1. Consider the following schema:

Sailors (sid, sname, rating, age)

Boats (bid, bname, color)

Reserves (sid, bid, day(date))

CREATE TABLE Sailors (

sid INT PRIMARY KEY,

sname VARCHAR(50) NOT NULL,

rating INT CHECK (rating >= 1 AND rating <= 10),

age INT CHECK (age > 0)

);

CREATE TABLE Boats (

bid INT PRIMARY KEY,

bname VARCHAR(50) NOT NULL,

color VARCHAR(20) NOT NULL

);

CREATE TABLE Reserves (

sid INT,

bid INT,

day DATE,

PRIMARY KEY (sid, bid, day),

FOREIGN KEY (sid) REFERENCES Sailors(sid) ON DELETE CASCADE,

FOREIGN KEY (bid) REFERENCES Boats(bid) ON DELETE CASCADE

);

INSERT INTO Sailors (sid, sname, rating, age) VALUES

(1, 'Alice', 5, 25),

(2, 'Bob', 3, 30),

(3, 'Charlie', 4, 22),

(4, 'David', 3, 28),

(5, 'Eve', 4, 35),

(6, 'Frank', 5, 20),

(7, 'Grace', 3, 32);

INSERT INTO Boats (bid, bname, color) VALUES

(101, 'Titan', 'Red'),

(102, 'Wave', 'Blue'),

(103, 'Storm', 'Green'),

(104, 'Breeze', 'Red');

INSERT INTO Reserves (sid, bid, day) VALUES

(1, 101, '2025-03-01'),

(2, 102, '2025-03-02'),

(2, 103, '2025-03-03'),

(3, 104, '2025-03-04'),

(4, 101, '2025-03-05'),

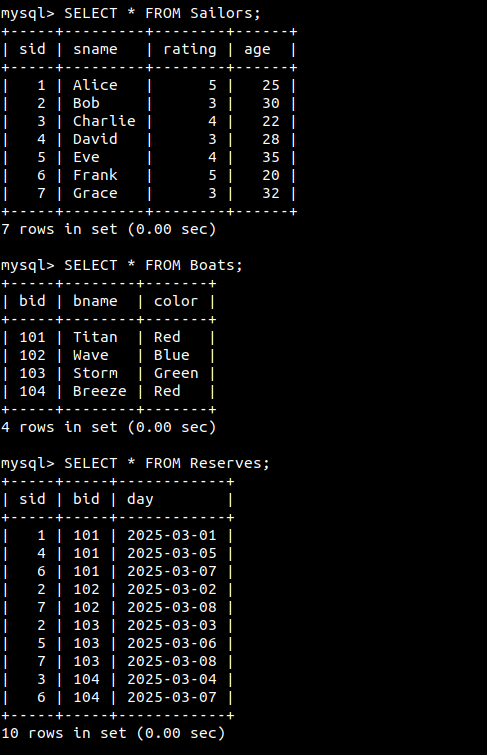
(5, 103, '2025-03-06'),

(6, 104, '2025-03-07'),

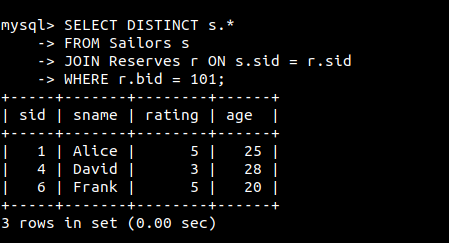
(6, 101, '2025-03-07'),

(7, 102, '2025-03-08'),

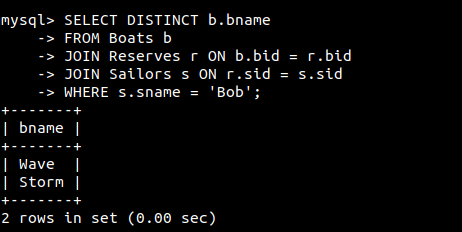
(7, 103, '2025-03-08');



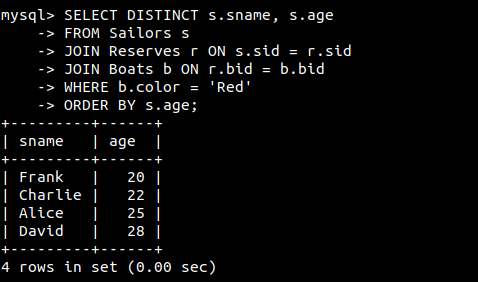
1. Find all information of sailors who have reserved boat number 101.



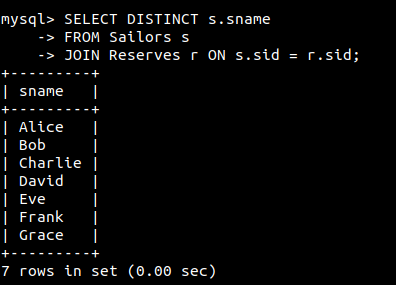
1. Find the name of the boat reserved by Bob.



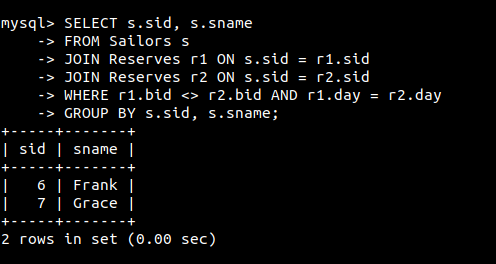
1. Find the names of sailors who have reserved a red boat, and list in the order of age.



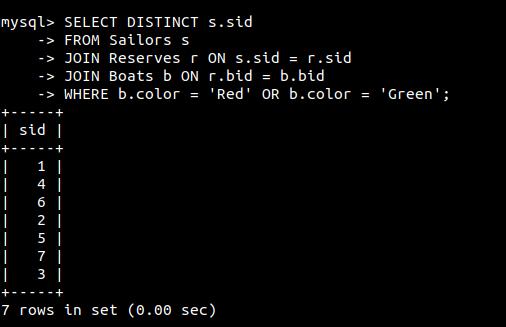
1. Find the names of sailors who have reserved at least one boat.



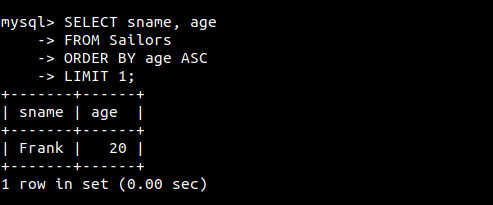
1. Find the ids and names of sailors who have reserved two different boats on the same day.



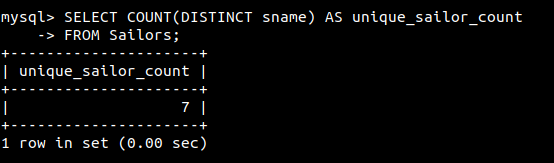
1. Find the ids of sailors who have reserved a red boat or a green boat.



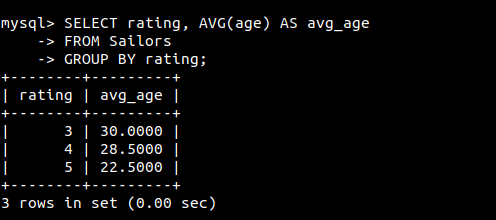
1. Find the name and the age of the youngest sailor.



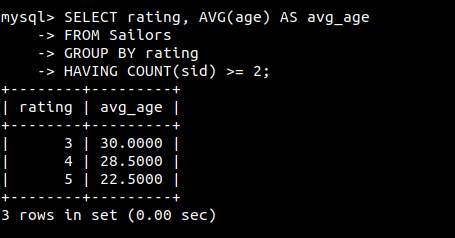
1. Count the number of different sailor names.



1. Find the average age of sailors for each rating level.



1. Find the average age of sailors for each rating level that has at least two sailors.



2. Answer the following questions based on given table.

Salesman Table

| salesman\_id | name | city | commission |
| --- | --- | --- | --- |
| 5001 | James Hoog | New York | 0.15 |
| 5002 | Nail Knite | Paris | 0.13 |
| 5005 | Pit Alex | London | 0.11 |
| 5006 | Mc Lyon | Paris | 0.14 |
| 5007 | Paul Adam | Rome | 0.13 |
| 5003 | Lauson Hen | San Jose | 0.12 |

Customer Table

| customer\_id | cust\_name | city | grade | salesman\_id |
| --- | --- | --- | --- | --- |
| 3002 | Nick Rimando | New York | 100 | 5001 |
| 3007 | Brad Davis | New York | 100 | 5001 |
| 3005 | Graham Zusi | California | 200 | 5002 |
| 3008 | Julian Green | London | 300 | 5002 |
| 3004 | Fabian Johnson | Paris | 200 | 5006 |
| 3009 | Geoff Cameron | Berlin | 100 | 5003 |
| 3003 | Jozy Altidore | Moscow | 200 | 5007 |
| 3001 | Brad Guzan | London |  | 5005 |

Orders Table

| ord\_no | purch\_amt | ord\_date | customer\_id | salesman\_id |
| --- | --- | --- | --- | --- |
| 70001 | 150.5 | 2012-10-05 | 3005 | 5002 |
| 70009 | 270.65 | 2012-09-10 | 3001 | 5005 |
| 70002 | 65.26 | 2012-10-05 | 3002 | 5001 |
| 70004 | 110.5 | 2012-08-17 | 3009 | 5003 |
| 70007 | 948.5 | 2012-09-10 | 3005 | 5002 |
| 70005 | 2400.6 | 2012-07-27 | 3007 | 5001 |
| 70008 | 5760 | 2012-09-10 | 3002 | 5001 |
| 70010 | 1983.43 | 2012-10-10 | 3004 | 5006 |
| 70003 | 2480.4 | 2012-10-10 | 3009 | 5003 |
| 70012 | 250.45 | 2012-06-27 | 3008 | 5002 |
| 70011 | 75.29 | 2012-08-17 | 3003 | 5007 |
| 70013 | 3045.6 | 2012-04-25 | 3002 | 5001 |

CREATE TABLE Salesman (

salesman\_id INT PRIMARY KEY,

name VARCHAR(50),

city VARCHAR(50),

commission DECIMAL(4,2)

);

CREATE TABLE Customer (

customer\_id INT PRIMARY KEY,

cust\_name VARCHAR(50),

city VARCHAR(50),

grade INT,

salesman\_id INT,

FOREIGN KEY (salesman\_id) REFERENCES Salesman(salesman\_id)

);

CREATE TABLE Orders (

ord\_no INT PRIMARY KEY,

purch\_amt DECIMAL(10,2),

ord\_date DATE,

customer\_id INT,

salesman\_id INT,

FOREIGN KEY (customer\_id) REFERENCES Customer(customer\_id),

FOREIGN KEY (salesman\_id) REFERENCES Salesman(salesman\_id)

);

INSERT INTO Salesman (salesman\_id, name, city, commission) VALUES

(5001, 'James Hoog', 'New York', 0.15),

(5002, 'Nail Knite', 'Paris', 0.13),

(5005, 'Pit Alex', 'London', 0.11),

(5006, 'Mc Lyon', 'Paris', 0.14),

(5007, 'Paul Adam', 'Rome', 0.13),

(5003, 'Lauson Hen', 'San Jose', 0.12);

INSERT INTO Customer (customer\_id, cust\_name, city, grade, salesman\_id) VALUES

(3002, 'Nick Rimando', 'New York', 100, 5001),

(3007, 'Brad Davis', 'New York', 100, 5001),

(3005, 'Graham Zusi', 'California', 200, 5002),

(3008, 'Julian Green', 'London', 300, 5002),

(3004, 'Fabian Johnson', 'Paris', 200, 5006),

(3009, 'Geoff Cameron', 'Berlin', NULL, 5003),

(3003, 'Jozy Altidore', 'Moscow', 200, 5007),

(3001, 'Brad Guzan', 'London', NULL, 5005);

INSERT INTO Orders (ord\_no, purch\_amt, ord\_date, customer\_id, salesman\_id) VALUES

(70001, 150.50, '2012-10-05', 3005, 5002),

(70009, 270.65, '2012-09-10', 3001, 5005),

(70002, 65.26, '2012-10-05', 3002, 5001),

(70004, 110.50, '2012-08-17', 3009, 5003),

(70007, 948.50, '2012-09-10', 3005, 5002),

(70005, 2400.60, '2012-07-27', 3007, 5001),

(70008, 5760.00, '2012-09-10', 3002, 5001),

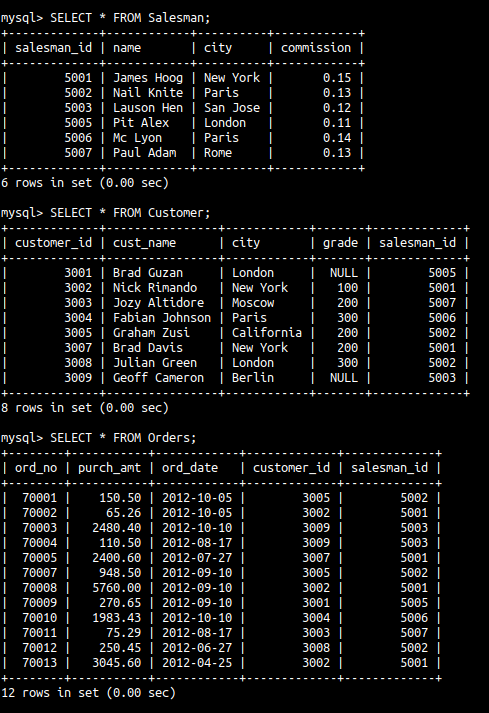
(70010, 1983.43, '2012-10-10', 3004, 5006),

(70003, 2480.40, '2012-10-10', 3009, 5003),

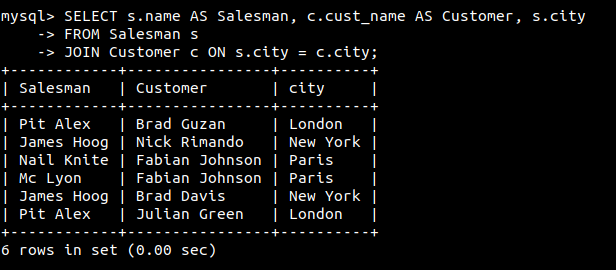
(70012, 250.45, '2012-06-27', 3008, 5002),

(70011, 75.29, '2012-08-17', 3003, 5007),

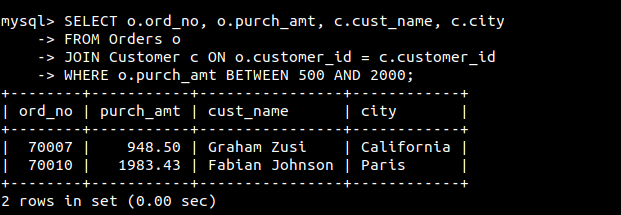
(70013, 3045.60, '2012-04-25', 3002, 5001);



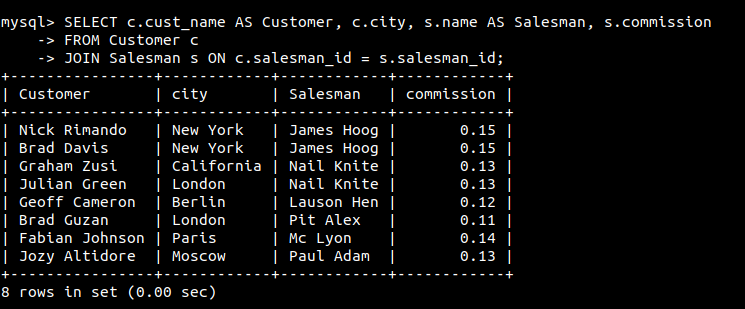
1. From Salesman and Customer tables, write a SQL query to find the salesperson and customer who belongs to the same city. Return Salesman, cust\_name and city.



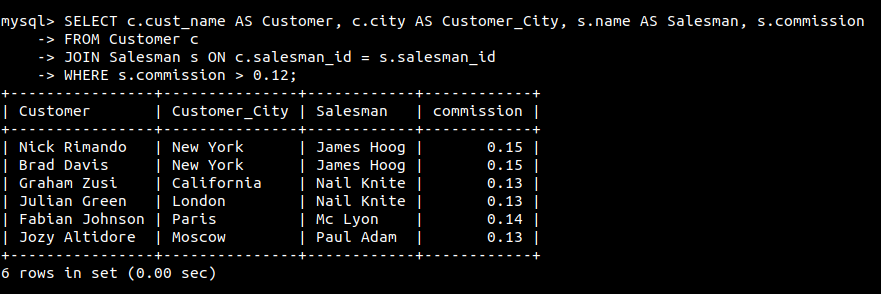
1. From Orders and Customer tables, write a SQL query to find those orders where order amount exists between 500 and 2000. Return ord\_no, purch\_amt, cust\_name, city.



1. From Customer and Salesman tables, write a SQL query to find the salesperson(s) and the customer(s) he handles. Return Customer Name, city, Salesman, commission.



1. From Customer and Salesman tables, write a SQL query to find those salespersons who received a commission from the company more than 12%. Return Customer Name, customer city, Salesman, commission.



1. From Orders and Customer tables, write a SQL statement to make a report with customer name, city, order number, order date, and order amount in ascending order according to the order date to find that either any of the existing customers have placed no order or placed one or more orders.

