billet` | `id_navire` |
| EMPLOYE | `id_employe` | `id_superieur (FK auto-référencée)` |
| NAVIRE_COMMERCE | `id_navire` (PK/FK) | `id_navire` |
| NAVIRE_PASSAGER | `id_navire` (PK/FK) | `id_navire` |
| NAVIRE_PECHE | `id_navire` (PK/FK) | `id_navire` |

TABLE 2.1 – Structure du Modèle Logique de Données - Entités et Relations

L'implémentation des spécialisations de navires s'effectue par des tables filles partageant la même clé primaire que la table mère `NAVIRE`. Cette approche, appelée "Table par Sous-classe", garantit l'intégrité référentielle tout en isolant les attributs spécifiques à chaque type de navire.

# Chapitre 3

# Implémentation de la Base de Données (DDL)

Ce chapitre expose les choix techniques lors de la création physique de la base de données Oracle. Il met l'accent sur la robustesse du schéma via des contraintes d'intégrité et une gestion optimisée des identifiants. L'implémentation physique traduit directement le modèle logique en constructs Oracle tout en intégrant les exigences non-fonctionnelles de performance et de sécurité.

## 3.1 Création des Tables et Contraintes

La définition des tables a été réalisée en intégrant des mécanismes de validation automatique pour garantir l'intégrité des données dès le niveau de stockage.

### 3.1.1 Gestion Automatique des Identifiants (Identity)

Pour simplifier la gestion des clés primaires et éliminer le besoin de déclencheurs manuels pour l'incrémentation, nous avons utilisé la fonctionnalité native d'identité offerte par Oracle :

- **Syntaxe :** `GENERATED BY DEFAULT AS IDENTITY` est appliquée aux colonnes de type `PK` comme `id_navire` ou `id_escale`. Cette syntaxe permet une génération automatique des identifiants tout en préservant la flexibilité lors des tests d'intégration.
- **Avantages :** (1) Élimination des risques de doublon par concurrence, (2) Simplification du code applicatif, (3) Conformité avec les standards ANSI SQL.
- **Impact :** Ce mécanisme permet au système de générer automatiquement les identifiants techniques tout en conservant la possibilité d'insérer des valeurs spécifiques lors de tests ou de migrations de données.

### 3.1.2 Contraintes d'Intégrité Complexes

Plusieurs types de contraintes ont été déployés pour sécuriser les données et prévenir les incohérences :

- **Contrainte UNIQUE :** Appliquée sur la colonne `num_imo` de la table `NAVIRE`. Le numéro IMO (International Maritime Organization) étant un identifiant maritime mondial unique, cette contrainte garantit qu'un navire physique ne peut être enregistré qu'une seule fois dans le système, prévenant ainsi les doublons.
- **Contrainte PRIMARY KEY :** Définit l'identifiant unique de chaque entité. Dans les tables de spécialisation (NAVIRE_COMMERCE, etc.), la PK sert également de FK vers la table parent pour garantir la cohérence de l'héritage.
- **Contraintes CHECK :**
  - Pour la table `CONTENEUR`, une contrainte limite les types autorisés (ex : 'DRY', 'REEFER', 'TANK') afin de respecter les normes de manutention du port et les standards ISO de conteneurisation.
  - Pour la table `EMPLOYE`, le `taux_horaire` est soumis à une contrainte `CHECK (taux_horaire > 0)` pour empêcher toute erreur de saisie salariale lors du recrutement.
  - Pour la table `ESCALE`, les statuts autorisés sont définis de manière exhaustive : 'PLANIFIEE', 'EN$_COURS$','$TERMINEE$','$ANNULEE$'.
  - Pour la table `FACTURE`, le montant total doit être non-négatif : `CHECK (montant_total` $\geq 0$).
- **Contraintes FOREIGN KEY :** Elles garantissent la cohérence référentielle. Par exemple, un `id_quai` dans la table ESCALE doit correspondre à un `id_quai` existant dans la table QUAI. Toutes les FK sont configurées avec `ON DELETE RESTRICT` pour prévenir les suppressions accidentelles d'entités référencées.

## 3.2 Héritage et Spécialisation

La diversité de la flotte portuaire est gérée par un mécanisme d'héritage de type "Table par Sous-classe". Cette approche permet de normaliser les données communes tout en isolant les attributs spécifiques à chaque métier maritime. Elle offre un excellent équilibre entre flexibilité et performance.

### 3.2.1 Implémentation de la Flotte

- **Table Mère (NAVIRE) :** Regroupe les attributs partagés par tous les bâtiments tels que le nom, l'IMO, le pavillon, la longueur et le tirant d'eau. Cette centralisation garantit une source unique de vérité pour les données structurelles communes.
- **Tables Filles :**
  - `NAVIRE_COMMERCE` : Spécifique à la logistique avec le `tonnage_max` (poids maximal en charge) et la `capacite_conteneurs` (nombre maximal de conteneurs). Ces navires représentent le cœur du trafic portuaire moderne.
  - `NAVIRE_PASSAGER` : Orientée transport de personnes avec `nb_cabines` et `capacite_passagers` Ces navires requièrent des procédures de sécurité renforcées.
  - `NAVIRE_PECHE` : Dédiée à l'activité halieutique avec `capacite_stockage` (volumes frigorifiques) et `type_peche` (chalut, filet, etc.).
- **Mécanisme de Cohérence :** Techniquement, les tables filles utilisent l'attribut `id_navire` à la fois comme Clé Primaire (PK) et comme Clé Étrangère (FK) vers la table mère. Cette double utilisation assure une intégrité référentielle stricte : chaque

navire ne peut appartenir qu'à une seule sous-classe, et tout enregistrement dans une table fille doit correspondre à un navire parent existant.

## 3.3 Indexation et Optimisation

Pour garantir des performances de lecture optimales lors des jointures fréquentes entre les modules, une stratégie d'indexation ciblée a été mise en place. Cette approche suit le principe de penser en termes de cas d'usage réels plutôt que de généraliser.

### 3.3.1 Index Automatiques

Oracle Database génère automatiquement des index B-tree sur chaque Clé Primaire (`PK`) et chaque contrainte `UNIQUE`. Ces index accélèrent les recherches unitaires critiques, notamment sur les identifiants de navires et d'escales. L'index sur `num_imo` en est un excellent exemple : chaque recherche d'un navire par son numéro IMO est servie en temps logarithmique.

### 3.3.2 Optimisation des Jointures Métier

Des index explicites ont été ajoutés sur les Clés Étrangères (`FK`) les plus sollicitées par les acteurs du système :

- **ESCALE (id_navire)** : Pour l'affichage instantané du planning d'un navire spécifique par le Manager Quai. Cette jointure est fréquente lors de la consultation de l'historique des escales d'un armateur.
- **AFFECTATION (id_employe)** : Crucial pour l'agrégation rapide des heures travaillées lors du calcul de la paie mensuelle par le Responsable RH. Les agrégatios `SUM(heures_travaillees)` bénéficient grandement de cet index.
- **OPERATION (id_escale)** : Indispensable pour la consolidation financière du coût total d'une escale par le Responsable Finance. Cette jointure synthétise les opérations pour produire les montants facturables.
- **ESCALE (id_quai)** : Permet l'interrogation rapide de l'occupation d'un quai sur une plage temporelle donnée, essentiel pour la planification prospective.

### 3.3.3 Considérations de maintenance

L'équipe d'administration de base de données doit surveiller régulièrement (mensuellement) les statistiques d'utilisation via `v$sql` et `dba_hist_sqlstat` pour identifier les chemins d'accès non optimisés. Des requêtes fréquentes sans support d'index devraient déclencher des investigations.

# Chapitre 4

# Architecture PL/SQL et Logique Métier

## 4.1 Introduction

Ce chapitre détaille l'intelligence métier du système de gestion portuaire. L'utilisation du langage PL/SQL a permis de déporter la logique de contrôle directement au sein du SGBD Oracle. Cette approche garantit une intégrité absolue des données, une sécurité accrue par la gestion des rôles et une centralisation des règles métier indépendamment des applications clientes. Les procédures stockées et les triggers constituent l'ossature comportementale du système.

## 4.2 Règles de Gestion (Business Rules)

Les règles de gestion constituent le socle de cohérence du système. Elles traduisent les contraintes physiques (quais), opérationnelles (manutention), financières (facturation) et humaines (RH). Cette section synthétise les principales règles implémentées.

### 4.2.1 Infrastructure et Quais

| Code | Règle | Implémentation |
|---|---|---|
| **RG-QUAI-01** | Compatibilité physique : longueur_navire ≤ longueur_max_quai | Fonction `FN_EST_QUAI_COMPATIBLE` + Trigger |
| **RG-QUAI-02** | Tirant d'eau ≤ (profondeur_bassin - 0.5m) | Trigger `TRG_SEC_VERIF_QUAI` - Erreur -20001 |
| **RG-QUAI-03** | Disponibilité temporelle (pas de chevauchement) | Fonction `FN_VERIFIER_DISPO_QUAI` |
| **RG-QUAI-04** | Traçabilité des modifications de planning | Trigger `TRG_AUDIT_PLANNING` |
| **RG-QUAI-05** | Référence unique automatique (ESC-YYYY-NNNN) | Trigger `TRG_GENERATE_REF_ESCALE` |

| Code | Règle | Implémentation |
|---|---|---|
| **RG-QUAI-06** | Capacité tonnage du quai suffisante | Procédure de validation lors de l'affectation |

TABLE 4.1: Règles de gestion - Infrastructure et Quais

### 4.2.2 Opérations et Logistique

| Code | Règle | Implémentation |
|---|---|---|
| **RG-OPS-01** | Création d'opération impossible si le navire n'est pas présent | Erreur -20002 si `ATA IS NULL` dans procédure |
| **RG-OPS-02** | Date opération $\geq$ date arrivée réelle | Trigger `TRG_OPS_DATE_COHERENCE` |
| **RG-OPS-03** | Poids total cargaison $\leq$ tonnage max navire | Fonction `FN_VERIFIER_POIDS` - Erreur -20003 |
| **RG-OPS-04** | Signalement automatique des cargaisons dangereuses | Trigger `TRG_OPS_SEC_DANGER` |
| **RG-OPS-05** | Employé actif obligatoire pour affectation | Erreur -20001 dans `AFFECTER_EMPLOYE` |
| **RG-OPS-06** | Interdiction d'opération après départ du navire | Trigger `TRG_OPS_DATE_COHERENCE` |

TABLE 4.2: Règles de gestion - Opérations Logistiques

### 4.2.3 Finance, Facturation et RH

| Code | Règle | Implémentation |
|---|---|---|
| **RG-FIN-01** | Pas de facture pour une escale annulée | Erreur -20010 dans `PR_CREER_FACTURE` |
| **RG-FIN-02** | Facture payée non supprimable | Trigger `TRG_SECURITE_FACTURE` - Erreur -20005 |
| **RG-FIN-03** | Facture payée non modifiable | Contrainte de statut terminal |
| **RG-FIN-04** | Facture en retard si échéance dépassée | Trigger `TRG_CHECK_RETARD_PAIEMENT` |
| **RG-RH-01** | Taux horaire strictement positif | Trigger `TRG_RH_INTEGRITE_EMPLOYE` - Erreur -20020 |

| Code | Règle | Implémentation |
|---|---|---|
| **RG-RH-02** | Interdiction de hiérarchie cyclique | Trigger `TRG_RH_INTEGRITE_EMPLOYE` - Erreur -20021 |
| **RG-RH-03** | Archivage obligatoire (Soft Delete) | Procédure `PR_ARCHIVER_EMPLOYE` |

TABLE 4.3: Règles de gestion - Finance et RH

## 4.3 Structure en Packages

L'architecture logicielle regroupe les fonctionnalités par domaine métier dans des packages, facilitant ainsi la maintenance et l'évolution du système. Les packages offrent une encapsulation naturelle et une interface claire entre les applicatifs clients et la base de données.

— **PKG_MANAGER_QUAI** : Centralise la planification et la gestion des infrastructures. Il utilise `FN_EST_QUAI_COMPATIBLE` pour valider physiquement chaque escale avant assignation. Les procédures principales incluent l'enregistrement de navires et la création de réservations.

— **PKG_RH** : Gère le cycle de vie complet du personnel portuaire. La procédure `PR_RECRUTER_EMPLOYE` ajoute un nouvel agent avec validations de salaire, tandis que `PR_MAJ_SALAIRE` permet une gestion sécurisée des ajustements de taux horaires. L'archivage (soft delete) préserve l'historique.

— **PKG_FINANCE** : Automatise la chaîne comptable complète. Il inclut `FN_CALCULER_COUT_TOT` qui agrège le coût matériel (opérations) et la main-d'œuvre (affectations), ainsi que `FN_CHIFFRE_AFFAIRES_PERIODE` pour les reportings financiers.

— **PKG_OPERATIONS** : Pilote le flux logistique et l'affectation du personnel. Les procédures `ENREGISTRER_OPERATION`, `AFFECTER_EMPLOYE` et `AJOUTER_CARGAISON` forment le cœur des activités quotidiennes. `FN_VERIFIER_POIDS` assure le contrôle de surcharge.

— **PKG_CLIENT (Optionnel)** : Interface sécurisée pour les clients externes (armateurs). Elle permet l'authentification via `FN_LOGIN` et la réservation autonome sans exposer les données sensibles internes.

## 4.4 Gestion des Exceptions Personnalisées

Nous avons implémenté une plage d'erreurs métier spécifiques (-20000 à -20100). Grâce à `PRAGMA EXCEPTION_INIT`, ces erreurs Oracle sont associées à des noms métier explicites, facilitant le débogage et la gestion d'erreurs applicative.

| Code d'Erreur | Exception | Contexte |
|---|---|---|
| -20001 | EXC_QUAI_TROP_PEU_PROFOND | Tirant d'eau incompatible |
| -20002 | EXC_DATE_INVALIDE | Opération avant arrivée du navire |
| -20003 | EXC_CAPACITE_DEPASSEE | Surcharge du navire |
| -20004 | EXC_DONNEE_INTROUVABLE | Entité non trouvée |
| -20005 | EXC_DOUBLON_AFFECTATION | Employé déjà affecté |
| -20010 | EXC_ESCALE_ANNULEE | Tentative de facturation d'escale annul |
| -20020 | EXC_SALAIRE_INVALIDE | Taux horaire non-positif |
| -20021 | EXC_HIERARCHIE_CYCLIQUE | Employé = son propre supérieur |
| -20022 | EXC_SUPPRESSION_INTERDITE | Suppression d'employé avec historique |
| -20023 | EXC_DOUBLON_RH | Matricule ou email en doublon |

TABLE 4.4 – Codes d'erreur personnalisés du système

## 4.5 Automatisation par Triggers

Les triggers assurent une surveillance automatique des données sans intervention manuelle. Chacun répond à une nécessité métier spécifique :

1. **Sécurité :** `TRG_SECURITE_FACTURE` interdit le `DELETE` sur une facture au statut 'Payée', protégeant l'intégrité comptable. `TRG_RH_PREVENT_DELETE` empêche la suppression physique d'employés ayant un historique.
2. **Cohérence :** `TRG_OPS_DATE_COHERENCE` vérifie que les opérations se déroulent temporellement entre l'ATA (arrivée réelle) et l'ATD (départ réel). `TRG_CHECK_RETARD_PAIEMENT` bascule les factures en retard de paiement.
3. **Audit :** `TRG_AUDIT_PLANNING` trace chaque modification de planning d'escale dans un journal interne.
4. **Automatisation métier :** `TRG_GENERATE_REF_ESCALE` génère les références uniques au format ESC-YYYY-NNNN. `TRG_OPS_SEC_DANGER` ajoute automatiquement le tag [DANGER] pour les cargaisons réglementées.

# Chapitre 5

# Sécurité et Administration (DBA)

## 5.1 Introduction

En tant que projet orienté "Database Administration" (DBA), la sécurité repose sur une correspondance stricte entre les responsabilités métier et les privilèges techniques. Cette section détaille comment les rôles définis dans le cahier des charges ont été implémentés via des rôles Oracle (`ROLES`), des vues sécurisées et des packages encapsulés. L'approche privilégie le principe du **moindre privilège** : chaque utilisateur n'accède qu'aux données strictement nécessaires à sa mission.

## 5.2 Gestion des Utilisateurs et Rôles Métier

Le système repose sur une séparation des tâches (*Separation of Duties - SoD*) où chaque utilisateur n'accède qu'aux fonctionnalités nécessaires à sa mission. Cette architecture prévient les conflits d'intérêts et les fraudes internes.

### 5.2.1 Manager Quai & Capitainerie (`ROLE_MANAGER_QUAI`)

**Rôle principal :** Planification et optimisation de l'utilisation des infrastructures portuaires.

- **Responsabilités fonctionnelles :**
  - **Enregistrement des navires** : Implémenté via `PKG_MANAGER_QUAI.SP_ENREGISTRER_NAVIR` L'enregistrement capture les caractéristiques techniques critiques (longueur, tirant d'eau, type, IMO).
  - **Gestion des réservations** : Prise en charge par `PKG_MANAGER_QUAI.SP_CREER_RESERVATION` Les réservations deviennent des escales après validation et assignation de quai.
  - **Planification des escales** : Utilisation de la procédure `PR_PLANIFIER_ESCALE` qui valide la compatibilité du quai et l'absence de chevauchement temporel.
- **Contraintes métier et sécurité :**
  - **Vérification de compatibilité** : Assurée par les fonctions `FN_EST_QUAI_COMPATIBLE` et `FN_VERIFIER_DISPO_QUAI`.
  - **Prévention des risques** : Le trigger `TRG_SEC_VERIF_QUAI` bloque toute affectation présentant un risque d'échouage (Tirant d'eau > Profondeur).

  - **Non-accès aux données sensibles** : Le Manager Quai n'a pas accès aux tables FACTURE, EMPLOYE ou AFFECTATION.
- **Droits SQL accordés :**
  - CRUD complet sur : NAVIRE, NAVIRE_COMMERCE, NAVIRE_PASSAGER, NAVIRE_PECHE, ESCALE, QUAI, RESERVATION
  - SELECT uniquement sur : EMPLOYE (pour les recherches simples), OPERATION (pour le monitoring)
  - EXECUTE sur : PKG_MANAGER_QUAI

### 5.2.2 Responsable Opérations (ROLE_OPERATIONS)

**Rôle principal :** Coordination des opérations de manutention et gestion des équipes.

- **Responsabilités fonctionnelles :**
  - **Enregistrement des opérations** : Création via `PKG_OPERATIONS.ENREGISTRER_OPERATION`. Chaque opération doit être liée à une escale active.
  - **Affectation du personnel** : Gestion des équipes via `PKG_OPERATIONS.AFFECTER_EMPLOYE`. Les affectations enregistrent le rôle de l'employé et les heures travaillées.
  - **Gestion des cargaisons** : Déclaration et vérification via `PKG_OPERATIONS.AJOUTER_CARGAIS` Chaque cargaison est vérifiée contre la capacité du navire.
- **Contraintes métier et sécurité :**
  - **Cohérence temporelle** : Le trigger `TRG_OPS_DATE_COHERENCE` interdit toute opération avant l'arrivée réelle ou après le départ du navire.
  - **Contrôle de charge** : La fonction `FN_VERIFIER_POIDS` empêche de dépasser le `tonnage_max` du navire.
  - **Sécurité des données RH** : Accès via la vue `V_EMPLOYE_OPS` qui masque les salaires. Les opérateurs ne voient que nom, prénom, poste, statut, pas le taux horaire.
- **Droits SQL accordés :**
  - CRUD sur : OPERATION, AFFECTATION, CARGAISON, CONTENEUR
  - SELECT sur : ESCALE, NAVIRE, QUAI, V_EMPLOYE_OPS (vue sécurisée)
  - EXECUTE sur : PKG_OPERATIONS

### 5.2.3 Responsable Finance (ROLE_FINANCE)

**Rôle principal :** Gestion de la facturation et suivi de la performance financière.

- **Responsabilités fonctionnelles :**
  - **Création de factures** : Génération automatisée via `PKG_FINANCE.PR_CREER_FACTURE`. Les factures sont produites une fois l'escale planifiée et validée.
  - **Calcul des coûts** : Agrégation dynamique via `FN_CALCULER_COUT_TOTAL_ESCALE` (Matériel + Heures travaillées × taux horaire).
  - **Enregistrement de paiement** : Via `PR_PAYER_FACTURE` qui valide que la facture n'est pas déjà payée.
  - **Analyse financière** : Reporting du chiffre d'affaires via `FN_CHIFFRE_AFFAIRES_PERIODE`.
- **Contraintes métier et sécurité :**

  - **Intégrité comptable** : Le trigger `TRG_SECURITE_FACTURE` interdit la suppression d'une facture au statut 'Payée'. Les modifications ne sont possibles que sur les factures 'En attente'.
  - **Suivi des retards** : Le trigger `TRG_CHECK_RETARD_PAIEMENT` met à jour automatiquement le statut en 'En retard' si l'échéance est dépassée.
  - **Confidentialité** : Le Responsable Finance a accès aux taux horaires uniquement pour le calcul des coûts, via une vue dédiée `V_EMPLOYE_FINANCE`.
- **Droits SQL accordés :**
  - CRUD complet sur : FACTURE, BILLET
  - SELECT sur : ESCALE, OPERATION, NAVIRE, V_EMPLOYE_FINANCE (vue sécurisée avec taux horaires)
  - EXECUTE sur : PKG_FINANCE

### 5.2.4 Responsable RH (`ROLE_RH`)

**Rôle principal :** Administration du personnel et gestion de la masse salariale.

- **Responsabilités fonctionnelles :**
  - **Recrutement** : Enregistrement propre via `PKG_RH.PR_RECRUTER_EMPLOYE`. Chaque recrutement requiert matricule, email, taux horaire et rattachement hiérarchique.
  - **Gestion des salaires** : Mise à jour sécurisée via `PR_MAJ_SALAIRE` et calcul via `FN_CALCULER_PAIE_MENSUELLE`. Les augmentations sont tracées.
  - **Archivage (Soft Delete)** : Sortie des effectifs via la procédure `PR_ARCHIVER_EMPLOYE`. L'employé passe en statut 'ARCHIVE', préservant l'historique.
  - **Gestion hiérarchique** : Maintien de l'organigramme via le champ `id_superieur`. Les relations de supervision sont validées.
- **Contraintes métier et sécurité :**
  - **Cohérence hiérarchique** : Le trigger `TRG_RH_INTEGRITE_EMPLOYE` interdit qu'un employé soit son propre supérieur ou celui de son supérieur (chaîne cyclique).
  - **Protection des données** : Le trigger `TRG_RH_PREVENT_DELETE` bloque toute suppression physique d'un employé possédant un historique d'affectations.
  - **Contrôle de la paie** : Validation des taux horaires positifs avant insertion/-mise à jour.
- **Droits SQL accordés :**
  - CRUD complet sur : EMPLOYE, AFFECTATION (pour la gestion du lien hiérarchique et des allocations)
  - SELECT sur : OPERATION (pour audit des activités)
  - EXECUTE sur : PKG_RH

## 5.3 Matrice de Contrôle d'Accès et Modèle Hybride

Le SGP adopte une stratégie de **Défense en Profondeur**. Contrairement à une approche monolithique, la sécurité est assurée par trois couches superposées :

- **Niveau 1 : Les Privilèges (CRUD) :** Définit ce que l'utilisateur est techniquement capable de faire sur les objets (Tables/Vues).

— **Niveau 2 : L'Abstraction (Packages) :** Fournit des interfaces standardisées pour les opérations complexes, garantissant la cohérence des transactions (ACID).
— **Niveau 3 : Le Garde-Fou (Triggers et Contraintes) :** Agit comme une dernière ligne de défense. Si un utilisateur privilège utilise ses droits d'accès directs (SQL) pour contourner un package, les triggers bloquent toute opération violant les règles métier critiques (suppression de facture, incohérence de dates).

Le tableau suivant résume les droits accordés à chaque rôle sur les entités principales. C = Create, R = Read, U = Update, D = Delete.

| **Entité / Rôle** | **Manager Quai** | **Opérations** | **Finance** | **RH** |
|---|---|---|---|---|
| NAVIRE | C R U D | R | R | - |
| QUAI | C R U | R | R | - |
| RESERVATION | C R U | R | R | - |
| ESCALE | C R U | R | R | - |
| OPERATION | R | C R U | R | - |
| CARGAISON | R | C R U | R | - |
| CONTENEUR | R | C R U | - | - |
| AFFECTATION | - | C R U | R | C R U |
| FACTURE | - | - | C R U D | - |
| EMPLOYE | R (limité) | R (vue) | R (vue) | C R U D |

TABLE 5.1 – Matrice de droits d'accès par rôle (CRUD)

## 5.4 Principe du Moindre Privilège et Vues Sécurisées

Pour garantir la confidentialité des données sensibles (notamment les salaires), le système utilise des vues qui filtrent l'accès aux tables brutes. Cette approche offre une flexibilité maximale tout en respectant les contraintes de sécurité.

— **V_EMPLOYE_OPS** : Vue destinée au `ROLE_OPERATIONS`. Elle expose id_employe, matricule, nom, prenom, poste, statut, telephone, **mais masque** taux_horaire, email personnel et date_embauche.
— **V_EMPLOYE_FINANCE** : Vue destinée au `ROLE_FINANCE`. Elle expose id_employe, taux_horaire uniquement, permettant le calcul des coûts sans accès aux données personnelles (nom, email).
— **V_MES_FACTURES** : Vue destinée aux clients externes. Basée sur le contexte d'authentification (USER), elle filtre les factures pour afficher uniquement celles de l'armateur connecté. Paramétrage via :
  — WHERE id_armateur = SYS_CONTEXT('USERENV', 'CLIENT_IDENTIFIER')
— **Bénéfices** : (1) Isolation des données sensibles, (2) Implémentation sûre du contrôle d'accès au niveau données (DAC), (3) Impossible de contourner via requêtes SQL directes.

## 5.5 Sécurité Transactionnelle et Abstraction

Le système garantit l'intégrité des données à travers plusieurs mécanismes techniques standards :

- **Atomicité ACID** : Toutes les procédures (ex : recrutement, paiement, planification) utilisent des blocs `COMMIT` et `ROLLBACK` implicites ou explicites pour garantir que les données ne sont jamais partiellement enregistrées en cas d'erreur. Les procédures critiques utilisent des savepoints pour les opérations multi-étapes.
- **Abstraction par Packages** : Bien que les droits directs existent, l'architecture privilégie l'utilisation des packages (PKG_OPERATIONS, PKG_RH, etc.) comme interface standard. Cela crée une couche intermédiaire qui isole la logique métier de la structure de stockage, facilitant la maintenance sans casser les interfaces applicatives.
- **Abstraction par Synonymes** : Des synonymes publics ont été créés pour masquer la complexité des schémas et permettre une invocation directe des objets métiers. Par exemple : `CREATE PUBLIC SYNONYM PKG_OPERATIONS FOR SYSTEM.PKG_OPERATIONS`.
- **Gestion des erreurs** : Les exceptions levées par les packages sont attrapées au niveau applicatif, permettant un traitement métier cohérent et des messages d'erreur localisés.

## 5.6 Défense en Profondeur : Complémentarité Triggers et Packages

L'architecture de sécurité du SGP repose sur une approche hybride où les Triggers et les Packages ne s'opposent pas, mais se complètent pour offrir une sécurité maximale :

- **Niveau Applicatif (Packages)** : Les packages facilitent le travail des utilisateurs en automatisant les processus complexes et en effectuant les contrôles métier en amont (ex : PKG_OPERATIONS vérifie les dates avant d'essayer d'insérer).
- **Niveau Données (Triggers)** : Les triggers agissent comme une **dernière ligne de défense**. Si un utilisateur disposant de droits directs (CRUD) tente de contourner les procédures standards ou commet une erreur de manipulation SQL directe, les triggers de sécurité bloquent immédiatement la transaction.

Exemple concret : Le responsable Opérations dispose techniquement du droit INSERT sur la table OPERATION. S'il exécute une requête manuelle avec une date incohérente, le trigger TRG_OPS_DATE_COHERENCE rejettera l'opération, garantissant l'intégrité du système même en dehors des packages.

# Chapitre 6

# Architecture et Implémentation du Backend

Le développement de la partie serveur (Backend) de l'application *Gestion Portuaire* repose sur le framework *Spring Boot* (Java). Ce choix garantit une architecture robuste, modulaire et capable de gérer la logique complexe liée aux opérations portuaires, à la finance et aux ressources humaines.

## 6.1 Architecture en Couches

Pour assurer la maintenabilité et la séparation des responsabilités, nous avons adopté une architecture classique en trois tiers (N-tier) :

1. **Couche Contrôleur (Controller) :** Point d'entrée de l'API REST. Les contrôleurs (annotés `@RestController`) reçoivent les requêtes HTTP (GET, POST, etc.) provenant du Frontend React, valident les paramètres entrants (comme les IDs ou les dates) et délèguent le traitement à la couche service. Ils renvoient ensuite les réponses au format JSON.
2. **Couche Service (Business Logic) :** C'est le cœur de l'application. Cette couche contient toute la logique métier. Elle orchestre les appels, effectue les calculs (ex : estimation des coûts de facture) et gère les transactions. Si une règle métier est violée (ex : surcharge navire), le service capture l'exception et renvoie un message d'erreur clair.
3. **Couche d'Accès aux Données (Repository) :** Cette couche interagit directement avec la base de données Oracle. Nous utilisons l'interface `JpaRepository` de Spring Data, qui permet d'effectuer des opérations CRUD standards sans écrire de SQL, tout en permettant l'appel de procédures stockées complexes.

## 6.2 Technologies et Outils Clés

**Spring Boot 3 :** Utilisé pour sa configuration automatique et son serveur web embarqué (Tomcat). Il facilite la création d'API RESTful performantes.

**Hibernate / Spring Data JPA :** Ces technologies ORM (Object-Relational Mapping) permettent de manipuler les tables de la base de données (Navire, Escale, Employé) comme des objets Java. Cela simplifie grandement la gestion des entités.

**Oracle Database :** Le système de gestion de base de données choisi pour sa robustesse. Une particularité de notre implémentation est l'utilisation intensive de **PL/SQL** (Procédures stockées et Triggers) pour déléguer les règles de gestion critiques (vérification de profondeur de quai, unicité des matricules) directement au niveau de la base.

**Jackson :** Librairie utilisée implicitement pour la sérialisation/désérialisation des objets JSON, assurant la communication fluide entre le Frontend (React) et le Backend (Java), notamment via l'utilisation d'annotations comme `@JsonProperty` pour mapper les données correctement.

**Lombok :** Bibliothèque utilisée pour réduire le code "boilerplate" (Getters, Setters, Constructeurs) grâce à des annotations, rendant le code plus lisible et concis.

## 6.3 Interaction Java - Oracle (Procédures Stockées)

Contrairement à une approche standard où toute la logique réside dans le code Java, nous avons opté pour une approche hybride performante. Les opérations sensibles (comme `PR_ENREGISTRER_OPERATION` ou `PR_AJOUTER_CARGAISON`) sont exécutées via des appels natifs depuis les Repositories Java :

```
@Procedure(procedureName = "PR_AJOUTER_CARGAISON")
void ajouterCargaison(...);
```

Cette méthode permet de profiter de la puissance du moteur Oracle pour les contrôles d'intégrité tout en gardant une interface API moderne en Java.

# Chapitre 7

# Jeux d'Essai et Validation

## 7.1 Validation Technique et Sécurité (Environnement SQL Developer)

Cette section présente la validation des règles métier critiques via l'exécution de scripts PL/SQL directs. L'objectif est de prouver que la logique implémentée dans la base de données (Packages et Triggers) fonctionne indépendamment de l'interface utilisateur.

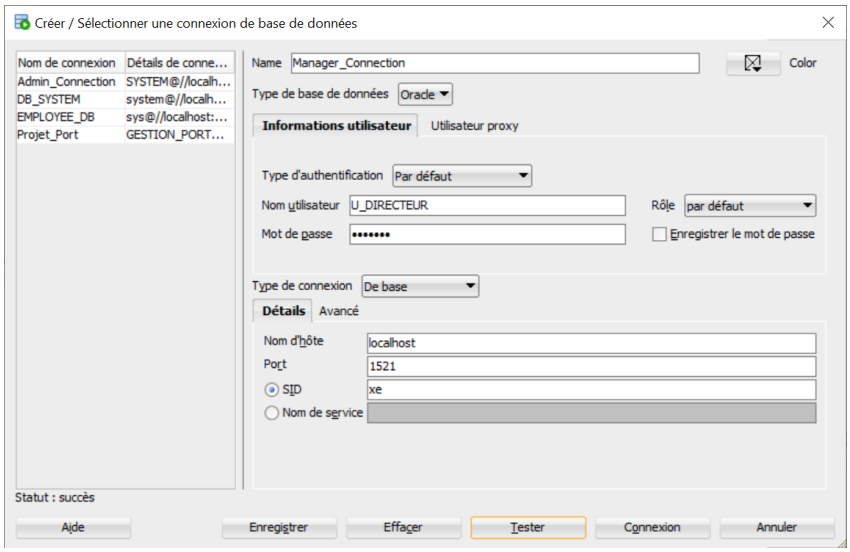


FIGURE 7.1 – Connexion de l'utilisateur U_DIRECTEUR (Manager Quai)

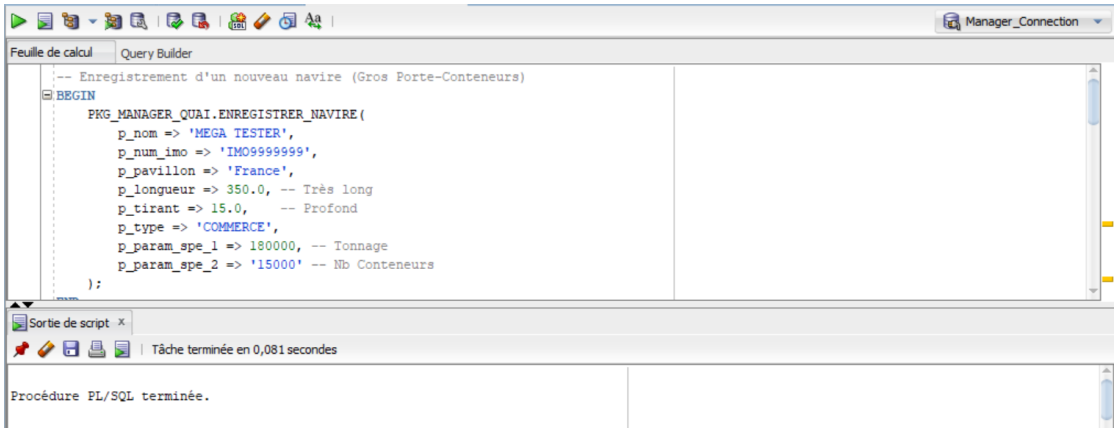


FIGURE 7.2 – Exécution de la procédure ENREGISTRER_NAVIRE

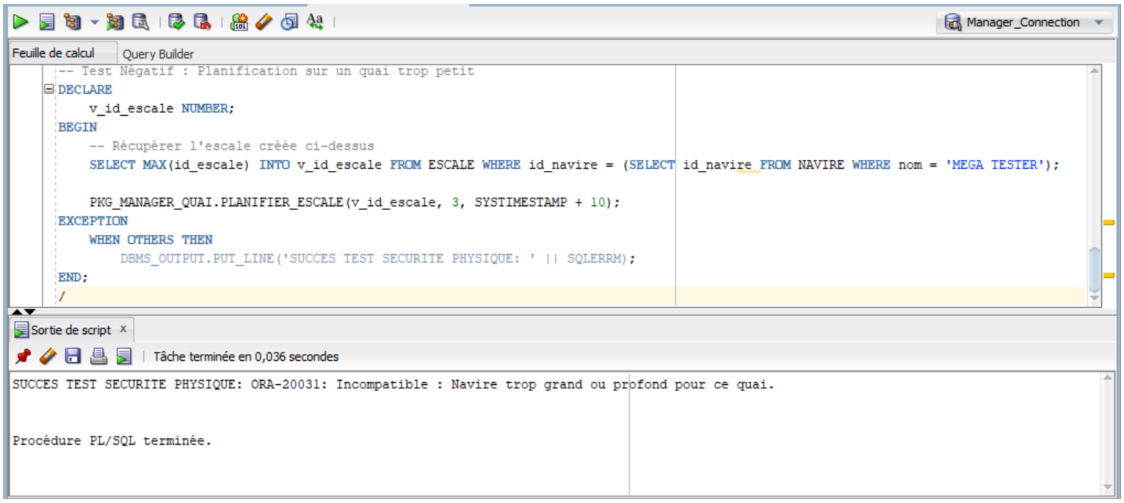


FIGURE 7.3 – Blocage d'une planification dangereuse par le Trigger

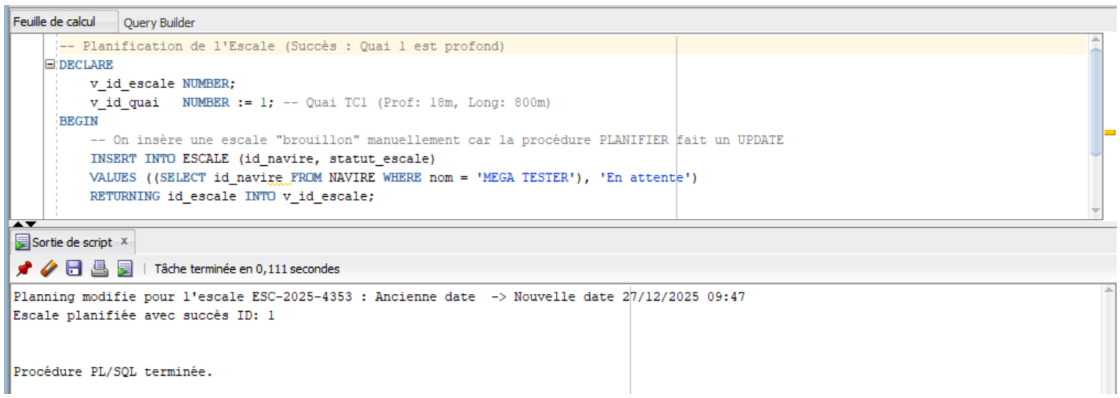


FIGURE 7.4 – Planification réussie et déclenchement de l'Audit

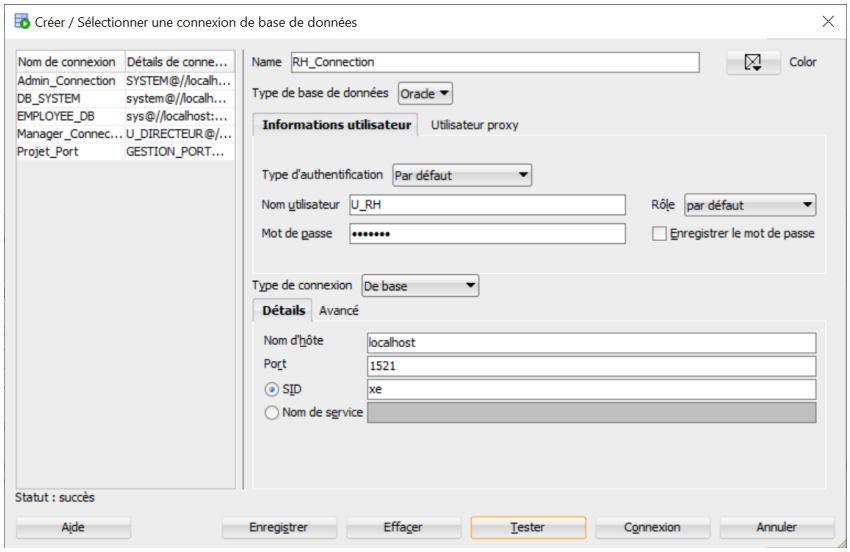


FIGURE 7.5 – Connexion de l'utilisateur U_RH

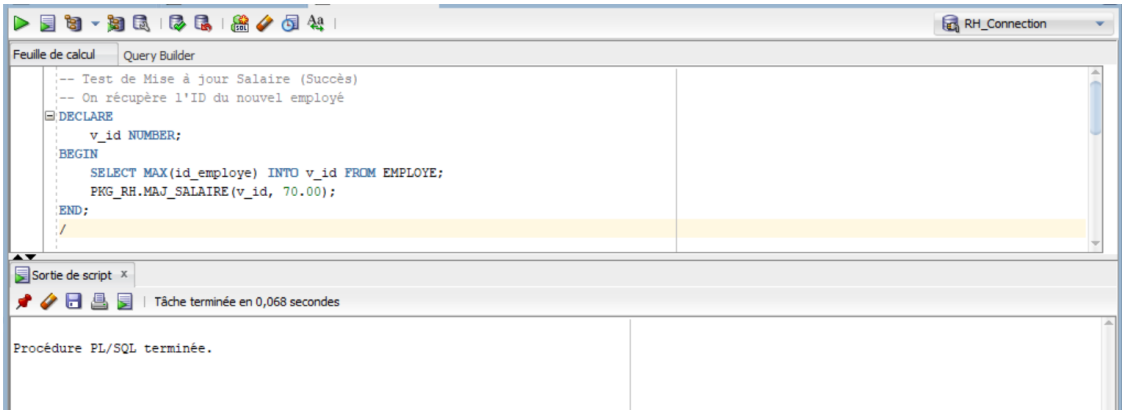


FIGURE 7.6 – Mise à jour salaire

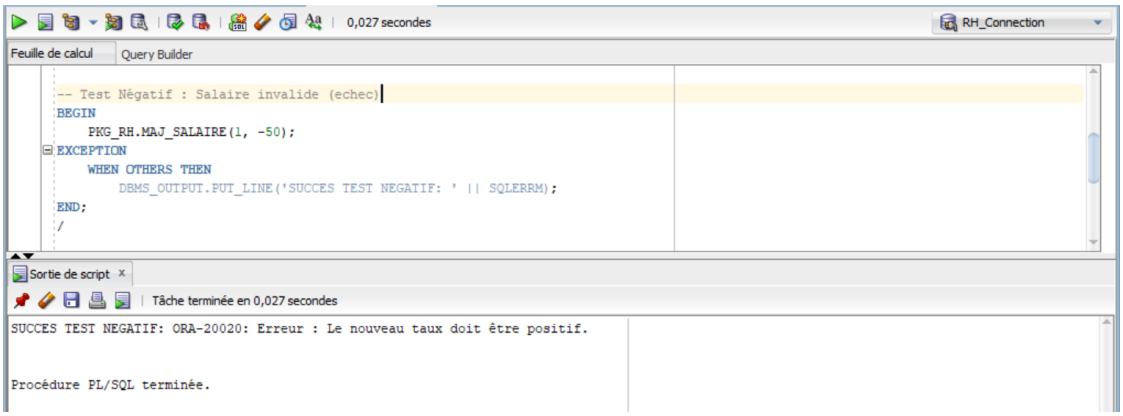


FIGURE 7.7 – Rejet d'une mise à jour de salaire invalide

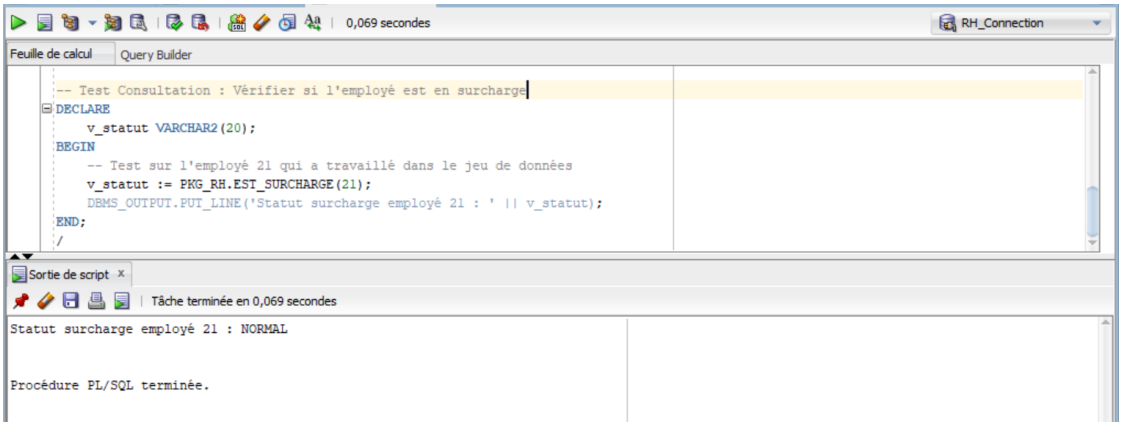


FIGURE 7.8 – Calcul de surcharge

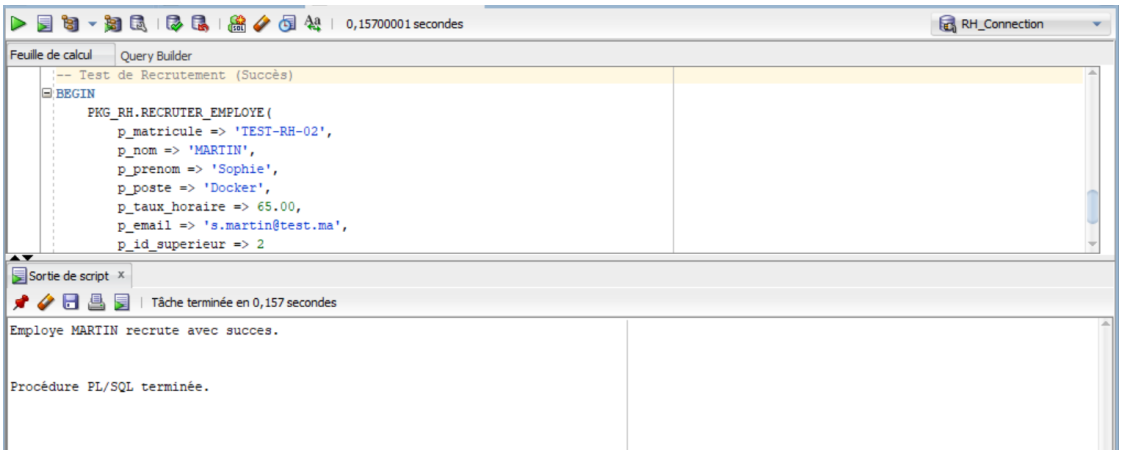


FIGURE 7.9 – Recrutement d'un nouvel employé "Martin Sophie"

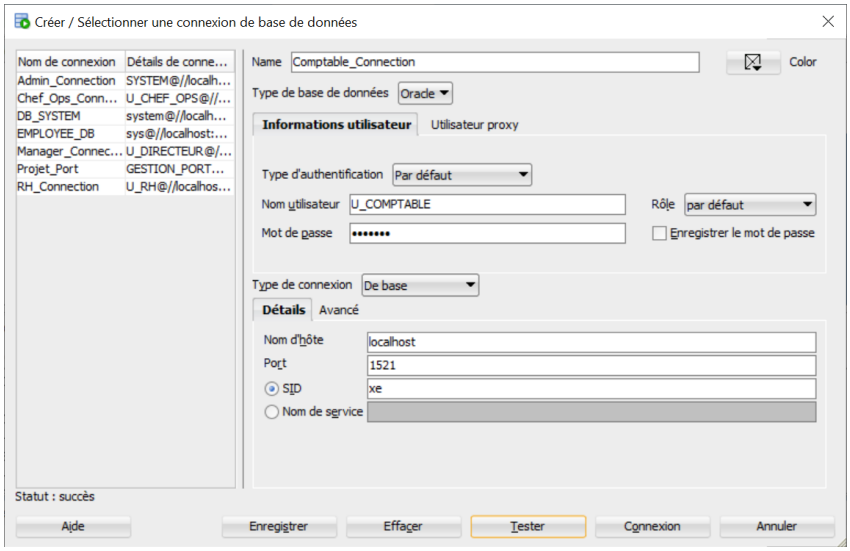


FIGURE 7.10 – Connexion de l'utilisateur U_COMPTABLE

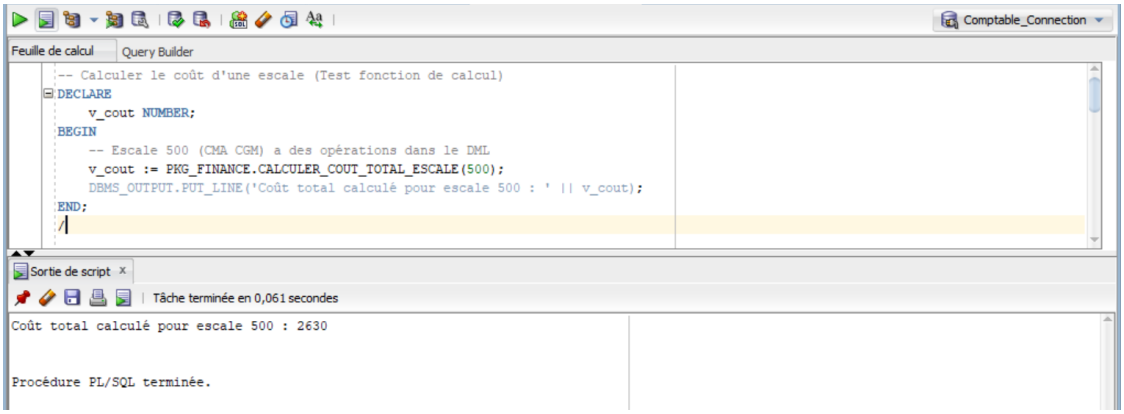


FIGURE 7.11 – Calcul du coût total

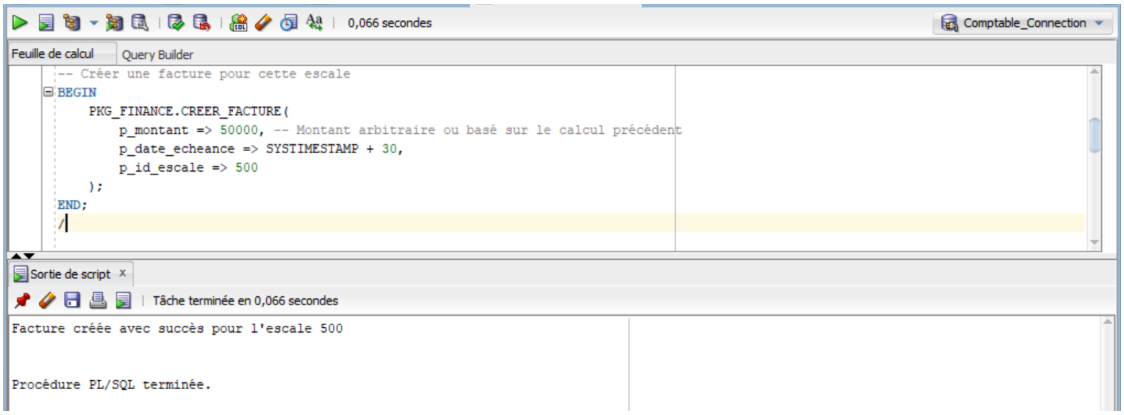


FIGURE 7.12 – Création d'une facture

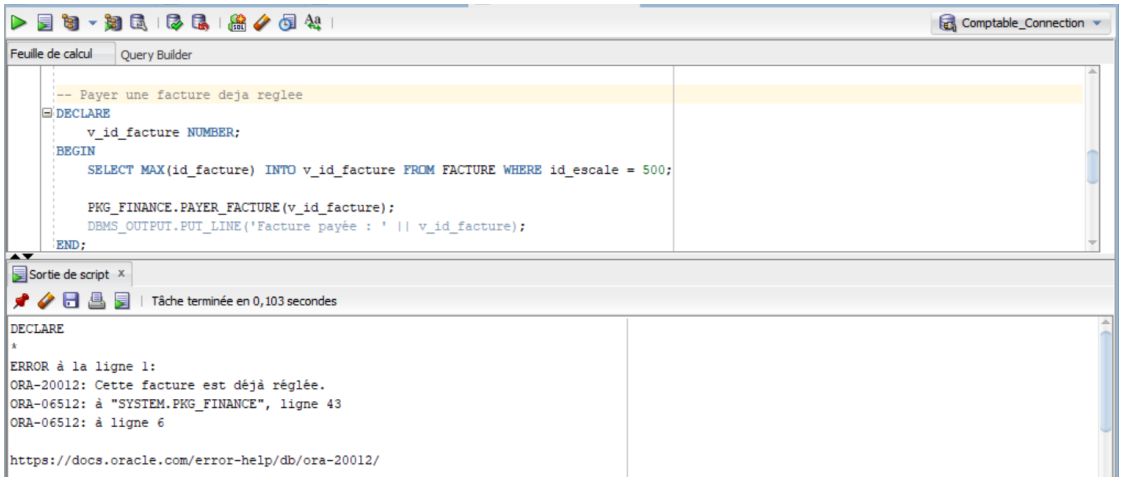


FIGURE 7.13 – Tentative de double paiement

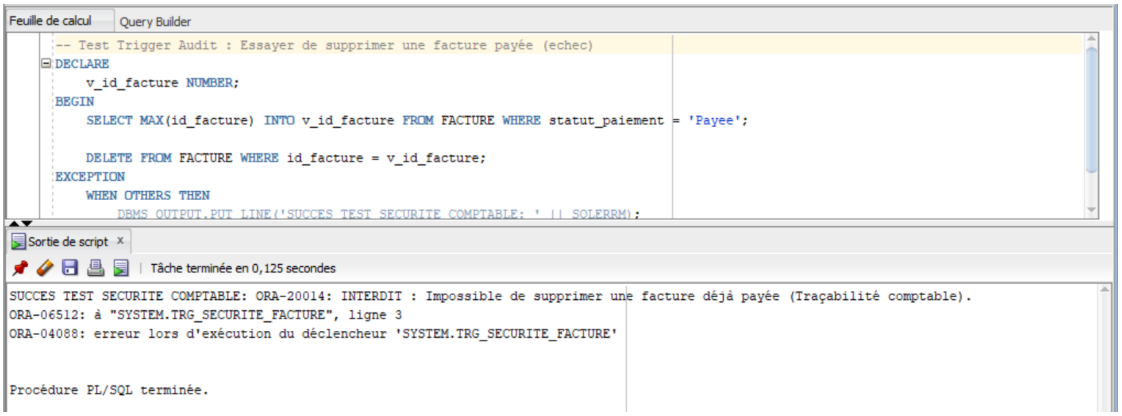


FIGURE 7.14 – Blocage de la suppression d'une facture payée

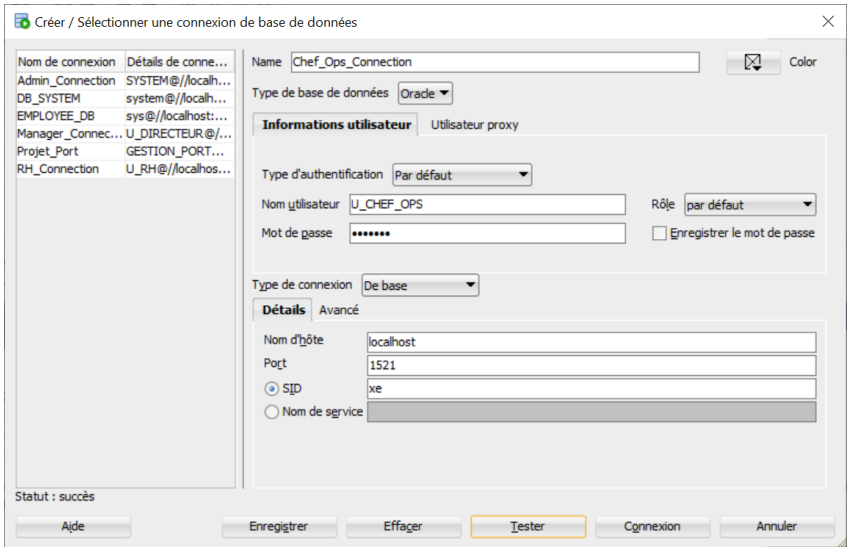


FIGURE 7.15 – Connexion de l'utilisateur U_CHEF_OPS

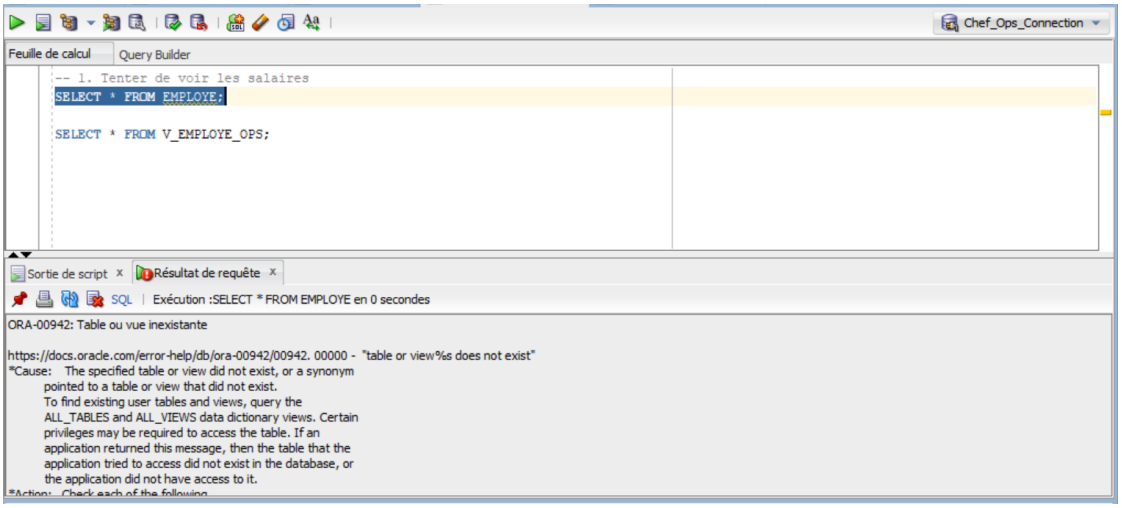


FIGURE 7.16 – Comparaison d'accès Table vs Vue

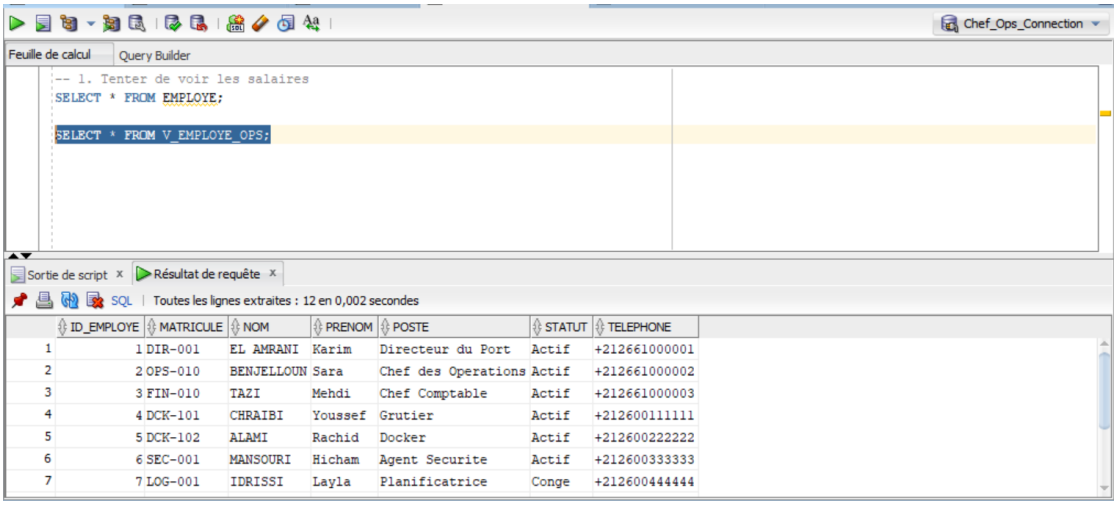


FIGURE 7.17 – Comparaison d'accès Table vs Vue

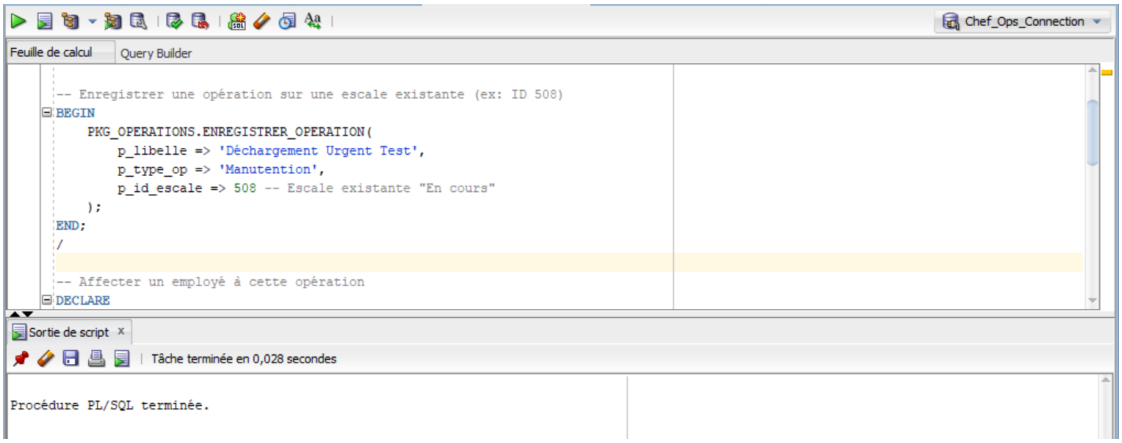


FIGURE 7.18 – Enregistrement d'une Opération

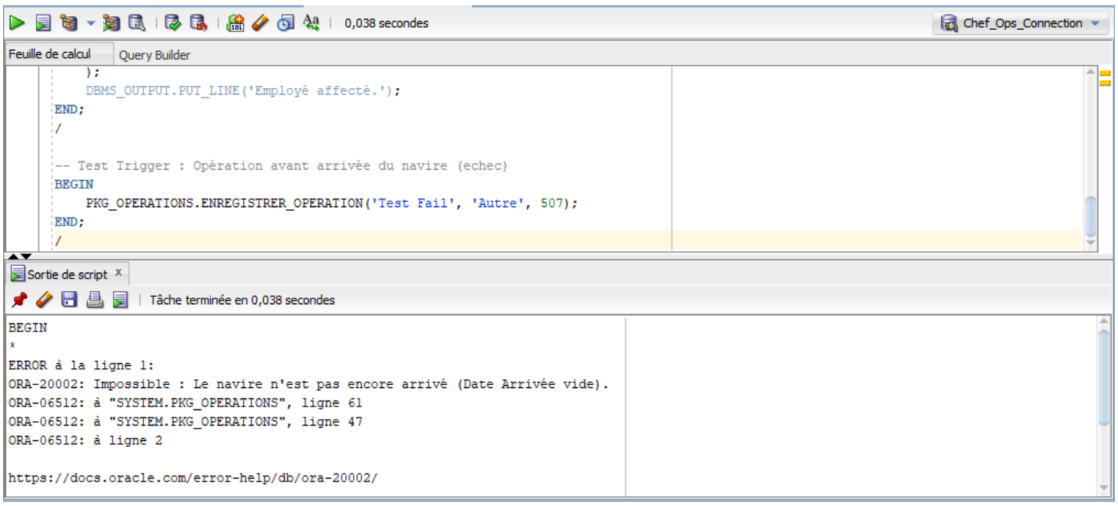


FIGURE 7.19 – Blocage d'une opération anticipée

## 7.2 Validation Fonctionnelle et Ergonomique (Prototype Applicatif)

Voir le documents consacré au tests.

# Chapitre 8

# Recommandations et Améliorations Futures

## 8.1 Optimisations à Court Terme

### 8.1.1 Amélioration de la Performance

— **Ajout d'index supplémentaires** : Sur les colonnes de dates (date_arrivee_prevue, date_depart_prevue) pour les requêtes de planification sur des plages.
— **Partitionnement de FACTURE** : Partition par année pour améliorer les performances des requêtes financières annuelles. Syntaxe : `PARTITION BY RANGE (YEAR(date_emissio`
— **Matérialisation de vues** : Créer des vues matérialisées (snapshots) pour les rapports financiers lourds, rafraîchies nuitamment via job scheduler.

### 8.1.2 Renforcement de la Sécurité

— **Audit complet** : Implémenter Oracle Unified Audit Trail pour tracer tous les accès sensibles (FACTURE, EMPLOYE, salaires).
— **Chiffrement des données** : Appliquer Transparent Data Encryption (TDE) sur les colonnes sensibles comme taux_horaire et montants financiers.
— **Authentification multi-facteur** : Intégrer une authentification externe (LDAP/Active Directory) et ajouter un second facteur (OTP).

## 8.2 Améliorations Fonctionnelles

### 8.2.1 Gestion de la Disponibilité des Ressources

— **Calendrier de maintenance** : Ajouter une table MAINTENANCE_QUAI pour exclure les quais de la disponibilité lors des travaux.
— **Gestion de congés** : Ajouter une table CONGE_EMPLOYE pour connaître la disponibilité réelle du personnel avant affectation.
— **Surcoûts saisonniers** : Table TARIF_QUAI_SAISONNIER pour gérer les variations de prix selon la saison.

### 8.2.2 Traçabilité et Audit Avancé

— **Versioning** : Implémenter une table d'audit (AUDIT_LOG) avec trigger généralisé pour tracer toutes les modifications de tables critiques.
— **Historique des prix** : Garder un historique des modifications de taux horaires et tarifs quai avec dates de validité.
— **Justification des modifications** : Ajouter un champ `motif_modification` pour documenter les changements importants.

### 8.2.3 Intégration Applicative

— **API REST** : Créer une couche API (via Oracle ORDS ou similaire) pour exposer les packages PL/SQL de manière standard.
— **Dashboard temps réel** : Intégrer un outil BI (Power BI, Tableau) connecté en direct pour le monitoring d'occupation des quais.
— **EDI/XML** : Implémenter un module d'échange de données avec les autorités portuaires (manifest navires, douane).

## 8.3 Considérations Organisationnelles

### 8.3.1 Formation et Documentation

— **Manuels utilisateur** : Créer des guides par rôle détaillant les workflows quotidiens.
— **Runbooks** : Documenter les procédures opérationnelles (backup, recovery, incident response).
— **Formation du personnel** : Planifier des sessions de formation par rôle, notamment sur les vues sécurisées et les limitations d'accès.

### 8.3.2 Plan de Continuité

— **Stratégie de backup** : Défaut quotidien + backup horaire des tables critiques (FACTURE, ESCALE).
— **Disaster Recovery** : Mettre en place une réplication vers un site secondaire (Data Guard Oracle).
— **Monitoring** : Alertes sur l'espace disque, les erreurs de triggers, et les tentatives d'accès non autorisé.

# Chapitre 9

# Conclusion

Le Système de Gestion Portuaire présenté dans ce rapport constitue une solution intégrée et sécurisée répondant aux enjeux majeurs de la modernisation portuaire. En déplaçant l'intelligence métier directement dans la base de données Oracle, nous avons atteint plusieurs objectifs stratégiques :

- **Robustesse** : Les règles de gestion implémentées via triggers et procédures garantissent une cohérence absolue des données, quels que soient les erreurs applicatives.
- **Sécurité** : L'architecture à base de rôles, vues et packages encapsulés prévient les accès non autorisés et protège les données sensibles.
- **Performance** : L'indexation stratégique et l'agrégation des coûts en base de données offrent des réponses rapides aux requêtes métier.
- **Conformité** : La traçabilité complète des opérations et la protection des factures payées satisfont les exigences comptables et légales.
- **Maintenabilité** : L'organisation en packages et en schémas propres facilite l'évolution future du système.

Les trois phases du projet (modélisation, implémentation, sécurisation) ont été réalisées en synergie, créant un ensemble cohérent et testable. Les améliorations futures proposées in fine permettront une montée en charge progressive et une adaptation aux évolutions métier.

Ce rapport, associé à la documentation technique fournie en parallèle via GitHub, offre une base solide pour le déploiement, la maintenance et l'évolution du système. L'équipe de projet reste disponible pour les clarifications et les améliorations itératives.