

1. Develop a UDF to receive a product number and returns the total number product sold/

```
create or replace function udf_total_product(p_prod_number int)
return int
as
v_prod_sold int;
v_prod_no int;
cursor c1 is
select ol.product_no ,count(o.order_no) from orders o join orderline ol
on o.order_no=ol.order_no
group by ol.product_no;
begin
open c1;
loop
fetch c1 into v_prod_no,v_prod_sold;
exit when c1%notfound;

if v_prod_no=p_prod_number then
return v_prod_sold;
end if;
end loop;
close c1;
end;
/
-- select udf total product(120) from dual;
```

Script Output x



Task completed in 0.032 seconds

Function UDF\_TOTAL\_PRODUCT compiled

Develop a UDF to receive a product number and year, and returns the total number product sold for that year

```

9  -- select udf_total_product_yearly(180,2001) from dual
10 create or replace function udf_total_product_yearly(p_prod_number int,p_year int)
11 return int
12 as
13 v_prod_sold int;
14 v_year int;
15 v_prod_no int;
16 cursor c1 is
17 select ol.product_no ,extract(year from o.order_date),count(o.order_no)  from orders  o join orderline ol
18 on o.order_no=ol.order_no
19 join returnprod rp on ol.orderline_no=rp.orderline_no
20 group by ol.product_no,extract(year from o.order_date)
21 having ol.product_no = p_prod_number  ;
22
23 begin

```

Script Output x Query Result x

SQL | All Rows Fetched: 1 in 0 seconds

UDF\_TOTAL\_PRODUCT\_YEARLY(180,2001)

1	4
---	---

Develop a UDF that will receive a product number, a month, a year, and returns the sales for that year

```

70 select ol.product_no ,extract(year from o.order_date) ,(sum(o.subtotal)-sum(nvl(amount_refunded,0 ))) as sale
71 from orders  o join orderline ol
72 on o.order_no=ol.order_no
73 join returnprod rp on ol.orderline_no=rp.orderline_no
74 group by ol.product_no,extract(year from o.order_date)
75 having ol.product_no=p_prod_no;
76
77 begin
78 open c1;
79
80 loop
81 fetch c1 into v_prod_no,v_year,v_sales;
82 exit when c1%notfound;
83 if v_prod_no=p_prod_no and v_year=p_year then
84 return v_sales;
85 else return null;
86 end if;
87 end loop;
88 close c1;
89 exception WHEN others then
90 return null;
91 end;
92

```

Script Output x Query Result x

Task completed in 0.032 seconds

Function UDF\_TOTAL\_PRODUCT\_DETAI compiled

Develop a UDF that will receive a product number, a month, a year, and returns the sales for that year

Develop a SQL program to call these UDF and produce the following report

Product#, product\_name, year, Number\_Product\_sold, total\_sale

The screenshot shows a SQL IDE with a 'Query Builder' tab. The code in the editor is as follows:

```
126 dbms_output.put_line(rpad(v_prod_no,5) || rpad( v_prodname,15) ||rpad( v_year,15) ||rpad( v_prod_sold,15) ||rpad( v_total_sale,15)
127 end loop;
128 close cl;
129 end;
130 /
131 -- Develop udf to calculate subtotal tax and shipping chagre..
132 create or replace function gross_sale(p_order_no int)
133 return int
134 IS
135 v_orderno int ;
136 v_grosssale int;
137 begin
138 select o.order_no, sum((p.unit_price*ol.qty)-nvl(rp.AMOUNT_REFUNDED,0)) into v_orderno ,v_grosssale
139 from orders o
140 join orderline ol
141 on o.order_no=ol.order_no
142 join product p
143 on p.product_no=ol.product_no
144 left join returnprod rp on
145 ol.ORDERLINE_NO=rp.ORDERLINE_NO
146 group by o.order_no
147 having o.order_no = p_order_no;
148 return v_grosssale;
```

Below the code editor, the 'Script Output' tab shows the message: 'Function GROSS\_SALE compiled'. The 'Query Result' tab is also visible but empty.

The shipping charge has been update and then on selecting the select \* form orders;

The screenshot shows a SQL IDE with a 'Query Builder' tab. The code in the editor is as follows:

```
149 end;
150
151 -- to calculate the tax
152
153 create or replace function udf_tax_calculation(p_customer_no int)
154 return int
155 as
156 v_custnomer int ;
157 v_order_no int;
158
159 v_tax float;
160 select * from orders;
161
162 cursor cl is
```

Below the code editor, the 'Query Result' tab shows the results of the query 'select \* from orders;'. The results are displayed in a table with 11 columns: ORDER\_NO, ORDER\_DATE, SHIP\_DATE, SHIPPING\_METHOD, TAX\_STATUS, SUBTOTAL, TAX\_AMT, SHIPPING\_CHARGE, TOTAL\_AMT, CUSTOMER\_NO, and EMPLOYEE\_NO. The table contains 10 rows of data.

ORDER_NO	ORDER_DATE	SHIP_DATE	SHIPPING_METHOD	TAX_STATUS	SUBTOTAL	TAX_AMT	SHIPPING_CHARGE	TOTAL_AMT	CUSTOMER_NO	EMPLOYEE_NO
1	1193 10-MAY-02	13-MAY-02	1-day	n	17800	0	1651.65	20470	1099	1042
2	1194 11-MAY-02	14-MAY-02	1day-air	y	64870	726	2022	75326.9...	1100	1042
3	1196 13-MAY-02	16-MAY-02	Ground	n	483605	0	1112.8	507785.25	1000	1047
4	1197 14-MAY-02	17-MAY-02	1-day	n	344720	0	1493.1	396428	1023	1047
5	1200 17-MAY-02	20-MAY-02	1-day	y	590070	30569	2889.6	709149....	1068	1047
6	1201 18-MAY-02	21-MAY-02	Ground	y	13200	714	145.2	14574.3...	1072	1047
7	1202 19-MAY-02	22-MAY-02	2-day	n	6725	0	346.9	7397.5	1076	1047
8	1203 20-MAY-02	23-MAY-02	Ground	y	350430	22970	453.55	390921....	1000	1047
9	1204 21-MAY-02	24-MAY-02	1-day	n	436155	0	806.25	501578.25	1023	1047
10	1205 22-MAY-02	25-MAY-02	1-day	y	31610	2094	1876.8	38445.9...	1060	1047

## Tax calculation

The product no is the parameter and it will return the total tax for that product/

```
152 |
153 | create or replace function udf_tax_calculation(p_customer_no int)
154 | return int
155 | as
156 | v_custnumber int ;
157 | v_order_no int;
158 | v_tax float;
159 | cursor cl is
160 | select o.order_no,c.customer_no,sum(t.tax_rate*o.subtotal) as tax_amount from customer c
161 | join orders o
162 | on o.customer_no=c.customer_no
163 | join tax t
164 | on c.state=t.state
165 | group by o.order_no,c.customer_no
166 | HAVING c.customer_no =p_customer_no ;
167 | begin
168 | open cl;
169 | loop
170 |   fetch cl into v_order_no,v_custnumber,v_tax;
171 |   exit when cl%notfound;
172 |   return v_tax;
```

Script Output x Query Result x

SQL | All Rows Fetched: 1 in 0.016 seconds

UDF_TAX_CALCULATION(1099)	
1	199