SQL> CREATE TABLE bikes

(

bike\_id NUMBER primary key,

brand VARCHAR2(10),

model VARCHAR2(10),

manufa\_year NUMBER(4),

color VARCHAR2(10),

price NUMBER not null

);

Table created.

SQL> desc bikes;

Name Null? Type

----------------------------------------- -------- ----------------------------

BIKE\_ID NOT NULL NUMBER

BRAND VARCHAR2(10)

MODEL VARCHAR2(10)

MANUFA\_YEAR NUMBER(4)

COLOR VARCHAR2(10)

PRICE NOT NULL NUMBER

SQL> insert into bikes values (1, 'Hero', 'XPulse 200', 2022, 'Red', 100000);

1 row created

SQL> insert into bikes values (2, 'RE', 'Classic', 2021, 'Black', 90000);

1 row created.

SQL> insert into bikes values (3, 'Honda', 'CBR500R', 2020, 'Blue', 120000);

1 row created.

SQL> insert into bikes values (4, 'Yamaha', 'MT-07', 2021, 'Silver', 150000);

1 row created.

SQL> insert into bikes values (5, 'Ducati', 'PanigaleV4', 2022, 'Red', 200000);

1 row created.

DECLARE

BEGIN

dbms\_output.put\_line(‘Welcome to My DBMS Record’);

END;

/

DECLARE

n number

begin

dbms.output.put\_line('Multiplication Table ');

n:=&n;

if n<0 then

dbms.output.put\_line('Please Enter a positive number ');

return;

end if;

dbms.output.put\_line('Table'|| n);

for i in 1..12 loop

dbms.output.put\_line(i||'\*'|| n||'='||(n\*i));

end loop;

end;

/

SET SERVEROUTPUT ON;

DECLARE

a NUMBER := 0;

b NUMBER := 1;

c NUMBER;

n NUMBER;

i NUMBER := 0;

BEGIN

DBMS\_OUTPUT.PUT\_LINE('Fibonacci Sequence');

n := &n;

IF n < 0 THEN

DBMS\_OUTPUT.PUT\_LINE('Please Enter a positive number ');

RETURN;

END IF;

WHILE i <= n LOOP

DBMS\_OUTPUT.PUT\_LINE(a);

c := a + b;

a := b;

b := c;

i := i + 1;

END LOOP;

END;

/

Cursor

DECLARE

total\_price NUMBER := 0;

CURSOR bike\_cursor IS

SELECT price FROM bikes;

bike\_price bikes.price%TYPE;

BEGIN

OPEN bike\_cursor;

LOOP

FETCH bike\_cursor INTO bike\_price;

EXIT WHEN bike\_cursor%NOTFOUND;

total\_price := total\_price + bike\_price;

END LOOP;

CLOSE bike\_cursor;

DBMS\_OUTPUT.PUT\_LINE('Total Price of Bikes in the Shop: ' || total\_price);

END;

/

Trigger

CREATE TABLE shopsummary   
(

total\_price NUMBER,

total\_bikes\_count NUMBER

);

CREATE TRIGGER update\_shop\_summary

AFTER INSERT ON bikes

FOR EACH ROW

DECLARE

bike\_price bikes.price%TYPE;

BEGIN

SELECT price INTO bike\_price FROM bikes WHERE bike\_id = :NEW.bike\_id;

UPDATE shop\_summary

SET total\_price = NVL(total\_price, 0) + bike\_price,

total\_bikes\_count = NVL(total\_bikes\_count, 0) + 1;

EXCEPTION

WHEN NO\_DATA\_FOUND THEN

INSERT INTO shop\_summary(total\_price, total\_bikes\_count)

VALUES (bike\_price, 1);

END;

/

SQL> CREATE TABLE rental\_records

(

rental id NUMBER PRIMARY KEY,

bike\_id NUMBER REFERENCES bikes(bike\_id),

customer name VARCHAR2(10),

rental fee NUMBER

);

CREATE OR REPLACE TRIGGER logbikes

AFTER UPDATE OF price ON bikes

FOR EACH ROW

BEGIN

INSERT INTO rental\_records(rental\_id,customer\_name,bike\_id,rental\_fee)

VALUES(:NEW.bike\_id,:NEW.customer\_name,:NEW.price);

END;

/