DROP TABLE student;

TRIGGERS

CREATE TABLE student (

ROLLNO INT NOT NULL,

SNAME VARCHAR(45) NOT NULL,

AGE VARCHAR(45) NOT NULL,

COURSE VARCHAR(45) NOT NULL,

PRIMARY KEY (ROLLNO));

INSERT into STUDENT values(11, 'ANU', 20, 'BSC');

INSERT into STUDENT values(12, 'ASHA', 21, 'BCOM');

INSERT into STUDENT values(13, 'CHETAN', 20, 'BSC');

CURSORS

DECLARE

CURSOR student\_cursor IS SELECT sname FROM Student ;

snm Student.sname %type;

BEGIN

OPEN student\_cursor;

IF student\_cursor%ISOPEN = FALSE then

dbms\_output.put\_line('Cannot open cursor');

ELSE

LOOP

FETCH student\_cursor INTO snm;

IF student\_cursor%NOTFOUND then

Exit;

END IF;

dbms\_ output.put\_line('' ||snm);

END LOOP;

dbms\_output.put\_line('Total Records: ' ||student\_cursor%rowcount);

CLOSE student\_cursor;

END;

DECLARE

CURSOR student\_cursor IS SELECT sname FROM Student;

BEGIN

FOR snm IN student\_cursor LOOP

dbms\_output.put\_line('' || snm);

END LOOP;

END;

set serveroutput on;

DECLARE

CURSOR showRec(sno student.rollno%type) IS SELECT sname, course FROM student WHERE rollno=sno;

a student.sname%type;

b student.course%type;

c student.rollno%type;

BEGIN

d := &rollno;

OPEN showRec(d);

IF showRec%Isopen = FALSE then

dbms\_output.put\_line('Cannot open Cursor');

ELSE

LOOP

FETCH showRec into a,b;

EXIT WHEN showRec%NOTFOUND;

dbms\_output.put\_line(a|| '' ||b);

END LOOP;

End IF;

CLOSE showRec;

END;

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

DROP TABLE regions;

CREATE TABLE regions

(

region\_id NUMBER GENERATED BY DEFAULT AS IDENTITY

START WITH 5 PRIMARY KEY,

region\_name VARCHAR2( 50 ) NOT NULL

);

DROP TABLE countries;

-- countries table

CREATE TABLE countries

(

country\_id CHAR( 2 ) PRIMARY KEY ,

country\_name VARCHAR2( 40 ) NOT NULL,

region\_id NUMBER , -- fk

CONSTRAINT fk\_countries\_regions FOREIGN KEY( region\_id )

REFERENCES regions( region\_id )

ON DELETE CASCADE

);

DROP TABLE locations;

-- location

CREATE TABLE locations

(

location\_id NUMBER GENERATED BY DEFAULT AS IDENTITY START WITH 24

PRIMARY KEY ,

address VARCHAR2( 255 ) NOT NULL,

postal\_code VARCHAR2( 20 ) ,

city VARCHAR2( 50 ) ,

state VARCHAR2( 50 ) ,

country\_id CHAR( 2 ) , -- fk

CONSTRAINT fk\_locations\_countries

FOREIGN KEY( country\_id )

REFERENCES countries( country\_id )

ON DELETE CASCADE

);

DROP TABLE warehouses;

-- warehouses

CREATE TABLE warehouses

(

warehouse\_id NUMBER

GENERATED BY DEFAULT AS IDENTITY START WITH 10

PRIMARY KEY,

warehouse\_name VARCHAR( 255 ) ,

location\_id NUMBER( 12, 0 ), -- fk

CONSTRAINT fk\_warehouses\_locations

FOREIGN KEY( location\_id )

REFERENCES locations( location\_id )

ON DELETE CASCADE

);

DROP TABLE employees;

-- employees

CREATE TABLE employees

(

employee\_id NUMBER

GENERATED BY DEFAULT AS IDENTITY START WITH 108

PRIMARY KEY,

first\_name VARCHAR( 255 ) NOT NULL,

last\_name VARCHAR( 255 ) NOT NULL,

email VARCHAR( 255 ) NOT NULL,

phone VARCHAR( 50 ) NOT NULL ,

hire\_date DATE NOT NULL ,

manager\_id NUMBER( 12, 0 ) , -- fk

job\_title VARCHAR( 255 ) NOT NULL,

CONSTRAINT fk\_employees\_manager

FOREIGN KEY( manager\_id )

REFERENCES employees( employee\_id )

ON DELETE CASCADE

);

-- product category

CREATE TABLE product\_categories

(

category\_id NUMBER

GENERATED BY DEFAULT AS IDENTITY START WITH 6

PRIMARY KEY,

category\_name VARCHAR2( 255 ) NOT NULL

);

-- products table

CREATE TABLE products

(

product\_id NUMBER

GENERATED BY DEFAULT AS IDENTITY START WITH 289

PRIMARY KEY,

product\_name VARCHAR2( 255 ) NOT NULL,

description VARCHAR2( 2000 ) ,

standard\_cost NUMBER( 9, 2 ) ,

list\_price NUMBER( 9, 2 ) ,

category\_id NUMBER NOT NULL ,

CONSTRAINT fk\_products\_categories

FOREIGN KEY( category\_id )

REFERENCES product\_categories( category\_id )

ON DELETE CASCADE

);

DROP TABLE customers;

-- customers

CREATE TABLE customers

(

customer\_id NUMBER

GENERATED BY DEFAULT AS IDENTITY START WITH 320

PRIMARY KEY,

name VARCHAR2( 255 ) NOT NULL,

address VARCHAR2( 255 ) ,

website VARCHAR2( 255 ) ,

credit\_limit NUMBER( 8, 2 )

);

-- contacts

CREATE TABLE contacts

(

contact\_id NUMBER

GENERATED BY DEFAULT AS IDENTITY START WITH 320

PRIMARY KEY,

first\_name VARCHAR2( 255 ) NOT NULL,

last\_name VARCHAR2( 255 ) NOT NULL,

email VARCHAR2( 255 ) NOT NULL,

phone VARCHAR2( 20 ) ,

customer\_id NUMBER ,

CONSTRAINT fk\_contacts\_customers

FOREIGN KEY( customer\_id )

REFERENCES customers( customer\_id )

ON DELETE CASCADE

);

DROP TABLE orders;

-- orders table

CREATE TABLE orders

(

order\_id NUMBER

GENERATED BY DEFAULT AS IDENTITY START WITH 106

PRIMARY KEY,

customer\_id NUMBER( 6, 0 ) NOT NULL, -- fk

status VARCHAR( 20 ) NOT NULL ,

salesman\_id NUMBER( 6, 0 ) , -- fk

order\_date DATE NOT NULL ,

CONSTRAINT fk\_orders\_customers

FOREIGN KEY( customer\_id )

REFERENCES customers( customer\_id )

ON DELETE CASCADE,

CONSTRAINT fk\_orders\_employees

FOREIGN KEY( salesman\_id )

REFERENCES employees( employee\_id )

ON DELETE SET NULL

);

-- order items

CREATE TABLE order\_items

(

order\_id NUMBER( 12, 0 ) , -- fk

item\_id NUMBER( 12, 0 ) ,

product\_id NUMBER( 12, 0 ) NOT NULL , -- fk

quantity NUMBER( 8, 2 ) NOT NULL ,

unit\_price NUMBER( 8, 2 ) NOT NULL ,

CONSTRAINT pk\_order\_items

PRIMARY KEY( order\_id, item\_id ),

CONSTRAINT fk\_order\_items\_products

FOREIGN KEY( product\_id )

REFERENCES products( product\_id )

ON DELETE CASCADE,

CONSTRAINT fk\_order\_items\_orders

FOREIGN KEY( order\_id )

REFERENCES orders( order\_id )

ON DELETE CASCADE

);

-- inventories

CREATE TABLE inventories

(

product\_id NUMBER( 12, 0 ) , -- fk

warehouse\_id NUMBER( 12, 0 ) , -- fk

quantity NUMBER( 8, 0 ) NOT NULL,

CONSTRAINT pk\_inventories

PRIMARY KEY( product\_id, warehouse\_id ),

CONSTRAINT fk\_inventories\_products

FOREIGN KEY( product\_id )

REFERENCES products( product\_id )

ON DELETE CASCADE,

CONSTRAINT fk\_inventories\_warehouses

FOREIGN KEY( warehouse\_id )

REFERENCES warehouses( warehouse\_id )

ON DELETE CASCADE

);

CREATE VIEW sales AS

SELECT customer\_id,

SUM(unit\_price \* quantity) total,

ROUND(SUM(unit\_price \* quantity) \* 0.05) credit

FROM order\_items

INNER JOIN orders USING (order\_id)

WHERE status = 'Shipped'

GROUP BY customer\_id;

DECLARE

l\_budget NUMBER := 1000000;

-- cursor

CURSOR c\_sales IS

SELECT \* FROM sales

ORDER BY total DESC;

-- record

r\_sales c\_sales%ROWTYPE;

BEGIN

-- reset credit limit of all customers

UPDATE customers SET credit\_limit = 0;

OPEN c\_sales;

LOOP

FETCH c\_sales INTO r\_sales;

EXIT WHEN c\_sales%NOTFOUND;

-- update credit for the current customer

UPDATE

customers

SET

credit\_limit =

CASE WHEN l\_budget > r\_sales.credit

THEN r\_sales.credit

ELSE l\_budget

END

WHERE

customer\_id = r\_sales.customer\_id;

-- reduce the budget for credit limit

l\_budget := l\_budget - r\_sales.credit;

DBMS\_OUTPUT.PUT\_LINE( 'Customer id: ' ||r\_sales.customer\_id ||

' Credit: ' || r\_sales.credit || ' Remaining Budget: ' || l\_budget );

-- check the budget

EXIT WHEN l\_budget <= 0;

END LOOP;

CLOSE c\_sales;

END;

CURSOR c\_sales IS

SELECT \* FROM sales

ORDER BY total DESC;