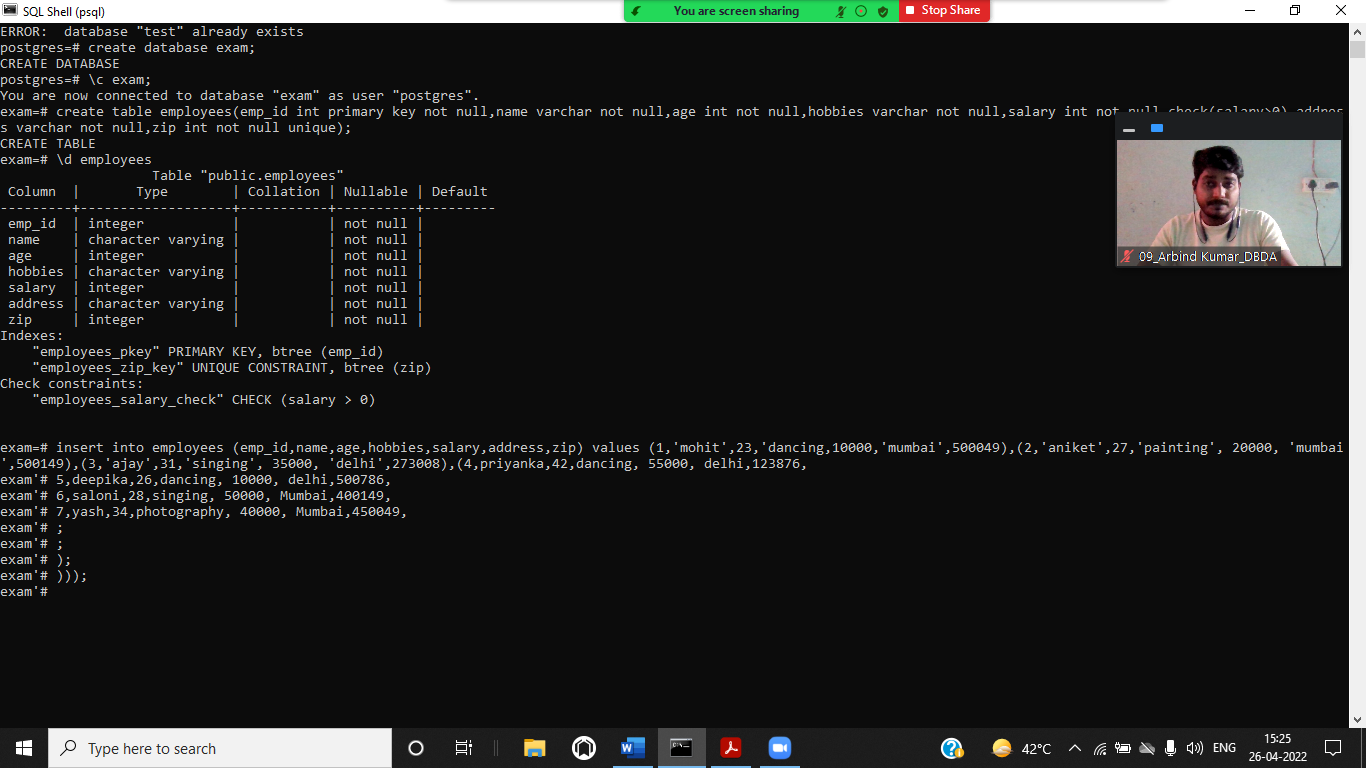
**Q1 Create table employee,dept with following column and insert given data**



You are now connected to database "exam" as user "postgres".

exam=# \d employees

Table "public.employees"

Column | Type | Collation | Nullable | Default

---------+-------------------+-----------+----------+---------

emp\_id | integer | | not null |

name | character varying | | not null |

age | integer | | not null |

hobbies | character varying | | not null |

salary | integer | | not null |

address | character varying | | not null |

zip | integer | | not null |

Indexes:

"employees\_pkey" PRIMARY KEY, btree (emp\_id)

"employees\_zip\_key" UNIQUE CONSTRAINT, btree (zip)

Check constraints:

"employees\_salary\_check" CHECK (salary > 0)

exam=# insert into employees(emp\_id,name,age,hobbies,salary,address,zip) values (1,'mohit',23,'dancing', 10000, 'Mumbai',500049);

INSERT 0 1

exam=# insert into employees(emp\_id,name,age,hobbies,salary,address,zip) values (2,'aniket',27,'painting', 20000, 'mumbai',500149);

INSERT 0 1

exam=# insert into employees(emp\_id,name,age,hobbies,salary,address,zip) values (3,'ajay',31,'singing', 35000, 'delhi',273008);

INSERT 0 1

exam=# insert into employees(emp\_id,name,age,hobbies,salary,address,zip) values (4,'priyanka',42,'dancing', 55000, 'delhi',123876);

INSERT 0 1

exam=# insert into employees(emp\_id,name,age,hobbies,salary,address,zip) values (5,'deepika',26,'dancing', 10000, 'delhi',500786);

INSERT 0 1

exam=# insert into employees(emp\_id,name,age,hobbies,salary,address,zip) values (6,'saloni',28,'singing', 50000, 'Mumbai',400149);

INSERT 0 1

exam=# insert into employees(emp\_id,name,age,hobbies,salary,address,zip) values (7,'yash',34,'photography', 40000, 'Mumbai',450049);

INSERT 0 1

exam=# insert into employees(emp\_id,name,age,hobbies,salary,address,zip) values (8,'vinay',45,'painting', 70000, 'Mumbai',273006);

INSERT 0 1

exam=# table employees;

emp\_id | name | age | hobbies | salary | address | zip

--------+----------+-----+-------------+--------+---------+--------

1 | mohit | 23 | dancing | 10000 | Mumbai | 500049

2 | aniket | 27 | painting | 20000 | mumbai | 500149

3 | ajay | 31 | singing | 35000 | delhi | 273008

4 | priyanka | 42 | dancing | 55000 | delhi | 123876

5 | deepika | 26 | dancing | 10000 | delhi | 500786

6 | saloni | 28 | singing | 50000 | Mumbai | 400149

7 | yash | 34 | photography | 40000 | Mumbai | 450049

8 | vinay | 45 | painting | 70000 | Mumbai | 273006

(8 rows)

FOR DEPT

exam=# create table dept(dept\_id integer primary key,dept\_name varchar,e\_id int references employees(emp\_id));

CREATE TABLE

exam=# \d dept

Table "public.dept"

Column | Type | Collation | Nullable | Default

-----------+-------------------+-----------+----------+---------

dept\_id | integer | | not null |

dept\_name | character varying | | |

e\_id | integer | | |

Indexes:

"dept\_pkey" PRIMARY KEY, btree (dept\_id)

Foreign-key constraints:

"dept\_e\_id\_fkey" FOREIGN KEY (e\_id) REFERENCES employees(emp\_id)

exam=# alter table dept add column manager varchar;

ALTER TABLE

exam=# \d dept

Table "public.dept"

Column | Type | Collation | Nullable | Default

-----------+-------------------+-----------+----------+---------

dept\_id | integer | | not null |

dept\_name | character varying | | |

e\_id | integer | | |

manager | character varying | | |

Indexes:

"dept\_pkey" PRIMARY KEY, btree (dept\_id)

Foreign-key constraints:

"dept\_e\_id\_fkey" FOREIGN KEY (e\_id) REFERENCES employees(emp\_id)

exam=# insert into dept values (1,'ec',8, 'virat');

INSERT 0 1

exam=# insert into dept values (2,'cs',7, 'sachin');

INSERT 0 1

exam=# insert into dept values (3,'it',6, 'rahul');

INSERT 0 1

exam=# insert into dept values (4,'it',5, 'rahul');

INSERT 0 1

exam=# insert into dept values (5,'cs',4, 'sachin');

INSERT 0 1

exam=# insert into dept values (6,'ec',3, 'virat');

INSERT 0 1

exam=# insert into dept values (7,'ec',2, 'virat');

INSERT 0 1

exam=# insert into dept values (8,'ec',1, 'virat');

INSERT 0 1

exam=# table dept;

dept\_id | dept\_name | e\_id | manager

---------+-----------+------+---------

1 | ec | 8 | virat

2 | cs | 7 | sachin

3 | it | 6 | rahul

4 | it | 5 | rahul

5 | cs | 4 | sachin

6 | ec | 3 | virat

7 | ec | 2 | virat

8 | ec | 1 | virat

(8 rows)

Write a Query to count No. of employees

exam=# select count(name) from employees;

count

-------

8

(1 row)

Write a Query to get min,max,avg,sum of salary for all employees

select min(salary),max(salary),avg(salary),sum(salary) from employees;

min | max | avg | sum

-------+-------+--------------------+--------

10000 | 70000 | 36250.000000000000 | 290000

(1 row)

Write a Query to get highest salary of an individual based on hobbies

exam=# select name,hobbies from employees where salary=(select max(salary) from employees);

name | hobbies

-------+----------

vinay | painting

(1 row)

Write a Query for sum of salary where address starts with 'M' or 'd'

select sum(salary) from employees where address like 'M%' or address like 'd%';

sum

--------

270000

(1 row)

Write a Query to Get all employee details with their department details

select \* from employees e, dept d where emp\_id=dept\_id;

emp\_id | name | age | hobbies | salary | address | zip | dept\_id | dept\_name | e\_id | manager

--------+----------+-----+-------------+--------+---------+--------+---------+-----------+------+---------

1 | mohit | 23 | dancing | 10000 | Mumbai | 500049 | 1 | ec | 8 | virat

2 | aniket | 27 | painting | 20000 | mumbai | 500149 | 2 | cs | 7 | sachin

3 | ajay | 31 | singing | 35000 | delhi | 273008 | 3 | it | 6 | rahul

4 | priyanka | 42 | dancing | 55000 | delhi | 123876 | 4 | it | 5 | rahul

5 | deepika | 26 | dancing | 10000 | delhi | 500786 | 5 | cs | 4 | sachin

6 | saloni | 28 | singing | 50000 | Mumbai | 400149 | 6 | ec | 3 | virat

7 | yash | 34 | photography | 40000 | Mumbai | 450049 | 7 | ec | 2 | virat

8 | vinay | 45 | painting | 70000 | Mumbai | 273006 | 8 | ec | 1 | vira

Write a QUERY TO FIND employees age between 20 and 30

exam=# select \* from employees where age between 20 and 30;

emp\_id | name | age | hobbies | salary | address | zip

--------+---------+-----+----------+--------+---------+--------

1 | mohit | 23 | dancing | 10000 | Mumbai | 500049

2 | aniket | 27 | painting | 20000 | mumbai | 500149

5 | deepika | 26 | dancing | 10000 | delhi | 500786

6 | saloni | 28 | singing | 50000 | Mumbai | 400149

(4 rows)

**Q3 Write a function to return name,emp\_id,dept\_name,hobbies,age by passing manager name**

exam=# create or replace function manager (a varchar)

exam-# returns table(name varchar,emp\_id integer,dept\_name varchar,hobbies varchar,age int)

exam-# as $$

exam$# begin

exam$# return query select e.name,e.emp\_id,d.dept\_name,e.hobbies,e.age from employees as e join dept as d on e.emp\_id=d.dept\_id where d.manager=a;

exam$# end;

exam$# $$ language plpgsql;

CREATE FUNCTION

exam=# select \* from manager('virat');

name | emp\_id | dept\_name | hobbies | age

--------+--------+-----------+-------------+-----

mohit | 1 | ec | dancing | 23

saloni | 6 | ec | singing | 28

yash | 7 | ec | photography | 34

vinay | 8 | ec | painting | 45

(4 rows)

CREATE MONGO DB COLLECTIONS with following details and insert data

> use mongoexam

switched to db mongoexam

> db.assignment.insertOne({ item: "journal", qty: 25, tags: ["blank", "red"], size: { h: 14, w: 21, uom: "cm" } })

{

"acknowledged" : true,

"insertedId" : ObjectId("6267d0ff080962512d26a42d")

}

> db.assignment.insertOne({ item: "mat", qty: 85, tags: ["gray"], size: { h: 27.9, w: 35.5, uom: "cm" } })

{

"acknowledged" : true,

"insertedId" : ObjectId("6267d122080962512d26a42e")

}

> db.assignment.insertOne({ item: "mousepad", qty: 25, tags: ["gel", "blue"], size: { h: 19, w: 22.85, uom: "cm" } })

{

"acknowledged" : true,

"insertedId" : ObjectId("6267d13f080962512d26a42f")

}

For inventory

> db.assignment.insertOne({ item: "journal", qty: 25, tags: ["blank", "red"], size: { h: 14, w: 21, uom: "cm" } })

{

"acknowledged" : true,

"insertedId" : ObjectId("6267d0ff080962512d26a42d")

}

> db.assignment.insertOne({ item: "mat", qty: 85, tags: ["gray"], size: { h: 27.9, w: 35.5, uom: "cm" } })

{

"acknowledged" : true,

"insertedId" : ObjectId("6267d122080962512d26a42e")

}

> db.assignment.insertOne({ item: "mousepad", qty: 25, tags: ["gel", "blue"], size: { h: 19, w: 22.85, uom: "cm" } })

{

"acknowledged" : true,

"insertedId" : ObjectId("6267d13f080962512d26a42f")

}

> db.inventry.insertOne({ item: "journal", qty: 25, tags: ["blank", "red"], dim\_cm: [ 14, 21 ] })

{

"acknowledged" : true,

"insertedId" : ObjectId("6267d1a8080962512d26a430")

}

> db.inventry.insertOne({ item: "notebook", qty: 50, tags: ["red", "blank"], dim\_cm: [ 14, 21 ] })

{

"acknowledged" : true,

"insertedId" : ObjectId("6267d1bf080962512d26a431")

}

> db.inventry.insertOne({ item: "paper", qty: 100, tags: ["red", "blank", "plain"], dim\_cm: [ 14, 21 ] })

{

"acknowledged" : true,

"insertedId" : ObjectId("6267d1e2080962512d26a432")

}

> db.inventry.insertOne({ item: "planner", qty: 75, tags: ["blank", "red"], dim\_cm: [ 22.85, 30 ] })

{

"acknowledged" : true,

"insertedId" : ObjectId("6267d1fc080962512d26a433")

}

> db.inventry.insertOne({ item: "postcard", qty: 45, tags: ["blue"], dim\_cm: [ 10, 15.25 ] })

{

"acknowledged" : true,

"insertedId" : ObjectId("6267d213080962512d26a434")

}

>

get assignment documents having tags = gray

> db.assignment.find({},{tags:"gray"})

{ "\_id" : ObjectId("6267d0ff080962512d26a42d"), "tags" : "gray" }

{ "\_id" : ObjectId("6267d122080962512d26a42e"), "tags" : "gray" }

{ "\_id" : ObjectId("6267d13f080962512d26a42f"), "tags" : "gray" }

>

2. Get inventory details whose dim\_cm > 10 , sorted by qty descending order and print only

db.inventry.find({dim\_cm:{$gt:10}}).sort({qty:-1}).limit(3)

{ "\_id" : ObjectId("6267d1e2080962512d26a432"), "item" : "paper", "qty" : 100, "tags" : [ "red", "blank", "plain" ], "dim\_cm" : [ 14, 21 ] }

{ "\_id" : ObjectId("6267d1fc080962512d26a433"), "item" : "name", "qty" : 75, "tags" : [ "blank", "red" ], "dim\_cm" : [ 22.85, 30 ] }

{ "\_id" : ObjectId("6267d1bf080962512d26a431"), "item" : "notebook", "qty" : 50, "tags" : [ "red", "blank" ], "dim\_cm" : [ 14, 21 ] }

>

3. Create index on inventory in descending order of qty

db.inventry.createIndex({"qty":-1})

{

"numIndexesBefore" : 1,

"numIndexesAfter" : 2,

"createdCollectionAutomatically" : false,

"ok" : 1

5. query to update inventory collection item name where qty:75 and dim\_cm > 22

db.inventry.updateOne({$and:[{"qty":75},{dim\_cm:{$gt:22}}]},{$set:{item:"name"}})

{ "acknowledged" : true, "matchedCount" : 1, "modifiedCount" : 1 }

> db.inventry.find().pretty()

{

"\_id" : ObjectId("6267d1a8080962512d26a430"),

"item" : "journal",

"qty" : 25,

"tags" : [

"blank",

"red"

],

"dim\_cm" : [

14,

21

]

}

{

"\_id" : ObjectId("6267d1bf080962512d26a431"),

"item" : "notebook",

"qty" : 50,

"tags" : [

"red",

"blank"

],

"dim\_cm" : [

14,

21

]

}

{

"\_id" : ObjectId("6267d1e2080962512d26a432"),

"item" : "paper",

"qty" : 100,

"tags" : [

"red",

"blank",

"plain"

],

"dim\_cm" : [

14,

21

]

}