

## **PRACTICAL – 6**

**Aim:** To solve various queries related to grouping and aggregate functions by manipulating data in the product and emp\_company tables.

**Constraints –**

- **Not Null Constraints:** Ensure critical fields are not null.
- **Unique Constraints:** Ensure data integrity by limiting column values.
- **Check Constraints:** Ensure columns have unique values where required.

**Test Cases –**

1) Insert the following values into the product table.

Test Case: Verify that the values are inserted correctly into the product table.

<b>Detorder_no</b>	<b>Product_no</b>	<b>Qty_order</b>
O19001	P00001	10
O19001	P00002	3
O19002	P00001	4
O19003	P00004	2
O19004	P00003	6
O19005	P00005	2
O19006	P00004	7

ORACLE Database Express Edition

User: 23DIT056

Home > SQL > SQL Commands

Autocommit Display 10 Save Run

```
create table product(Detorder_no Varchar2(15),Product_no Varchar2(15),Qty_order Number(3))
insert into product(Detorder_no,Product_no,Qty_order)
values('019001','P00001','10')
insert into product values('019001','P00002','3')
insert into product values('019002','P00001','4')
insert into product values('019003','P00004','2')
insert into product values('019004','P00003','6')
insert into product values('019005','P00005','2')
insert into product values('019006','P00004','7')
select * from product
```

Results Explain Describe Saved SQL History

DETORDER_NO	PRODUCT_NO	QTY_ORDER
O19001	P00001	10
O19001	P00002	3
O19002	P00001	4
O19003	P00004	2
O19004	P00003	6
O19005	P00005	2
O19006	P00004	7

7 rows returned in 0.00 seconds CSV Export

Language: en-us Application Express 2.1.0.00.39 Copyright © 1999, 2006, Oracle. All rights reserved.

2) Retrieve the product numbers and total quantity ordered for each product from the product table.

Test Case: Verify that the sum of quantities ordered for each product number is calculated correctly.

ORACLE Database Express Edition

User: 23DIT056

Home > SQL > SQL Commands

Autocommit Display 10 Save Run

```
create table product(Detorder_no Varchar2(15),Product_no Varchar2(15),Qty_order Number(3))
insert into product(Detorder_no,Product_no,Qty_order)
values('019001','P00001','10')
insert into product values('019001','P00002','3')
insert into product values('019002','P00001','4')
insert into product values('019003','P00004','2')
insert into product values('019004','P00003','6')
insert into product values('019005','P00005','2')
insert into product values('019006','P00004','7')
select * from product

select PRODUCT_NO,SUM(QTY_ORDER) AS total_quantity from product group by PRODUCT_NO order by PRODUCT_NO
```

Results Explain Describe Saved SQL History

PRODUCT_NO	TOTAL_QUANTITY
P00001	14
P00002	3
P00003	6
P00004	9
P00005	2

5 rows returned in 0.00 seconds CSV Export

Language: en-us Application Express 2.1.0.00.39 Copyright © 1999, 2006, Oracle. All rights reserved.

3) Retrieve the product no and the total quantity ordered for product's 'P00001' and 'P00004' from product table.

Test Case: Verify that the sum of quantities ordered for product numbers 'P00001' and 'P00004' is calculated correctly.

ORACLE Database Express Edition

User: 23DIT056

Home > SQL > SQL Commands

Autocommit Display 10 Save Run

```

create table product(Detorder_no Varchar2(15),Product_no Varchar2(15),Qty_order Number(3))
insert into product(Detorder_no,Product_no,Qty_order)
values('019001','P00001','10')
insert into product values('019001','P00002','3')
insert into product values('019002','P00001','4')
insert into product values('019003','P00004','2')
insert into product values('019004','P00003','6')
insert into product values('019005','P00005','2')
insert into product values('019006','P00004','7')
select * from product

select PRODUCT_NO,SUM(QTY_ORDER) AS total_quantity from product group by PRODUCT_NO order by PRODUCT_NO
select PRODUCT_NO,SUM(QTY_ORDER) AS total_quantity from product where PRODUCT_NO IN ('P00001','P00004') group by PRODUCT_NO order by PRODUCT_NO
  
```

Results Explain Describe Saved SQL History

PRODUCT_NO	TOTAL_QUANTITY
P00001	14
P00004	9

2 rows returned in 0.00 seconds CSV Export

Application Express 2.1.0.00.39  
Copyright © 1999, 2006, Oracle. All rights reserved.

4) Insert the following values into emp\_company

ENAME	CNAME	SALARY
Anil	ACC	1500
Shankar	TATA	2000
Jay	WIPRO	1800
Sunil	WIPRO	1700
Vijay	TATA	5000
Prakash	TATA	3000
Ajay	ACC	8000

Abhay	ACC	1800
-------	-----	------

Test Case: Verify that the values are inserted correctly into the emp\_company table.

ORACLE Database Express Edition

User: 23DIT056

Home > SQL > SQL Commands

Autocommit Display 10 Save Run

```

select PRODUCT_NO,SUM(QTY_ORDER) AS total_quantity from product group by PRODUCT_NO order by PRODUCT_NO
select PRODUCT_NO,SUM(QTY_ORDER) AS total_quantity from product where PRODUCT_NO IN ('P00001','P00004') group by PRODUCT_NO order by PRODUCT_NO

create table emp_company(ENAME Varchar2(30),CNAME Varchar2(30),SALARY Number(8,2))
Insert into emp_company(ENAME,CNAME,SALARY)
Values('Anil','ACC',1500)
insert into emp_company values('Shankar','TATA',2000)
insert into emp_company values('Jay','WIPRO',1800)
insert into emp_company values('Sunil','WIPRO',1700)
insert into emp_company values('Vijay','TATA',5000)
insert into emp_company values('Prakash','TATA',3000)
insert into emp_company values('Ajay','ACC',8000)
select * from emp_company

```

Results Explain Describe Saved SQL History

ENAME	CNAME	SALARY
Anil	ACC	1500
Shankar	TATA	2000
Jay	WIPRO	1800
Sunil	WIPRO	1700
Vijay	TATA	5000
Prakash	TATA	3000
Ajay	ACC	8000

7 rows returned in 0.00 seconds CSV Export

Language: en-us Application Express 2.1.0.00.39 Copyright © 1999, 2006, Oracle. All rights reserved.

5) List the name of the company and the maximum salary in that company.

Test Case: Verify that the maximum salary for each company is calculated correctly.

ORACLE Database Express Edition

User: 23DIT056

Home > SQL > SQL Commands

Autocommit Display 10 Save Run

```

select PRODUCT_NO,SUM(QTY_ORDER) AS total_quantity from product where PRODUCT_NO IN ('P00001','P00004') group by PRODUCT_NO order by PRODUCT_NO

create table emp_company(ENAME Varchar2(30),CNAME Varchar2(30),SALARY Number(8,2))
Insert into emp_company(ENAME,CNAME,SALARY)
Values('Anil','ACC',1500)
insert into emp_company values('Shankar','TATA',2000)
insert into emp_company values('Jay','WIPRO',1800)
insert into emp_company values('Sunil','WIPRO',1700)
insert into emp_company values('Vijay','TATA',5000)
insert into emp_company values('Prakash','TATA',3000)
insert into emp_company values('Ajay','ACC',8000)
select * from emp_company
select CNAME,MAX(SALARY) AS MAX_SALARY from emp_company group by CNAME order by CNAME

```

Results Explain Describe Saved SQL History

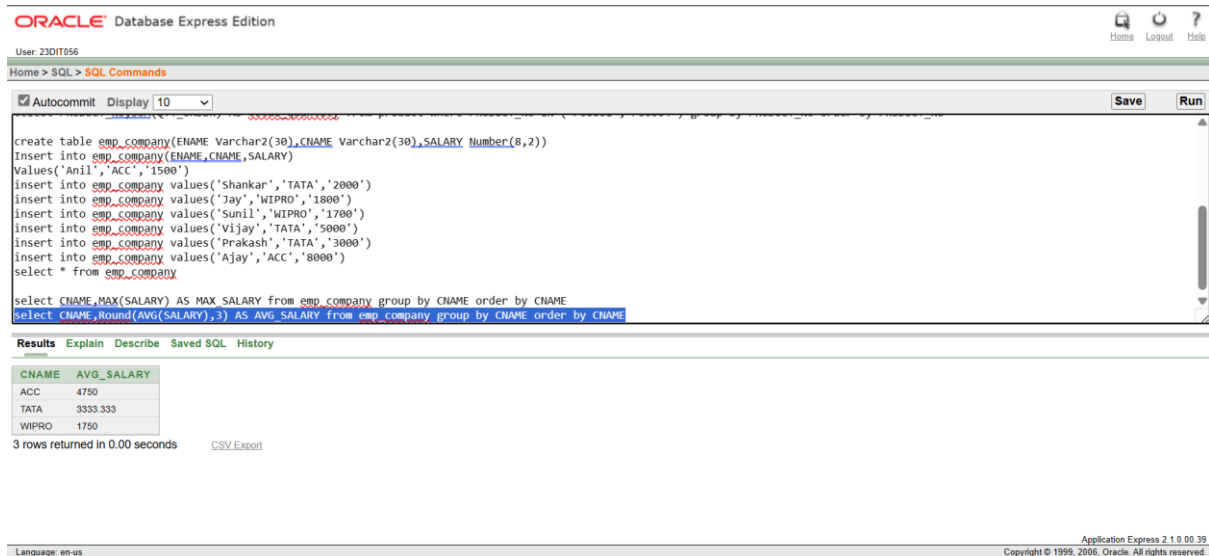
CNAME	MAX_SALARY
ACC	8000
TATA	5000
WIPRO	1800

3 rows returned in 0.00 seconds CSV Export

Language: en-us Application Express 2.1.0.00.39 Copyright © 1999, 2006, Oracle. All rights reserved.

6) Find out the average salary of each company.

Test Case: Verify that the average salary for each company is calculated correctly.



The screenshot shows the Oracle Database Express Edition interface. The SQL Commands window contains the following code:

```
create table emp_company(ENAME Varchar2(30),CNAME Varchar2(30),SALARY Number(8,2))
insert into emp_company(ENAME,CNAME,SALARY)
values('Anil','ACC',1500)
insert into emp_company values('Shankar','TATA',2000)
insert into emp_company values('Jay','WIPRO',1800)
insert into emp_company values('Sunil','WIPRO',1700)
insert into emp_company values('Vijay','TATA',5000)
insert into emp_company values('Prakash','TATA',3000)
insert into emp_company values('Ajay','ACC',8000)
select * from emp_company

select CNAME,MAX(SALARY) AS MAX_SALARY from emp_company group by CNAME order by CNAME
select CNAME,round(AVG(SALARY),3) AS AVG_SALARY from emp_company group by CNAME order by CNAME
```

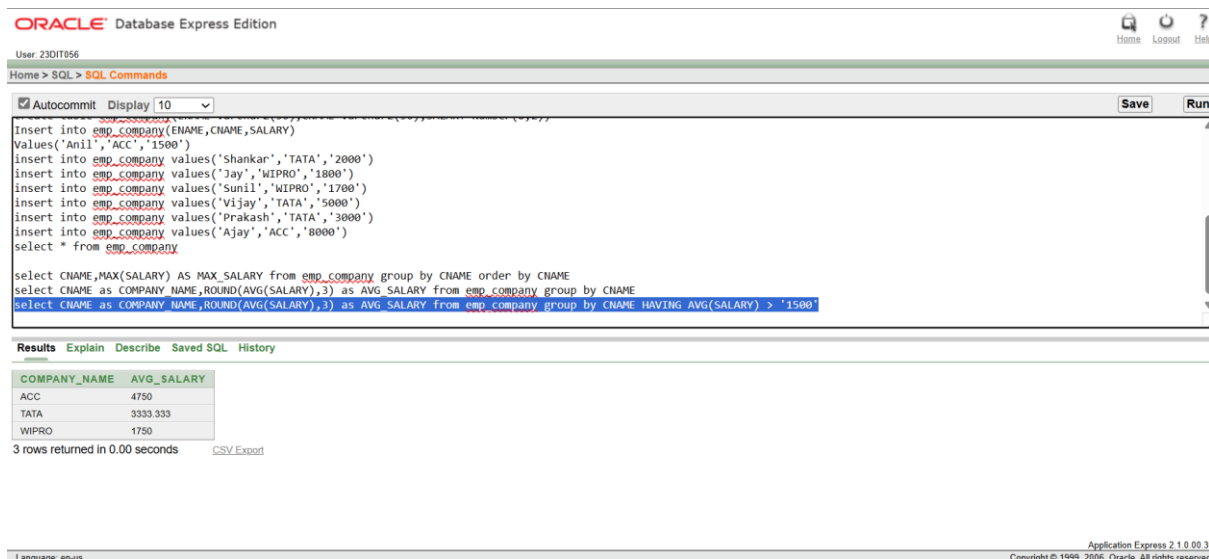
The Results window shows the output of the second query:

CNAME	AVG_SALARY
ACC	4750
TATA	3333.333
WIPRO	1750

3 rows returned in 0.00 seconds

7) Find out the names of companies having an average salary of more than 1500.

Test Case: Verify that the companies with an average salary greater than 1500 are listed correctly.



The screenshot shows the Oracle Database Express Edition interface. The SQL Commands window contains the following code:

```
insert into emp_company(ENAME,CNAME,SALARY)
values('Anil','ACC',1500)
insert into emp_company values('Shankar','TATA',2000)
insert into emp_company values('Jay','WIPRO',1800)
insert into emp_company values('Sunil','WIPRO',1700)
insert into emp_company values('Vijay','TATA',5000)
insert into emp_company values('Prakash','TATA',3000)
insert into emp_company values('Ajay','ACC',8000)
select * from emp_company

select CNAME,MAX(SALARY) AS MAX_SALARY from emp_company group by CNAME order by CNAME
select CNAME as COMPANY_NAME,ROUND(AVG(SALARY),3) as AVG_SALARY from emp_company group by CNAME
select CNAME as COMPANY_NAME,ROUND(AVG(SALARY),3) as AVG_SALARY from emp_company group by CNAME HAVING AVG(SALARY) > 1500
```

The Results window shows the output of the third query:

COMPANY_NAME	AVG_SALARY
ACC	4750
TATA	3333.333
WIPRO	1750

3 rows returned in 0.00 seconds

8) Find out the average salary of each company except 'ACC'.

Test Case: Verify that the average salary for each company, excluding 'ACC', is calculated correctly.

The screenshot shows the SQL Developer interface with the following SQL commands entered in the 'SQL Commands' window:

```

Values('Anil','ACC','1500')
insert into emp_company values('Shankar','TATA','2000')
insert into emp_company values('Jay','WIPRO','1800')
insert into emp_company values('Sunil','WIPRO','1700')
insert into emp_company values('Vijay','TATA','5000')
insert into emp_company values('Prakash','TATA','3000')
insert into emp_company values('Ajay','ACC','8000')
select * from emp_company

select CNAME,MAX(SALARY) AS MAX_SALARY from emp_company group by CNAME order by CNAME
select CNAME as COMPANY_NAME,ROUND(AVG(SALARY),3) as AVG_SALARY from emp_company group by CNAME
select CNAME as COMPANY_NAME,ROUND(AVG(SALARY),3) as AVG_SALARY from emp_company group by CNAME HAVING AVG(SALARY) > '1500'
select CNAME as COMPANY_NAME,ROUND(AVG(SALARY),3) as AVG_SALARY from emp_company where CNAME != 'ACC' group by CNAME
  
```

The 'Results' window shows the output of the last query:

COMPANY_NAME	AVG_SALARY
TATA	3333.333
WIPRO	1750

2 rows returned in 0.00 seconds

### ➤ DESCRIPTION (Theory):

**SUM():** Calculates the total sum of a numeric column. Used to get the total quantity ordered for each product.

**AVG():** Computes the average value of a numeric column. Used to find the average salary for each company.

**MAX():** Determines the maximum value in a numeric column. Used to get the highest salary in each company.

**GROUP BY:** The GROUP BY clause groups rows that have the same values in specified columns into aggregated data. It's used with aggregate functions to perform calculations on each group.

**HAVING:** The HAVING clause is used to filter groups of rows that meet certain conditions after aggregation. It's similar to the WHERE clause but operates on aggregated data.

### ➤ CONCLUSION:

From this practical 6, I learn about SQL skills by perform some functions like `SUM()`, `AVG()`, and `MAX()`, and `GROUP BY` and `HAVING` clauses for advanced data analysis. solve most of queries related to grouping and aggregate functions by manipulating data in the product and emp\_company tables that s the learn.