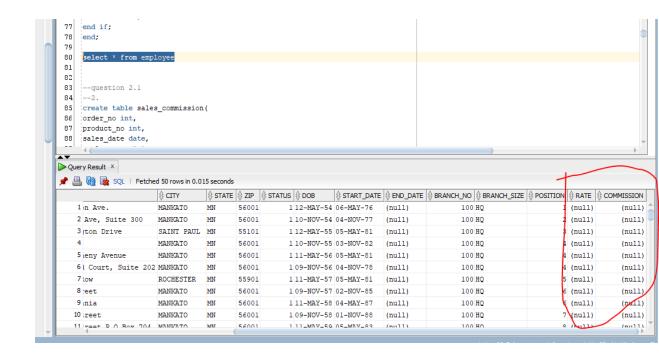## 2.1 Add employee commission to db:

1. update the commission rate for the sales related employees only
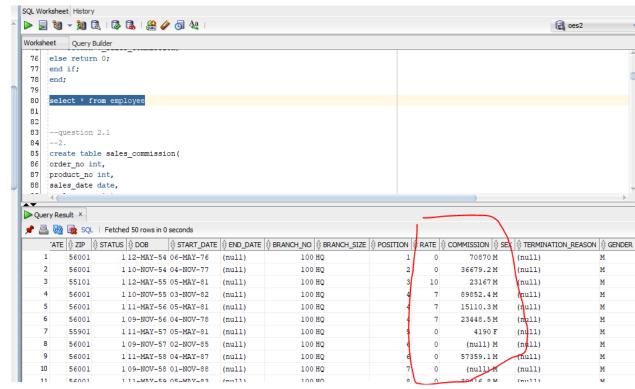
the sp will check if the position is 3 or 4 , if it is then the commission is applied else not a.pplied.

The procedure is in the .txt file and the

execute sp_update_commission_rate; will add the rate to the employee and

udf_comm_amt is used to find the commission amount for those employee only

After execution of the sp and function

Worksheet | Query Builder

```
76   else return 0;
77   end if;
78   end;
79
80   select * from employee
81
82
83   --question 2.1
84   --2.
85   create table sales_commission(
86   order_no int,
87   product_no int,
88   sales_date date,
```

Query Result ×

SQL | Fetched 50 rows in 0 seconds

ATE	ZIP	STATUS	DOB	START_DATE	END_DATE	BRANCH_NO	BRANCH_SIZE	POSITION	RATE	COMMISSION	SEX	TERMINATION_REASON	GENDER
1	56001	1	12-MAY-54	06-MAY-76	(null)	100	HQ	1	0	70870	M	(null)	M
2	56001	1	10-NOV-54	04-NOV-77	(null)	100	HQ	2	0	36679.2	M	(null)	M
3	55101	1	12-MAY-55	05-MAY-81	(null)	100	HQ	3	10	23167	M	(null)	M
4	56001	1	10-NOV-55	03-NOV-82	(null)	100	HQ	4	7	89852.4	M	(null)	M
5	56001	1	11-MAY-56	05-MAY-81	(null)	100	HQ	4	7	15110.3	M	(null)	M
6	56001	1	09-NOV-56	04-NOV-78	(null)	100	HQ	4	7	23448.5	M	(null)	M
7	55901	1	11-MAY-57	05-MAY-81	(null)	100	HQ	5	0	4190	F	(null)	M
8	56001	1	09-NOV-57	02-NOV-85	(null)	100	HQ	6	0	(null)	M	(null)	M
9	56001	1	11-MAY-58	04-MAY-87	(null)	100	HQ	6	0	57359.1	M	(null)	M
10	56001	1	09-NOV-58	01-NOV-88	(null)	100	HQ	7	0	(null)	M	(null)	M
11	56001	1	11-MAY-59	05-MAY-83	(null)	100	HQ	8	0	3016.8	M	(null)	M

Here we see the rate is non zero only for position is 3 or 4 and commission is applied to those only.

Question 2.2

Table return_commission and sales_commission ,orders_mv created.

trigger sales_commission is fired when new rows are inserted in orders_mv;

```
SELECT * FROM SALES_COMMISSION;

EXECUTE p_commission_sales;
```

ript Output ×   Query Result ×

SQL | All Rows Fetched: 0 in 0 seconds

| ORDER_NO | PRODUCT... | SALES_D... | EMPLOYE... | COMMISS... |

AFTER

AFTER executing sp

```
 97
 98   SELECT * FROM SALES_COMMISSION;
 99
100   EXECUTE p_commission_sales;
101
102
```

Script Output ×   Query Result ×

| Task completed in 0.625 seconds

```
order no : 2360
order no : 2371
order no : 2372
order no : 2432
order no : 2434
order no : 2444
order no : 2453
order no : 2465
order no : 2468
order no : 2477


PL/SQL procedure successfully completed.
```

```
     EXECUTE p_commission_sales;
L    SELECT * FROM SALES_COMMISSION;
2
```

Script Output ×   ▷ Query Result ×

🖨 ⏪ 🗙 SQL | Fetched 50 rows in 0 seconds

	ORDER_NO	PRODUCT_NO	SALES_DATE	EMPLOYEE_NO	COMMISSION_AMT
1	1193	190	10-MAY-02	1042	890
2	1193	120	10-MAY-02	1042	890
3	1193	130	10-MAY-02	1042	890
4	1193	140	10-MAY-02	1042	890
5	1193	150	10-MAY-02	1042	890
6	1194	120	11-MAY-02	1042	3243.5
7	1194	130	11-MAY-02	1042	3243.5
8	1194	140	11-MAY-02	1042	3243.5
9	1194	110	11-MAY-02	1042	3243.5
10	1194	130	11-MAY-02	1042	3243.5
11	1194	140	11-MAY-02	1042	3243.5
12	1196	140	13-MAY-02	1047	24180.25

Here the sales commission table is populated after the sp is executed which in turn fire the trigger..

Similarly, the return sales trigger and sp is created.The script is in script file..

Report1:

```
97
98 □select sc.order_no,sc.product_no,sc.employee_no,extract(year from sales_date)as years,extract(month from sales_date) as months
99     ,sum(sc.commission_amt)as commission,sum(rc.commission_amt) as return_commission,
00       sum(sc.commission_amt)-sum(rc.commission_amt) as "net commission"
01  from sales_commission sc
02  left join RETURN_COMMISSION rc on
03       sc.order_no=rc.order_no
04  group by sc.order_no,sc.product_no,sc.employee_no,
05         extract(year from sales_date),
06         extract(month from sales_date);
07  drop table commission_info;
08 □create table commission_info as
09
10     select sc.order_no,sc.product_no,sc.employee_no,extract(year from sales_date) as years,to_char(sales_date,'MON')as months
11       ,sum(NVL(sc.commission_amt,0))as commission,sum(NVL(rc.commission_amt,0)) as return_commission,
12        sum(NVL(sc.commission_amt,0))-sum(NVL(rc.commission_amt,0)) as "net commission"
13       from sales_commission sc
```

Script Output × | Query Result × | Query Result 1 ×

SQL | Fetched 50 rows in 0.094 seconds

	ORDER_NO	PRODUCT_NO	EMPLOYEE_NO	YEARS	MONTHS	COMMISSION	RETURN_COMMISSION	net commission
1	1193	140	1042	2002	5	890	(null)	(null)
2	1197	180	1047	2002	5	17236	(null)	(null)
3	1201	120	1047	2002	5	660	(null)	(null)
4	1202	130	1047	2002	5	336.25	(null)	(null)
5	1203	160	1047	2002	5	17521.5	(null)	(null)
6	1204	170	1047	2002	5	21807.75	(null)	(null)

ignment 1.sql | Line 298 Column 1 | Insert | | Window

Report 2.5

-- 2.5 report2 Here we use the oracle pivot function for cross tabular work
select * from
(select employee_no,months,"net commission"
from commission_info
)
pivot(
sum("net commission") for months in
('JAN','FEB','MAR','APR','MAY','JUN','JLY','AUG','SEP','OCT','NOV','DEC')

)

```
-- 2.5 report2
select * from
(select  employee_no,months,"net commission"
from commission_info
)
pivot(
sum( "net commission") for months  in ('JAN','FEB','MAR','APR','MAY','JUN','JLY','AUG','SEP','OCT','NOV','DEC')

)
```

script Output × ▶ Query Result × ▶ Query Result 1 ×

🖳 🕸 📄 SQL | All Rows Fetched: 70 in 0.125 seconds

EMPLOYEE_NO	'JAN'	'FEB'	'MAR'	'APR'	'MAY'	'JUN'	'JLY'	'AUG'	'SEP'	'OCT'	'NOV'	'DEC'
1	1000	0	(null)	(null)	(null)	(null)	(null)	(null)	(null)	(null)	(null)	(null)	(null)
2	1003	517552	402515.4	28954.8	619716.3	930048.7	24307.5	(null)	15080.8	13928.6	(null)	(null)	(null)
3	1013	(null)	1983.5	(null)	(null)	(null)	(null)	(null)	(null)	(null)	(null)	(null)	(null)
4	1036	(null)	8693.3	467686.8	935049.5	1059464.35	19051487	(null)	27506.85	847086.45	39962.65	69052.2	42579.25
5	1041	(null)	0	0	0	0	0	(null)	0	0	0	0	0
6	1011	(null)	(null)	(null)	(null)	(null)	(null)	(null)	0	0	0	0	0
7	1030	(null)	0	0	0	0	0	(null)	(null)	(null)	(null)	(null)	(null)
8	1042	(null)	995	581.75	1155637.5	1015678.25	9651.5	(null)	64404.25	35232.5	94270.25	52508.75	131490.75
9	1062	(null)	(null)	(null)	(null)	(null)	(null)	(null)	0	0	0	0	0
10	1074	(null)	(null)	(null)	(null)	(null)	(null)	(null)	0	0	0	0	0
11	1046	(null)	(null)	(null)	(null)	212100.25	995667.5	(null)	(null)	(null)	(null)	(null)	(null)

calculate the **employee_commission**
For this the rate is updated in the employee table for the given business rule using the function.
1.function **udf_commission_rate** to calculate the year of experience.
2. procedure **sp_update_commission_rate** to update the rate for the emplpoyee based on year of experience returned by udf_commission_rate
Table employee_rate_audit is created to verify the rate has been update in the employee table after executing the procedure sp_update_commission_rate.
The trigger **rate_audit** is created to insert the record in employee rate_audit table that holds the before and after insert value for rate.

      create or replace trigger rate_audit
      after update on employee
      for each row
      declare
      v_employee_no int;
      begin
      -- select employee_no into v_employee_no from dual;
      insert into employee_rate_audit
      (employee_no,rate_before_update,rate_after_update,update_time)values
          (:new.employee_no,:old.rate,:new.rate,sysdate) ;
      end;

Now execute select * from employee_rate_audit; before executing the procedure

```sql
          end;
57
58    select * from employee_rate_audit;
59    select * from employee;
60    execute sp_update_commission_rate;
61    select * from employee;
62    select * from employee_rate_audit;
63    -- commission table creation
64    create table commission as (select e.employee_no,nvl
```

Script Output ×   ▶ Query Result ×

SQL | All Rows Fetched: 0 in 0 seconds

EMPLOYE…	RATE_BE…	RATE_AF…	UPDATE_…

```sql
7
8     select * from employee_rate_audit;
9     select * from employee;
0     execute sp_update_commission_rate;
1     select * from employee;
2     select * from employee_rate_audit;
3     -- commission table creation
4     create table commission as (select e.employee_no,nvl(extract(year from o.order_date),0)as years ,nvl(sum(subtotal),0) as subtotal
```

Script Output ×   ▶ Query Result ×

SQL | Fetched 50 rows in 0 seconds

	CITY	STATE	ZIP	STATUS	DOB	START_DATE	END_DATE	BRANCH_NO	BRANCH_SIZE	POSITION	RATE	COMMIS
1	Street	MANKATO	MN	56001	3	12-MAY-54	06-MAY-76	(null)	100	HQ	2	(null)	(n
2	lvania Ave	SAINT PAUL	MN	55101	3	12-MAY-54	06-MAY-76	(null)	101	BG	3	(null)	(n
3	:reet	ROCHESTER	MN	55901	3	12-MAY-54	06-MAY-76	(null)	102	MD	6	(null)	(n
4	Point Road	SPRINGFIELD	IL	62701	3	12-MAY-54	06-MAY-76	(null)	103	MD	7	(null)	(n
5	gton Blvd	WASHINGTON	DC	20009	3	12-MAY-54	06-MAY-76	(null)	114	MD	5	(null)	(n
6	/lvania Avenue	TALLAHASSEE	FL	32301	2	12-MAY-54	06-MAY-76	(null)	112	MD	3	(null)	(n
7	ia Ave	SACRAMENTO	CA	95814	2	12-MAY-54	06-MAY-76	(null)	109	BG	7	(null)	(n
8	s Street	CARSON CITY	NV	89701	2	12-MAY-54	06-MAY-76	(null)	110	BG	4	(null)	(n
9	eke Avenue	PHOENIX	AZ	85003	2	12-MAY-54	06-MAY-76	(null)	112	MD	2	(null)	(n
10	:gheny Avenue	MANKATO	MN	56001	2	12-MAY-54	06-MAY-76	(null)	100	HQ	3	(null)	(n