

**Experiment No.: 05**

**Title:** To implement aggregate functions with order

by, group by, like and having clauses.

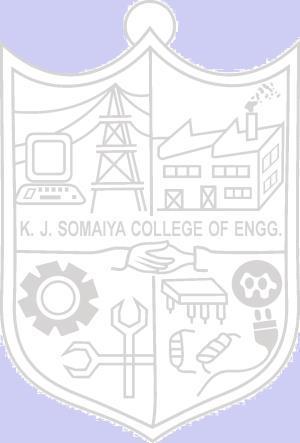
**Batch:** A3 **Roll No.:** 16010421075 **Experiment No: 05**

**Aim:** To implement aggregate functions with order by, group by, like and having clause.

**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

**Resources needed:** PostgreSQL PgAdmin4

**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

**Theory:**

The ORDER BY clause is used to sort the data in ascending or descending order, based on one or more columns.

SELECT column-list

FROM table\_name

[WHERE condition]

[ORDER BY column1, column2, .. columnN] [ASC | DESC];

The GROUP BY clause is used in collaboration with the SELECT statement to group together those rows in a table that have identical data. This is done to eliminate redundancy in the output and/or compute aggregates that apply to these groups.

The GROUP BY clause follows the WHERE clause in a SELECT statement and precedes the ORDER BY clause.

SELECT column-list

FROM table\_name

WHERE [ conditions ]

GROUP BY column1, column2....columnN

ORDER BY column1, column2....columnN

The LIKE operator is used to match text values against a pattern using wildcards. If the search expression can be matched to the pattern expression, the LIKE operator will return true, which is 1. There are two wildcards used in conjunction with the LIKE operator:

* The percent sign (%)
* The underscore (\_)

The percent sign represents zero, one, or multiple numbers or characters. The underscore represents a single number or character. These symbols can be used in combinations.

If either of these two signs is not used in conjunction with the LIKE clause, then the LIKE acts like the equals operator.

SELECT FROM table\_name

WHERE column LIKE 'XXXX%'

or

SELECT FROM table\_name

WHERE column LIKE '%XXXX%'

or

SELECT FROM table\_name

WHERE column LIKE 'XXXX\_'

or

SELECT FROM table\_name

WHERE column LIKE '\_XXXX'

or

SELECT FROM table\_name

WHERE column LIKE '\_XXXX\_'

Here are examples showing WHERE part having different LIKE clause with '%' and '\_' operators:

| **Statement** | **Description** |
| --- | --- |
| WHERE SALARY::text LIKE '200%' | Finds any values that start with 200 |
| WHERE SALARY::text LIKE '%200%' | Finds any values that have 200 in any position |
| WHERE SALARY::text LIKE '\_00%' | Finds any values that have 00 in the second and third positions |
| WHERE SALARY::text LIKE '2\_%\_%' | Finds any values that start with 2 and are at least 3 characters in length |
| WHERE SALARY::text LIKE '%2' | Finds any values that end with 2 |
| WHERE SALARY::text LIKE '\_2%3' | Finds any values that have a 2 in the second position and end with a 3 |
| WHERE SALARY::text LIKE '2\_\_\_3' | Finds any values in a five-digit number that start with 2 and end with 3 |

The HAVING clause allows us to pick out particular rows where the function's result meets some condition.

The WHERE clause places conditions on the selected columns, whereas the HAVING clause places conditions on groups created by the GROUP BY clause.

SELECT column1, column2

FROM table1, table2

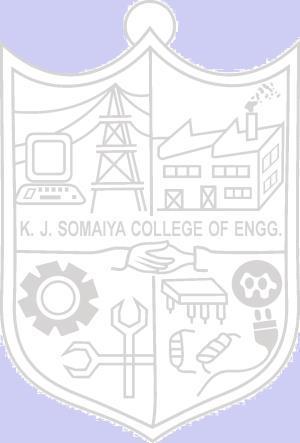
WHERE [ conditions ]

GROUP BY column1, column2

HAVING [ conditions ]

ORDER BY column1, column2

**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**



**Results: (Queries printout with output)**

1. Write 13 queries using ‘order by’, ‘group by’, ‘like’ and ‘having’ clause.

5 with normal aggregate fun,3 with clauses and aggregate function and 5 with like operator

CREATE TABLE AIRPORT(

    Airport\_Name text PRIMARY KEY NOT NULL,

    Country text NOT NULL,

    City text NOT NULL,

    Statee text NOT NULL

);

insert into AIRPORT values('CSMIA', 'India','Mumbai', 'Maharashtra');

insert into AIRPORT values('IGIA', 'India','Delhi', 'Delhi');

insert into AIRPORT values('SVPI', 'India','Ahmedabad', 'Gujarat');

insert into AIRPORT values('RGIA', 'India','Hyderabad', 'Telangana');

insert into AIRPORT values('NSCBIA', 'India','Kolkata', 'West Bengal');

drop table AIRPORT

select\*from AIRPORT

CREATE TABLE COMPANY(

    Company\_ID text PRIMARY KEY NOT NULL,

    Company\_Name text NOT NULL

);

insert into COMPANY values('6E', 'Indigo');

insert into COMPANY values('UK', 'Vistara');

insert into COMPANY values('AI', 'Air India');

insert into COMPANY values('G8', 'Go First');

insert into COMPANY values('SG', 'Spice Jet');

drop table COMPANY

select\*from COMPANY

CREATE TABLE PASSENGER(

    Passenger\_ID INT PRIMARY KEY NOT NULL,

    CHECK(Passenger\_ID>9999 or Passenger\_ID<100000),

    Passenger\_Name text NOT NULL,

    Passenger\_Age int NOT NULL,

    Passenger\_Address text NOT NULL,

    Passenger\_Gender char(1)

    CHECK (Passenger\_Gender='F' or Passenger\_Gender='M'),

    Passenger\_DOB DATE NOT NULL,

    Passenger\_Phone\_Number varchar(10) NOT NULL

);

insert into PASSENGER values(10000, 'Tirth',18, 'Maharashtra', 'M','2003-10-25',9082198360);

insert into PASSENGER values(10001, 'Keyur',18, 'Delhi', 'M','2003-08-20',9884796324);

insert into PASSENGER values(10002, 'Arsalan',18, 'Gujarat', 'M','2003-11-25',9745148752);

insert into PASSENGER values(10003, 'Harsh',18, 'Telangana', 'M','2003-09-25',9658312047);

insert into PASSENGER values(10004, 'Krishiv',18, 'West Bengal', 'M','2003-05-25',9754321984);

drop table PASSENGER

select \* from PASSENGER

CREATE TABLE EMPLOYEE(

    Employee\_ID INT PRIMARY KEY NOT NULL,

    CHECK(Employee\_ID>999 or Employee\_ID<10000),

    Employee\_Name text NOT NULL,

    Employee\_Age int NOT NULL,

    Employee\_\_Address text NOT NULL,

    Employee\_Gender char(1)

    CHECK (Employee\_Gender='F' or Employee\_Gender='M'),

    Employee\_DOB text NOT NULL,

    Employee\_Salary int NOT NULL,

    CHECK (Employee\_Salary>0),

    Employee\_Designation text NOT NULL,

    Employee\_Phone\_Number varchar(10) NOT NULL

);

insert into EMPLOYEE values(1000, 'Avery Peters',26, 'Maharashtra', 'M','2003-10-25', 15000 , 'Senior',9754321984);

insert into EMPLOYEE values(1001, 'Kayleigh Soto',26, 'Maharashtra', 'M','2003-08-20', 17000 , 'Senior',9658312047);

insert into EMPLOYEE values(1002, 'Savanna Horne',26, 'Maharashtra', 'F','2003-11-25', 14000 , 'Junior',9745148752);

insert into EMPLOYEE values(1003, 'Matteo Crosby',26, 'Maharashtra', 'M','2003-09-25', 13000 , 'Senior',9658312047);

insert into EMPLOYEE values(1004, 'Fatma Sheldon',26, 'Maharashtra', 'F','2003-05-25', 11000 , 'Junior',8745147534);

drop table EMPLOYEE

select \* from EMPLOYEE

CREATE TABLE FLIGHT(

    Flight\_Number text PRIMARY KEY NOT NULL,

    Sourcee text NOT NULL,

    Destination text NOT NULL,

    Status text NOT NULL,

    CHECK(Status='Departed' or Status='Arrived' or Status='Delayed' or Status='Cancelled'),

    Departure\_Time TIME NOT NULL,

    Arrival\_Time TIME NOT NULL,

    Connected char(1) NOT NULL,

    CHECK(Connected='Y' or Connected='N')

);

INSERT INTO FLIGHT VALUES('12345','Mumbai','Delhi','Departed', '09:00:00', '11:00:00','N');

INSERT INTO FLIGHT VALUES('12356','Delhi', 'Mumbai','Departed', '09:10:00', '11:05:00','N');

INSERT INTO FLIGHT VALUES('12389','Delhi', 'Mumbai','Departed', '09:15:00', '11:15:00','N');

INSERT INTO FLIGHT VALUES('12384','Ahmedabad', 'Mumbai','Departed', '19:15:00', '21:15:00','N');

INSERT INTO FLIGHT VALUES('12383','Kolkata', 'Mumbai','Departed', '15:55:00', '17:45:00','N');

drop table FLIGHT

select \* from FLIGHT

CREATE TABLE TICKET(

    PASSENGER\_NAME VARCHAR(20) NOT NULL,

    Ticket\_Number VARCHAR(20) NOT NULL,

    Company\_NAME VARCHAR(20) NOT NULL,

    Seat\_Number VARCHAR(20) PRIMARY KEY NOT NULL,

    Classs text NOT NULL,

    Price INT NOT NULL,

    Departure\_Time TIME NOT NULL,

    Arrival\_Time TIME NOT NULL,

    Destination VARCHAR(20) NOT NULL,

    Sourcee VARCHAR(20) NOT NULL

);

INSERT INTO TICKET VALUES('Keyur',6000, 'Spice Jet', 'A-2' , 'First Class', '2500', '09:10:00',  '11:05:00', 'Delhi', 'Mumbai');

INSERT INTO TICKET VALUES('Kushal',6700, 'Indigo', 'B-5' , 'First Class', '3000', '09:10:00',  '11:05:00', 'Kolkata', 'Delhi');

INSERT INTO TICKET VALUES('Srajan',6800, 'Vistara', 'A-24' , 'First Class', '1500', '09:10:00',  '11:05:00', 'Ahmedabad', 'Mumbai');

INSERT INTO TICKET VALUES('Arsalan',9000, 'Spice Jet', 'D-2' , 'First Class', '1500', '09:10:00',  '11:05:00', 'Delhi', 'Mumbai');

INSERT INTO TICKET VALUES('Shubh',6300, 'Go First', 'F-2' , 'First Class', '1500', '21:10:00',  '23:05:00', 'Delhi', 'Mumbai');

drop table TICKET

select \* from TICKET

CREATE TABLE CONTAIN(

    Airport\_Name VARCHAR(20),

    Company\_ID text PRIMARY KEY NOT NULL,

    FOREIGN KEY (Airport\_Name) REFERENCES AIRPORT(Airport\_Name),

    FOREIGN KEY (Company\_ID) REFERENCES COMPANY(Company\_ID)

);

INSERT INTO CONTAIN VALUES('CSMIA','6E');

INSERT INTO CONTAIN VALUES('CSMIA','UK');

INSERT INTO CONTAIN VALUES('RGIA','AI');

INSERT INTO CONTAIN VALUES('RGIA','G8');

INSERT INTO CONTAIN VALUES('SVPI','SG');

drop table CONTAIN

select \* from CONTAIN

CREATE TABLE HAS(

    Employee\_ID INT PRIMARY KEY NOT NULL,

    Airport\_Name VARCHAR(20),

    FOREIGN KEY (Airport\_Name) REFERENCES AIRPORT(Airport\_Name),

    FOREIGN KEY (Employee\_ID) REFERENCES EMPLOYEE(Employee\_ID)

);

INSERT INTO HAS VALUES(1000,'CSMIA');

INSERT INTO HAS VALUES(1001,'CSMIA');

INSERT INTO HAS VALUES(1002,'SVPI');

INSERT INTO HAS VALUES(1003,'RGIA');

INSERT INTO HAS VALUES(1004,'CSMIA');

drop table HAS

select \* from HAS

CREATE TABLE SERVES(

    Employee\_ID INT,

    Passenger\_ID  INT ,

    FOREIGN KEY (Employee\_ID) REFERENCES EMPLOYEE(Employee\_ID),

    FOREIGN KEY (Passenger\_ID) REFERENCES PASSENGER(Passenger\_ID)

);

INSERT INTO SERVES VALUES(1004,10001);

INSERT INTO SERVES VALUES(1001,10002);

INSERT INTO SERVES VALUES(1003,10003);

INSERT INTO SERVES VALUES(1000,10000);

INSERT INTO SERVES VALUES(1001,10004);

drop table SERVES

select \* from SERVES

CREATE TABLE BOOKS(

    Passenger\_ID INT NOT NULL,

    Seat\_Number VARCHAR(20),

    FOREIGN KEY (Passenger\_ID) REFERENCES PASSENGER(Passenger\_ID),

    FOREIGN KEY (Seat\_Number) REFERENCES TICKET(Seat\_Number)

);

INSERT INTO BOOKS VALUES(10000, 'A-2');

INSERT INTO BOOKS VALUES(10001, 'B-5');

INSERT INTO BOOKS VALUES(10002, 'A-24');

INSERT INTO BOOKS VALUES(10003, 'D-2');

INSERT INTO BOOKS VALUES(10004, 'F-2');

drop table BOOKS

select \* from BOOKS

CREATE TABLE CANCELS(

    Passenger\_ID INT NOT NULL,

    Seat\_Number VARCHAR(20),

    FOREIGN KEY (Passenger\_ID) REFERENCES PASSENGER(Passenger\_ID),

    FOREIGN KEY (Seat\_Number) REFERENCES TICKET(Seat\_Number)

);

INSERT INTO CANCELS VALUES(10000, 'A-2');

INSERT INTO CANCELS VALUES(10001, 'B-5');

INSERT INTO CANCELS VALUES(10002, 'A-24');

INSERT INTO CANCELS VALUES(10003, 'D-2');

INSERT INTO CANCELS VALUES(10004, 'F-2');

drop table CANCELS

select \* from CANCELS

select Employee\_gender, count(Employee\_Gender) from Employee group by Employee\_gender

select AVG(Employee\_salary) from Employee

select sourcee, count(Flight\_Number) from FLIGHT group by sourcee

select sourcee, max(destination) from TICKET group by sourcee

select sourcee, min(destination) from TICKET group by sourcee

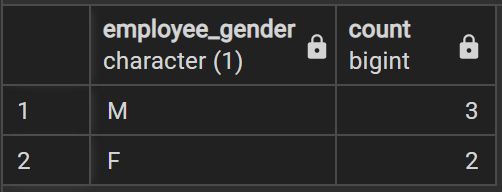
SELECT city from AIRPORT ORDER BY city DESC;

SELECT city from AIRPORT where airport\_name='CSMIA' ORDER BY country

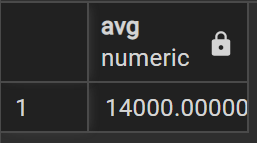
select sum(Employee\_salary) from Employee

select Employee\_gender, count(Employee\_Gender) from Employee group by

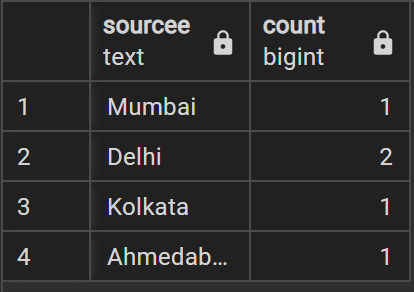
Employee\_gender



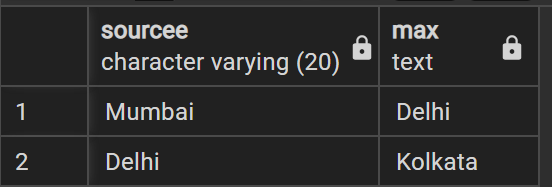
select AVG(Employee\_salary) from Employee



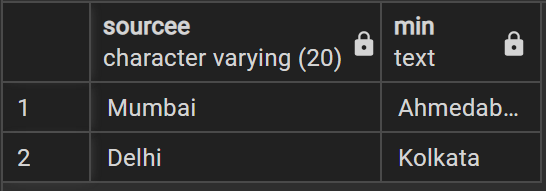
select sourcee, count(Flight\_Number) from FLIGHT group by sourcee



select sourcee, max(destination) from TICKET group by source



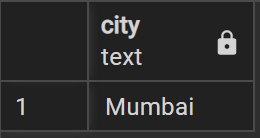
select sourcee, min(destination) from TICKET group by sourcee



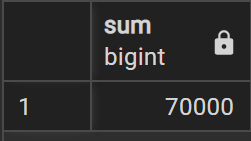
SELECT city from AIRPORT ORDER BY city DESC;



SELECT city from AIRPORT where airport\_name='CSMIA' ORDER BY country



select sum(Employee\_salary) from Employee



**Example:**

1. SELECT \* FROM COMPANY ORDER BY NAME, SALARY ASC;
2. SELECT NAME, SUM(SALARY) FROM COMPANY GROUP BY NAME;
3. SELECT \* FROM COMPANY WHERE AGE::text LIKE '2%';
4. SELECT \* FROM COMPANY WHERE ADDRESS LIKE '%-%';
5. SELECT NAME FROM COMPANY GROUP BY name HAVING count(name) > 1;

**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

**Outcomes:**

**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

**Questions:**

**Q1 Can you apply an operator on integer value? explain with an example how?**

**Ans :** We cannot directly use **LIKE** operator on **INTEGER** type columns. We have to explicitly

cast the **INTEGER** column as **VARCHAR** and then use **LIKE** operator.

For example, for an Employee table we want information of all employees having start of their phone number as XXX.

So, query statement for the above problem will

SELECT \*

FROM Employee

WHERE CAST(phone\_no\_column as VARCHAR(10)) LIKE 'XXX%'

**Q2 Why aggregate functions are more used with order by, group by and having clauses? Can we change order of these clauses when used in single query**

Aggregate functions cannot be used with WHERE clause. HAVING clause is the equivalent of WHERE to be used with aggregate functions. HAVING expects Boolean value as well. Aggregate functions like max, min, sum can be appliead to groups formed by particular attributes. In such cases GROUP BY clause is used.

**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

**Conclusion:**

From the above experiment we learnt the use of Aggregate Functions, **GROUP BY, HAVING** **BY** and **ORDER BY** clauses.

**Grade: AA / AB / BB / BC / CC / CD /DD**

**Signature of faculty in-charge with date**

**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

**References:**

**Books:**

1. Elmasri and Navathe, “Fundamentals of Database Systems”, 6th Edition, Pearson Education
2. Korth, Slberchatz,Sudarshan, :”Database System Concepts”, 6th Edition, McGraw –

Hill.

**WebSite:**

1. http://www.tutorialspoint.com/postgresql/