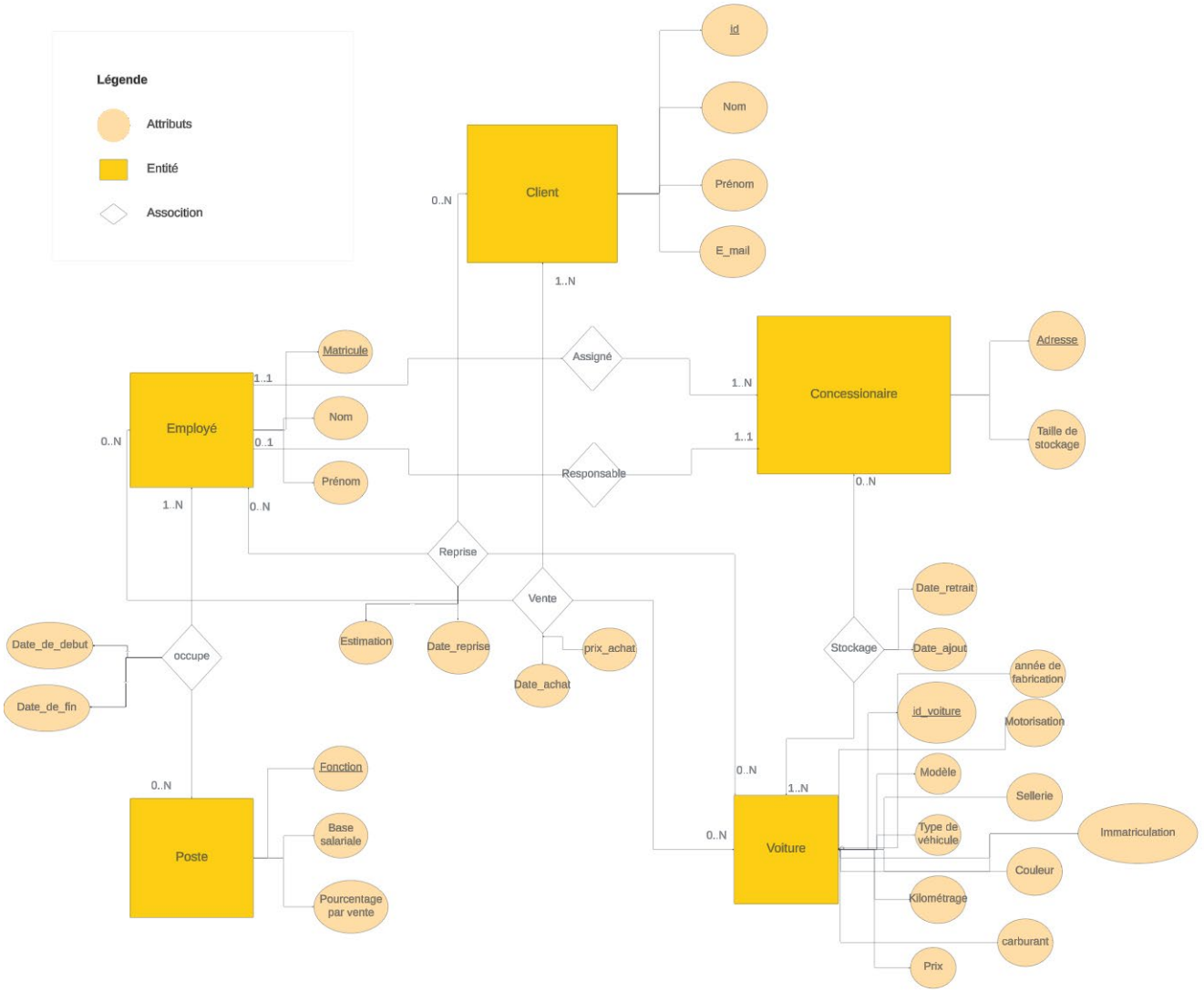
Compte rendu Phase 1 projet IN513

TD1 : Kanga Elie, assam mohamed Gaya

Dans le cadre de la création d'une base de données sous oracle, nous avons choisi de nous intéresser à la gestion de concessionnaires.
En effet, nous devons gérer une entreprise, ayant plusieurs concessionnaires. Ses concessionnaires sont autant spécialisés dans le rachat que la vente de véhicules. Ils proposent plusieurs gammes de véhicules avec différentes options. L'entreprise a une offre d'emploi variée, et tous les concessionnaires ont un responsable qui est le référent de celui-ci.
Nous devons donc construire une application nous permettant de gérer les données de cette entreprise tout en respectant les principes et les fonctions de celle-ci. Les concessionnaires font de l'achat et vente de véhicules ils peuvent donc revendre des véhicules d'occasion.
Voici les objectifs que nous devons respecter :

- **Gestion de l'inventaire :**
    - Les véhicules en stock et disponibles pour la vente sont indiqués dans la base de données.
- **Gestion des ventes :**
    - L'historique des opérations commerciales permet de suivre l'évolution des tendances de la clientèle.
    - En enregistrant les ventes de chaque vendeur, cela permet d'évaluer les performances individuelles des employés.
- **Relation Client :**
    - Stockage des informations clients, permettant un service client personnalisé et efficace.
    - Conservation des achats de chaque client, favorisant les offres promotionnelles ciblées et des recommandations de véhicules basées sur les préférences précédentes.
- **Gestion des Marques et Modèles :**
    - Suivie des performances des différentes marques en fonction des ventes, des préférences des clients, etc.
    - Stockage des informations détaillées sur chaque modèle, l'année de fabrication, le type de carburant et le prix de vente, etc.

Nous avons donc modélisé le problème dans le modèle entité association ci-dessous.



Pour comprendre les associations entres les entités, voici leurs définitions :

Assigné : est une relation entre les tables Concessionnaire et Employé qui a pour objectif de désigner quel employé travaille dans quel concessionnaire
Responsable : est une relation entre les tables Concessionnaire et employé qui a pour objective de désigner un responsable pour chaque concessionnaire
Vente : est une relation entre les tables client, employé et véhicule qui a pour but de désigner quel employé a vendu à quel client quelle voiture
Reprise : est une relation entre les tables Client, Employé et Véhicule qui a pour but de désigner quel employé a racheté à quel client quelle voiture
Stockage : est une relation entre les tables véhicule et concessionnaire qui a pour but de désigner quelle voiture est stocké dans quel concessionnaire
Occupe : est une relation entre les tables employés et poste qui a pour but de désigner quel employé occupe quel poste en ce moment, ainsi que l'historique des

Nous transformons donc le modèle Entité association en modèle relationnel.

- Les attributs soulignés sont les clés primaires de leur table.

Client (id, e_mail, nom, prenom)
Employe (Matricule, nom, prenom, lieu_de_travail)
Poste (Fonction, base_salariale, pourcentage_par_vente)
Occupe (Fonction, matricule, Début, Fin)
Concessionnaire (Adresse, taille_stockage, mat_responsable)
Voiture (id_voiture, immatriculation, modèle, type_vehicule, kilometrage, prix, motorisation, sellerie, couleur, annee_frabication, carburant)
Vente (id_client, mat_vendeur, id_vehicule, date_achat, prix d'achat)
Reprise (id_client, Mat_vendeur, id_vehicule, date_reprise, estimation)
Stockage (Id_vehicule, adr_concessionnaire, date_exe, date_retrait)

Nous déclarons les clés étrangères ci-dessous.

Employe.lieu_de_travail référence concessionaire.Adresse

Concessionnaire.mat_responsable référence Employe.Matricule

Occupe.fonction reference Poste.fonction

Occupe.matricule référence Employe.Matricule

Vente.id_client référence Client.id

Vente.mat_vendeur référence Employe.Matricule

Vente.id_vehicule référence Voiture.id_voiture

Reprise.id_client référence Client.id

Reprise.mat_vendeur référence Employe.Matricule

Reprise.id_vehicule référence Voiture.id_voiture

Pour garder la logiques et l'exactitude de notre Base de données nous avons décidé d'ajouter des contraintes d'intégrité sur les attributs des entités. Celle-ci seront effectuées lors de l'implémentation en PL/SQL, mais nous les exprimons ci-dessous en langages naturel.

Clients.e_mail doit être une email valide (ex : xyz@abc.def)
Poste.base_salariale doit être supérieur ou égal à 1400
occupe.debut doit etre inférieur a occupe.fin (si CDD)
Concessionnaire.taille_stockage doit être supérieur à 1
Voiture.kilometrage doit être supérieur ou égal à 0
Voiture.prix doit être supérieur à 0
Vente.date_achat est par défaut la date actuelle

Vente.mat_vendeur(reprise.mat_vendeur)doit être le matricule d'un employé ayant la fonction vendeur

Une fois la vente (reprise) effectuer la date de retrait(exécution) dans la table stockage doit être mis à jour

Reprise.date_reprise est par défaut la date actuelle

Stockage.date_exe est par défaut la date actuelle

concessionnaire.mat_responsable doit être un employé avec au moins 4 ans d'ancienneté

Question a posé à la base de données :

1. Quels sont les clients ayant acheté et revendu leurs véhicules avant 3 ans ?
2. Calculez les primes de ventes pour chaque vendeur pour l'année 2023.
3. Quelles voitures ont été acheté dans un concessionnaire et revendu dans un autre ?
4. Calculer le total des ventes de chaque concessionnaire sur l'année 2023.
5. Afficher le stock actuel pour chaque concessionnaire.
6. Quels concessionnaires sont remplis ?
7. Quels sont les vendeurs qui ont vendu tous les types de véhicules ?
8. Quels véhicules n'ont pas été vendu pendant l'année 2023 ?
9. Quels véhicules n'a pas changer de prix entre sa vente et sa reprise ?
10. Quels vendeurs ont repris un véhicule qu'ils avaient eux-mêmes vendu ?
11. Quelle est la moyenne des ventes pour chaque concessionnaire ?
12. Quel est le meilleur vendeur pour chaque concessionnaire ?
13. Quel est l'employé qui a touché le plus gros salaire en octobre 2023 ?
14. Quel type carburant a été le moins vendu en 2023 ?

A/

Scripts de création des tables et des contraintes de notre base de données.
générateur table et contraintes

B/ jeu de données

Document Sheets regroupant toutes les données BDD_concessionaire

1. Fichier SQL contenant tous les inserts pour générer la base de données fichier_insert.sql
2. Dossier contenant tous les fichiers a l'usage de la commande sql*load dossier sql*load

C/ manipulations des données

1. Quels sont les clients ayant acheté et revendu leurs véhicules avant 3 ans ?

```
SELECT distinct(c.id),c.nom, c.prenom
FROM CLIENT c, reprise r, vente v
where c.id = r.id_client
and c.id = v.id_client
and v.id_vehicule = r.id_vehicule
and v.date_achat - r.Date_reprise <= 3*365;
```

2. Calculez les primes de ventes pour chaque vendeur pour l'année 2023.

```
select e.nom, e.prenom, sum(v.prix_achat)*0.1 prime_ventes
    from Employe e, vente v
    where e.matricule = v.mat_vendeur and v.date_achat BETWEEN '01-jan-2023' and '31-Dec-2023'
    GROUP by (e.nom,e.prenom);
```

3. Quelles voitures ont été achetées dans un concessionnaire et revendu dans un autre ?

```
select v.id_vehicule, voit.modele, voit.immatriculation
    from vente v, reprise r, Employe e1, Employe e2, voiture voit
    where v.id_vehicule = r.id_vehicule and v.date_achat < r.Date_reprise
    and voit.id_voiture = v.id_vehicule
    and e1.matricule = v.mat_vendeur and e2.matricule = r.mat_vendeur
    and e1.lieu_de_travail != e2.lieu_de_travail;
```

4. Calculer le total des ventes de chaque concessionnaire sur l'année 2023.

```
select e.lieu_de_travail, sum(v.prix_achat) as profit
    from vente v,Employe e
    where e.matricule = v.mat_vendeur
    group by e.lieu_de_travail;
```

5. Afficher le stock actuel pour chaque concessionnaire.

```
 select adr_concessionnaire, count(Id_vehicule)
 from stockage
     where date_retrait is null
 group by adr_concessionnaire;
```

6. Quels concessionnaires sont remplis ?

```
select s.adr_concessionnaire, c.taille_stockage, count(s.Id_vehicule)
        from stockage s, concessionnaire c
        where s.date_retrait is null
        and s.adr_concessionnaire = C.adresse
        group by s.adr_concessionnaire, c.taille_stockage
        having count(s.id_vehicule) = c.taille_stockage;
```

7. Quels sont les vendeurs qui ont vendu tous les types de véhicules ?

```
SELECT e.nom , COUNT(DISTINCT vo.type_vehicule), COUNT(DISTINCT voit.type_vehicule)
            FROM Vente ve, voiture vo, employe e, voiture voit
            WHERE ve.mat_vendeur = e.matricule
            and ve.id_vehicule = vo.id_voiture
            group by e.nom
            having COUNT(DISTINCT vo.type_vehicule) = COUNT(DISTINCT voit.type_vehicule);
```

8. Quels véhicules n'ont pas été vendu pendant l'année 2023 ?

```
select  voit.id_voiture, voit.immatriculation, voit.modele, voit.type_vehicule, voit.prix
        from  voiture voit
        where voit.id_voiture not in (select id_vehicule
            from vente v2
            where v2.date_achat between '01-jan-2023' and '31-dec-2023');
```

9. Quels véhicules n'a pas changer de prix entre sa vente et sa reprise ?

```
select voit.id_voiture, voit.immatriculation, voit.modele, voit.type_vehicule, voit.prix
        from vente v, reprise r, voiture voit
        where v.id_vehicule = r.id_vehicule
        and v.prix_achat = r.estimation
        and voit.id_voiture = v.id_vehicule;
```

10. Quels vendeurs ont repris un véhicule qu'ils avaient eux-mêmes vendu ?

```
select e.matricule, e.nom, e.prenom
        from employe e, reprise r, vente v
        where v.id_vehicule = r.id_vehicule
        and v.mat_vendeur = r.mat_vendeur
        and e.matricule = v.mat_vendeur;
```

11. Quelle est la moyenne des ventes pour chaque concessionnaire ?

```
select e.lieu_de_travail, avg(v.prix_achat)
        from vente v, employe e
        where v.mat_vendeur = e.matricule
        group by e.lieu_de_travail;
```

12. Quel est le meilleur vendeur pour chaque concessionnaire ?

```
select e.lieu_de_travail,e.nom,max(t.TTventes)
        from employe e, (select v.mat_vendeur vendeur, sum(v.prix_achat) TTventes
            from vente v
            group by v.mat_vendeur)  t
        where e.matricule  = t.vendeur
        group by e.lieu_de_travail, e.nom;
```

13. Quel est l'employé qui a touché le plus gros salaire en octobre 2023 ?

```
select e.nom, e.prenom, sum(v.prix_achat)*0.1 prime_ventes
    from Employe e, vente v
    where e.matricule = v.mat_vendeur and v.date_achat between '01-OCt-2023' and '31-oct-2023'
    GROUP by (e.nom,e.prenom);

select e.nom,e.prenom, p.base_salariale + COALESCE(prime.prime_ventes, 0)
from  poste p, occupe o, employe e  left  join (select em.matricule ,sum(ve.prix_achat)*0.1 prime_ventes
    from Employe em, vente ve
    where em.matricule = ve.mat_vendeur and ve.date_achat between '01-OCt-2023' and '31-oct-2023'
    GROUP by em.matricule) prime
on e.matricule = prime.matricule
where e.matricule = o.matricule
and o.fonction = p.fonction;
```

14. Quel type carburant a été le moins vendu en 2023 ?

```
select voit.carburant, count(ve.id_vehicule)
from voiture voit, vente ve
where ve.id_vehicule = voit.id_voiture
group by voit.carburant
having count(ve.id_vehicule) = (select min(c.nbr)
    from (select vo.carburant carb, count(v.id_vehicule) nbr
        from voiture vo, vente v
        where v.id_vehicule = vo.id_voiture
        group by vo.carburant) c);
```

D/ les vues

Pour la gestion des droits on utilisera plusieurs rôles :

- Viewer, qui représente tous les employés
- Responsable, qui représente les responsables des concessionnaires
- Seller, qui représente les vendeurs

```
CREATE ROLE viewer;
CREATE ROLE responsable;
CREATE ROLE seller;
```

- stock de chaque concessionnaires -> vendeur et responsable

```
create view stock_entrepot as
select adr_concessionnaire, count(Id_vehicule)  stock
        from stockage
    WHERE DATE_RETRAIT IS NULL
        group by adr concessionnaire;
GRANT all privileges ON stock_entrepot TO seller;
GRANT select ON stock entrepot TO responsable;
```

- nombre d'employé pour chaque concessionnaire -> responsable

```
create view employe_par_concessionnaire as
select lieu_de_travail, count(matricule)  effectif
        from employe
        group by lieu_de_travail;
GRANT select ON employe_par_concessionnaire TO responsable;
```

- employé avec leur salaire et lieu de travail -> responsable

```
create view donne_employe as
select lieu_de_travail,e.nom, e.prenom, p.base_salariale
from employe e, occupe o, poste p
where p.fonction = o.fonction
and e.matricule = o.matricule
GRANT select ON donne_employe TO responsable;
```

- véhicules vendu durant l'année 2023 -> vendeur et responsable

```
create view ventes_2023 as
select vo.id_voiture, vo.modele,vo.immatriculation
from vente v, voiture vo
where v.id_vehicule  = vo.id_voiture
and v.date_achat between '01-Jan-2023' and '31-DEC-2023'
GRANT select ON ventes_2023 TO responsable;
GRANT select ON ventes_2023 TO seller;
```

- nom et prénom des responsables pour chaque concessionnaire -> responsable,

```
create view ventes_2023 as
select c.adresse, e.nom, e.prenom
from concessionnaire c, employe e
where e.matricule  = c.mat_responsable
GRANT select ON ventes_2023 TO responsable;
GRANT select ON ventes_2023 TO seller;
GRANT select ON ventes_2023 TO viewer;
```

E/ Intégrité des données : les triggers

Mettre à jour la date de retrait du stock lorsqu'un véhicule est vendu

```
CREATE OR REPLACE TRIGGER trg_maj_date_retrait
AFTER insert or update ON vente
FOR EACH ROW
BEGIN
   UPDATE Stockage SET date_retrait = SYSDATE WHERE Id_vehicule = :OLD.Id_vehicule;
END;
/
```

Mettre à jour la date d'exécution du stock lorsqu'un véhicule est repris

```
CREATE OR REPLACE TRIGGER trg_maj_date_retrait
AFTER insert or update ON reprise
FOR EACH ROW
BEGIN
   UPDATE Stockage SET date_exe = SYSDATE WHERE Id_vehicule = :OLD.Id_vehicule;
END;
/
```

Vérifier si lors d'une vente l'employé l'exécutant soit bien un Vendeur

```
CREATE OR REPLACE TRIGGER vente_check_vendeur
BEFORE INSERT OR UPDATE ON Vente
FOR EACH ROW
DECLARE
    v_count NUMBER;
BEGIN
    IF :NEW.mat_vendeur IS NOT NULL THEN
        SELECT COUNT(*)
        INTO v_count
        FROM occupe
        WHERE Matricule = :NEW.mat_vendeur AND fonction = 'Vendeur';

        IF v_count = 0 THEN
            RAISE_APPLICATION_ERROR(-20001, 'Le matricule du vendeur doit correspondre à un employé ayant la fonction vendeur.');
        END IF;
    END IF;
END;
```

Vérifier si lors d'une reprise l'employé l'exécutant soit bien un Vendeur

```
CREATE OR REPLACE TRIGGER reprise_check_vendeur
BEFORE INSERT OR UPDATE ON reprise
FOR EACH ROW
DECLARE
    v_count NUMBER;
BEGIN
    IF :NEW.mat_vendeur IS NOT NULL THEN
        SELECT COUNT(*)
        INTO v_count
        FROM occupe
        WHERE Matricule = :NEW.mat_vendeur AND fonction = 'Vendeur';

        IF v_count = 0 THEN
            RAISE_APPLICATION_ERROR(-20001, 'Le matricule du vendeur doit correspondre à un employé ayant la fonction vendeur.');
        END IF;
    END IF;
END;
```

Vérifier si la personne a placer en responsable a une ancienneté >= a 4 ans

```
BEFORE INSERT OR UPDATE ON Concessionnaire
FOR EACH ROW
DECLARE
    anciennete NUMBER;
BEGIN
    SELECT 2024 - EXTRACT(YEAR FROM debut)
    INTO anciennete
    FROM occupe
    WHERE Matricule = :NEW.mat_responsable;
    IF anciennete < 4 THEN
        RAISE_APPLICATION_ERROR(-20001, 'Le responsable du concessionnaire doit avoir au moins 4 ans d''ancienneté.');
    END IF;
END;
```

F/ meta-données :

List_ora_constraints :

```
DECLARE
    v_constraint_name VARCHAR2(100);
    v_constraint_type VARCHAR2(20);
    v_search_condition VARCHAR2(4000);
    v_source_code CLOB;
BEGIN
    FOR c IN (SELECT table_name, constraint_name, constraint_type, search_condition
              FROM user_constraints
              WHERE table_name IS NOT NULL
              ORDER BY table_name, constraint_type)
    LOOP
        v_constraint_name := c.constraint_name;
        v_constraint_type := c.constraint_type;
        v_search_condition := c.search_condition;

        DBMS_OUTPUT.PUT_LINE('Table: ' || c.table_name);
        DBMS_OUTPUT.PUT_LINE('Constraint Name: ' || v_constraint_name);
        DBMS_OUTPUT.PUT_LINE('Constraint Type: ' || v_constraint_type);

        IF v_constraint_type IN ('C', 'U', 'P', 'R') THEN
            SELECT TEXT INTO v_source_code
            FROM user_source
            WHERE name = v_constraint_name AND type = 'CONSTRAINT';

            DBMS_OUTPUT.PUT_LINE('Source Code: ' || v_source_code);
        END IF;

        DBMS_OUTPUT.PUT_LINE('-------------------------');
    END LOOP;
END;
/
```

liste_ora_triggers :

```
DECLARE
   v_trigger_name VARCHAR2(100);
   v_table_name VARCHAR2(100);
   v_source_code CLOB;
BEGIN
   FOR t IN (SELECT trigger_name, table_name
             FROM user_triggers
             WHERE table_name IS NOT NULL
             ORDER BY table_name, trigger_name)
   LOOP
      v_trigger_name := t.trigger_name;
      v_table_name := t.table_name;

      DBMS_OUTPUT.PUT_LINE('Table: ' || v_table_name);
      DBMS_OUTPUT.PUT_LINE('Trigger Name: ' || v_trigger_name);

      SELECT TEXT INTO v_source_code
      FROM user_source
      WHERE name = v_trigger_name AND type = 'TRIGGER';

      DBMS_OUTPUT.PUT_LINE('Source Code: ' || v_source_code);
      DBMS_OUTPUT.PUT_LINE('-------------------------');
   END LOOP;
END;
/
```

Liste des colonnes avec leur type par table :

```
SELECT table_name, column_name, data_type
FROM user_tab_columns
ORDER BY table_name, column_name;
```

Liste des index par table :

```
SELECT table_name, index_name, uniqueness
FROM user_indexes
ORDER BY table_name, index_name;
```