

Exercise Four



4.1 Create a view called `Products_vw`, include the following tables and columns ...

`Products_Group` – `Pgr_Group_Name`

`Products` – `Pro_Name`, `Pro_Price`

`Products_Sales` – `Pss_Transaction_Date`, `Pss_Number_of_Units`, `Pss_Sales_Number`

Add an additional column which represents `Pss_Number_of_Units` multiplied by `Pro_Price`

Appendix 8NF10 is a graphical representation of the relationship between the above tables.

Save the create script as `c:\course\ex4_1.sql`. Running the view should produce 82150 records.

4.2 Create an Instead of trigger for `Product_vw` which when an insert statement is issued inserts into `Products_Sales` the Sales data, the `Pss_Sales_Number` is obtained from the `Prod_Sales_Seq` sequence. In addition, remember to populate the audit columns.

Call the trigger `pro_ii_trg`. Save the create script as `c:\course\ex4_2.sql`.

Exercise Four



4.3 Test the Pro_ii_trg by issuing the following statement ...

```
insert into products_vw
(group_name
,product_name
,sales_date
,units_sold)
values ('BUILDING', 'FINE SAND', to_date('01-FEB-2003','DD-MON-RRRR'),99);
```

Save as c:\course\ex4_3.sql

Select from the Products_Sales table to confirm that the insert occurred.

Exercise Four



4.4 Create an instead of trigger for Product_vw for update. Call the trigger Pro_iu_trg. Allow updates to transaction date, product number, number of units. Save as c:\course\ex4_4.sql. Test the trigger works by issuing the following statements ...

```
update products_vw
set product_name = 'GNOME'
where sales_number = <number from 4.3>
```

```
update products_vw
set units_sold = 888
where sales_number = <number from 4.3>
```

```
update products_vw
set pro_price = 888
where sales_number = <number from 4.3>
```

4.5 Create an instead of trigger for Product_vw for delete. Prevent any deletions and display a suitable error message. Call the trigger Pro_id_trg. Save the script as c:\course\ex4_5.sql.

Exercise Four



4.6 Create a temporary table called Test_Drop, create several columns of your own choosing. Create a System trigger which prevents a user from dropping the Test_Drop table and displays an appropriate error message.

Save the script as c:\course\4_6.sql

