

# Travail Pratique 3

INF3080 gr.20 E-23

STAMBOULI, Nouredine - STAN65090008

KHALLADI, Issam - KHAI27029800

DEBAY, Georges

2 Août, 2023

# Question 1

Voir le fichier *02\_declencheurs\_STAN65090008\_KHAI27029800.sql*

```
-- Trigger pour INSERT sur la table LOGEMENTS
CREATE OR REPLACE TRIGGER trg_logements_insert
AFTER INSERT ON LOGEMENTS
FOR EACH ROW
BEGIN
    UPDATE IMMEUBLES
    SET NB_LOGEMENTS = NB_LOGEMENTS + 1
    WHERE NO_IMMEUBLE = :new.NO_IMMEUBLE;
END;
/
```

```
-- Trigger pour UPDATE sur la table LOGEMENTS
CREATE OR REPLACE TRIGGER trg_logements_update
AFTER UPDATE ON LOGEMENTS
FOR EACH ROW
BEGIN
    IF :new.A_LOUER <> :old.A_LOUER THEN
        IF :new.A_LOUER = 'Y' THEN
            UPDATE IMMEUBLES
            SET NB_LOGEMENTS = NB_LOGEMENTS + 1
            WHERE NO_IMMEUBLE = :new.NO_IMMEUBLE;
        ELSE
            UPDATE IMMEUBLES
            SET NB_LOGEMENTS = NB_LOGEMENTS - 1
            WHERE NO_IMMEUBLE = :new.NO_IMMEUBLE;
        END IF;
    END IF;
END;
/
```

```
-- Trigger pour DELETE sur la table LOGEMENTS
CREATE OR REPLACE TRIGGER trg_logements_delete
AFTER DELETE ON LOGEMENTS
FOR EACH ROW
BEGIN
    UPDATE IMMEUBLES
    SET NB_LOGEMENTS = NB_LOGEMENTS - 1
    WHERE NO_IMMEUBLE = :old.NO_IMMEUBLE;
END;
/
```

**(Suite Question 1 - Image)**

## TRG\_LOGEMENTS\_INSERT

```
SELECT NO_IMMEUBLE,NB_LOGEMENTS FROM IMMEUBLES;

INSERT INTO LOGEMENTS (NO_IMMEUBLE, NO_LOGEMENT, LOYER, NB_CHAMBRES, NB_SALLE_BAINS, A_LOUER, CHAUFFAGE, FOYER, MEUBLE, SALLE_A_DINER, OCC_CODE)
VALUES (1001, 7, 1500 ,2, 1, '0', 'G', 'N', 'N', '0', 'L');
```

Script Output x Query Result x

SQL | All Rows Fetched: 11 in 0.045 seconds

	NO_IMMEUBLE	NB_LOGEMENTS
1	1001	33
2	1002	6
3	1003	33
4	1004	12
5	1005	4
6	1006	27
7	1007	10
8	1008	6
9	1009	27
10	1010	12
11	1297	23

```
SELECT NO_IMMEUBLE,NB_LOGEMENTS FROM IMMEUBLES;

INSERT INTO LOGEMENTS (NO_IMMEUBLE, NO_LOGEMENT, LOYER, NB_CHAMBRES, NB_SALLE_BAINS, A_LOUER, CHAUFFAGE, FOYER, MEUBLE, SALLE_A_DINER, OCC_CODE)
VALUES (1001, 7, 1500 ,2, 1, 'O', 'G', 'N', 'N', 'O', 'L');
```

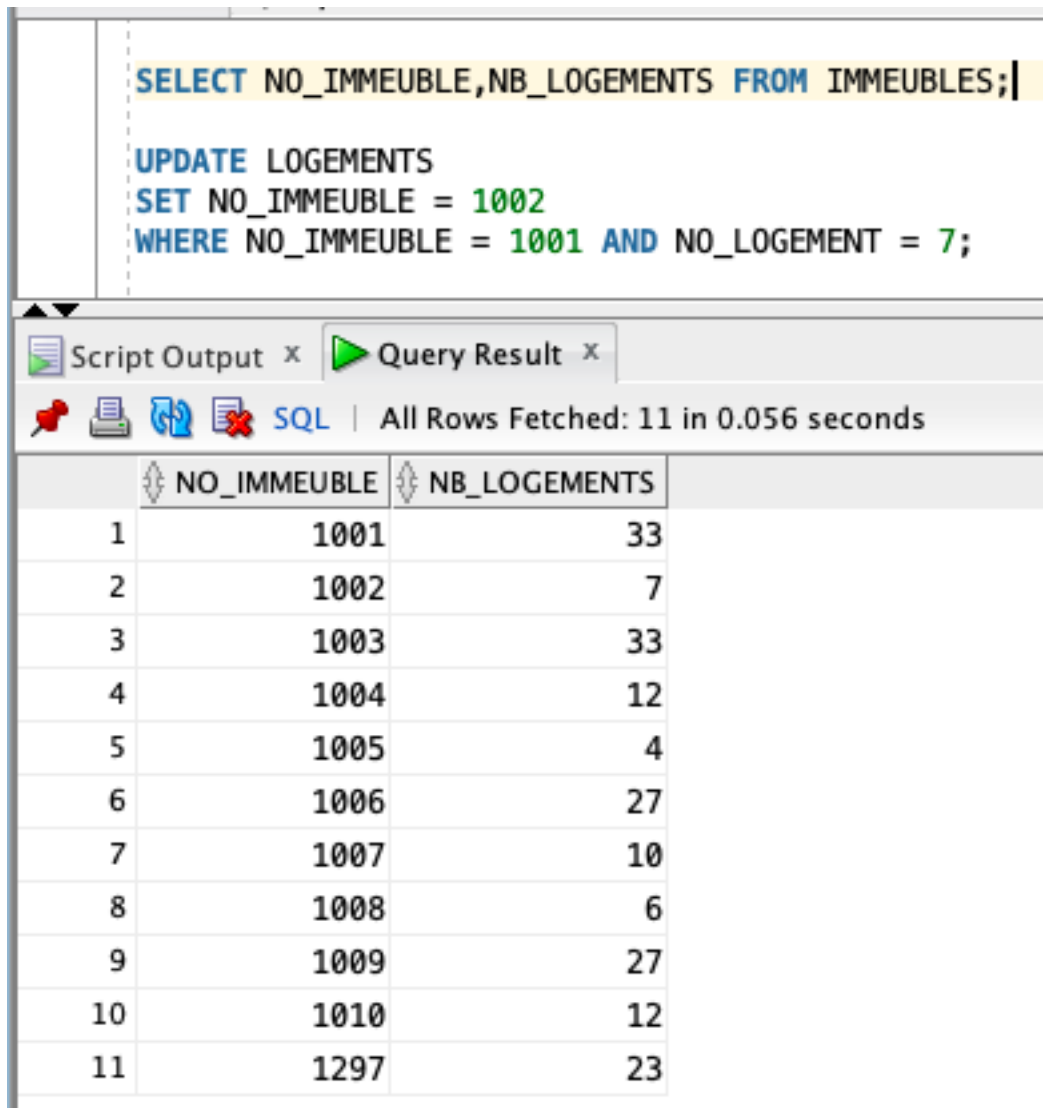
Script Output x Query Result x

SQL | All Rows Fetched: 11 in 0.021 seconds

	NO_IMMEUBLE	NB_LOGEMENTS
1	1001	34
2	1002	6
3	1003	33
4	1004	12
5	1005	4
6	1006	27
7	1007	10
8	1008	6
9	1009	27
10	1010	12
11	1297	23

## (Suite Question 1 - Image)

TRG\_LOGEMENTS\_UPDATE



The screenshot displays a SQL query editor with two tabs: 'Script Output' and 'Query Result'. The 'Query Result' tab is active, showing the results of a query. The query consists of a SELECT statement followed by an UPDATE statement. The SELECT statement retrieves 'NO\_IMMEUBLE' and 'NB\_LOGEMENTS' from the 'IMMEUBLES' table. The UPDATE statement sets 'NO\_IMMEUBLE' to 1002 for rows where 'NO\_IMMEUBLE' is 1001 and 'NO\_LOGEMENT' is 7.

```
SELECT NO_IMMEUBLE,NB_LOGEMENTS FROM IMMEUBLES;|

UPDATE LOGEMENTS
SET NO_IMMEUBLE = 1002
WHERE NO_IMMEUBLE = 1001 AND NO_LOGEMENT = 7;
```

Below the query, the 'Query Result' tab shows a table with 11 rows. The table has two columns: 'NO\_IMMEUBLE' and 'NB\_LOGEMENTS'. The data is as follows:

	NO_IMMEUBLE	NB_LOGEMENTS
1	1001	33
2	1002	7
3	1003	33
4	1004	12
5	1005	4
6	1006	27
7	1007	10
8	1008	6
9	1009	27
10	1010	12
11	1297	23

## (Suite Question 1 - Image)

TRG\_LOGEMENTS\_DELETE

```
SELECT NO_IMMEUBLE,NB_LOGEMENTS FROM IMMEUBLES;

DELETE FROM LOGEMENTS
WHERE NO_IMMEUBLE = 1002 AND NO_LOGEMENT = 7;
```

Script Output x Query Result x

SQL | All Rows Fetched: 11 in 0.012 seconds

	NO_IMMEUBLE	NB_LOGEMENTS
1	1001	33
2	1002	6
3	1003	33
4	1004	12
5	1005	4
6	1006	27
7	1007	10
8	1008	6
9	1009	27
10	1010	12
11	1297	23

## Question 2

Voir le fichier *02\_declencheurs\_STAN65090008\_KHAI27029800.sql*

```
-- Trigger pour INSERT sur la table ENTRETIENS
CREATE OR REPLACE TRIGGER trg_entretiens_insert
AFTER INSERT ON ENTRETIENS
FOR EACH ROW
BEGIN
    UPDATE IMMEUBLES
    SET ENTRETIEN = ENTRETIEN + :new.NB_HEURES
    WHERE NO_IMMEUBLE = :new.IMM_NO_IMMEUBLE;
END;
/

-- Trigger pour UPDATE sur la table ENTRETIENS
CREATE OR REPLACE TRIGGER trg_entretiens_update
AFTER UPDATE ON ENTRETIENS
FOR EACH ROW
BEGIN
    IF :new.NB_HEURES <> :old.NB_HEURES THEN
        UPDATE IMMEUBLES
        SET ENTRETIEN = ENTRETIEN - :old.NB_HEURES + :new.NB_HEURES
        WHERE NO_IMMEUBLE = :new.IMM_NO_IMMEUBLE;
    END IF;
END;
/

-- Trigger pour DELETE sur la table ENTRETIENS
CREATE OR REPLACE TRIGGER trg_entretiens_delete
AFTER DELETE ON ENTRETIENS
FOR EACH ROW
BEGIN
    UPDATE IMMEUBLES
    SET ENTRETIEN = ENTRETIEN - :old.NB_HEURES
    WHERE NO_IMMEUBLE = :old.IMM_NO_IMMEUBLE;
END;
/
```

## (Suite Question 2 - Image)

### TRG\_ENTRETIENS\_INSERT

```
SELECT NO_IMMEUBLE,ENTRETIEN FROM IMMEUBLES WHERE NO_IMMEUBLE = 1001;
```

Script Output x		Query Result x	
SQL   All Rows Fetched: 1 in 0.011 seconds			
	NO_IMMEUBLE	ENTRETIEN	
1	1001	5000	

```
SELECT NO_IMMEUBLE,ENTRETIEN FROM IMMEUBLES WHERE NO_IMMEUBLE = 1001;|  
INSERT INTO ENTRETIENS (NO_ENTRETIEN, ENT_DATE, NB_HEURES, IMM_NO_IMMEUBLE, LOG_NO_LOGEMENT, EMP_EMP_NOM)  
VALUES (11, TO_DATE('2023-06-01', 'YYYY-MM-DD'), 4.0, 1001, 1, 'John Smith');
```

Script Output x		Query Result x	
SQL   All Rows Fetched: 1 in 0.014 seconds			
	NO_IMMEUBLE	ENTRETIEN	
1	1001	5004	

## (Suite Question 2 - Image)

### TRG\_ENTRETIENS\_UPDATE

<pre>SELECT NO_IMMEUBLE,ENTRETIEN FROM IMMEUBLES WHERE NO_IMMEUBLE = 1001;  UPDATE ENTRETIENS SET NB_HEURES = 8.0 WHERE NO_ENTRETIEN = 11;</pre>		
Script Output x Query Result x		
SQL   All Rows Fetched: 1 in 0.014 seconds		
	NO_IMMEUBLE	ENTRETIEN
1	1001	5008

### TRG\_ENTRETIENS\_DELETE

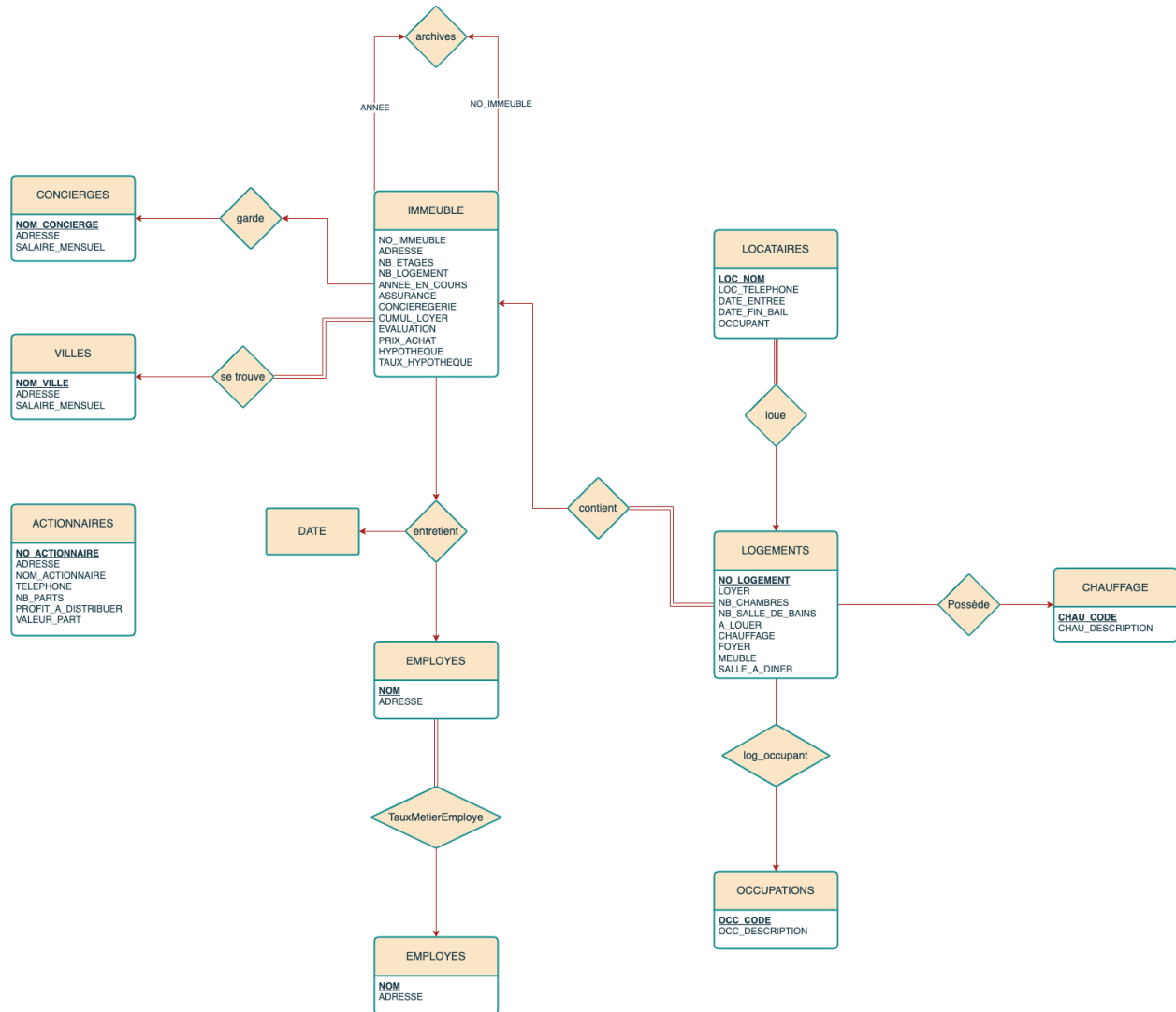
<pre>SELECT NO_IMMEUBLE,ENTRETIEN FROM IMMEUBLES WHERE NO_IMMEUBLE = 1001;  DELETE FROM ENTRETIENS WHERE NO_ENTRETIEN = 11;</pre>		
Script Output x Query Result x		
SQL   All Rows Fetched: 1 in 0.018 seconds		
	NO_IMMEUBLE	ENTRETIEN
1	1001	5000



## Question 3

### Diagramme ER #

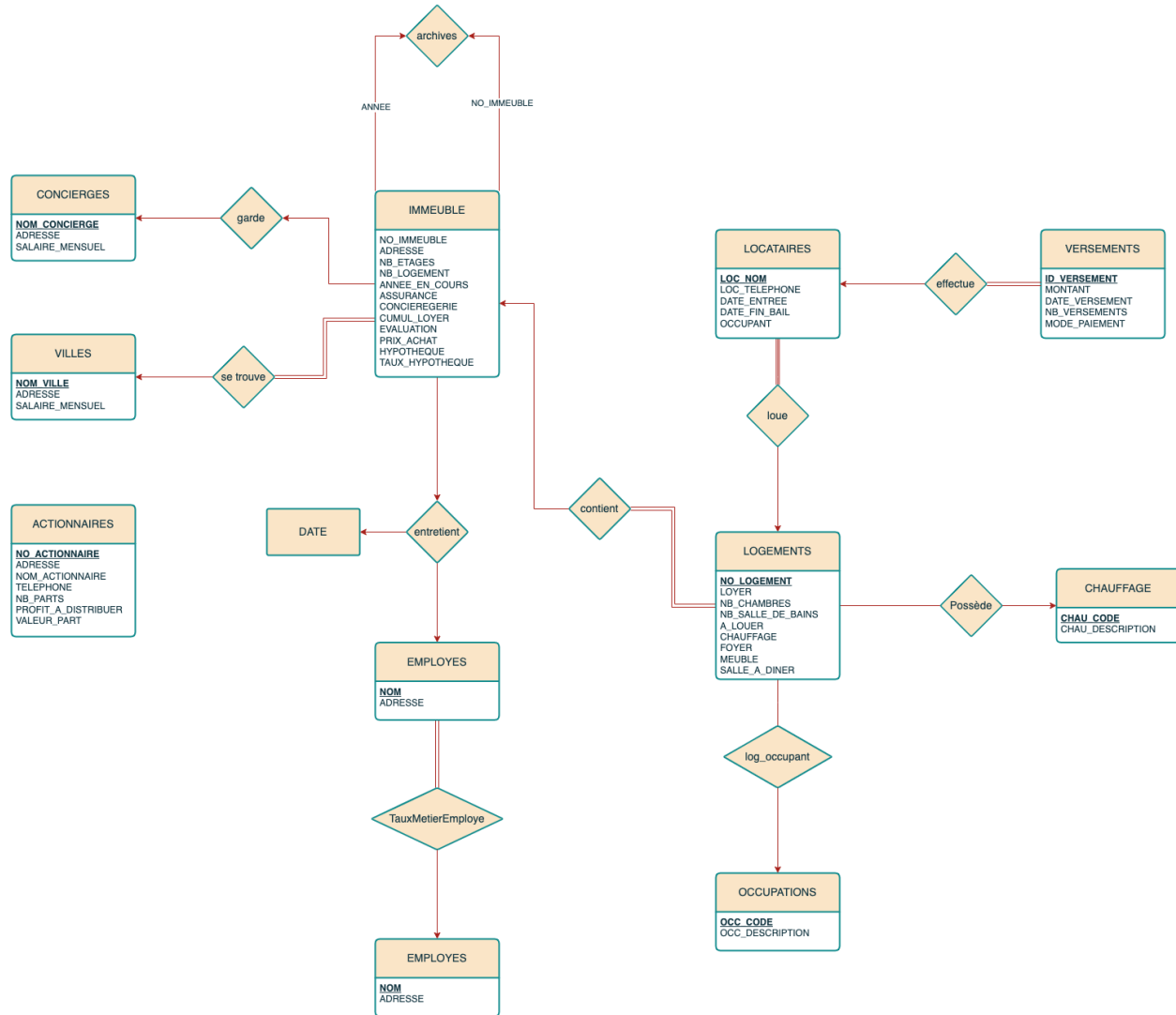
Diagramme ER incluant les clés, les cardinalités et les participations dont le modèle serait conforme au schéma de la page 2 de l'énoncé.



## Question 4

### Diagramme ER #2

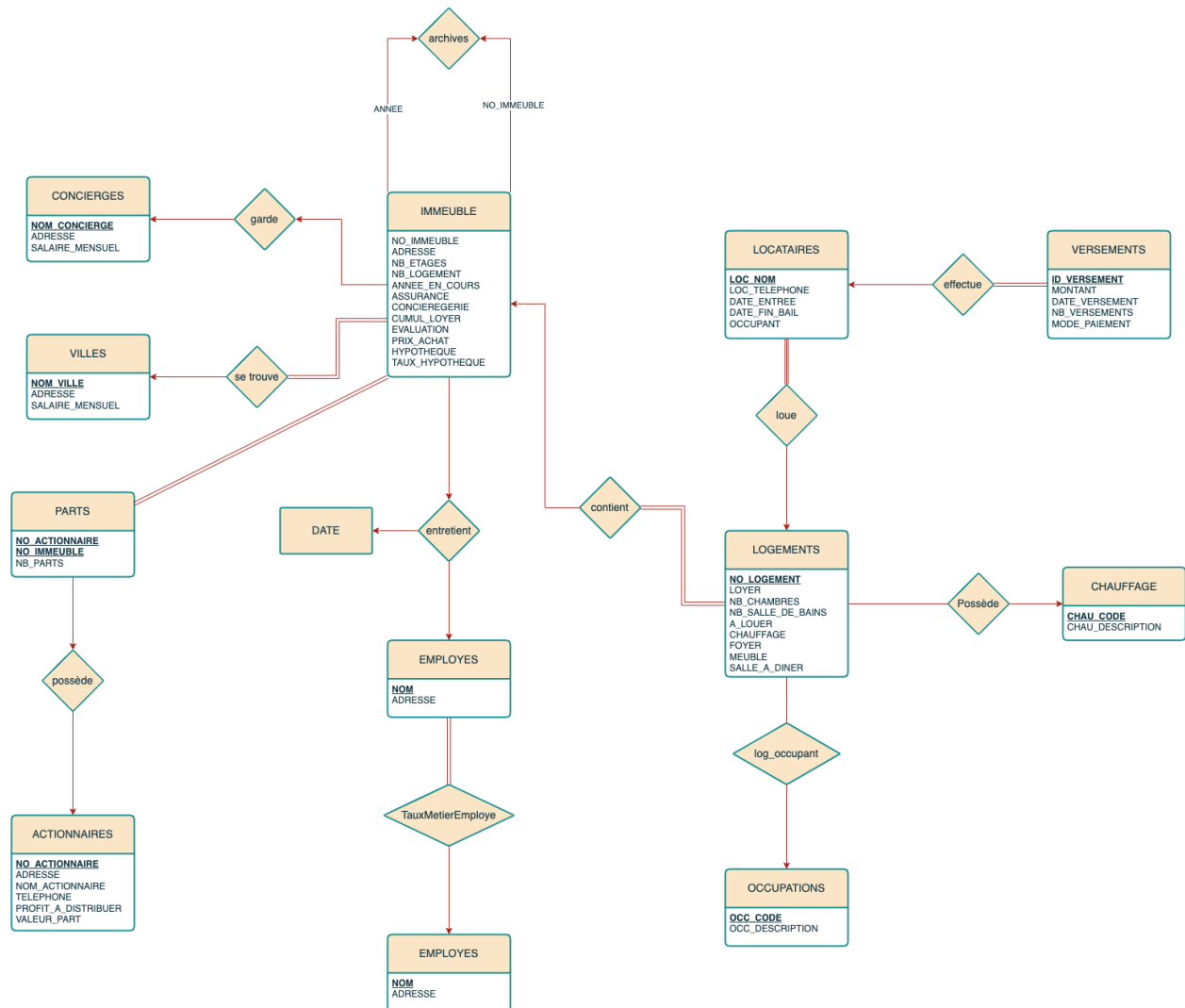
Faire un suivi de paiements des locataires en tenant compte qu'un locataire a le droit de payer en plusieurs versements durant le mois.



## Question 5

### Diagramme ER #3

Les parts des actionnaires changent selon les immeubles.



## Question 6

Voir le fichier *03\_create\_STAN65090008\_KHAI27029800.sql*

```
CREATE TABLE VERSEMENTS (  
    ID NUMBER(4),  
    LOC_NOM VARCHAR2(30),  
    MONTANT NUMBER(4),  
    DATE_VERSEMENT DATE,  
    MODE_PAIEMENT VARCHAR2(11),  
    NB_VERSEMENT NUMBER(2),  
    PRIMARY KEY(ID),  
    FOREIGN KEY(LOC_NOM) REFERENCES LOCATAIRES(LOC_NOM)  
);  
  
CREATE TABLE PARTS (  
    NO_ACTIONNAIRE NUMBER(5),  
    NO_IMMEUBLE NUMBER(4),  
    NB_PARTS NUMBER(7, 3),  
    PRIMARY KEY (NO_ACTIONNAIRE, NO_IMMEUBLE),  
    FOREIGN KEY (NO_ACTIONNAIRE) REFERENCES ACTIONNAIRES(NO_ACTIONNAIRE),  
    FOREIGN KEY (NO_IMMEUBLE) REFERENCES IMMEUBLES(NO_IMMEUBLE)  
);  
  
ALTER TABLE ACTIONNAIRES  
    DROP COLUMN NB_PARTS;
```

## Question 7

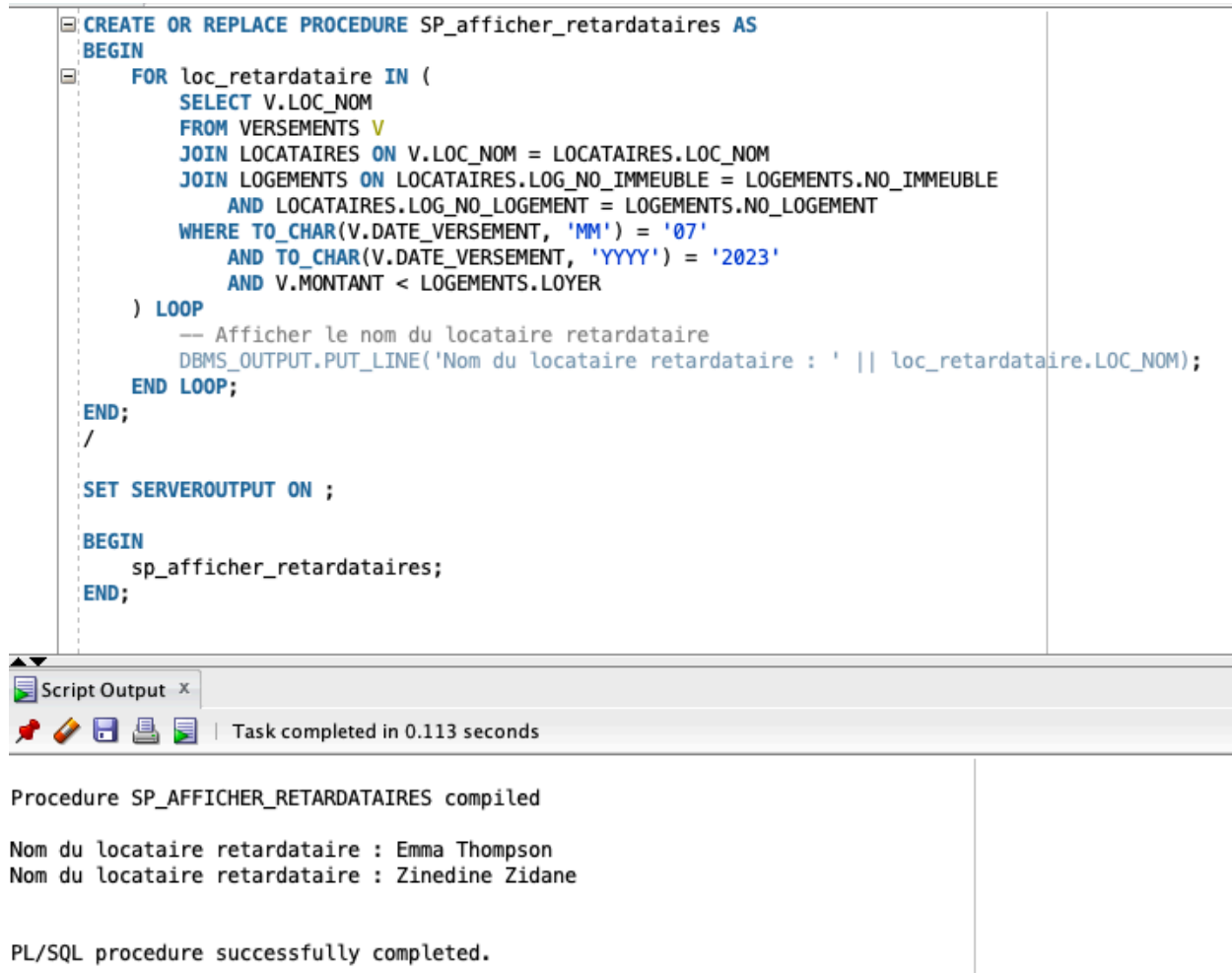
Voir le fichier *03\_create\_STAN65090008\_KHAI27029800.sql*

```
CREATE OR REPLACE PROCEDURE SP_afficher_retardataires AS
BEGIN
    FOR loc_retardataire IN (
        SELECT V.LOC_NOM
        FROM VERSEMENTS V
        JOIN LOCATAIRES ON V.LOC_NOM = LOCATAIRES.LOC_NOM
        JOIN LOGEMENTS ON LOCATAIRES.LOG_NO_IMMEUBLE = LOGEMENTS.NO_IMMEUBLE
            AND LOCATAIRES.LOG_NO_LOGEMENT = LOGEMENTS.NO_LOGEMENT
        WHERE TO_CHAR(V.DATE_VERSEMENT, 'MM') = '07'
            AND TO_CHAR(V.DATE_VERSEMENT, 'YYYY') = '2023'
            AND V.MONTANT < LOGEMENTS.LOYER
    ) LOOP
        -- Afficher le nom du locataire retardataire
        DBMS_OUTPUT.PUT_LINE('Nom du locataire retardataire : ' ||
loc_retardataire.LOC_NOM);
    END LOOP;
END;
/

SET SERVEROUTPUT ON ;

BEGIN
    sp_afficher_retardataires;
END;
```

## (Suite Question 7 - Image)



The screenshot displays the Oracle SQL Developer environment. The top pane shows a PL/SQL procedure named `SP_afficher_retardataires`. The procedure is designed to find tenants who are in arrears as of July 1, 2023. It uses a cursor to select tenant names from the `LOCATAIRES` table, joining it with `VERSEMENTS` and `LOGEMENTS` tables. The selection criteria include the tenant's name, the payment date (month 07, year 2023), and the condition that the payment amount is less than the rent (`LOGEMENTS.LOYER`). The procedure then loops through the results, displaying each tenant's name using `DBMS_OUTPUT.PUT_LINE`.

Below the code editor, the 'Script Output' window shows the results of the procedure's execution. It confirms that the procedure was compiled successfully and lists the names of the tenants in arrears: Emma Thompson and Zinedine Zidane. The overall execution of the PL/SQL procedure is also noted as successful.

```
CREATE OR REPLACE PROCEDURE SP_afficher_retardataires AS
BEGIN
    FOR loc_retardataire IN (
        SELECT V.LOC_NOM
        FROM VERSEMENTS V
        JOIN LOCATAIRES ON V.LOC_NOM = LOCATAIRES.LOC_NOM
        JOIN LOGEMENTS ON LOCATAIRES.LOG_NO_IMMEUBLE = LOGEMENTS.NO_IMMEUBLE
        AND LOCATAIRES.LOG_NO_LOGEMENT = LOGEMENTS.NO_LOGEMENT
        WHERE TO_CHAR(V.DATE_VERSEMENT, 'MM') = '07'
        AND TO_CHAR(V.DATE_VERSEMENT, 'YYYY') = '2023'
        AND V.MONTANT < LOGEMENTS.LOYER
    ) LOOP
        -- Afficher le nom du locataire retardataire
        DBMS_OUTPUT.PUT_LINE('Nom du locataire retardataire : ' || loc_retardataire.LOC_NOM);
    END LOOP;
END;
/

SET SERVEROUTPUT ON ;

BEGIN
    sp_afficher_retardataires;
END;
```

Script Output x

Task completed in 0.113 seconds

Procedure SP\_AFFICHER\_RETARDAIRES compiled

Nom du locataire retardataire : Emma Thompson  
Nom du locataire retardataire : Zinedine Zidane

PL/SQL procedure successfully completed.

## Question 8

## Application #1

Interroge la base de données et affiche la liste des logements à louer d'un immeuble d'où le numéro d'immeuble est passé en paramètre.

The screenshot displays an IDE with a Java project named 'inf3080-tp3'. The main class, 'Main', contains the following code:

```

public static void main(String[] args) {
    int buildingNumber = Integer.parseInt(args[0]);
    try {
        Class.forName("oracle.jdbc.driver.OracleDriver");

        Connection laConnectionJDBC = DriverManager.getConnection(
            "jdbc:oracle:thin@" + BD_URL_SERVER + ":" +
            BD_PORT + ":" + BD_SID, BD_UTILISATEUR, BD_MOT_PASSE);

        Statement requete = laConnectionJDBC.createStatement();

        ResultSet resultats = requete.executeQuery(
            "SELECT * FROM LOGEMENTS NATURAL JOIN IMMEUBLES WHERE A_LOUER = '0' AND NO_IMMEUBLE = " +
            buildingNumber);

        while (resultats.next()) {
            String immeuble = resultats.getString(columnLabel("NO_IMMEUBLE"));
            String nombre_logement = resultats.getString(columnLabel("NO_LOGEMENT"));
            String layer = resultats.getString(columnLabel("LOYER"));
            String nb_chambres = resultats.getString(columnLabel("NB_CHAMBRES"));
            String bains = resultats.getString(columnLabel("NB_SALLE_BAINS"));
            String alouer = resultats.getString(columnLabel("A_LOUER"));
            String chauffage = resultats.getString(columnLabel("CHAUFFAGE"));
            String foyer = resultats.getString(columnLabel("FOYER"));
            String meuble = resultats.getString(columnLabel("MEUBLE"));
            String diner = resultats.getString(columnLabel("SALLE_A_DINER"));

            System.out.println("IMMEUBLE: " + immeuble + "\t\t\tNO_LOGEMENT: " + nombre_logement + "\t\t\tLOYER: " + layer +
                "\t\t\tNOMBRE DE CHAMBRES: " + nb_chambres + "\t\t\tNB_SALLE_BAINS: " + bains + "\t\t\tA_LOUER: " + alouer +
                "\t\t\tCHAUFFAGE: " + chauffage + "\t\t\tFOYER: " + foyer + "\t\t\tMEUBLE: " + meuble + "\t\t\tSALLE_A_DINER: " + diner);
        }

        resultats.close();
        requete.close();
        laConnectionJDBC.close();
    } catch (ClassNotFoundException | SQLException e) {
        e.printStackTrace();
    }
}

```

The Run console shows the output of the application:

```

IMMEUBLE: 1001 NO_LOGEMENT: 1 LOYER: 1500 NOMBRE DE CHAMBRES: 2 NB_SALLE_BAINS: 1 A_LOUER: 0 CHAUFFAGE: 0 FOYER: 0 MEUBLE: 0
SALLE_A_DINER: 0
IMMEUBLE: 1001 NO_LOGEMENT: 2 LOYER: 1100 NOMBRE DE CHAMBRES: 3 NB_SALLE_BAINS: 1 A_LOUER: 0 CHAUFFAGE: 0 FOYER: 0 MEUBLE: 0
SALLE_A_DINER: 0

```

## Application #2 Ajoute un entretien à un logement.

The screenshot displays an IDE with a Java project named 'inf3080-tp3'. The main class 'Main' contains a method `ajouterEntretien` that inserts a new record into the 'ENTRETIENS' table. The database table structure is as follows:

NO_ENTRETIEN	ENT_DATE	NB_HEURES	IMM_NO_IMMEUBLE	LOG_NO_LOGEMENT	EMP_EMP_NOM
1	2023-06-01	3.5	1001		1 John Smith
2	2023-06-02	2.0	1002		2 Jane Doe
3	2023-06-03	1.5	1003		1 Michael Johnson
4	2023-06-04	2.5	1004		2 Emily Williams
5	2023-06-05	3.0	1005		1 Robert Brown
6	2023-06-06	1.0	1006		2 Jennifer Davis
7	2023-06-07	2.5	1007		1 David Wilson
8	2023-06-08	1.5	1008		1 Sarah Taylor
9	2023-06-09	2.0	1009		2 Daniel Anderson
10	2023-06-10	4.5	1010		1 Lisa Thomas

The application output shows the successful execution of the `ajouterEntretien` method, adding a new record to the 'ENTRETIENS' table. The database table is also visible in the 'Database' tab, showing the updated data.