Create the following tables with the data as shown below

CREATE TABLE Employee (

Employee\_id int AUTO\_INCREMENT PRIMARY KEY,

First\_name VARCHAR(50),

Last\_name VARCHAR(50),

Salary int,

Joining\_date Date,

Departement VARCHAR(50)

);

CREATE TABLE reward (

Employee\_ref\_id int,

date\_reward Date,

amount int,

FOREIGN KEY (Employee\_ref\_id) REFERENCES

Employee(Employee\_id)

);

INSERT INTO Employee (Employee\_id, First\_name, Last\_name, Salary, Joining\_date, Departement) VALUES (1, 'Bob', 'Kinto', 1000000, "2019-01-20", "Finance");

INSERT INTO Employee (Employee\_id, First\_name, Last\_name, Salary, Joining\_date, Departement) VALUES (2, 'Jerry', 'Kansxo', 6000000, "2019-01-15", "IT");

INSERT INTO Employee (Employee\_id, First\_name, Last\_name, Salary, Joining\_date, Departement) VALUES (3, 'Philip', 'Jose', 8900000, "2019-02-05", "Banking");

INSERT INTO Employee (Employee\_id, First\_name, Last\_name, Salary, Joining\_date, Departement) VALUES (4, 'John', 'Abraham', 2000000, "2019-02-25", "Insurance");

INSERT INTO Employee (Employee\_id, First\_name, Last\_name, Salary, Joining\_date, Departement) VALUES (5, 'Michael', 'Mathew', 2200000, "2019-02-28", "Finance");

INSERT INTO Employee (Employee\_id, First\_name, Last\_name, Salary, Joining\_date, Departement) VALUES (6, 'Alex', 'chreketo', 4000000, "2019-05-10", "IT");

INSERT INTO Employee (Employee\_id, First\_name, Last\_name, Salary, Joining\_date, Departement) VALUES (7, 'Yohan', 'Soso', 1230000, "2019-06-20", "Banking");

INSERT INTO reward (Employee\_ref\_id, date\_reward, amount) VALUES (1, '2019-05-11', '1000');

INSERT INTO reward (Employee\_ref\_id, date\_reward, amount) VALUES (2, '2019-02-15', '5000');

INSERT INTO reward (Employee\_ref\_id, date\_reward, amount) VALUES (3, '2019-04-22', '2000');

INSERT INTO reward (Employee\_ref\_id, date\_reward, amount) VALUES (1, '2019-06-20', '8000');

**1. Get all employees.**  
**2. Display the first name and last name of all employees.**  
**3. Display all the values of the “First\_Name” column using the alias “Employee Name”**  
**4. Get all “Last\_Name” in lowercase.**  
**5. Get all “Last\_Name” in uppercase.**  
**6. Get unique “DEPARTMENT”.**  
**7. Get the first 4 characters of “FIRST\_NAME” column.**  
**8. Get the position of the letter ‘h’ in ‘John’.**  
**9. Get all values from the “FIRST\_NAME” column after removing white space on the right.**  
**10. Get all values from the “FIRST\_NAME” column after removing white space on the left.**  
**11. Write the syntax to create the “employee” table.**

Questions

**1. Get the length of the text in the “First\_name” column.**

**2. Get the employee’s first name after replacing ‘o’ with ‘#’.**

**3. Get the employee’s last name and first name in a single column separated by a ‘\_’.**

**4. Get the year, month, and day from the “Joining\_date” column.**

**5. Get all employees in ascending order by first name.**

6. **Get all employees in descending order by first name.**

**7. Get all employees in ascending order by first name and descending order by salary.**

**8. Get employees whose first name is “Bob”.**

**9. Get employees whose first name is “Bob” or “Alex”.**

**10. Get employees whose first name is neither “Bob” nor “Alex”.**

**11. What is SQL injection?**

**1. Get all the details about employees whose first name begins with ‘B’.**

**2. Get all the details about employees whose first name contains ‘o’.**

**3. Get all the details of the employees whose first name ends with ‘n’.**

**4. Get all the details about employees whose first name ends with ‘n’ and contains 4 letters.**

**5. Get all the details about employees whose first name begins with ‘J’ and contains 4 letters.**

**6. Get all the details of employees whose salary is over 3,000,000.**

**7. Get all the details about employees whose salary is less than 3,000,000.**

**8. Get all the details about employees with a salary between 2,000,000 and 5,000,000.**

**9. Get all the details about employees whose first name is ‘Bob’ or ‘Alex’.**

**10. Get all the details about employees whose joining year is “2019”.**

Questions

**1. Get all the details on employees whose participation month (Joining\_date) is “January”**

**2. Get all the details of the employees who joined before March 1, 2019**

**3. Get all the details on employees who joined after March 31, 2019**

**4. Get the date and time of the employee’s enrollment.**

**5. Get the date and time, including milliseconds of the employee’s membership.**

**7. Get the current date and time.**

**8. Get the first names of employees who have the character ‘%’. Example: ‘Jack%’.**

**9. Get the employee name (Last\_name) after replacing the special character with white space.**

**10. Get the employee’s department and total salary, grouped by department.**

**1. Get the department and total salary, grouped by department, and sorted by total salary in descending order.**

**2. Get the department, the number of employees in that department, and the total salary grouped by department, and sorted by total salary in descending order.**

**3. Get the average salary by department in ascending order of salary.**

**4. Get the maximum salary by department in ascending order of salary.**

**5. Get the minimum salary by department in ascending order of salary.**

**6. Get the number of employees grouped by year and month of membership.**

**7. Get the department and total salary grouped by the department, where the total salary is greater than 1,000,000, and sorted in descending order of the total salary.**

**8. Get all the details of an employee if the employee exists in the Reward table? Or in other words, find employees with bonuses.**

**9. How to get common data in two query results?**

**10. Get the IDs of employees who did not receive rewards without using subqueries?**

###### Questions

**1. Get 20% of Bob’s salary, 10% of Alex’s salary, and 15% of other employees’ salaries.**

**2. Display the text**

**‘IT services’ instead of ‘IT’,**

**‘Financial services’ instead of ‘Finance’, and**

**‘Banking services’ instead of ‘Banking’**

**from the “Department” column.**

**3. Remove employees who have received rewards.**

**4. Insert employee whose name contains a single quote '**

**5. Get the name of the employees which contains only numbers.**

**6. Classify employees according to their reward for one month.**

**7. Update the reward of “Bob” to 1000.**

**8. Get the first name, the amount of the reward for the employees who have rewards.**

**9. Get the first name, the reward amount for employees who have rewards with an amount greater than 2000.**

**10. Get the first name, the amount of the reward for the employees even if they have not received any rewards.**

**1. Get the first name, the reward amount for employees even if they did not receive any rewards, and set a reward amount equal to 0 for the employees who did not receive rewards.**

**2. Get the employee’s first name, the reward amount for employees who have rewards.**

**3. Get the first name, the reward amount for employees who have rewards using “Right Join”.**

**4. Get the maximum reward per employee using subquery.**

**5. Get the TOP salary of two employees**

**6. Get the TOP salary of N employees.**

**7. Get the 2nd highest salary of an employee.**

**8. Get the Nth highest salary of an employee.**

**9. Get the employee’s first name and last name in separate lines.**

**10. What is the difference between UNION and UNION ALL?**

###### Cities table

**+------+---------------+------------+----------+------------+**

**| id | name | population | surface | city\_state |**

**+------+---------------+------------+----------+------------+**

**| 1 | New York | 463333 | 15.31667 | 01 |**

**| 2 | Albany | 391234 | 16.99663 | 01 |**

**| 3 | Buffalo | 402356 | 10.12345 | 01 |**

**| 4 | San Bruno | 153233 | 9.22147 | 02 |**

**| 5 | SAN-Francisco | 205689 | 11.99632 | 02 |**

**| 6 | SAN-Diego | 269988 | 19.89451 | 02 |**

**| 7 | Houston | 197009 | 18.00001 | 03 |**

**| 8 | Chicago | 239878 | 22.01250 | 04 |**

**+------+---------------+------------+----------+------------+**

###### States table

**+----+------------+------------+**

**| id | state\_code | state\_name |**

**+----+------------+------------+**

**| 1 | 01 | New York |**

**| 2 | 02 | California |**

**| 3 | 03 | Texas |**

**| 4 | 04 | Illinois |**

**| 5 | 978 | Florida |**

**| 6 | 971 | Indiana |**

**+----+------------+------------+**

Questions

**1. Get the list of the 3 most populated cities.**

**2. Get the list of the 3 cities with the smallest surface.**

**3. Get the list of states whose department number starts with “97”.**

**4. Get the names of the 3 most populated cities, as well as the name of the associated state.**

**5. Get the list of the name of each State, associated with its code and the number of cities within these States, by sorting in order to get in priority the States which have the largest number of cities.**

**6. Get the list of the 3 largest States, in terms of surface area.**

**7. Count the number of cities whose names begin with “San”.**

**8. Get the list of cities whose surface is greater than the average surface.**

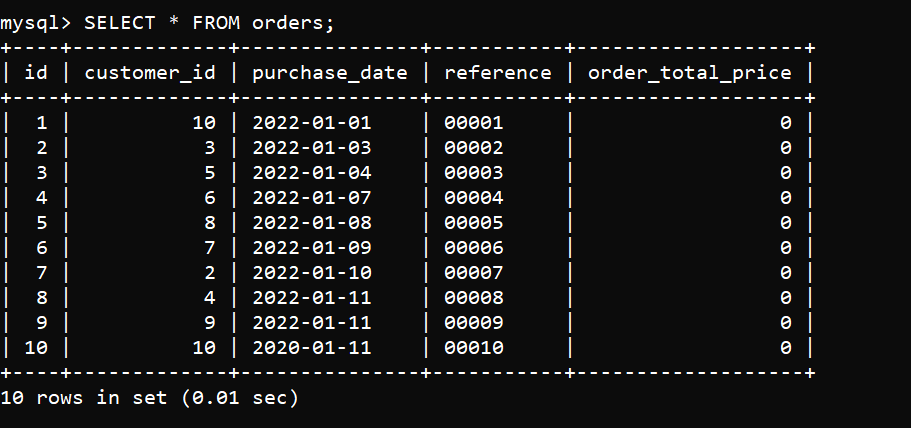
**9. Get the list of States with more than 1 million residents.**

**10. Replace the dashes with a blank space, for all cities beginning with “SAN-” (inside the column containing the upper case names).**

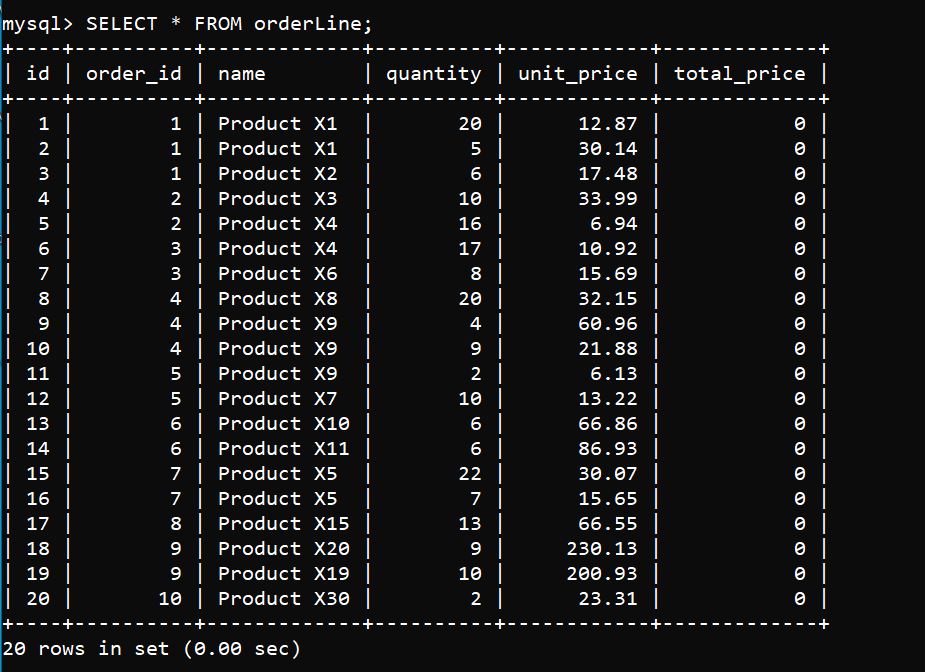
###### Customer table



###### Orders table



###### OrderLine table



**1. Get the user with the first name “Ivan” and the password “password2”, considering that the encoding of the password is done with the Sha1 algorithm.**

**2. Get the list of all products that are present on several orders.**

**3. Get the list of all the products that are present on several orders and add a column that lists the id of the associated orders.**

**4. Store the total price within each order line, based on the unit price and quantity.**

**5. Get the total price for each order and the date associated with that order as well as the first and last name of the associated customer.**

**6. Store the total price of each order in the field named “order\_total\_price”.**

**7. Get the total price of all orders, for each month.**

**8. Get a list of the 10 customers who made the largest amount of orders, and get this total price for each customer.**

**9. Get the total price of orders for each date.**

**10. Add a column named “category” to the table containing the orders. This column will contain a numerical value.**

**11. Enter the value of the category, according to the following rules:**

**“1” for orders under 200€.**

**“2” for orders between 200€ and 500€.**

**“3” for orders between 500€ and 1.000€.**

**“4” for orders over 1.000€.**

**12. Create a table called “category\_order” which will contain the description of these categories.**

**13. Insert the 4 descriptions of each category in the table previously created.**

**“1” for orders under 200€.**

**“2” for orders between 200€ and 500€.**

**“3” for orders between 500€ and 1.000€.**

**“4” for orders over 1.000€.**

**14. Delete all orders (and orderLines) below January 7, 2022.**