Q)Write a Query to add a column package_stat to the table orders.

alter table orders add column package_stat varchar(40);

Q)Write a Query to change the package_stat column of orders table with 'not available' for all orders.

SET SQL_SAFE_UPDATES = 0;
update orders set package_stat = 'not availabe' where package_stat is null;

```
SET SQL_SAFE_UPDATES = 0;
update orders set package_stat = 'not availabe' where package_stat is null;

SELECT * FROM practice.orders;
```

Result Grid | Filter Rows: | Edit: | Export/Import:

order_Id	order_Date	shipped_Date	Deliver	customer_Id	package_stat
10100	2003-01-06	2003-01-13	Shipped	114	not availabe
10101	2003-01-09	2003-01-18	Shipped	125	not availabe
10102	2003-01-10	2003-01-18	Shipped	129	not availabe
10103	2003-01-29	2003-02-07	Shipped	121	not availabe
10104	2003-01-31	2003-02-09	Shipped	141	not availabe
10105	2003-02-11	2003-02-21	Shipped	145	not availabe

orders 52

Q)Write a Query to delete a row from customers table where credit_limit is 0.00

SET FOREIGN_KEY_CHECKS=0; -- to disable them
DELETE FROM customers WHERE creditlimit=0.00;
SET FOREIGN_KEY_CHECKS=1; -- to re-enable them

Write SELECT statements to achieve the following:

Q)Write a Query to display the first_name with the occurrence of 'el' in the customers tables.

SELECT first_name FROM customers WHERE first_name like '%el%';

Result Grid

first_name
Atelier
Nelson
Keitel
Saveley

Q)Write a Query to prepare a list with customer name ,customer_id ,order_id for the customers whose delivery status is shipped.

select c.First_Name,c.Last_Name,c.customer_Id,o.order_Id
from customers c
join orders o
on c.customer_Id=o.customer_Id
where o.Deliver='Shipped';

```
29 *    select c.First_Name,c.Last_Name,c.customer_Id,o.order_Id
30      from customers c
31      join orders o
32      on c.customer_Id=o.customer_Id
33      where o.Deliver='Shipped';
34
35
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

First_Name	Last_Name	customer_Id	order_Id
Ferguson	Peter	114	10100
Murphy	Julie	129	10102
Bergulfsen	Jonas	121	10103
Freyre	Diego	141	10104
Petersen	Jytte	145	10105

Q)Write a Query to get the number of customers with the creditLimit greater than 50000.

select count(customer_id) from customers where creditLimit > 50000;

```
29 *    select count(customer_id) from customers where creditLimit > 50000;
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

count(customer_id)
14

Q)Write a Query to display the customer_id, name (first name and last name), order_id and deliver for all customers.

select concat(c.First_Name,' ',c.Last_Name) as name,c.customer_Id,o.order_Id, o.Deliver

from customers c
join orders o
on c.customer_Id=o.customer_Id;

```
select concat(c.First_Name,' ',c.Last_Name) as name,c.customer_Id,o.order_Id, o.Deliver
from customers c
join orders o
on c.customer_Id=o.customer_Id;
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

name	customer_Id	order_Id	Deliver
Ferguson Peter	114	10100	Shipped
Murphy Julie	129	10102	Shipped
Bergulfsen Jonas	121	10103	Shipped
Freyre Diego	141	10104	Shipped
Petersen Jytte	145	10105	Shipped

Q)Write a Query to customer name in order of creditLimit smallest to highest.

select concat(first_name,' ',last_name) as name, creditLimit
from customers order by creditLimit asc;

```
select concat(first_name,' ',last_name) as name, creditLimit
from customers order by creditLimit asc;
```

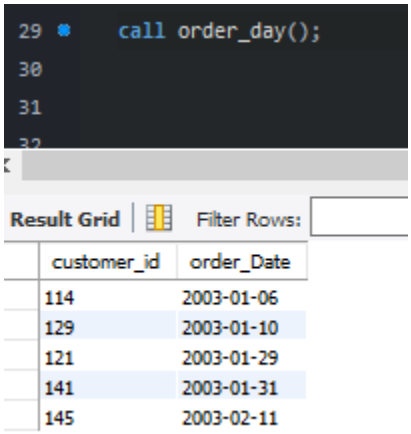
Result Grid | Filter Rows: | Export: | Wrap Cell Content:

name	creditLimit
Atelier Schmitt	21000.00
Berglund Christina	53100.00
Keitel Roland	59700.00
Murphy Julie	64600.00
Signal King	71800.00
Bergulfsen Jonas	81700.00
Petersen Jytte	83400.00
Eric Jecob	103800.00
Lee Kwai	114900.00
Ferguson Peter	117300.00
Labrune Janine	118200.00
Saveley Mary	123900.00
Young Jeff	138500.00

Q)Write a stored procedure by name order_day. The procedure should show the customer_id and the day on which he had made the order.

```
delimiter /
create procedure order_day ()
begin
select c.customer_id , o.order_Date from customers c JOIN orders o on c.customer_Id =
o.customer_Id;
end /
delimiter ;

call order_day();
```



Q)Write a stored function by the name of cutomer_search. The stored function should return the maximum creditLimit made by any customer.

```
delimiter /
create PROCEDURE customer_search (in cid int)
begin
select creditlimit from customers where customer_Id = cid;
end /
delimiter ;

call customer_search(119);
```

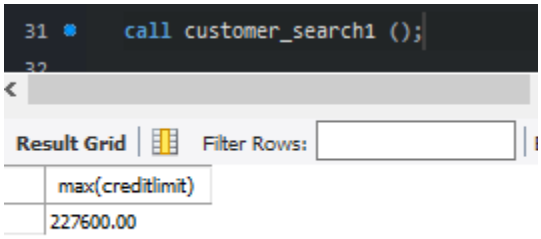
or

===

```
delimiter /
create PROCEDURE customer_search1 ()
begin
```

```
select max(creditlimit) from customers ;
end /
delimiter ;

call customer_search1 ();
```



==========

```
create table dept(
deptno int(2),
dname varchar(14),
loc varchar(13)
);

insert into dept values (10,'accounting', 'new york');
insert into dept values (20, 'RESEARCH', 'DALLAS');
insert into dept values (30, 'SALES', 'CHICAGO');
insert into dept values  (40, 'OPERATIONS',  'BOSTON');


create table emp(
EMPNO INT(4),
ENAME VARCHAR (10),
JOB VARCHAR (9),
HIREDATE DATE,
SAL FLOAT(7,2),
COMM FLOAT(7,2),
DEPTNO INT(2)
);



insert into emp values(7369, 'SMITH', 'CLERK', str_to_date('17-12-80 ','%d-%m-%y'), 800, null,
20);
```

insert into emp values(7499, 'ALLEN', 'SALESMAN', str_to_date('20-02-81','%d-%m-%y'), 1600, 300, 30);
insert into emp values(7521, 'WARD', 'SALESMAN', str_to_date('22-02-81','%d-%m-%y'), 1250, 500, 30);
insert into emp values(7566, 'JONES', 'MANAGER', str_to_date('02-04-81','%d-%m-%y'), 2975, null, 20);
insert into emp values(7654, 'MARTIN', 'SALESMAN', str_to_date('28-09-81','%d-%m-%y'), 1250, null, 30);
insert into emp values(7698, 'BLAKE', 'MANAGER', str_to_date('01-05-81','%d-%m-%y'), 2850, null, 30);
insert into emp values(7782, 'CLARK', 'MANAGER', str_to_date('09-06-81','%d-%m-%y'), 2450, null, 10);
insert into emp values(7788, 'SCOTT', 'ANALYST', str_to_date('09-12-82','%d-%m-%y'), 3000, null, 20);
insert into emp values(7839, 'KING', 'PRESIDENT', str_to_date('17-11-81','%d-%m-%y'), 5000, null, 10);
insert into emp values(7844, 'TURNER', 'SALESMAN', str_to_date('08-09-81','%d-%m-%y'), 1500, 0, 30);
insert into emp values(7876, 'ADAMS', 'CLERK', str_to_date('12-01-83','%d-%m-%y'), 1100, null, 20);
insert into emp values(7900, 'JAMES', 'CLERK', str_to_date('03-12-81','%d-%m-%y'), 950, null, 30);
insert into emp values(7902, 'FORD', 'ANALYST', str_to_date('03-12-81','%d-%m-%y'), 3000, null, 20);
insert into emp values(7934, 'MILLER', 'CLERK', str_to_date('23-01-82','%d-%m-%y'), 1300, null, 10);

==========

Write SELECT statements to achieve the following:-
Q)Display only the EMPNO and ENAME columns from EMP table.

select empno, ename from emp;

```
5 *   select empno, ename from emp;
```

Result Grid | Filter Rows: | Export:

empno	ename
7369	SMITH
7499	ALLEN
7521	WARD
7566	JONES
7654	MARTIN
7698	BLAKE
7782	CLARK
7788	SCOTT
7839	KING
7844	TURNER
7876	ADAMS
7900	JAMES
7902	FORD

Q)Display all employees who are CLERKs and the MANAGERs.
select * from emp where job in ('clerk','manager');

```
35 *   select * from emp where job in ('clerk','manager');
```

Result Grid | Filter Rows: | Export: | Wrap

EMPNO	ENAME	JOB	HIREDATE	SAL	COMM	DEPTNO
7369	SMITH	CLERK	1980-12-17	800.00	NULL	20
7566	JONES	MANAGER	1981-04-02	2975.00	NULL	20
7698	BLAKE	MANAGER	1981-05-01	2850.00	NULL	30
7782	CLARK	MANAGER	1981-06-09	2450.00	NULL	10
7876	ADAMS	CLERK	1983-01-12	1100.00	NULL	20
7900	JAMES	CLERK	1981-12-03	950.00	NULL	30
7934	MILLER	CLERK	1982-01-23	1300.00	NULL	10

Q)Display the ENAME and JOB for all employees who belong to the same DEPTNO as employee 'KING'.
select ename, job from emp where deptno =(select deptno from emp where ename='KING');

```
select ename, job from emp where deptno =(select deptno from emp where ename='KING');
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

ename	job
CLARK	MANAGER
KING	PRESIDENT
MILLER	CLERK

Q)Find the names of all employees hired in the month of February (of any year).
select ename from emp where month(hiredate) =2;

```
select ename from emp where month(hiredate) =2;
```

Result Grid | Filter Rows: | Export:

ename
ALLEN
WARD

Q)Display the employees in descending order of DEPTNO.
select * from emp order by deptno desc;

```
select * from emp order by deptno desc;
```

Result Grid | Filter Rows: | Export: | Wrap Cell

EMPNO	ENAME	JOB	HIREDATE	SAL	COMM	DEPTNO
7499	ALLEN	SALESMAN	1981-02-20	1600.00	300.00	30
7521	WARD	SALESMAN	1981-02-22	1250.00	500.00	30
7654	MARTIN	SALESMAN	1981-09-28	1250.00	NULL	30

Q)Display the employee name and employee number of the employees with the headings as NUMBER and NAME.

select ename as 'Employee name', EMPNO as 'Employee number' from emp;

```
select ename as 'Employee name', EMPNO as 'Employee number' from emp;
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

Employee name	Employee number
SMITH	7369
ALLEN	7499
WARD	7521
JONES	7566
MARTIN	7654
BLAKE	7698

Q)Find the names of all employees who were hired on the last day of the month.

select ename,hiredate,last_day(hiredate) as lastday from emp where hiredate=last_day(hiredate);

```
select ename,hiredate,last_day(hiredate) as lastday from emp where hiredate=last_day(hiredate);
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

ename	hiredate	lastday

Q)Find the name of the employee who is receiving the maximum salary.

select ename from emp where SAL = (select max(sal) from emp);

```
select ename from emp where SAL = (select max(sal) from emp);
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

ename
KING

Q)Display the sum of SAL for all the employees belonging to DEPTNO 10.
select sum(sal) from emp where DEPTNO = 10 group by deptno;

```
select sum(sal) from emp where DEPTNO = 10 group by deptno;
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

sum(sal)
8750.00

Q)Display the rows where JOB column ends with the letter 'T'.

select * from emp where job like '%t';

SELECT *,
ROW_NUMBER() OVER() AS row_num
FROM emp where job like '%t';

```
SELECT *,
    ROW_NUMBER() OVER() AS row_num
FROM emp where job like '%t';
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

EMPNO	ENAME	JOB	HIREDATE	SAL	COMM	DEPTNO	row_num
7788	SCOTT	ANALYST	1982-12-09	3000.00	NULL	20	1
7839	KING	PRESIDENT	1981-11-17	5000.00	NULL	10	2
7902	FORD	ANALYST	1981-12-03	3000.00	NULL	20	3

11.Write a stored procedure to convert a temperature in Fahrenheit (F) to its equivalent in Celsius (C). The required formula is:- C= (F-32)*5/9
Insert the temperature in Centigrade into TEMPP table. Calling program for the stored procedure need not be written.

Delimiter @@
Create procedure fahr_to_cel(in f decimal, out c decimal)
begin
select ((F-32)*5/9) into c;
end @@
Delimiter ;

call fahr_to_cel(98,@c);
select @c;

```
Delimiter @@
Create procedure fahr_to_cel(in f decimal, out c decimal)
begin
select ((F-32)*5/9) into c;
end @@
Delimiter ;

call fahr_to_cel(98,@c);
select @c;
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

@c
37

12. Write a stored function by the name of Num_cube. The stored function should return the cube of a number 'N'. The number 'N' should be passed to the stored function as a parameter. Calling program for the stored function need not be written.

```
Delimiter /
create procedure Num_cube(IN n int, out c decimal)
begin
select (n*n*n) into c;
end /
delimiter ;

call Num_cube(5, @c);
```

Result Grid | Filter Rows: | Export: | Wrap C

@c
125

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

create table employee (
emp_id int not null,

```
ename varchar(40) not null,
age int not null,
hobbies varchar(40) not null,
salary int not null,
address varchar(40) not null,
zip int not null,
primary key (emp_id),
unique key (zip),
check (salary > 0)
);


create table dept2(
dept_id int,
dept_name varchar(40),
eid int,
manager varchar(40),
primary key (dept_id),
foreign key (eid) REFERENCES employee (emp_id)
);


insert into employee values (1,'mohit',23,'dancing', 10000, 'Mumbai',500049);
insert into employee values (2,'aniket',27,'painting', 20000, 'mumbai',500149);
insert into employee values (3,'ajay',31,'singing', 35000, 'delhi',273008);
insert into employee values (4,'priyanka',42,'dancing', 55000, 'delhi',123876);
insert into employee values (5,'deepika',26,'dancing', 10000, 'delhi',500786);
insert into employee values (6,'saloni',28,'singing', 50000, 'Mumbai',400149);
insert into employee values (7,'yash',34,'photography', 40000, 'Mumbai',450049);
insert into employee values (8,'vinay',45,'painting', 70000, 'Mumbai',273006);

insert into dept2 values
(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');


========================================
--Query to count No. of employees
```

select count(emp_id) as no_of_employee from employee;

```
select count(emp_id) as no_of_employee from employee;
```

Result Grid | Filter Rows: | Export: | Wrap Cell Conter

no_of_employee
8

--Query to get unique department of employees
select distinct dept_id, eid from dept2;

```
select distinct dept_id, eid from dept2;
```

Result Grid | Filter Rows: | Edit:

dept_id	eid
8	1
7	2
6	3
5	4
4	5
3	6

dept2 47

--Query to get min,max,avg,sum of salary for all employees --get highest salary of an individual based on hobbies
select min(salary), max(salary), avg(salary), sum(salary)
from employee ;

```
select min(salary), max(salary), avg(salary), sum(salary)
from employee ;
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content

min(salary)	max(salary)	avg(salary)	sum(salary)
10000	70000	36250.0000	290000

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

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

```
select sum(salary)
from employee
where
salary in (select salary from employee
      where address like ('M%') or
            address like ('d%'));
```

Result Grid | Filter Rows: | Export:

sum(salary)
290000

Get all employee details with their department details

select emp_id,ename,age,hobbies,salary,address,zip,dept_id,dept_name,manager
from employee e
join dept2 d on
e.emp_id = d.dept_id

```
select emp_id,ename,age,hobbies,salary,address,zip,dept_id,dept_name,manager
from employee e
join dept2 d on
e.emp_id = d.dept_id
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

emp_id	ename	age	hobbies	salary	address	zip	dept_id	dept_name	manager
1	mohit	23	dancing	10000	Mumbai	500049	1	ec	virat
2	aniket	27	painting	20000	mumbai	500149	2	cs	sachin
3	ajay	31	singing	35000	delhi	273008	3	it	rahul
4	priyanka	42	dancing	55000	delhi	123876	4	it	rahul
5	deepika	26	dancing	10000	delhi	500786	5	cs	sachin

QUERY TO FIND employees age between 20 and 30

select * from employee
where age between 20 and 30;

```
82 •   select * from employee
83     where age between 20 and 30;
84
```

Result Grid | Filter Rows: | Edit:

emp_id	ename	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
NULL	NULL	NULL	NULL	NULL	NULL	NULL

Q) function to return name,emp_id,dept_name,hobbies,age by passing manager name

```
Delimiter /
create procedure empdetails(IN manager varchar(30))
begin
select ename,emp_id,dept_name,hobbies,age from employee e
join dept2 d on e.emp_id=d.eid where d.manager=manager;
end /
delimiter

call empdetails ('sachin');
```

```
Delimiter /
create procedure empdetails(IN manager varchar(30))
begin
select ename,emp_id,dept_name,hobbies,age from employee e
join dept2 d on e.emp_id=d.eid where d.manager=manager;
end /
delimiter

call empdetails ('sachin');
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

ename	emp_id	dept_name	hobbies	age
yash	7	cs	photography	34
priyanka	4	cs	dancing	42

Q) FUNCTION RETURNS BELOW DETAILS FROM EMPLOYEE AND DEPARTMENT TABLE
by passing manager name

CREATE MONGO DB COLLECTIONS with following details and insert data
--DB = mongo exam
--Collection = assignment,inventory

--assigment data
{ item: "journal", qty: 25, tags: ["blank", "red"], size: { h: 14, w: 21, uom: "cm" } },
{ item: "mat", qty: 85, tags: ["gray"], size: { h: 27.9, w: 35.5, uom: "cm" } },

{ item: "mousepad", qty: 25, tags: ["gel", "blue"], size: { h: 19, w: 22.85, uom: "cm" } }

--inventory data

{ item: "journal", qty: 25, tags: ["blank", "red"], dim_cm: [14, 21] },
{ item: "notebook", qty: 50, tags: ["red", "blank"], dim_cm: [14, 21] },
{ item: "paper", qty: 100, tags: ["red", "blank", "plain"], dim_cm: [14, 21] },
{ item: "planner", qty: 75, tags: ["blank", "red"], dim_cm: [22.85, 30] },
{ item: "postcard", qty: 45, tags: ["blue"], dim_cm: [10, 15.25] }

```
> db.assignment.insertMany([
    {
    item:"journal",
    qty: 25,
    tags: ["blank","red"],
    size: {h: 14, w: 21, uom: "cm" }
    },
    {
    item:"mat",
    qty:85,
    tags:["grey"],
    size:{h: 27.9, w: 35.5, uom: "cm" }
    },
    {
    items:"mousepad",
    qty:25,
    tags:["gel","blue"],
    size:{h: 19, w: 22.85, uom: "cm"}
    }
  ])
< { acknowledged: true,
    insertedIds:
     { '0': ObjectId("635e60d5f72c6ac34719048c"),
       '1': ObjectId("635e60d5f72c6ac34719048d"),
       '2': ObjectId("635e60d5f72c6ac34719048e") } }
mongo_exam>
```

```
> db.assignment.find();
< { _id: ObjectId("635e60d5f72c6ac34719048c"),
    item: 'journal',
    qty: 25,
    tags: [ 'blank', 'red' ],
    size: { h: 14, w: 21, uom: 'cm' } }
  { _id: ObjectId("635e60d5f72c6ac34719048d"),
    item: 'mat',
    qty: 85,
    tags: [ 'grey' ],
    size: { h: 27.9, w: 35.5, uom: 'cm' } }
  { _id: ObjectId("635e60d5f72c6ac34719048e"),
    items: 'mousepad',
    qty: 25,
    tags: [ 'gel', 'blue' ],
    size: { h: 19, w: 22.85, uom: 'cm' } }
```

```
> db.inventory_data.insertMany(
    [
    { item: "journal", qty: 25, tags: ["blank", "red"], dim_cm: [ 14, 21 ] },
     { item: "notebook", qty: 50, tags: ["red", "blank"], dim_cm: [ 14, 21 ] },
     { item: "paper", qty: 100, tags: ["red", "blank", "plain"], dim_cm: [ 14, 21 ] },
     { item: "planner", qty: 75, tags: ["blank", "red"], dim_cm: [ 22.85, 30 ] },
     { item: "postcard", qty: 45, tags: ["blue"], dim_cm: [ 10, 15.25 ] }

    ]
    )
< { acknowledged: true,
    insertedIds:
     { '0': ObjectId("635e6360f72c6ac34719048f"),
       '1': ObjectId("635e6360f72c6ac347190490"),
       '2': ObjectId("635e6360f72c6ac347190491"),
       '3': ObjectId("635e6360f72c6ac347190492"),
       '4': ObjectId("635e6360f72c6ac347190493") } }
```

```
> db.inventory_data.find();
< { _id: ObjectId("635e6360f72c6ac34719048f"),
    item: 'journal',
    qty: 25,
    tags: [ 'blank', 'red' ],
    dim_cm: [ 14, 21 ] }
  { _id: ObjectId("635e6360f72c6ac347190490"),
    item: 'notebook',
    qty: 50,
    tags: [ 'red', 'blank' ],
    dim_cm: [ 14, 21 ] }
  { _id: ObjectId("635e6360f72c6ac347190491"),
    item: 'paper',
    qty: 100,
    tags: [ 'red', 'blank', 'plain' ],
    dim_cm: [ 14, 21 ] }
  { _id: ObjectId("635e6360f72c6ac347190492"),
    item: 'planner',
    qty: 75,
    tags: [ 'blank', 'red' ],
    dim_cm: [ 22.85, 30 ] }
  { _id: ObjectId("635e6360f72c6ac347190493"),
    item: 'postcard',
    qty: 45,
    tags: [ 'blue' ],
    dim_cm: [ 10, 15.25 ] }
```

Q) get assignment documents having tags = gray

```
> db.assignment.find({tags:"grey"});
< { _id: ObjectId("635e60d5f72c6ac34719048d"),
    item: 'mat',
    qty: 85,
    tags: [ 'grey' ],
    size: { h: 27.9, w: 35.5, uom: 'cm' } }
```

Q) get inventory details whose dim_cm > 10 , sorted by qty descending order and print only 3 documents

```
> db.inventory_data.find({dim_cm:{$gt:10}}).sort({"qty":-1}).limit(3)
< { _id: ObjectId("635e6360f72c6ac347190491"),
    item: 'paper',
    qty: 100,
    tags: [ 'red', 'blank', 'plain' ],
    dim_cm: [ 14, 21 ] }
  { _id: ObjectId("635e6360f72c6ac347190492"),
    item: 'planner',
    qty: 75,
    tags: [ 'blank', 'red' ],
    dim_cm: [ 22.85, 30 ] }
  { _id: ObjectId("635e6360f72c6ac347190490"),
    item: 'notebook',
    qty: 50,
    tags: [ 'red', 'blank' ],
    dim_cm: [ 14, 21 ] }
```

Q) Create index on inventory in descending order of qty

```
> db.inventory_data.getIndexes()
< [ { v: 2, key: { _id: 1 }, name: '_id_' } ]
> db.inventory_data.createIndex({qty:-1})
< 'qty_-1'
> db.inventory_data.getIndexes()
< [
    { v: 2, key: { _id: 1 }, name: '_id_' },
    { v: 2, key: { qty: -1 }, name: 'qty_-1' }
  ]
```

Q) Query to aggregate sum of qty in inventory collection

```
> db.inventory_data.aggregate({$group:{_id:1,total:{$sum:'$qty'}}})
< { _id: 1, total: 295 }
mongo_exam>
```

Q) query to update inventory collection item name where qty:75 and dim_cm > 22

```
> db.inventory_data.updateMany({qty:75,dim_cm:{$gt:22}},{$set:{item:"anything"}})
< { acknowledged: true,
    insertedId: null,
    matchedCount: 1,
    modifiedCount: 1,
    upsertedCount: 0 }
mongo_exam>
```