**Nesma AbdElBaqi – Final Project**

**(Steps and output)**

**ـــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــ**

**CREATE CLIENTS TABLE**

CREATE TABLE CLIENTS

(CLIENT\_ID NUMBER(6) NOT NULL CONSTRAINT CLIENT\_PK PRIMARY KEY,

CLIENT\_NAME VARCHAR2(50) NOT NULL,

CLIENT\_ADDRESS VARCHAR2(100),

CLIENT\_NOTES VARCHAR2(500) )

**CREATE Seq/Trigger pair**

This pair is used to make a valid and sequential client id on table CLIENTS

-- Sequence

CREATE SEQUENCE ACCOUNTANT.CLIENTS\_SEQ

START WITH 1

MAXVALUE 999999999999999999999999999

MINVALUE 1

NOCYCLE

CACHE 20

NOORDER;

-- Trigger

CREATE TRIGGER ACCOUNTANT.CLIENTS\_TRG

BEFORE INSERT

ON ACCOUNTANT.CLIENTS

REFERENCING NEW AS New OLD AS Old

FOR EACH ROW

BEGIN

:new.CLIENT\_ID := CLIENTS\_SEQ.nextval;

END CLIENTS\_TRG;

**Inserting data into CLIENTS table**

INSERT INTO CLIENTS(CLIENT\_NAME, CLIENT\_ADDRESS)

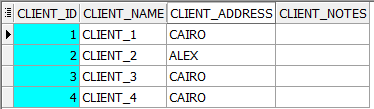
VALUES

('CLIENT\_1', 'CAIRO'),

('CLIENT\_2', 'ALEX'),

('CLIENT\_3', 'CAIRO'),

('CLIENT\_4', 'CAIRO');



**CREATE CONTRACTS TABLE**

CREATE TABLE CONTRACTS

(CONTRACT\_ID NUMBER(6) NOT NULL CONSTRAINT CONTRACT\_PK PRIMARY KEY,

CONTRACT\_STARTDATE DATE,

CONTRACT\_ENDDATE DATE,

PAYMENT\_INSTALLMENT\_NO NUMBER(6),

CONTRACT\_TOTAL\_FEES NUMBER(8,2),

CONTRACT\_DEPOSIT\_FEES NUMBER(8,2),

CLIENT\_ID NUMBER(6),

CONTRACT\_PAYMENT\_TYPE VARCHAR2(25),

NOTES VARCHAR2(1000),

CONSTRAINT CLIENT\_CONTRACT\_FK FOREIGN KEY (CLIENT\_ID) REFERENCES CLIENTS (CLIENT\_ID)

);

**CREATE Seq/Trigger pair**

This pair is used to make a valid and sequential contract id on table CONTRACTS.

-- Sequence

CREATE SEQUENCE ACCOUNTANT.CONTRACTS\_SEQ

START WITH 101

MAXVALUE 999999999999999999999999999

MINVALUE 1

NOCYCLE

CACHE 20

NOORDER;

-- Trigger

CREATE TRIGGER ACCOUNTANT.CONTRACTS\_TRG

BEFORE INSERT

ON ACCOUNTANT.CONTRACTS

REFERENCING NEW AS New OLD AS Old

FOR EACH ROW

BEGIN

:new.CONTRACT\_ID := CONTRACTS\_SEQ.nextval;

END CONTRACTS\_TRG;

/

**CREATE INSTALLMENT PAID TABLE**

CREATE TABLE INSTALLMENT\_PAID

(INSTALLMENT\_ID NUMBER(6) NOT NULL CONSTRAINT INSTALLMENT\_PK PRIMARY KEY,

CONTRACT\_ID NUMBER(6) NOT NULL,

INSTALLMENT\_DATE DATE,

INSTALLMENT\_AMOUNT NUMBER(10,2),

PAID NUMBER(1) ,

CONSTRAINT INSTALL\_CONTRACT\_FK FOREIGN KEY (CONTRACT\_ID) REFERENCES CONTRACTS (CONTRACT\_ID));

**CREATE Seq/Trigger pair**

This pair is used to make a valid and sequential INSTALLMENT\_ID on table INSTALLMENT\_PAID.

-- Sequence

CREATE SEQUENCE ACCOUNTANT.INSTALLMENT\_PAID\_SEQ

START WITH 1

MAXVALUE 999999999999999999999999999

MINVALUE 1

NOCYCLE

CACHE 20

NOORDER;

-- Trigger

CREATE TRIGGER ACCOUNTANT.INSTALLMENT\_PAID\_TRG

BEFORE INSERT

ON ACCOUNTANT.INSTALLMENT\_PAID

REFERENCING NEW AS New OLD AS Old

FOR EACH ROW

BEGIN

:new.INSTALLMENT\_ID := INSTALLMENT\_PAID\_SEQ.nextval;

END INSTALLMENT\_PAID\_TRG;

/

**FUNCTION TO CALC NUMBER OF INSTALLMENTS FOR EACH CLIENT**

CREATE OR REPLACE FUNCTION PAYMENT\_NUMBER (V\_CONTRACT\_ENDDATE DATE ,V\_CONTRACT\_STARTDATE DATE , V\_CONTRACT\_PAYMENT\_TYPE VARCHAR2)

RETURN NUMBER

IS

V\_MONTH NUMBER(6);

BEGIN

V\_MONTH := MONTHS\_BETWEEN(V\_CONTRACT\_ENDDATE,V\_CONTRACT\_STARTDATE);

IF (V\_CONTRACT\_PAYMENT\_TYPE = 'ANNUAL')

THEN

V\_MONTH := V\_MONTH / 12;

ELSIF (V\_CONTRACT\_PAYMENT\_TYPE = 'HALF\_ANNUAL')

THEN

V\_MONTH := (V\_MONTH / 12) \* 2;

ELSIF (V\_CONTRACT\_PAYMENT\_TYPE = 'QUARTER')

THEN

V\_MONTH := (V\_MONTH / 12) \* 4;

ELSIF (V\_CONTRACT\_PAYMENT\_TYPE = 'MONTHLY')

THEN

V\_MONTH := V\_MONTH;

END IF;

RETURN V\_MONTH;

END;

**TRIGGER TO UPDATE THE NUMBER OF INSTALLMENTS FOR EACH CLIENT**

CREATE OR REPLACE TRIGGER CONTRACTS\_PAYMENT\_NUM\_TRG

FOR INSERT OR UPDATE OF CONTRACT\_STARTDATE, CONTRACT\_ENDDATE, CONTRACT\_PAYMENT\_TYPE

ON CONTRACTS

COMPOUND TRIGGER

V\_MONTH NUMBER(6);

V\_CONTRACT\_ID NUMBER(6);

AFTER EACH ROW IS

BEGIN

V\_CONTRACT\_ID := :NEW.CONTRACT\_ID;

V\_MONTH := PAYMENT\_NUMBER (:NEW.CONTRACT\_ENDDATE , :NEW.CONTRACT\_STARTDATE , :NEW.CONTRACT\_PAYMENT\_TYPE);

END AFTER EACH ROW;

AFTER STATEMENT IS

BEGIN

UPDATE CONTRACTS

SET PAYMENT\_INSTALLMENT\_NO = V\_MONTH

WHERE CONTRACT\_ID = V\_CONTRACT\_ID;

END AFTER STATEMENT;

END;

**FUNCTION TO CALC AMOUNT OF MONEY FOR EACH INSTALLMENT**

CREATE OR REPLACE FUNCTION INSTALLMENTS\_AMOUNT (V\_CONTRACT\_TOTAL\_FEES NUMBER , V\_CONTRACT\_DEPOSIT\_FEES NUMBER,V\_PAYMENT\_INSTALLMENT\_NO NUMBER)

RETURN NUMBER

IS

V\_MONEY NUMBER(8,2);

BEGIN

V\_MONEY := (V\_CONTRACT\_TOTAL\_FEES - NVL(V\_CONTRACT\_DEPOSIT\_FEES,0)) /V\_PAYMENT\_INSTALLMENT\_NO;

RETURN V\_MONEY;

END;

**FUNCTION TO CALC THE NEXT INSTALLMENT DATE**

CREATE OR REPLACE FUNCTION NEXT\_INSTALLMENT\_DATE(V\_CONTRACT\_PAYMENT\_TYPE VARCHAR2, V\_CONTRACT\_STARTDATE DATE)

RETURN DATE

IS

V\_RESULT DATE;

BEGIN

IF (V\_CONTRACT\_PAYMENT\_TYPE = 'ANNUAL')

THEN

V\_RESULT := ADD\_MONTHS( V\_CONTRACT\_STARTDATE,12);

ELSIF (V\_CONTRACT\_PAYMENT\_TYPE = 'HALF\_ANNUAL')

THEN

V\_RESULT := ADD\_MONTHS( V\_CONTRACT\_STARTDATE,6);

ELSIF (V\_CONTRACT\_PAYMENT\_TYPE = 'QUARTER')

THEN

V\_RESULT := ADD\_MONTHS( V\_CONTRACT\_STARTDATE,3);

ELSIF (V\_CONTRACT\_PAYMENT\_TYPE = 'MONTHLY')

THEN

V\_RESULT := ADD\_MONTHS( V\_CONTRACT\_STARTDATE,1);

END IF;

RETURN V\_RESULT;

END;

**TRIGGER TO AUTOFILL THE INSTALLMENT\_PAID TABLE AND UPDATING THE CLIENT NOTES.**

CREATE OR REPLACE TRIGGER INSTALLMENT\_AUTOFILL\_TRG

AFTER UPDATE ON CONTRACTS

FOR EACH ROW

DECLARE

V\_MONTHS\_COUNT NUMBER(6);

V\_MONEY NUMBER(8,2);

V\_NEXT\_DATE DATE;

V\_NOTES VARCHAR2(1000);

BEGIN

V\_MONTHS\_COUNT := PAYMENT\_NUMBER (:NEW.CONTRACT\_ENDDATE , :NEW.CONTRACT\_STARTDATE , :NEW.CONTRACT\_PAYMENT\_TYPE);

V\_MONEY := INSTALLMENTS\_AMOUNT (:NEW.CONTRACT\_TOTAL\_FEES , :NEW.CONTRACT\_DEPOSIT\_FEES ,V\_MONTHS\_COUNT ,:NEW.CONTRACT\_PAYMENT\_TYPE);

V\_NEXT\_DATE := :NEW.CONTRACT\_STARTDATE;

--TO UPDATE CLINED NOTES

SELECT CLIENT\_NOTES INTO V\_NOTES

FROM CLIENTS

WHERE CLIENT\_ID = :NEW.CLIENT\_ID;

FOR I IN 1..V\_MONTHS\_COUNT LOOP

INSERT INTO INSTALLMENT\_PAID

(CONTRACT\_ID, INSTALLMENT\_DATE, INSTALLMENT\_AMOUNT)

VALUES(:NEW.CONTRACT\_ID,V\_NEXT\_DATE, V\_MONEY);

V\_NEXT\_DATE := NEXT\_INSTALLMENT\_DATE(:NEW.CONTRACT\_PAYMENT\_TYPE , V\_NEXT\_DATE);

END LOOP;

IF (V\_NOTES = NULL)

THEN

UPDATE CLIENTS SET CLIENT\_NOTES = 'AMOUNT OF PAYMENT: ' || V\_MONEY || ' PAID FOR ' || V\_MONTHS\_COUNT || ' TIMES FOR CONTRACT ID ' || :NEW.CONTRACT\_ID

WHERE CLIENT\_ID = :NEW.CLIENT\_ID;

ELSE

UPDATE CLIENTS SET CLIENT\_NOTES = V\_NOTES||' / '|| 'AMOUNT OF PAYMENT: ' || V\_MONEY || ' PAID FOR ' || V\_MONTHS\_COUNT || ' TIMES FOR CONTRACT ID ' || :NEW.CONTRACT\_ID

WHERE CLIENT\_ID = :NEW.CLIENT\_ID;

END IF;

END;

**AFTER INSERTING:**

